# Summary of Chinese Celebrities Data Collecting & Cleaning

Na Zhang

## Outline

1. Create Name List
2. Download Images

# 1. Create Name List

- Goal
  - ✔ Find Chinese Celebrities
  - ✔ Contain persons in
    - Mainland China
    - Hong Kong
    - Taiwan

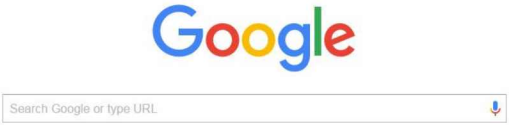


# Occupation includes

- Singers
- Actors/Actresses
- Models
- Host/Hostess
- Directors
- Athletes
- Dancers
- Entertainers
- Musician
- ……

# How to find?

- Mainly Chinese Websites
  - Use some famous Search Engines, like Google, Baidu.
  - Baidu，largest Chinese search engine
  - Search by keywords







# Name List Sources

- Tencent entertainment 腾讯娱乐资料库
- 6Kstar 乐酷明星网
- TTpaihang 天天排行榜
- StarRank 百度百科人气榜
- n63.com 明星写真馆
- Zhidao 百度知道
- TopBaidu 百度搜索风云榜
- Other sources…

# (1) 腾讯娱乐资料库 ( Tencent Entertainment)

- http://ent.qq.com/c/dalu_star.shtml




# (2) 乐酷明星网 ( 6Kstar)

- http://star.6k.com/diqu/zhongguo/

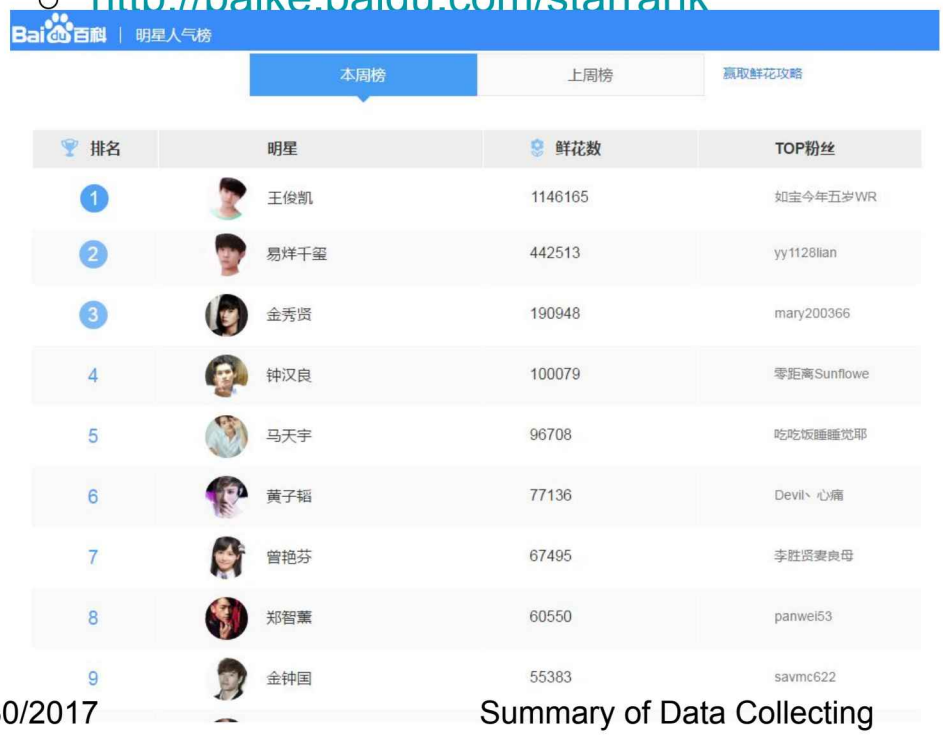
# (3) 天天排行榜 (ttpaihang)

- http://www.ttpaihang.com/subjects/starworld/




# (4) 百度百科人气榜 (star rank)

- http://baike.baidu.com/starrank



# (5) 明星写真馆 (n63)

- http://www.n63.com/




# (6) 百度知道 ( zhidao )

https://zhidao.baidu.com/question/369594442.html

# (7) 百度搜索风云榜 ( top baidu)

http://top.baidu.com/buzz?b=258&c=9&fr=topbuzz_b454_c9




# (8) Other sources

**athletes**: http://blog.sina.com.cn/s/blog_5f05e98f0102x6yt.html

http://www.xuexila.com/news/1243489.html


# How to get the list?

- Copy names from webpage to document (.txt/.doc/.xls/…)
- Format the list
  - Manually
  - Automatically:  write code
- Check name list
  - remove band name, repeated names, non Chinese…
  - Write code + manually



