Instagram Crawling for Bigdata Analysis

2019. 01. 14.

Lee Sungjae
Woo Hyeonwoong

Overview

1. Rawdata Crawling Pipeline

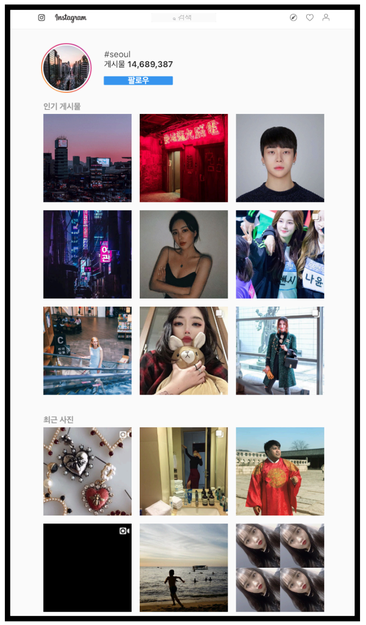
2. Multiprocessing in Python

3. Scenarios

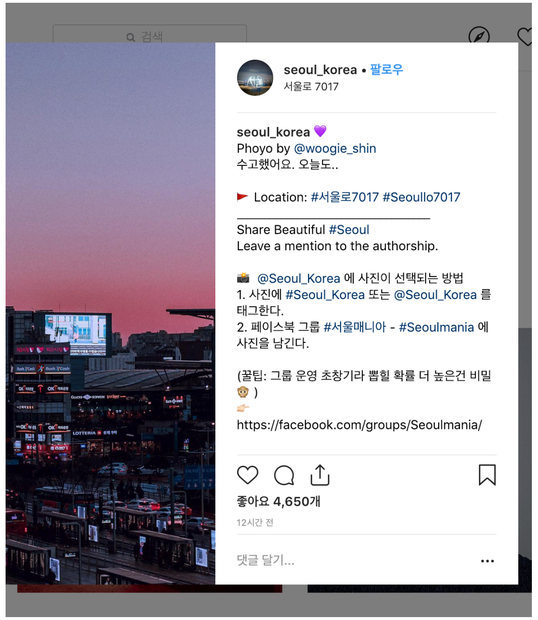
1. Rawdata Crawling Pipeline

Crawling Pipeline

Link Crawling with Selenium



Rawdata Crawling with Requests & BS4

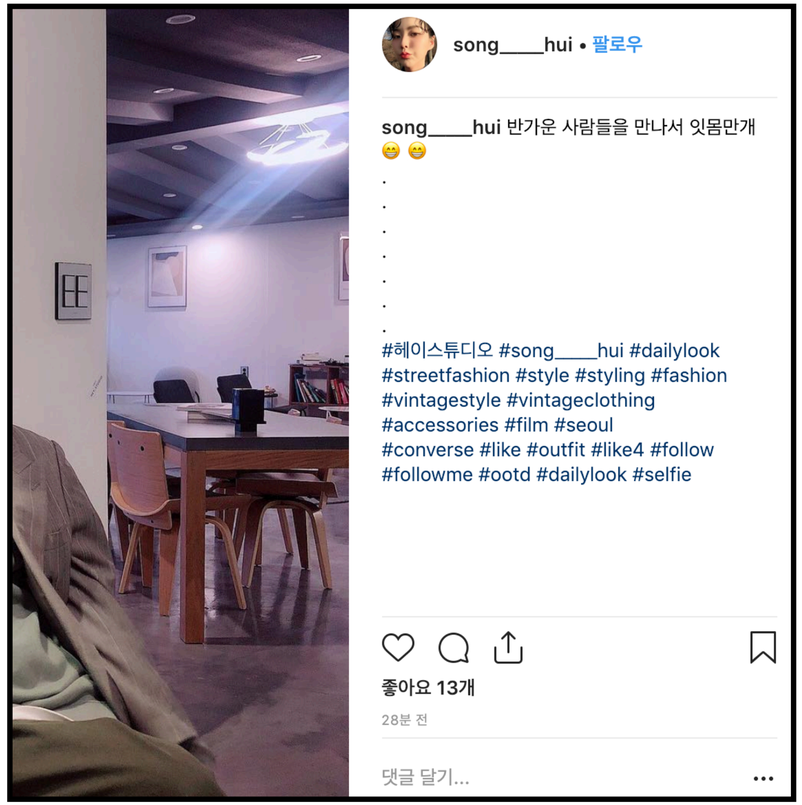


find_tag
id
username
date
contents
hashtags
image_links
likes
comments

mytag_tag_link.csv

mytag_rawdata.json

JSON Structure of Instagram Dataset



save highest resolution images

delete emoji from tags

delete special char from contents

delete tags from contents


```
{'comments': '0',
 'contents': '반가운 사람들을 만나서 잇몸만개
 'date': '2019-01-13T11:37:58',
 'find_tag': 'seoul',
 'hashtags': ['streetfashion',
              '헤이스튜디오',
              'style',
              'styling',
              'outfit',
              'vintageclothing',
              'seoul',
              'dailylook',
              'song_____hui',
              'fashion',
              'converse',
              'accessories',
              'like4',
              'like',
              'selfie',
              'film',
              'vintagestyle',
              'followme',
              'follow',
              'ootd'],
 'id': 'BsksXvCH3Sg',
 'imagelinks': ['https://scontent-icn1-1.cdninstagra
5/47692675_2209715202575458_8940295736636529883_n.jp
 'likes': '11',
 'username': 'song_____hui'}
```

Crawling Codes

```python
def crawling_links(my_tag, num_of_crawling_pages):
    options = ChromeOptions()
    options.add_argument('--headless')
    options.add_argument('--disable-extensions')
    options.add_argument('--no-sandbox')

    driver = Chrome(options = options)
    driver.implicitly_wait(1)

    base_url = "https://www.instagram.com/explore/tags/"
    url = base_url + my_tag

    driver.get(url)
    elem = driver.find_element_by_tag_name('body')
    link_list = []

    pagedowns = 0
