画像データを集める

1　WEB サイトから API を使ってデータを集める

画像の分類を行うにあたって、画像データ等のデータを多くを集める必要が出てくる場合があります。今回は、WEB サイトの API を使って画像データを集める例を説明致します。WEB サイトから集めることができるデータには、画像データ以外にも多くのデータがあります。

2　「フォト蔵」からダウンロードする

2.1　概要

フォト蔵（フォトぞう）は簡単に写真を投稿・共有できる無料のフォトアルバムサービスです。スマートフォンやデジタルカメラで撮った写真をみんなに見てもらったり、友達や家族でお互いの写真を見せあったりすることができます。フォト蔵の URL は以下の通りです。

`http://photozou.jp/`

2.2　API

フォト蔵 API とは、他のプログラムからフォト蔵にアクセスするために提供している外部プログラムインターフェースです。API によるアクセスは 1 リクエスト/秒を目安とすることとなっています。このように、API にアクセスする際には、サービスに影響の出る過剰なアクセスは避ける必要があります。

フォト蔵 API の仕様は、

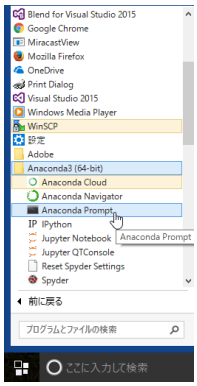
`http://photozou.jp/basic/api`

に記載されています。今回は、これらの API の中から、「search_public」という API を利用しています。

`http://photozou.jp/basic/api_method_search_public`

API の仕様について各自確認してみて下さい。

2.3 Anaconda Prompt の立ち上げ



Anaconda Prompt を選択します。

2.4 実行用のツールのダウンロード

実行用のツールをダウンロードします。そして、関連ファイルの入っているディレクトリに移動します。

```
cd %homepath%
cd Lesson
git clone https://github.com/k1nk/ch8.git
cd ch8
```

2.5 モジュールのインストール

実行に必要なモジュールをインストールします。以前に授業で作成した tensorenv 環境で作業を行います。

```
activate tensorenv
conda install urllib3
conda install beautifulsoup4
conda install requests
```

授業に欠席したなどの理由で、「activate tensorenv」で環境に入れない場合には、このテキストの「(参考) tensorenv 環境の作成」を行ってから続けて下さい。