- Problems?
  - More than one persons have same name
  - One person has more than one names
  - Name has incorrect characters
  - Person does not exist
  - There are non-Chinese persons
  - Is not a name (bands,…)

# About Name List

- Assign each name an ID number
- 10,109 names in total


NameList



# 2. Download Images

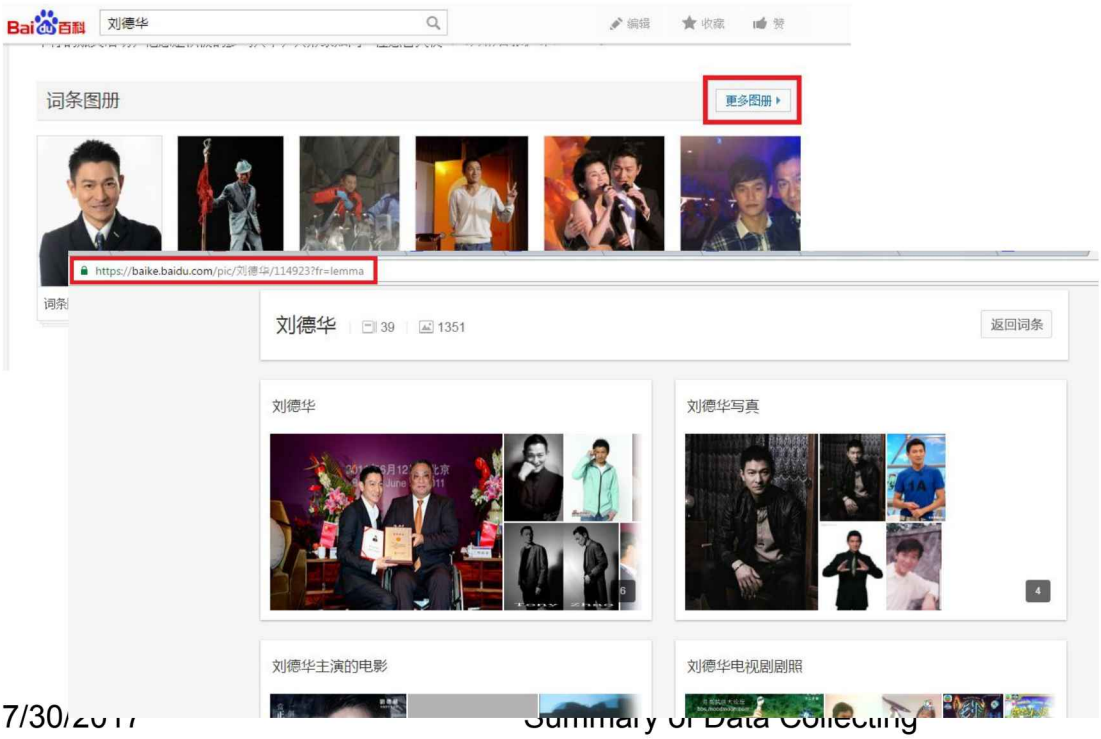
- Method
  - Automatically download images from Internet by running python script
- Procedures
  1. Analyze pattern of websites
  2. Preprocess name list (It depends)
  3. Write python script
  4. Read name list and download images
  5. Save images

# Image sources

- Baidu Baike 百度百科
- Baidu Tieba 百度贴吧
- Mingxing 明星图库
- n63 明星写真馆
- Tupianzj 图片之家
- Google Images 谷歌图片



## (1)百度百科 https://baike.baidu.com



- Pattern of URL

  http://baike.baidu.com/item/刘德华

  https://baike.baidu.com/pic/刘德华/114923?fr=lemma

```
for i in range(0, len(text)-1):
    print '%d/%d\n' % (i+1, len(text))
    subject_no = text[i].split()[0].strip()
    subject_name = text[i].split()[1].strip()
    save_dir = 'F:/zn_Chinese_Celebrities/bd_images/' + str(subject_no) # subject_name.decode('utf-8')
    URL = 'http://baike.baidu.com/item/' + subject_name

    try:
        page = requests.get(URL)
        m = re.search(urllib.quote(subject_name) + '/[0-9]+\?fr=lemma', page.content)
        number = m.group(0).split('?')[0].split('/')[1]
        updated_URL = 'http://baike.baidu.com/pic/' + urllib.quote(subject_name) + '/' + number + '?fr=lemma'
        if not os.path.isdir(save_dir):
            os.makedirs(save_dir)
        call(["image-scraper", "-s", save_dir, updated_URL])
    except Exception:
        print subject_name + ' skipped.\n'
        continue
```