    while pagedowns < num_of_crawling_pages:

        time.sleep(0.5)
        links = driver.find_elements_by_css_selector('div.v1Nh3 > a')
        for i in links:
            link_list.append(i.get_attribute('href'))

        for i in range(6):
            elem.send_keys(Keys.PAGE_DOWN)
            time.sleep(0.5)

        pagedowns += 1
        print(pagedowns, '페이지 크롤링 완료')

    set_link_list = list(set(link_list))
    num_link_list = len(link_list)
    num_set_link_list = len(set_link_list)

    print('중복링크 개수', num_link_list)
    print('유니크링크 개수', num_set_link_list)
    print('유니크링크 / 중복링크 : ', round((num_set_link_list/num_link_list) * 100, 2), '%')
    print('유니크링크 구성')
    print(set_link_list[:10])

    driver.quit()

    save_list_csv(set_link_list, my_tag)
```

```python
def crawling_rawdata(my_tag):
    my_links = []
    read_csv_list(my_links, my_tag)

    json_list = []
    emoji_keys = emoji.UNICODE_EMOJI.keys()

    for i in range(len(my_links)-1):
        url = my_links[i]
        req = requests.get(url)
        html = req.text
        header = req.headers
        status = req.status_code
        soup = BeautifulSoup(html, 'html.parser')

        script_contents = soup.find_all('script')

        ## id
        id_p = re.compile("\/p\/(.*?)\/")
        id = id_p.findall(url)[0]

        ## username
        user_content = soup.find(rel = "canonical")
        if not user_content:
            username = ''
        else:
            username_href = user_content.get('href')
            username_p = re.compile("instagram.com/(.*)/p")
            username = username_p.findall(str(username_href))
            if len(username) != 0:
                username = username[0]
            else:
                username = ''

        ## date
        date_p = re.compile("uploadDate\":\"(.{19})\"")
        date = date_p.findall(str(script_contents))
        if len(date) != 0:
            date = date[0]
        else:
            date = ''
```

