法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

- 课程详情请咨询
  - 微信公众号：大数据分析挖掘
  - 新浪微博：ChinaHadoop

分布式爬虫

大纲

- Tesseract-OCR
- 图片相似度匹配
- 在线接口的使用
- 微博数据抓取

Tesseract-Ocr

Pillow

Pillow 是一个图像工具包，包含了一个 Image 类用来做图像的处理

```
pip install pillow
```

```
from PIL import Image
def extract_image(html):
    tree = lxml.html.fromstring(html)
    img_data = tree.cssselect('div#recaptcha img')[0].get('src')
    img_data = img_data.partition(',')[-1]
    binary_img_data = img_data.decode('base64')
    img_data = BytesIO(binary_img_data)
    img = Image.open(img_data)
    img.save('test.png')
    return img
```

Tesseract-Ocr

Tesseract-Ocr 是一个 Google 主导的开源 OCR (Optical Character Recongnition) 引擎。Tesseract-Ocr 有很多的 python 开源版本

pip install pytesseract

```
import pytesseract

pytesseract.image_to_string(bw)
```

识别过程

大量验证码都是添加了干扰元素的，因此第一步要找出噪声并去除掉

http://www.bjhjyd.gov.cn/

找出验证码的色彩

对色彩像素进行统计

```
pixdata = img.load()
colors = {}
# 统计字符颜色像素情况
for y in range(img.size[1]):
    for x in range(img.size[0]):
        if colors.has_key(pixdata[x,y]):
            colors[pixdata[x, y]] += 1
        else:
            colors[pixdata[x,y]] = 1

# 排名第一的是背景色，第二的是主要颜色
colors = sorted(colors.items(), key=lambda d:d[1], reverse=True)
```

((240, 240, 240), 1996) - 排名第一的是背景色
((51, 153, 0), 645) – 排名第二的是验证码字体颜色
((241, 244, 237), 168),
((192, 168, 185), 37),
((161, 250, 53), 1)

去噪

把验证码色彩设置为黑色，其余颜色设置为白色

```
significant = colors[1][0]
for y in range(img.size[1]):
   for x in range(img.size[0]):
      if pixdata[x,y] != significant:
         pixdata[x,y] = (255,255,255)
      else:
         pixdata[x, y] = (0,0,0)
```

调用 TesseractOcr 进行识别

```
word = pytesseract.image_to_string(img, lang='eng', config='ocr.conf')
```

lang 指定识别的语言

config 指定配置文件，我们设置了有效字符仅包含A~Za~z0~9

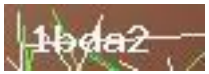
tessedit_char_whitelist
abdefghijklmnoprstuvwxyzABDEFGHIJKLMNOPQRSTUVWXYZ12
34567890

一共 12 个验证码，识别正确了4个，正确率 33%

图片匹配

标准字体的图片

考虑下面这种类型的图片

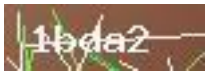


用 Tesseract-Ocr 完全不能识别，干扰信息太多，而且干扰笔画的色彩与验证码一样

仔细观察，这些字体都比较标准，我们可以考虑用图片相似度匹配的方式来识别

标准字体的图片

考虑下面这种类型的图片



用 Tesseract-Ocr 完全不能识别，干扰信息太多，而且干扰笔画的色彩与验证码一样

仔细观察，这些字体都比较标准，我们可以考虑用图片相似度匹配的方式来识别

图片匹配

- 把所有的图片找出来，裁剪并拼接成如下的样子

- 把验证码图片中文字部分剪裁出来

- 把验证码图片转化为黑白，设定一个阈值200，小于200的处理为白色

- 将参考字体与验证码每个字体比对，计算它们的距离，计算方式为每个像素的色彩差之和

$$distance = \sum_{i=0}^{n} p_i - l_i$$

图片匹配

- 把所有的图片找出来，裁剪并拼接成如下的样子

- 把验证码图片中文字部分剪裁出来

- 把验证码图片转化为黑白，设定一个阈值200，小于200的处理为白色

- 将参考字体与验证码每个字体比对，计算它们的距离，计算方式为每个像素的色彩差之和

$$distance = \sum_{i=0}^{n} p_i - l_i$$

在线人工服务

将图片发送到注册的在线服务，由它们人工判别并返回

```
data = {
    'action': 'usercaptchaupload',
    'apikey': api_key,
    'file-upload-01': img_data.encode('base64'),
    'base64': '1',
    'selfsolve': '1',
    'maxtimeout': str(self.timeout)
}
encoded_data = urllib.urlencode(data)
request = urllib2.Request(self.url, encoded_data)
response = urllib2.urlopen(request)
result = response.read()
```