- Install dependency python packages before running the script
  - "imagescraper"
    - Download and install

      https://github.com/sananth12/ImageScraper/

      go to your chosen folder and then apply the following command:

      python setup.py install
  - Other dependency packages
    - setproctitle / future/ requests/ selenium/ lxml …

- Problems

## When encounter the following situations, it fails.

❖ One name shows more than one persons






❖ No album exist


❖ There is no such person in this website

- 6,979 persons / 184,580 images

- Image quality: big noise
  - Many persons in one subjects
  - Complex background
  - Blur
  - Illumination
  - Big pose angle
  - Irrelative images

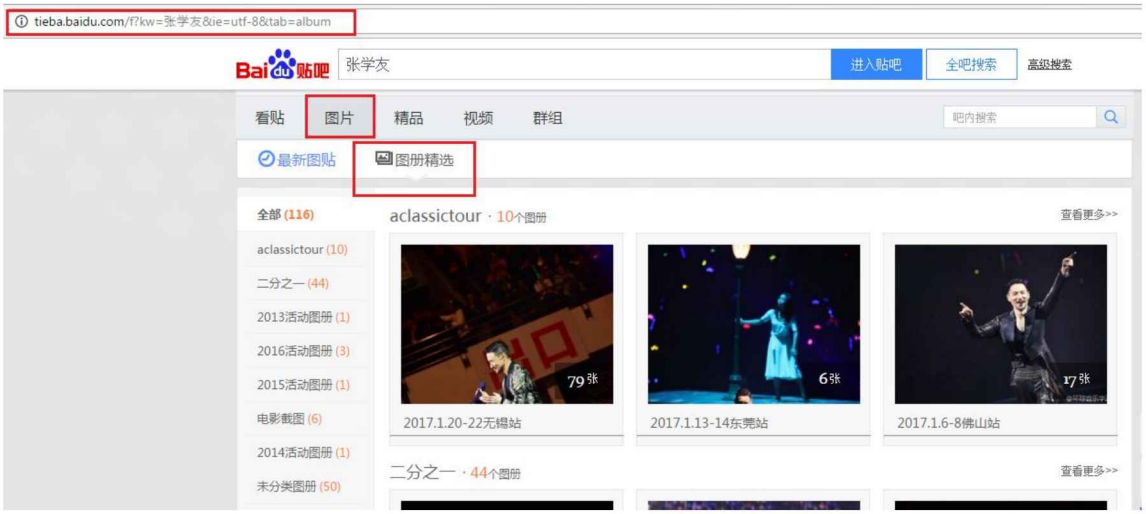








## (2)百度贴吧:http://tieba.baidu.com/



- Pattern of URL
  - http://tieba.baidu.com/f?kw=张学友&ie=utf-8&tab=album

- Use similar python script and environment

```
for i in range(0, len(text)-1):
    print '%d/%d\n' % (i+1, len(text))
    subject_no = text[i].split()[0].strip()
    subject_name = text[i].split()[1].strip()
    save_dir = 'F:/zn Chinese Celebrities/bd images/' + str(subject no)
    URL = 'http://tieba.baidu.com/f?kw='+ urllib.quote(subject_name) + '&ie=utf-8&tab=album'
    try:
        if not os.path.isdir(save_dir):
            os.makedirs(save_dir)
        call(["image-scraper", "-s", save_dir, URL])
    except Exception:
        print subject_name + ' skipped.\n'
        continue
```



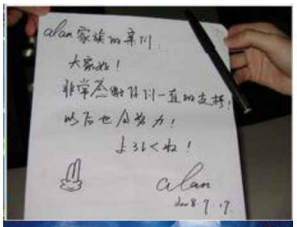
- Problems

  When encounter the following situations, it fails.

  - One name shows more than one persons
  - No album exist
  - The person does not exist in this website

- 5,330 persons / 70,233 images

- Image quality: big noise
  - Many persons in one subject
  - Complex background
  - Cartoon images
  - Wrong identity
  - Blur
  - Illumination
  - Big pose angle
  - Irrelative images



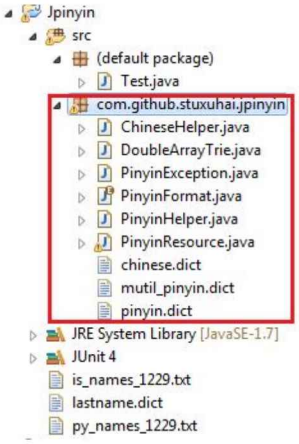




# (3) 明星图库：http://www.mingxing.com/ziliao/

- Pattern of URL
  - Mainland China
    - http://www.mingxing.com/neidi/chenlong/tupian.html
  - Hong Kong & Taiwan
    - http://www.mingxing.com/gangtai/lizongsheng/tupian.html