Crawling Codes

```python
## contents
contents_p = re.compile("\"caption\":\"(.*?)\"")
contents = contents_p.findall(str(script_contents))
if len(contents) == 0:
    contents_p = re.compile("\"edge_media_to_caption\".*?text\":\"(.*?)\"")
    contents = contents_p.findall(str(script_contents))
    if len(contents) == 0:
        contents = ['']

if script_contents:
    contents_p = re.compile("\"caption\":\"(.*?)\"")
    contents = contents_p.findall(str(script_contents))

    if len(contents) > 0:
        contents = contents[0]
        contents = contents.encode('utf-8')
        contents = contents.decode('unicode_escape')
        contents = contents.encode('utf-8','ignore')
        contents = contents.decode('utf-8')
    else:
        contents = ''

## hashtags
meta_content = soup.find_all(property = "instapp:hashtags")
hashtags = []
if meta_content:
    hash_tags_p = re.compile("content=\"(.*?)\"")
    emoji_hashtag = hash_tags_p.findall(str(meta_content))

    for tag in emoji_hashtag:
        for e in emoji_keys:
            emoji_have = tag.find(e)
            if emoji_have > -1:
                tag = tag.replace(e, '')
        hashtags.append(tag)
    hashtags = list(filter(None, hashtags))

## remove hashtag from contents
for tag in hashtags:
    tag = '#' + tag
    contents = contents.replace(tag, '', 1)
```

```python
## preprocessing of contents
contents = contents.replace('#','')
contents = re.sub("\\\\u[0-9A-Fa-f]{4}", "", contents)
contents = re.sub("[-()\"#/@;:<>{}`+=~|.!?,]", "", contents)
contents = re.sub('\n', '', contents)
contents = re.sub('\s+', ' ', contents)

## image links
image_links_p = re.compile("\"display_resources\":\[.*?\]")
image_links = image_links_p.findall(str(script_contents))

final_image_link = []
single_image_link_p = re.compile("\"src\":\"(.*?)\"")
for dif_img in image_links:
    link = single_image_link_p.findall(str(dif_img))[2]
    final_image_link.append(link)

if len(final_image_link) != 1:
    final_image_link = final_image_link[1:]


## likes num
likes_num_p = re.compile("\"description\":\"(.*?)Likes")
likes_num = likes_num_p.findall(str(script_contents))

if len(likes_num) > 0:
    likes_num = likes_num[0]
    likes_num = re.sub('[,\s]', '', likes_num)
else:
    likes_num = '0'

## comments num
comments_num_p = re.compile("\"description\":\".*?,(.*?)Comments")
comments_num = comments_num_p.findall(str(script_contents))

if len(comments_num) > 0:
    comments_num = comments_num[0]
    comments_num = re.sub('[,\s]', '', comments_num)
else:
    comments_num = '0'

## save data as json
single_json = data2json(my_tag, id, username, date, contents, hashtags, final_image
json_list.append(single_json)
time.sleep(0.05)
```

Outputs : seoul_tag_links.csv / seoul_rawdata.json

```
https://www.instagram.com/p/BskrzKLh-ze/,
https://www.instagram.com/p/Bskr-Vfncuc/,
https://www.instagram.com/p/Bskrxxagbkn/,
https://www.instagram.com/p/BsksNq8HuoC/,
https://www.instagram.com/p/BsksbLoHz-m/,
https://www.instagram.com/p/BsksJ7TgCQH/,
https://www.instagram.com/p/BskpzHag3Vt/,
https://www.instagram.com/p/BskfDA2AqUw/,
https://www.instagram.com/p/BskrcFJFp6p/,
https://www.instagram.com/p/BsksuVfhvbr/,
https://www.instagram.com/p/BsksxeQhQuI/,
https://www.instagram.com/p/BskPCKrHiAy/,
https://www.instagram.com/p/BsksMwyArQU/,
https://www.instagram.com/p/Bskp20NHS-N/,
https://www.instagram.com/p/BskrucuAIi6/,
https://www.instagram.com/p/Bskr45RnfH0/,
https://www.instagram.com/p/BskTTKFh7M2/,
https://www.instagram.com/p/BsksWNbB4Rn/,
https://www.instagram.com/p/BnNqVcunCQq/,
https://www.instagram.com/p/Bskr0P3gXZv/,
https://www.instagram.com/p/BsksSI9BrVF/,
https://www.instagram.com/p/BsksGGpAwt6/,
https://www.instagram.com/p/BsksqrDH62D/,
https://www.instagram.com/p/BsksJqvFoF-/,
https://www.instagram.com/p/BsksylsDkpv/,
https://www.instagram.com/p/BsksVbsASz6/,
https://www.instagram.com/p/BsksANiAei8/,
```


```
    {
        "find_tag": "seoul",
        "id": "BDmtN4FrjPY",
        "username": "cabigaobrian",
        "date": "2016-03-31T04:56:41",
        "contents": "Gyeongbokgung Palace&x27s doorway to it&x27s restored legacy ",
        "hashtags": [
            "scenery",
            "travelogue",
            "traveljournal",
            "latergram",
            "potd",
            "travelog",
            "vscocam",
            "korea",
            "explore",
            "followgram",
            "travelbug",
            "travels",
            "seoul",
            "uploadkorea",
            "vscophoto",
            "vscogram",
            "vsco",
            "view",
            "vscopic",
            "followtrain",
            "goingplaces",
            "wanderlust",
            "vscophile",
            "followme",
            "scenic",
            "follow",
            "vscoplaces",
            "potd",
            "stockphotos"
        ],
        "imagelinks": [
            "https://scontent-icn1-1.cdninstagram.com/vp/cb63d191dcfa043a89cbb4892bae432c/5CB52197/t51.2885-15/e35/12519454_2099
56456040486_48641593_n.jpg?_nc_ht=scontent-icn1-1.cdninstagram.com"
        ],
        "likes": "43",
        "comments": "3"
    }
]
jovyan@ded7ac83932a:~/2019_Bigdatalab_Project/instagram/Module_and_Multiprocessing$
```