新浪微博

登录

最重要的是设置 User-Agent，否则无法跳转链接

```
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities

user_agent = (
    "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_4) " +
    "AppleWebKit/537.36 (KHTML, like Gecko) Chrome/29.0.1547.57 Safari/537.36"
)

dcap = dict(DesiredCapabilities.PHANTOMJS)
dcap["phantomjs.page.settings.userAgent"] = user_agent

driver = webdriver.PhantomJS(desired_capabilities=dcap)
```

输入用户名与密码


```
<input id="loginname"
type="text"
class="W_input " maxlength="128"
autocomplete="off"
action-data="text=邮箱/会员帐号/手机号"
action-type="text_copy"
name="username"
node-type="username" tabindex="1">
```

输入用户名与密码

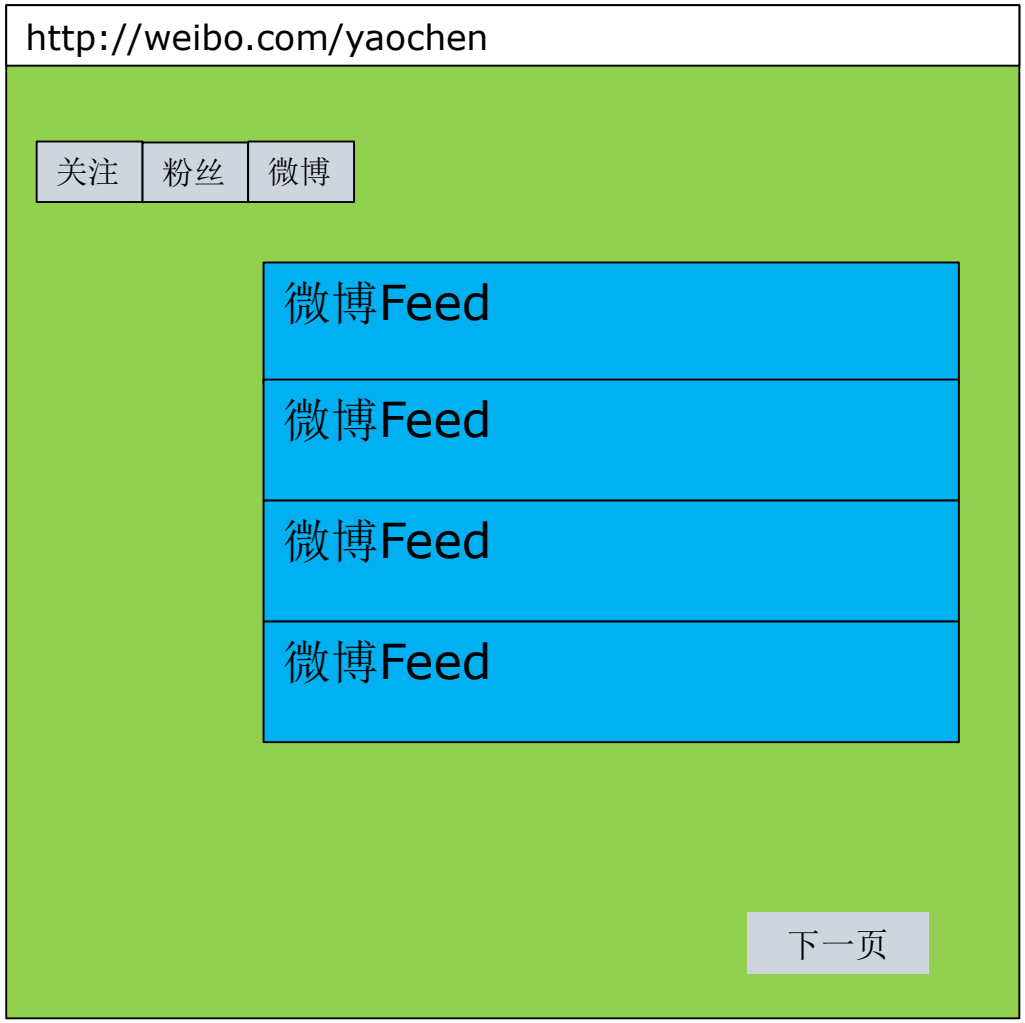
标准 javascript 是

```
document.getElementById('loginname').value='abc'
document.getElementsByName('password')[0].value='abc'
```

