Widzenie Komputerowe

Laboratorium nr 5.

Trenowanie sieci neuronowych uczenia głębokiego

Autor: Magdalena Mazur Milecka

Gdańsk, 2019

WSTĘP

Zadania należy wykonać w Pythonie (wersja 3.6.4) korzystając z darmowej biblioteki OpenCV (wersja 3.4.0). Wymagane jest także zainstalowanie pakietów: *numpy*, *matplotlib*, *tensorflow* oraz *certifi*.

Celem zadania będzie klasyfikacja obrazów przy użyciu sieci neuronowych głębokiego uczenia. Zastosujemy uczenie transferowe, które oznacza, że punktem początkowym trenowania jest model, który został wcześniej wytrenowany na różnych obrazach. Takie trenowanie tylko ostatniej warstwy złożonego (Rys.) modelu przy użyciu danego zbioru uczącego okazuje się osiągać dużą poprawność klasyfikacji oraz ogromną przewagę jeśli chodzi o czas trenowania.

Modele używane w instrukcji (MobileNet - https://ai.googleblog.com/2017/06/mobilenets-open-source-models-for.html oraz Inception V1 - https://arxiv.org/abs/1409.4842v1) są wytrenowane na bazie obrazów ImageNet (http://www.image-net.org/).

Cały proces trenowania i klasyfikacji w systemie Linux został opisany przez twórców pakietu TensorFlow na stronie:

https://codelabs.developers.google.com/codelabs/tensorflow-for-poets/#0

Instrukcja została przygotowana dla pojedynczej maszyny bez procesora GPU, na system Windows 10, bez dodatkowego IDE Pythona.

Rys. Złożoność modelu Inception

Trenowanie przy użyciu gotowego zbioru uczącego zawierającego obrazy 5-ciu klas kwiatów

1. W dowolnej lokalizacji rozpakować plik *tensorflow_example.zip*
2. W katalogu *tensorflow-for-poets-2/tf_files* rozpakować plik *flower_photos.zip*
3. Uruchomić wiersz poleceń z uprawnieniami administratora oraz przejść do wypakowanego pliku *tensorflow-for-poets-2*
4. Uruchomić trenowanie na obrazach kwiatów używając modelu MobileNet o połowie rozmiaru największej sieci oraz rozdzielczości obrazu=224 poleceniem:

```
python.exe scripts\retrain.py
--bottleneck_dir=tf_files/bottlenecks
--how_many_training_steps=500
--model_dir=tf_files/models/
--summaries_dir=tf_files/training_summaries/"mobilenet_0.50_224"
--output_graph=tf_files/retrained_graph.pb
--output_labels=tf_files/retrained_labels.txt
--architecture="mobilenet_0.50_224"
--image_dir=tf_files/flower_photos
```

Jeżeli nie dodano ścieżki pythona do ścieżek systemu należy podać całą ścieżkę dostępu do pliku pyton.exe

Klasyfikacja kwiatów

Uruchomić klasyfikację dowolnego obrazu poleceniem

```
python.exe scripts\label_image.py --graph=tf_files\retrained_graph.pb
--image=tf_files\flower_photos\daisy\5547758_eea9edfd54_n.jpg
--labels=tf_files\retrained_labels.txt
```

Zadanie 1.

Przetrenować dowolną sieć MobileNet oraz Inception na obrazach kwiatów, wybrać kilka obrazów kwiatów (innych niż ze zbioru treningowego) oraz obrazów przedstawiających inne rzeczy niż kwiaty i przeprowadzić klasyfikację.

W sprawozdaniu podać parametry trenowanych sieci, wyniki trenowania oraz klasyfikacji. Sprawdzić dla jakiej minimalnej liczby kroków trenowania sieć osiąga wyniki **final test accuracy** powyżej 80%

Zadanie 2.