Shell Script to 'instacrawler' package

Shell Script Version

```python
## server 에서 chromedriver 를 실행하기 위해, 기존의 패키지에서 webdriver.ChromeOption 을 추가하여 가져옵니다.
import os
import time
from selenium.webdriver import Chrome
from selenium.webdriver import ChromeOptions
from selenium.webdriver.common.keys import Keys

## --headless / --disable-extensions / --no-sandbox 옵션으로 드라이버를 생성합니다.
options = ChromeOptions()
options.add_argument('--headless')
options.add_argument('--disable-extensions')
options.add_argument('--no-sandbox')

driver = Chrome(options = options)
driver.implicitly_wait(3)

## 앞에서와 마찬가지로 크롤링 할 url 을 특정 태그 기준으로 생성합니다.
tags_url = "https://www.instagram.com/explore/tags/"
my_tag = input('Enter your tag :')
url = tags_url + my_tag

## 더 빠른 크롤링을 위해 0.25 배율로 진행하고자 했으나, chrome.settingPrivate 을 인식하지 못해 실패하였습니다
## local 환경에서 chrome browser 를 사용하던 것에서 chromium browser 를 사용하며 문제가 생긴 것으로 보입니다.
# driver.get('chrome://settings/')
# driver.execute_script('chrome.settingsPrivate.setDefaultZoom(0.25);')

## 나머지 부분은 local 환경과 마찬가지로 크롤링을 진행합니다
driver.get(url)
elem = driver.find_element_by_tag_name('body')
link_list = []
num_of_crawling_pages = int(input('Enter num of crawling pages :'))

pagedowns = 0
while pagedowns < num_of_crawling_pages:

    ## 놓치는 데이터가 없도록 크롤링 후 PAGE_DOWN 방식으로 변경하였습니다.
    time.sleep(0.5)
    links = driver.find_elements_by_css_selector('div.v1Nh3 > a')
    for i in links:
        link_list.append(i.get_attribute('href'))

    ## Local 환경에서 테스트 한 대로 최적화된 PAGE_DOWN 방법을 사용합니다.
    for i in range(6):
        elem.send_keys(Keys.PAGE_DOWN)
        time.sleep(0.5)

    pagedowns += 1
    print(pagedowns, '페이지 크롤링 완료')
```