通过 Selenium 提供的 send_keys 来传递 value

```
driver.find_element_by_id('loginname').send_keys(username)
driver.find_element_by_name('password').send_keys(password)
```

微博 Web 图分析


http://weibo.com/yaochen
关注
粉丝
微博
微博Feed
微博Feed
微博Feed
微博Feed
下一页

关注、粉丝

关注列表、粉丝列表，作为漫游Weibo的外链

获得微博外链

driver.find_element_by_xpath('//a[@class="t_link S_txt1"]')

0 是关注，1是粉丝，2是微博，我们只需要 0，关注的微博一般是有质量的，而粉丝的数量太多，并且有太多僵尸粉

打开关注列表页：

driver.find_element_by_xpath('//a[@class="t_link S_txt1"]').get_attribute('href')

获取所有关注的微博号的地址：

driver.find_elements_by_xpath('//*[contains(@class, "follow_item")]//a[@class="S_txt1"]')

获取微博用户信息

- 提取用户的基本信息
  - 链接：用正则表达式把用户的链接参数都去掉

  /u/1634431184?refer_flag=1005050006_
  - 微博昵称及头像
  - 关注、粉丝及微博数量

- 过滤质量差的用户。对于微博数量少于阈值，或者关注数超过粉丝数 N 倍以上的，判定为僵尸粉或广告微博，直接跳过
  - 僵尸粉：微博数量极少
  - 纯广告、营销微博：关注数远远超过粉丝数量

- 提取下一页，可以继续查找更多的user

获得微博外链

driver.find_element_by_xpath('//a[@class="t_link S_txt1"]')

0 是关注，1是粉丝，2是微博，我们只需要 0，关注的微博一般是有质量的，而粉丝的数量太多，并且有太多僵尸粉

打开关注列表页：

driver.find_element_by_xpath('//a[@class="t_link S_txt1"]').get_attribute('href')

获取所有关注的微博号的地址：

driver.find_elements_by_xpath('//*[contains(@class, "follow_item")]//a[@class="S_txt1"]')

微博信息抽取

微博名：driver.find_element_by_tag_name('h1')

所有的Feed：driver.find_elements_by_class_name('WB_detail')

```
feed = {}
feed['time'] = element.find_element_by_xpath('.//div[@class="WB_from S_txt2"]').text
feed['content'] = element.find_element_by_class_name('WB_text').text
feed['image_names'] = []
for image in element.find_elements_by_xpath('.//li[contains(@class,"WB_pic")]/img'):
    feed['image_names'].append(re.findall('/([^/]+)$', image.get_attribute('src')))
```

微博的图片，只需要保存图片名

http://wx2.sinaimg.cn/thumb150/4b7a8989ly1fcws2sryvrj22p81sub2a.jpg

http://存储域名/分辨率/文件名

微博图片信息

re.findall('/([^/]+)$', image.get_attribute('src'))

微博的图片，只需要保存图片名

http://wx2.sinaimg.cn/thumb150/4b7a8989ly1fcws2sryvrj22p81sub2a.jpg

http://存储域名/分辨率/文件名

名称	宽度	定义
thumb150	150 像素	缩略图
mw690	690 像素	中图
mw1024	1024像素	大图

滚频与翻页

每次滚动后，检查是否已经出现了

- 微博的下一页的 class：

点击重新载入

```
page next S_txt1 S_line1
driver.find_element_by_xpath('//a[@class="page next S_txt1 S_line1"]').click()
```

- 翻页命令

```
driver.execute_script('window.scrollTo(0, document.body.scrollHeight)')
```