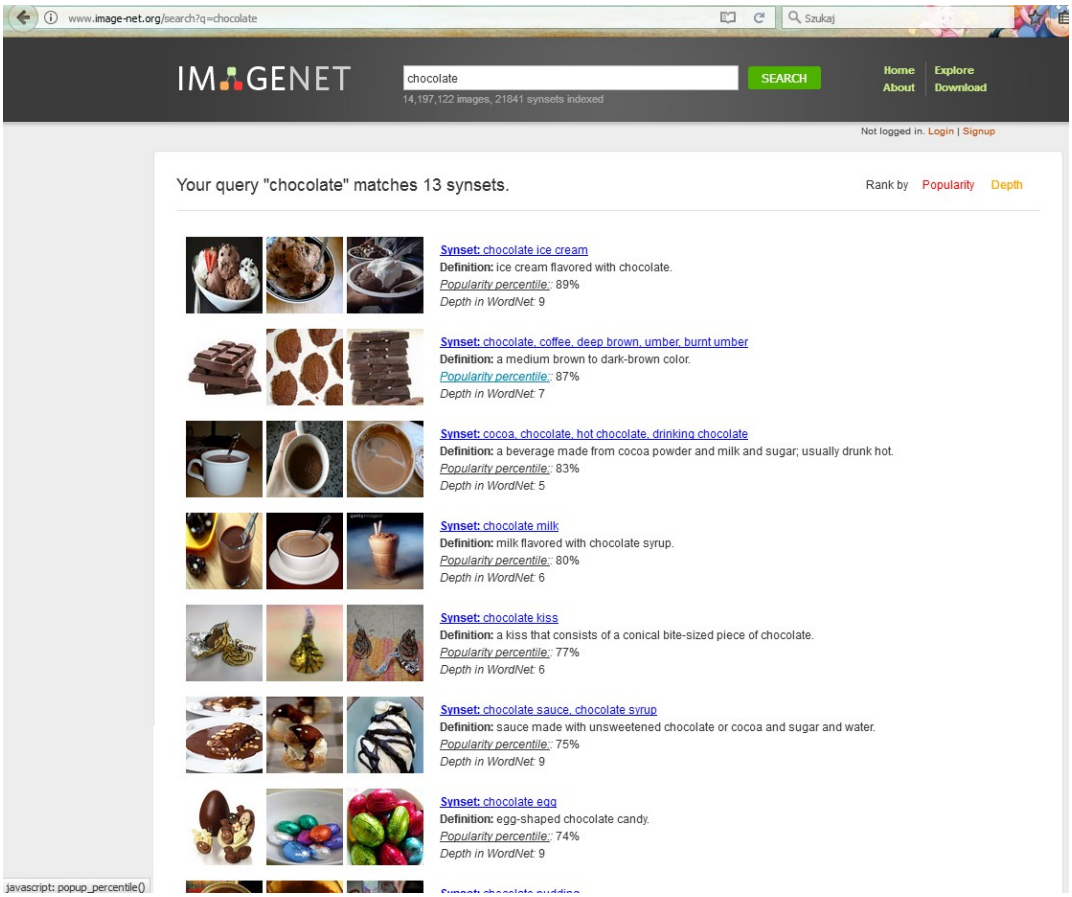
Przetrenować dowolną sieć na swoich obrazach. W sprawozdaniu opisać zbiór uczący, jego klasy, liczność itp. oraz załączyć kilka przykładowych obrazów. Podać wyniki trenowania oraz klasyfikacji.

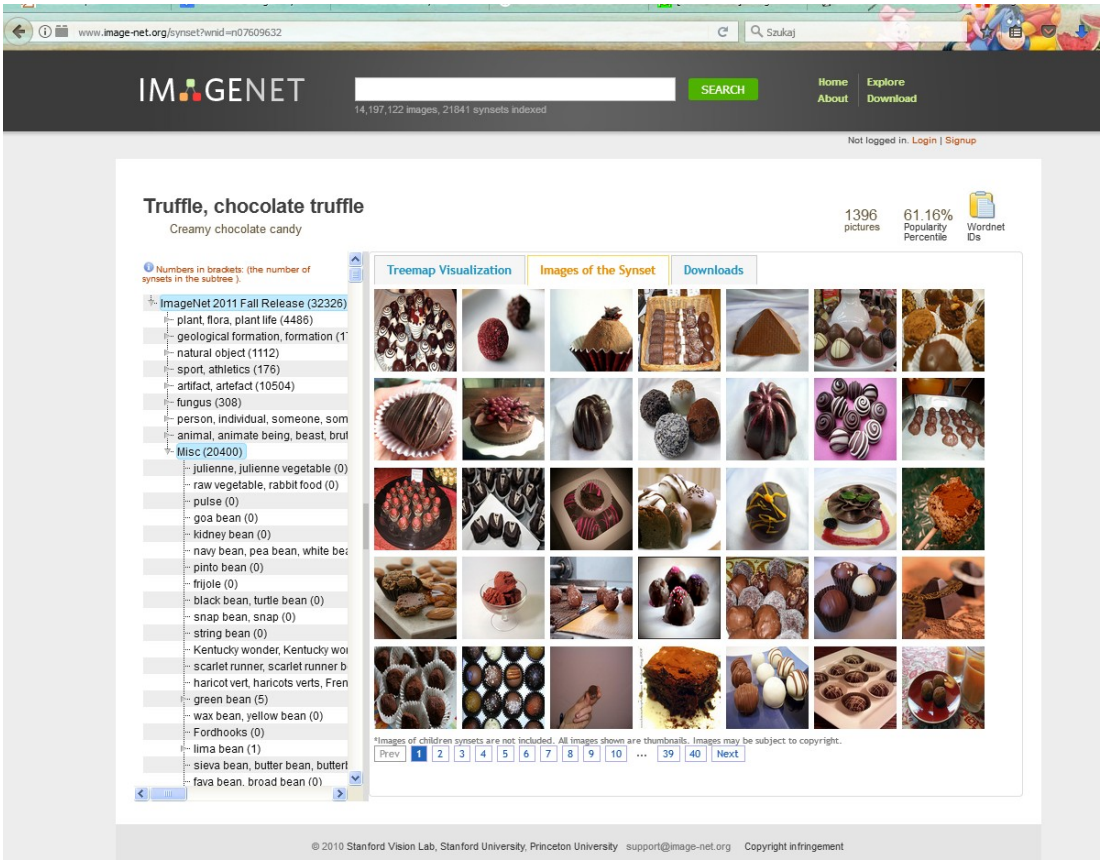
Uwagi do tworzenia zbiorów uczących:

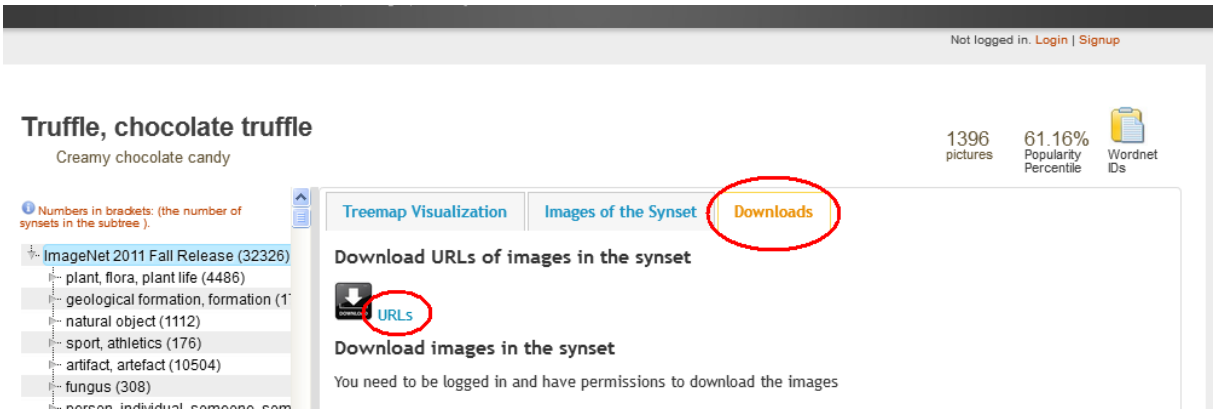
Ze względu na wymaganie dużej liczby obrazów treningowych, tworzenie zbioru uczącego należy zautomatyzować i skorzystać z gotowych baz danych obrazów. Jedną z takich baz jest ImageNet, na której używane w tej instrukcji sieci były trenowane.

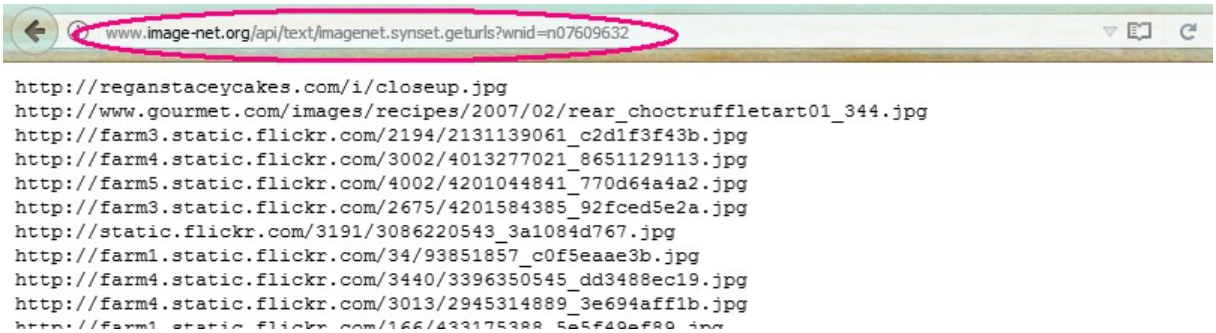
http://www.image-net.org/

Można w niej wyszukać określone obrazy oraz pobrać ich url.









Można też korzystać z innych baz.

Plik pyton zapisujący obrazy z adresów url:

```
import urllib.request
import cv2
import urllib
import os

images_link = 'http://image-net.org/api/text/imagenet.synset.geturls?
wnid=n07609632'
image_urls = urllib.request.urlopen(images_link).read().decode()
pic_num = 1
folder_name='chocolate'


if not os.path.exists(folder_name):
    os.makedirs(folder_name)

for i in image_urls.split('\n'):
    try:
        print(i)
        urllib.request.urlretrieve(i, folder_name+"/"+str(pic_num)+'.jpg')
        pic_num += 1

    except Exception as e:
        print(str(e))
```