link_crawling.py

```python
import requests
import re
import time
import json
import emoji
from bs4 import BeautifulSoup

def read_csv_list(my_list, my_tag):
    tag_name = my_tag
    file_name = tag_name + '_tag_link.csv'

    with open(file_name) as file:
        for line in file:
            read_data = line
            my_list.append(read_data[:-2])
        file.close()
    print('load ', len(my_list), 'data from csv to list')

## 찾아낸 데이터를 JSON 객체로 생성하는 함수를 정의합니다.
def data2json(my_tag, id, username, date, contents, hash_tags, final_image_link, likes_num, com
    single_data = {
        "find_tag" : my_tag,
        "id" : id,
        "username" : username,
        "date" : date,
        "contents" : contents,
        "hashtags" : hash_tags,
        "imagelinks" : final_image_link,
        "likes" : likes_num,
        "comments" : comments_num
    }
    return single_data

my_tag = input('Enter your tag :')
my_links = []
read_csv_list(my_links, my_tag)

json_list = []
emoji_keys = emoji.UNICODE_EMOJI.keys()

for i in range(len(my_links)-1):
# for i in range(1):

    test_url = my_links[i]
    req = requests.get(test_url)
    html = req.text
    header = req.headers
    status = req.status_code
    soup = BeautifulSoup(html, 'html.parser')
```

rawdata_crawling.py

```shell
#!/bin/sh
python link_crawling.py
python rawdata_crawling.py
```

pipe.sh

"instacrawler" Package Version

```python
import os
import time
from selenium.webdriver import Chrome
from selenium.webdriver import ChromeOptions
from selenium.webdriver.common.keys import Keys

import requests
import re
import json
import emoji
from bs4 import BeautifulSoup

def save_list_csv(mylist, tag_name):
    file_name = tag_name + '_tag_link.csv'
    with open(file_name, "w") as file:
        for line in mylist:
            file.write(line + ',\n')

def read_csv_list(my_list, my_tag):
    tag_name = my_tag
    file_name = tag_name + '_tag_link.csv'

    with open(file_name) as file:
        for line in file:
            read_data = line
            my_list.append(read_data[:-2])
        file.close()
    print('load ', len(my_list), 'data from csv to list')

def crawling_links(my_tag, num_of_crawling_pages):
    options = ChromeOptions()
    options.add_argument('--headless')
    options.add_argument('--disable-extensions')
    options.add_argument('--no-sandbox')
```

instacrawler.py

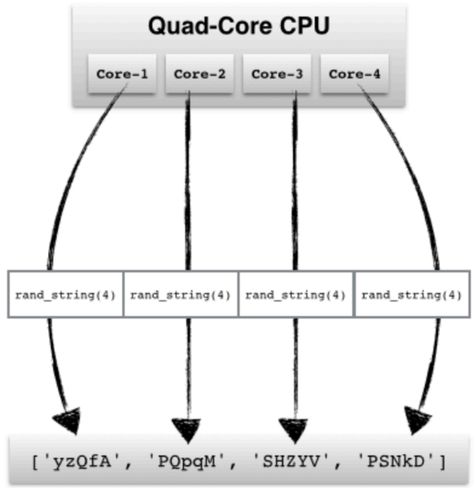
```python
import instacrawler as insta

my_tags = ['seoul', 'newyork', 'tokyo']
for my_tag in my_tags:
    insta.crawling_links(my_tag, 5)
```

project.py

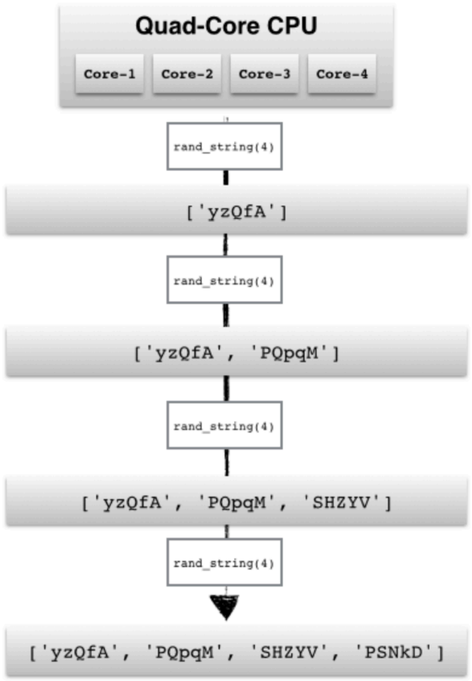
2. Multiprocessing in Python

Multiprocessing in Python

[parallel processing]



[serial processing]




```python
from multiprocessing import Pool

def f(x):
    return x*x

if __name__ == '__main__':
    with Pool(5) as p:
        print(p.map(f, [1, 2, 3]))
```

map (function , data)

Pool (num of processor)