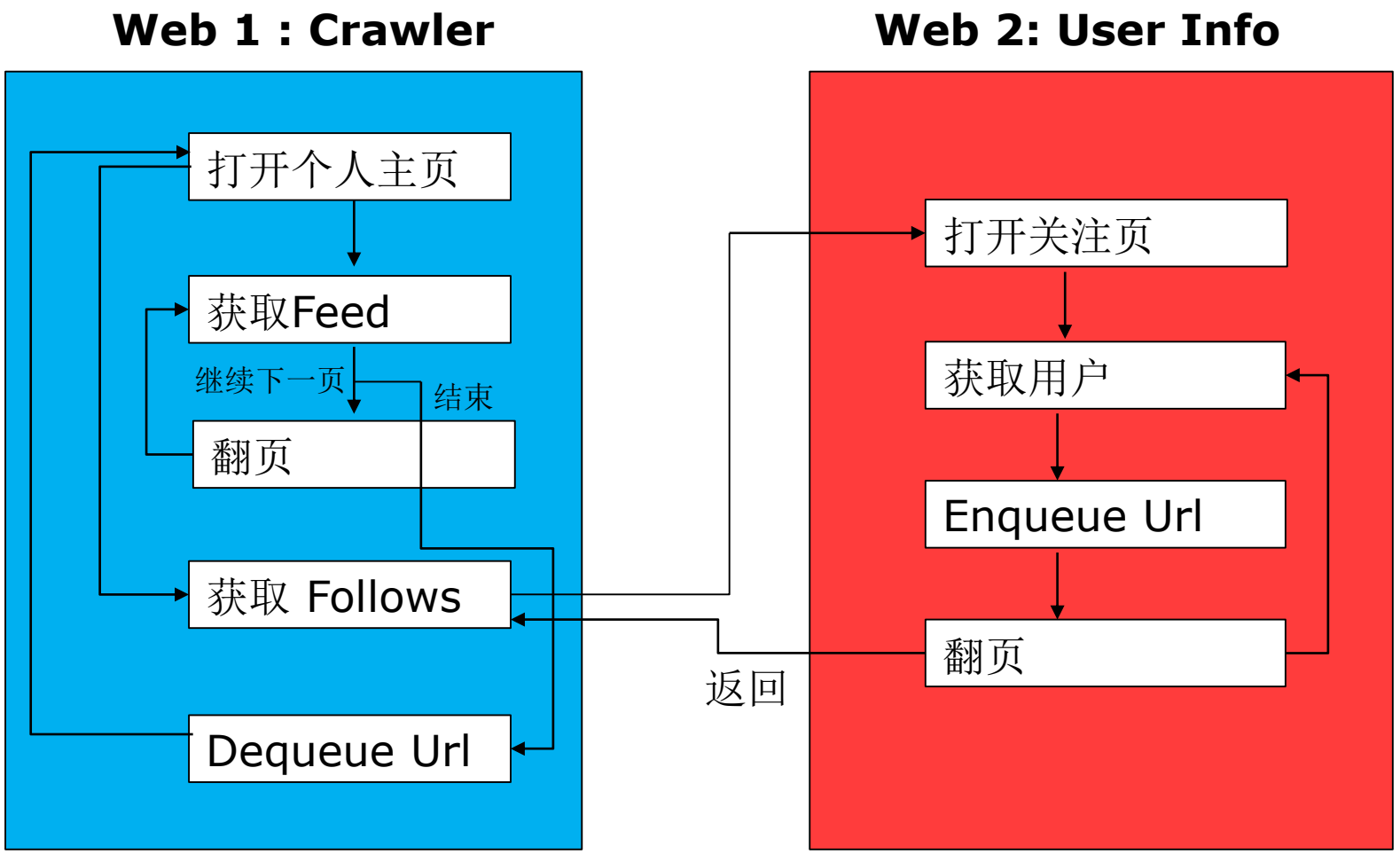
滚屏与翻页

每次滚动后，检查是否已经出现了“下一页”的按钮，如果是则可以停止翻页，否则检查是否出现了“网络超时”的链接，是的话，点击这个链接来重新加载

滚屏

```
for i in range(0,10):
    driver.execute_script('window.scrollTo(0, document.body.scrollHeight)')
    html = driver.page_source
    tr = etree.HTML(html)
    next_page_url = tr.xpath('//a[contains(@class,"page next")]')
    if len(next_page_url) > 0:
        return next_page_url[0].get_attribute('href')
    if len(re.findall('点击重新载入', html)) > 0:
        driver.find_element_by_link_text('点击重新载入').click()
```

微博抓取框架


Web 1 : Crawler
Web 2: User Info
打开个人主页
获取Feed
继续下一页
结束
翻页
获取 Follows
Dequeue Url
打开关注页
获取用户
Enqueue Url
翻页
返回

疑问

- 问题答疑： http://www.xxwenda.com/
  - 可邀请老师或者其他人回答问题

联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：大数据分析挖掘
- 新浪微博：ChinaHadoop