2.6 プログラム

phototkura_downloader.py

```
1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3
4 import sys, os, re, time
5 import urllib.request as req #python3
```

```
6  import urllib.parse as parse #python3
7
8  import json
9  import argparse
10
11 #args = sys.argv
12 parser = argparse.ArgumentParser(description='This script download
       files from photo_kura.')
13 parser.add_argument('kwd', \
14         action='store', \
15         nargs='?', \
16         const="apple", \
17         default="apple", \
18         type=str, \
19         choices=None, \
20         help='Keyword to search photos.', \
21         metavar=None)
22
23 parser.add_argument('dir', \
24         action='store', \
25         nargs='?', \
26         const="./image", \
27         default="./image", \
28         type=str, \
29         choices=None, \
30         help='Directory path where your taken photo files are located
              .', \
31         metavar=None)
32
33 args = parser.parse_args()
34
35 # APIのURLを指定
36 PHOTOZOU_API = "https://api.photozou.jp/rest/search_public.json"
37 CACHE_DIR = "./image/cache"
38
39 # フォト蔵のAPIを利用して画像を検索する --- (※1)
40 def search_photo(keyword, offset=0, limit=100):
41     # APIのクエリを組み立てる
42     keyword_enc = parse.quote_plus(keyword)
43     q = "keyword={0}&offset={1}&limit={2}".format(
```

```
44          keyword_enc, offset, limit)
45      url = PHOTOZOU_API + "?" + q
46      # キャッシュ用のディレクトリを作る
47      if not os.path.exists(CACHE_DIR):
48          os.makedirs(CACHE_DIR)
49      cache = CACHE_DIR + "/" + re.sub(r'[^a-zA-Z0-9\%\#]+', '_', url)
50      if os.path.exists(cache):
51          return json.load(open(cache, "r", encoding="utf-8"))
52      print("[API] " + url)
53      req.urlretrieve(url, cache)
54      time.sleep(1) # --- 礼儀として1秒スリープ
55      return json.load(open(cache, "r", encoding="utf-8"))
56
57  # 画像をダウンロードする --- (※2)
58  def download_thumb(info, save_dir):
59      if not os.path.exists(save_dir): os.makedirs(save_dir)
60      if info is None: return
61      if not "photo" in info["info"]:
62          print("[ERROR] broken info")
63          return
64      photolist = info["info"]["photo"]
65      for photo in photolist:
66          title = photo["photo_title"]
67          photo_id = photo["photo_id"]
68          url = photo["thumbnail_image_url"]
69          path = save_dir + "/" + str(photo_id) + "_thumb.jpg"
70          if os.path.exists(path): continue
71          try:
72              print("[download]", title, photo_id)
73              req.urlretrieve(url, path)
74              time.sleep(1) # --- 礼儀として1秒スリープ
75          except Exception as e:
76              print("[ERROR] failed to downlaod url=", url)
77
78  # 検索結果を全部取得する --- (※3)
79  def download_all(keyword, save_dir, maxphoto = 1000):
80      offset = 0
81      limit = 100
82      while True:
83          # APIを呼び出す
```

```
84          info = search_photo(keyword, offset=offset, limit=limit)
85          if info is None:
86              print("[ERROR] no result"); return
87          if (not "info" in info) or (not "photo_num" in info["info"]):
88              print("[ERROR] broken data"); return
89          photo_num = info["info"]["photo_num"]
90          if photo_num == 0:
91              print("photo_num = 0, offset=", offset)
92              return
93          # 写真情報が含まれていればダウンロード
94          print("*** download offset=", offset)
95          download_thumb(info, save_dir)
96          offset += limit
97          if offset >= maxphoto: break
98
99  if __name__ == '__main__':
100     # モジュールとして使わないで単独で実行する時
101     query = args.kwd
102
103     #save_dir
104     DIR = args.dir
105     query_p='+'.join(query.split())
106
107     if not os.path.exists(DIR):
108                 os.mkdir(DIR)
109     DIR = os.path.join(DIR, query_p.split()[0])
110     if not os.path.exists(DIR):
111                 os.mkdir(DIR)
112
113     download_all(query,DIR)
```

2.7 ダウンロードの方法

以下のコマンドで、写真をダウンロードします。

```
python phototkura_downloader.py "青リンゴ" green_apple_kura
```

この例では「青リンゴ」というキーワードで、「green_apple_kura」というディレクトリの中に画像をダウンロードします。デフォルトで maxphoto = 1000 まで、写真をダウンロードします。ダウンロードした画像から、学習に適したものを手作業で振り分けます。
キーワード等を自分の好きなものに変更して、ダウンロードを行って見ましょう。

3 （参考）tensorenv 環境の作成

ここでは、TensorFlow を自分のＰＣ上で使うために必要な環境設定を行います。以前の授業で tensorenv 環境を作成していない場合には、以下にしたがって、tensorenv 環境を作成して下さい。

3.1 tensorenv 環境の作成

作業ディレクトリへ移動します。ホームディレクトリの下に「Lesson」というディレクトリがあることを前提に作業します。ない場合には作成して下さい。

```
cd %homepath%
cd Lesson
```

python3.5 を使う環境を作成します。環境の名前を「tensorenv」とします。環境の作成が終ったら、その環境に入ります。

```
conda create --name=tensorenv python=3.5
activate tensorenv
```

環境に入ると、コマンドプロンプトの先頭に環境名（tensorenv）が表示されます。

3.2 TensorFlow のインストール

作成した環境に、TensorFlow やその他、プログラムの実行に必要なモジュール等をインストールします。この TensorFlow をバックエンドに、容易に素早くディープラーニングプロトタイプの迅速な実験を可能にすることに重点を置いて開発されたライブラリが Keras です。Keras もあわせてインストールしておきます。インストール中に、「Proceed ([y]/n)?」と聞かれたら「y」を入力します。

```
pip install --upgrade pip
CPU版
pip install tensorflow
pip install jupyter
conda install scipy
pip install keras
conda install matplotlib
conda install -c menpo opencv3
```