Multiprocessing by Tags

rawdata crawling of 'seoul' tag

```
import time

start_time = time.time()
insta.crawling_rawdata(my_tags[0])
print("--- %s seconds ---" % (time.time() - start_time))
```

```
ского проверяют на собеседовании по скайпу или видеозвонкуОбразовани
_____Агенство CHIN CHINAREM По всем вопросам пишите нам в What
_____Работавкорее Уфа Новосибирск южнаякорея ',
чительанглийского', 'москва', 'казань', 'хостес', 'вакансиивкорее',
'workinkorea', 'ищуработу', 'работахостес', 'урал', 'новосибирск
отадлядевушек', 'trip', 'работавкорее', 'southkorea', 'уфа', 'п
t-icn1-1.cdninstagram.com/vp/0091ca1450306895c57dce786413ca
813106762474433_n.jpg?_nc_ht=scontent-icn1-1.cdninstagram.c
------------------
150 번째 데이터
{'find_tag': 'seoul', 'id': 'Bskr3pHh6kz', 'username': 'ara
htags': ['빵스타그램', 'bread', 'cafehopping', 'cafe', 'lifest
ng', '파운드케이크', 'dessert', 'vegan', '카페투어', 'seoul', 'k
'디저트', 'tiramisu', '빵', '비건빵집', 'poundcake', '어니스크',
gram.com/vp/d2099f76cbb0ff64bf6f7c1e776c30ad/5CC3399D/t51.2
jpg?_nc_ht=scontent-icn1-1.cdninstagram.com'], 'likes': '27
------------------
Crawling 완료!
--- 99.47041869163513 seconds ---
```

153 links : 99.47 seconds

rawdata crawling of 'seoul', 'newyork', 'tokyo' tag

```
from multiprocessing import Pool
my_tags = ['seoul', 'newyork', 'tokyo']

start_time = time.time()
if __name__ == '__main__':
    with Pool(3) as p:
        p.map(insta.crawling_rawdata, my_tags)
print("--- %s seconds ---" % (time.time() - start_time))
```

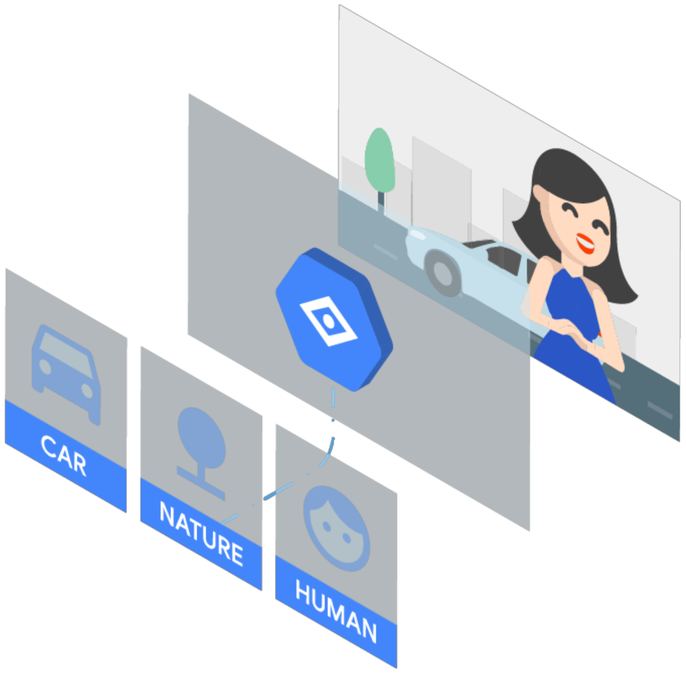
```
------------------
Crawling 완료!
150 번째 데이터
{'find_tag': 'tokyo', 'id': 'BsksuIFFdbq', 'username': 'pic
alon TOKYO autosalon chiba makuhari hino', 'hashtags': [],
p/8ae97947603c7e9e3b324b7e2c8b5f56/5C3D5ACA/t51.2885-15/e15
=scontent-icn1-1.cdninstagram.com'], 'likes': '2', 'comment
------------------
150 번째 데이터
{'find_tag': 'seoul', 'id': 'Bskr3pHh6kz', 'username': 'ara
htags': ['seoul', '교대카페', 'veganbaking', '비건빵집', '카페투어
'디저트', '카페', 'cafe', 'dessert', 'bread', '어니스크', 'korea
ramisu', '티라미수', '빵', '빵투어', '빵스타그램', 'bakery', 'poun
ram.com/vp/d2099f76cbb0ff64bf6f7c1e776c30ad/5CC3399D/t51.28
pg?_nc_ht=scontent-icn1-1.cdninstagram.com'], 'likes': '20'
------------------
Crawling 완료!
Crawling 완료!
--- 113.478266954422 seconds ---
```