- Name in the URL: pinyin (拼音)
- Need to convert Chinese Characters into Pinyin.
- Method: Java + Eclipse + Jpinyin
  - Jpinyin: Java Class Library




[pinyin] pinyin_names_1229_mingxing.txt



- Use similar python script and environment
  - ✔ “imagescraper”

```
for i in range(0, len(text)):
    print '%d/%d\n' % (i+1, len(text))
    subject_no = text[i].split()[0].strip()
    subject_name = text[i].split()[1].strip()
    save_dir = 'F:/zn Chinese Celebrities/all images/' + str(subject_no)
    URL = 'http://www.mingxing.com/neidi/' + urllib.quote(subject_name) + '/tupian.html'
    #URL = 'http://www.mingxing.com/gangtai/' + urllib.quote(subject_name) + '/tupian.html'
    try:
        if not os.path.isdir(save_dir):
            os.makedirs(save_dir)
        call(["image-scraper", "-s", save_dir, URL])
    except Exception:
        print subject_name + ' skipped.\n'
        continue
```

- Problems
  - names in URL are not always in strict conformity with pinyin




- pinyin names are not correct
  - Polyphone
  - Some words as last name have different pinyin

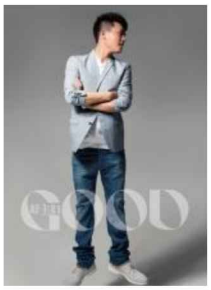
- For bands, fail to download by individual names
  - tfboys


- Name does not exist in this website



- Different words has same pinyin
  - 杨紫（actress） yangzi
  - 杨梓（host） yangzi

- 2,931 persons / 30,324 images
- Image quality: a few noise
  - Wrong identity
  - Big pose angle
  - Irrelative images
  - Blur
  - Illumination







# (4) 明星写真馆:http://www.n63.com/

- Pattern of URL
  - Female
    - http://www.n63.com/n_china/chenhuixian
  - Male
    - http://www.n63.com/n_chinam/chenchuhe
- Also pinyin in URL
- Use pinyin name list



- Use similar python script and environment:

✔ "imagescraper"

```
for i in range(0, len(text)):
    print '%d/%d\n' % (i+1, len(text))
    subject_no = text[i].split()[0].strip()
    subject_name = text[i].split()[1].strip()
    save_dir = 'F:/zn_Chinese_Celebrities/all_images/' + str(subject_no)
    URL = 'http://www.n63.com/n_china/'+ urllib.quote(subject_name)
    #URL = http://www.n63.com/n_chinam/'+ urllib.quote(subject_name)

    try:
        if not os.path.isdir(save_dir):
            os.makedirs(save_dir)
        call(["image-scraper", "-s", save_dir, URL])
    except Exception:
        print subject_name + ' skipped.\n'
        continue
```

- Problems: similar with(3)
  - names in URL are not always in strict conformity with pinyin


Pinyin: guanzhibin


Pinyin: wujianhao



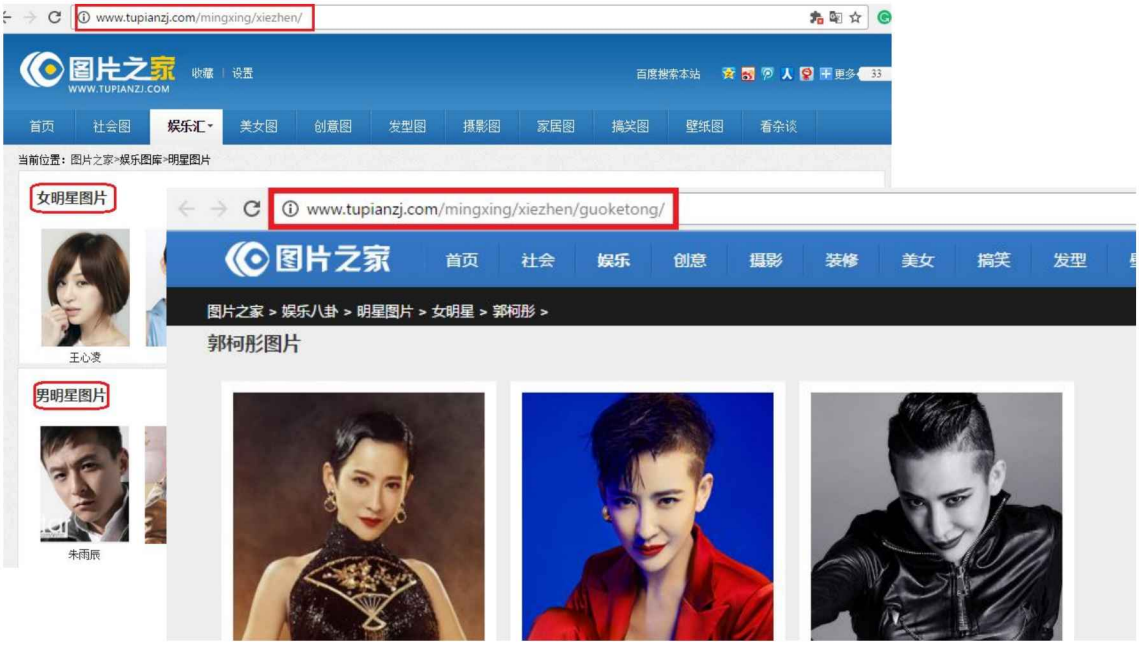
  - pinyin names are not correct
    - Polyphone
  - For bands, fail to download by individual names
    - Twins