471 links : 113.47 seconds

3. Scenarios

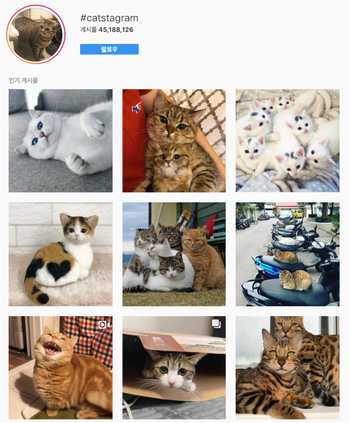
Scenario 1. Text Input AutoML

Google Cloud Vision AutoML
: User Input Image Dataset

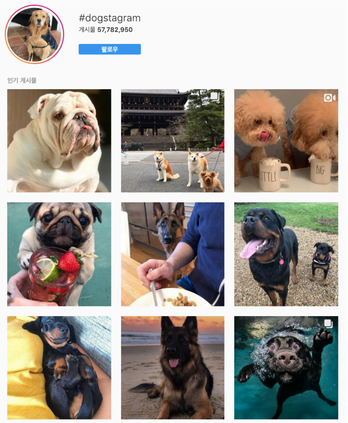


Input Text
'Cat - Dog Classification Model'

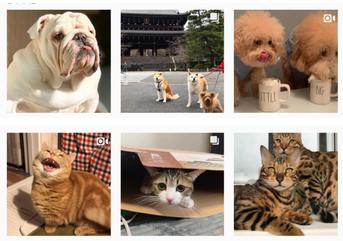
Crawling 'Cat' tag



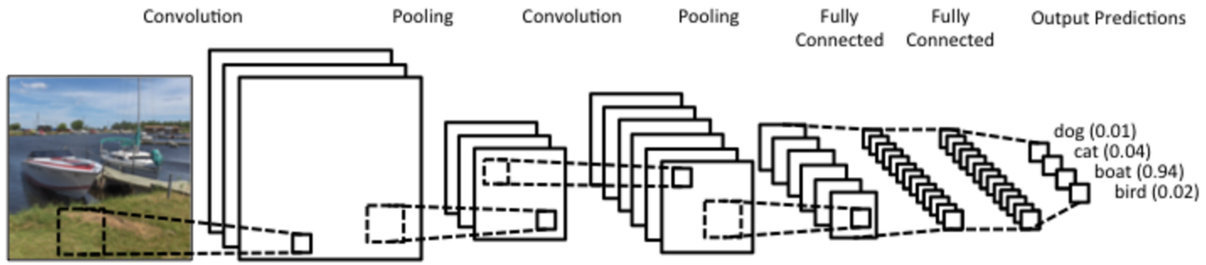
Crawling 'Dog' tag



Labeled Dataset



Train Classification Model with CNN

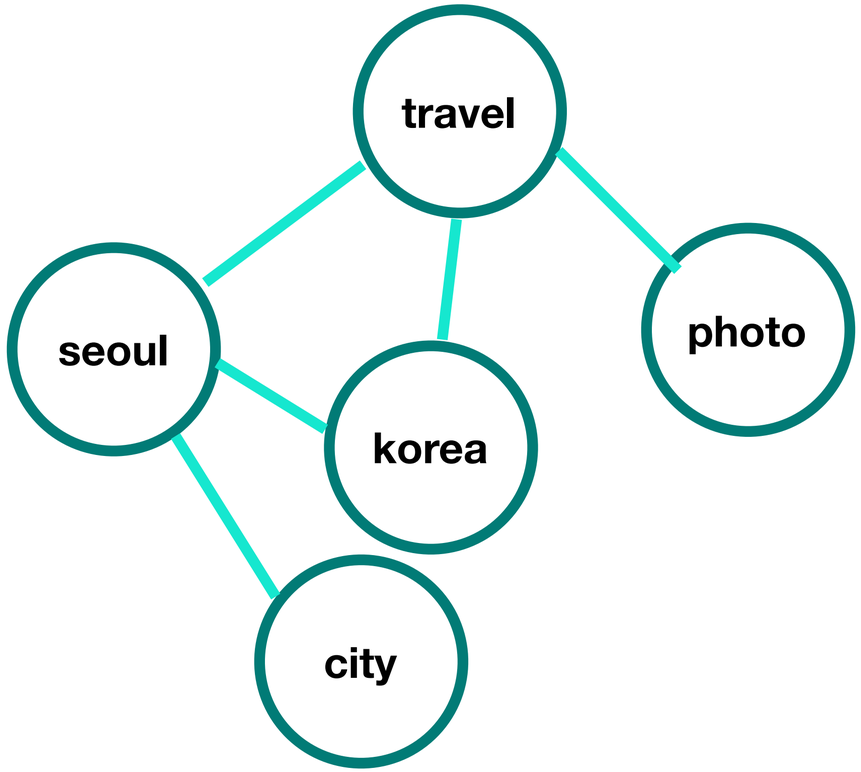


Scenario 2. Analyze Relation of Tags

Hashtags of 'seoul'


```
"hashtags": [
    "scenery",
    "travelogue",
    "traveljournal",
    "latergram",
    "potd",
    "travelog",
    "vscocam",
    "korea",
    "explore",
    "followgram",
    "travelbug",
    "travels",
    "seoul",
    "uploadkorea",
    "vscophoto",
    "vscogram",
    "vsco",
    "view",
    "vscopic",
    "followtrain",
    "goingplaces",
    "wanderlust",
    "vscophile",
    "followme",
    "scenic",
    "follow",
    "vscoplaces",
    "potd",
    "stockphotos"
```


Graph of 'seoul' tag



Sorting by Relation

1. seoul
2. korea
3. travel
4. city
5. photo
6. ...

Scenario 3. Real-Time Issue Tracker

Input Issue
: KMU

Input Date
: 1Day

Crawling 'KMU' tag
Recent 1 Day

Get 1Day Issue of KMU

1. 국민대학교
2. 수시 결과
3. 합격 통보
4. 대학원 입학
5. 정시박람회
6. ...

Github

https://github.com/odobenuskr/2019_Bigdatalab_Project

37 commits | 1 branch | 0 releases | 4 contributors

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