- 1,849 persons, 14,112 images

- Image quality: little noise
  - Big pose angle
  - Blur
  - Illumination
  - Two pictures appear in one single image



## (5) 图片之家：http://www.tupianzj.com/mingxing/xiezhen/



- Pattern of URL
  - http://www.tupianzj.com/mingxing/xiezhen/guoketong/
- Also pinyin in URL
- Use pinyin name list



- Use similar python script and environment
  - ✔ “imagescraper”

```
for i in range(0, len(text)-1):
    print '%d/%d\n' % (i+1, len(text))
    subject_no = text[i].split()[0].strip()
    subject_name = text[i].split()[1].strip()
    save_dir = 'F:/zn Chinese Celebrities/bd images/' + str(subject no)
    URL = 'http://www.tupianzj.com/mingxing/xiezhen/'+ subject_name +'/'

    try:
        if not os.path.isdir(save_dir):
            os.makedirs(save_dir)
        call(["image-scraper", "-s", save_dir, URL])
    except Exception:
        print subject_name + ' skipped.\n'
        continue
```

- Problems: similar with(3,4)
  - names in URL are not always in strict conformity with pinyin


章泽天
zhangzetian



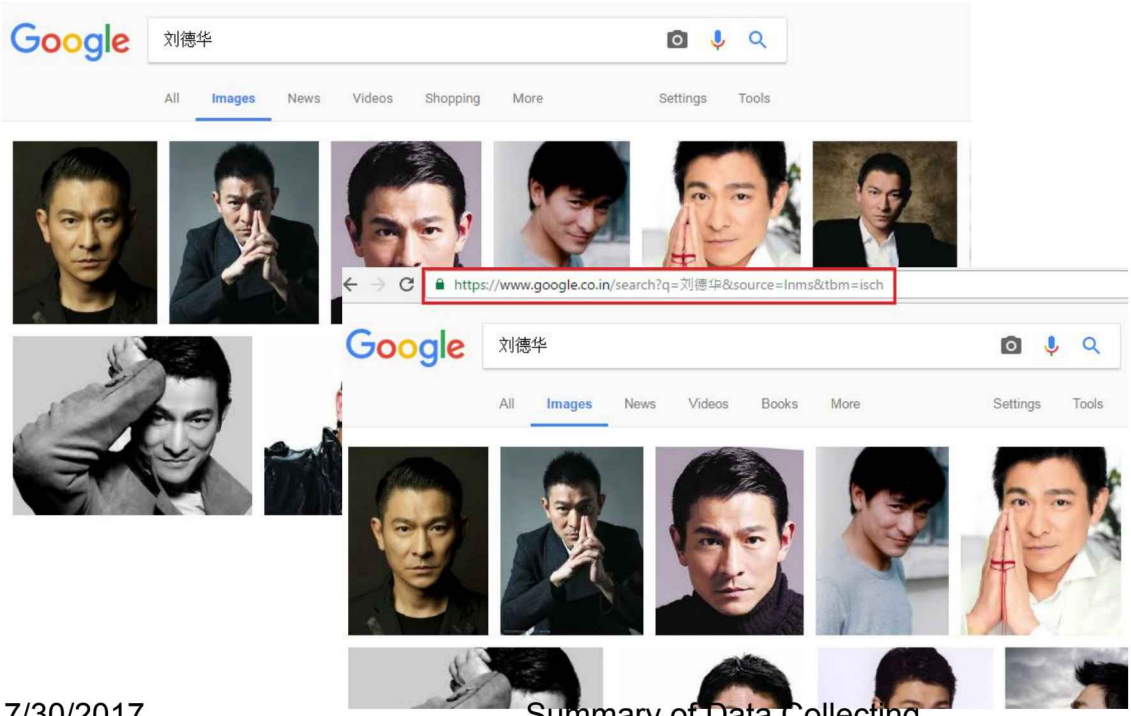
- pinyin names are not correct
  - Polyphone
- For bands, fail to download by individual names
  - Tfboys
- There is no such person in this website
- Different words has same pinyin

- 598 persons / 4,146 images

- Image quality: little noise
  - Big pose angle
  - Blur
  - Illumination
  - occlusion



# (6) Google Images: https://images.google.com/



- Pattern of URL:
  - https://www.google.co.in/search?q=刘德华&source=lnms&tbm=isch

- Use Chinese words in URL

- Not use ‘ImageScraper’ package to download
  - Fail to download from Google
- Write new script



- Download the first 25th images

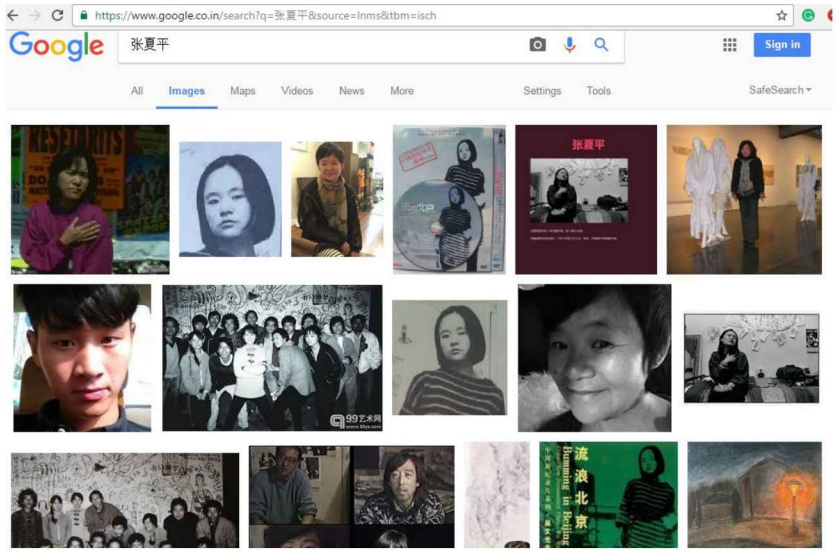
```
browser = webdriver.Firefox()
for i in range(0, len(text) - 1):
    print '%d/%d\n' % (i + 1, len(text))
    subject_no = text[i].split()[0].strip()
    subject_name = text[i].split()[1].strip()
    save_dir = 'F:/zn_Chinese_Celebrities/all_images/' + str(subject_no)
    url = "https://www.google.co.in/search?q="+subject_name+"&source=lnms&tbm=isch"

    browser.get(url)
    header={'User-Agent':"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
    counter = 0
    succounter = 0

    for _ in range(50):
        browser.execute_script("window.scrollBy(0,1000)")

    for x in browser.find_elements_by_xpath("//div[@class='rg_meta']"):
        counter = counter + 1
        if succounter > 25:
            break
        img = json.loads(x.get_attribute('innerHTML'))["ou"]
        imgtype = json.loads(x.get_attribute('innerHTML'))["ity"]
        try:
            req = urllib2.Request(img, headers={'User-Agent': header})
            raw_img = urllib2.urlopen(req).read()
            File = open(os.path.join(save_dir , save_dir + "_" + str(counter) + "." + imgtype), "wb")
            File.write(raw_img)
            File.close()
            succounter = succounter + 1
        except:
            print "can't get img"
browser.close()
```

- Problems
  - Unfamous names, get wrong & irrelative pictures







- 7,191 persons / 223,540 images

- Image quality: big noise in certain subjects
  - Big pose angle
  - Blur
  - Illumination
  - Occlusion
  - Complex background



persons

# Summary

- Table of each image source

| Image source | #subjects | #images | quality | Clean? |
|---|---|---|---|---|
| Baike | 6,979 | 184,580 | Big noise | Yes |
| Tieba | 5,330 | 70,233 | Big noise | Yes |
| Mingxing | 2,931 | 30,324 | a few noise | Yes |
| n63 | 1,849 | 14,112 | Little noise | No |
| tupianzj | 598 | 4,146 | Little noise | No |
| Google | **7,190** | **234,490** | Big noise | Yes |
| **In Total** | **9,128** | **537,885** | | |