odobenuskr fix unicode escape and multiprocess test — Latest commit 47ddd35 23 hours ago

amazon	extract keyWords from features	a day ago
instagram	fix unicode escape and multiprocess test	23 hours ago
README.md	Update README.md	a day ago

README.md

2019_Bigdatalab_Project

2019 Bigdatalab Undergrad Project :: Amazon / Instagram Crawling and Analysis.

참여자

이성재 / 우현웅 / 김재원 / 이수정 / 이연지

Commits on Jan 12, 2019

- fix unicode escape and multiprocess test — odobenuskr committed 23 hours ago — 47ddd35
- make crawling module and multiprocessing by tags — odobenuskr committed a day ago — b068849
- update all-in-one version — odobenuskr committed a day ago — 7b8ff37
- extract keyWords from features — nelumbotus committed a day ago — ef002b8
- Update README.md — Verified — odobenuskr committed a day ago — 3f64bde

Commits on Jan 11, 2019

- preprocessing in pipeline — odobenuskr committed 2 days ago — 97aa893
- large size test — odobenuskr committed 3 days ago — 2a3db68
- contents text preprocessing — odobenuskr committed 3 days ago — 90cea9c

Commits on Jan 10, 2019

- delete emoji from contents and hashtags — odobenuskr committed 3 days ago — f9746df
- caption,tags not included emoji — woomurf committed 3 days ago — 79be595
- fix unicode to utf-8 — odobenuskr committed 3 days ago — f6ae096
- fix csv load problem — odobenuskr committed 3 days ago — 905f4a2
- get caption,Location — woomurf committed 3 days ago — 900fa13

Commits on Jan 9, 2019

- shell with input — odobenuskr committed 4 days ago — f9b7616