# Data Cleaning

- Six subsets from six image sources
  - Baike
  - Tieba
  - Mingxing
  - N63
  - Tupianzj
  - Google

# Dataset Details

| Image source | #subjects | #images | quality | Clean? |
|---|---|---|---|---|
| Baike | 6,979 | 184,580 | Big noise | Yes |
| Tieba | 5,330 | 70,233 | Big noise | Yes |
| Mingxing | 2,931 | 30,324 | a few noise | Yes |
| N63 | 1,849 | 14,112 | Little noise | Yes,easy |
| Tupianzj | 598 | 4,146 | Little noise | Yes,easy |
| Google | **7,190** | **234,490** | Big noise | Yes |
| **In Total** | **9,128** | **537,885** | | |

- Noise distributions in them are not even
- For Baike and Tieba, each subject has high noisy images
- For Google, some subjects are pretty clear if the persons are very famous, and some others have too much noise if not famous
- Mingxing, N63 and Tupianzj have much less noise than the three subsets above

## Method: Clean manually

- All data will be cleaned by hand
- Our rule:

If you can easily detect the target face in the image, please keep it no matter what quality it is.

# Problems (Consider to remove)

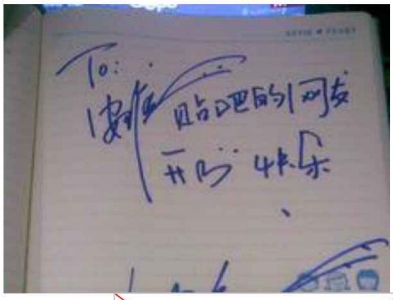
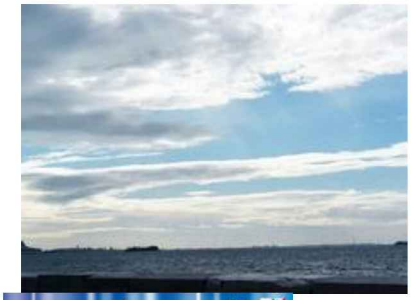
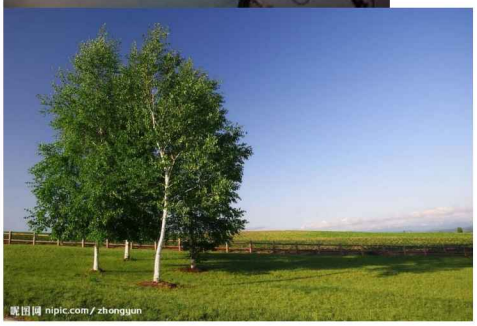
- 1. Occlusion - common problem
  - Heavily occluded, hardly recognize whether the face belongs to the target person (Remove)

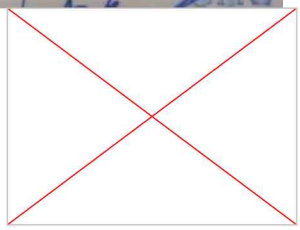



  - Although occluded, it is not so hard to recognize that the face belongs to the target person (Keep)




- 2. No faces in the image

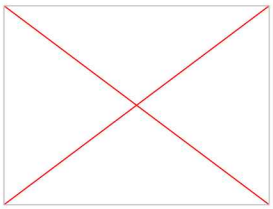



- 3. Too many faces in one image – common problem
  - So many persons, really hard for you to recognize the target face (Remove)




– Although many faces, easy for you to recognize the target face (Keep)

- 4. Crowded & Noisy Background
  - Background is so noisy that hard to find the target face (Remove)



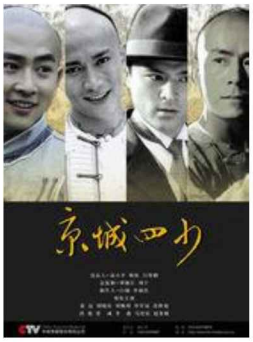
– Although background is noisy, the target face is easy for you to detect (Keep)




- 5. Multi-face image
  - Multiple faces of the target subject, clear to recognize (Keep)



– Multiple faces of different persons, the target face is in the image (Keep)

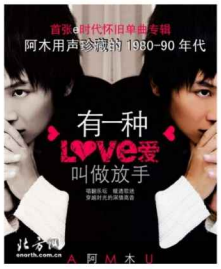


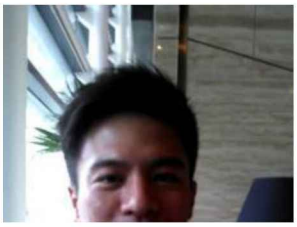



– Multiple faces of different persons, the target face is hard to detect in the image (Remove)


- 6. Partial faces  - common problem
  - The face in the image is cut by edge, hard to obtain facial landmarks (Remove)






  - Although the face is partial, more than half of the face is shown and easy to recognize it (Keep)

- 7. Cartoon & sketch
  - Some images are easy to be viewed as cartoon or sketch (Remove)



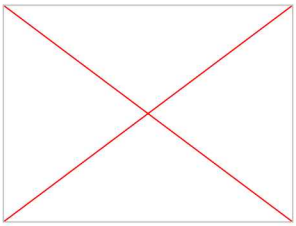




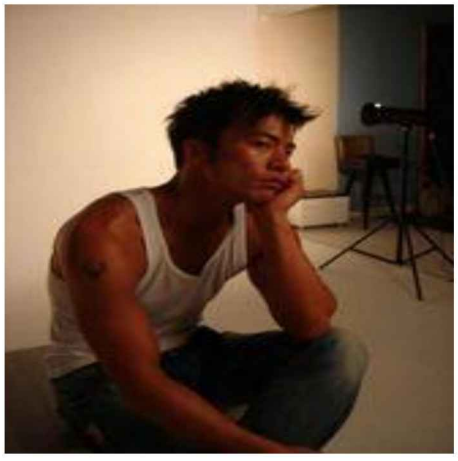



– Although it seems that the face in the image is like a cartoon face, if it is close to a real target face (Keep)


- 8. light problem
  - Too dark or too bright, hard to recognize the face (Remove)

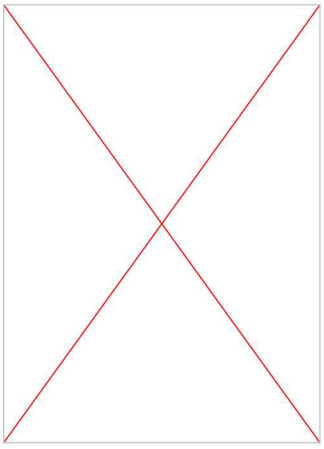



– Although there exists illumination problem, it is still easy to identify the target face (keep)



- 9. Poster
  - Same rule; Keep it if target face is easy for you to detect, or remove it.



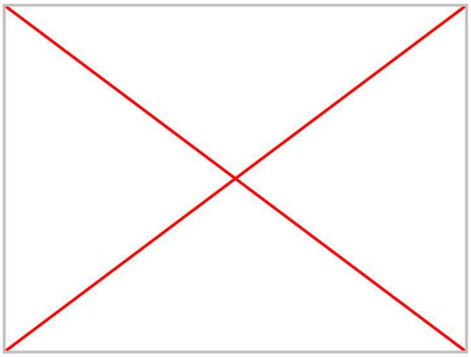


**Keep**

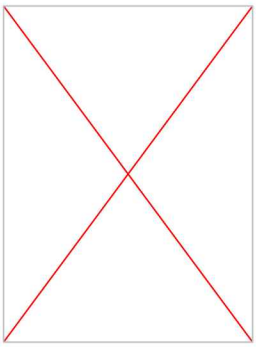
**Remove**



- 10. Large pose
  - Face contains heavy pose problem, hard to recognize it (Remove)



– The pose is not so large, it is still easy for you to verify whether the face is target person (Keep)



- 11. Too small/vague faces
  - Hard to detect faces (Remove)







– For you, can figure out the target face (Keep)

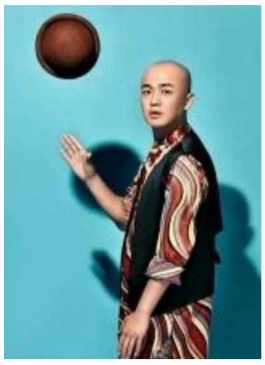


- 12. Faces in image are totally different with the target faces shown (Remove)

Target identity

Wrong identity, Remove

- 13. Wrong Gender



Target Identity

Wrong face, Remove

- 14. Damaged image
  - Fail to display (Remove)

0-aTt3RC.jpg

32857.jpg

# Consider to keep

✔ One complete face of right person

✔ More than one person

1. The correct person has full face
2. Others' faces are partial, small or vague


✔ More faces in one image

1. The right person with complete face

# Cleaning Result

- A total of 356,381 images of 7,676 subjects

| name | count |
|---|---|
| 0_New_Images | 14352 |
| 1_baike | 140261 |
| 2_tieba | 40264 |
| 3_mingxing | 19616 |
| 4_n63 | 10216 |
| 5_tupianzj | 3489 |
| 6_google | 127426 |
| 7_new | 757 |
| | 356381 |