Artificial Intelligence, Thomas More Geel 2022-2023

Task: Deep Learning Image Classification

by David Silva Troya

This code is also on my github

1. Scrape together your own image dataset:

The dataset in this Image Classification will be about **Bojack Horseman**.

The data will be dowloaded using a python script that allows to search the data from bing images:

In [ ]:
```
#first all the libraries needed to scrape the data

from bs4 import BeautifulSoup
from PIL import Image
from io import BytesIO
import requests
import json
import os
```

Now, the next code was modified by David Silva Troya, but belongs to:

stackoverflow

In [ ]:
```
def search_at_bing(search, to_folder):
    """This code was modified
    by David Silva Troya, but
    the real one was here:
    https://stackoverflow.com/questions/64226325/is-there-a-way-i-can-download-imag

    # search = input("Search for: ")

    url = "https://www.bing.com/images/search"

    headers = {
        "User-Agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:80.0) Gecko/2010
    }
    params = {"q": search, "form": "HDRSC2", "first": "1", "scenario": "ImageBasicH
    r = requests.get(url, headers=headers, params=params)

    soup = BeautifulSoup(r.text, "html.parser")
    links = soup.find_all("div", {"class": "img_cont hoff"})

    for data in soup.find_all("a", {"class": "iusc"}):
        json_data = json.loads(data["m"])
        img_link = json_data["murl"]
        img_object = requests.get(img_link, headers=headers)
        title = img_link.split("/")[-1]
        # print("Getting: ", img_link)
        # print("Title: ", title + "\n")
        try:
            img = Image.open(BytesIO(img_object.content))
            img_path =f'./photos/{to_folder}/'

            if not os.path.exists(img_path):
                os.makedirs(img_path)
                print(f'folder created for {to_folder}')

            img.save(img_path + title)

        except:
            #Because sometimes is just not possible to get the image.
            print(f'Not Possible to download one image')
```

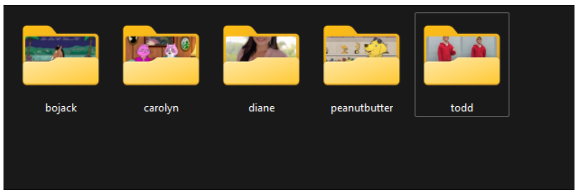
This is the code to get the images from the 5 different characters of Bojack Horseman

In [ ]:
```
#caracters to find as dictionary so a folder can be created with the key name
cast = {
    'bojack' : 'bojack horseman',
    'carolyn' : 'princess carolyn ',
    'diane' : 'diane nguyen',
    'peanutbutter' : 'mr. peanutbutter',
    'todd' : 'todd chavez',
}
#taking all the keys from the dictionary
cast_list = list(cast)


#for the UI /now deactivated for the notebook
#print(f'From the option in {cast_list}')


character = ""

while True:
    #for the UI /now deactivated for the notebook
    # number = input(f'Select a number from 0 to {len(cast_list)-1}: \n')
    number = 0 #here we just select the index of the list keys from the cast

    #this "try" is just to confirme the user wrote a number
    try:
        number = int(number)
        if number>=len(cast_list) or number<0:
            print('Wrong number')
        else:
            character = cast_list[number]
            break
    except:
        print(f'Please, write a number of the list')

#displaying what character was selected
print(f'from {character}:')

#Different options of searching to get different and more images
search_at_bing(cast[character],character)
search_at_bing(f'{cast[character]} happy',character)
search_at_bing(f'{cast[character]} sad',character)
search_at_bing(f'{cast[character]} only',character)
search_at_bing(f'{cast[character]} normal',character)
search_at_bing(f'{cast[character]} season 1',character)
search_at_bing(f'{cast[character]} season 2',character)
search_at_bing(f'{cast[character]} season 3',character)
search_at_bing(f'{cast[character]} season 4',character)
search_at_bing(f'{cast[character]} season 5',character)
```

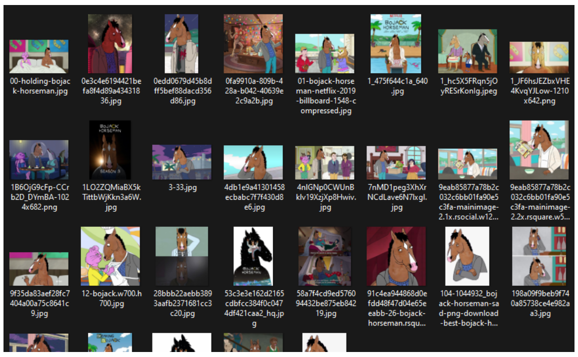
```
from bojack:
folder created for bojack
Not Possible to download one image
Not Possible to download one image
Not Possible to download one image
Not Possible to download one image
Not Possible to download one image
Not Possible to download one image
Not Possible to download one image
```

After to run the code above and change the number of character to be downloaded, the result is:

The data is organized by folders



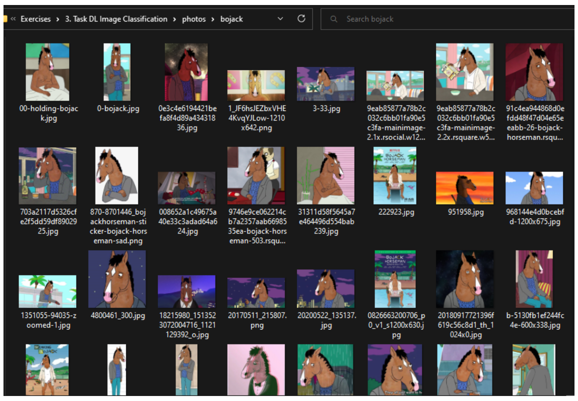
But all this is a dirty data, which means that it won't really helps to classify the different characters. The main reason is that some of the characters are in the same image for every different character:


00-holding-bojack-horseman.jpg
0e3c4e6194421befa8f4d89a43431836.jpg
0edd0679d45b8dff5bef88dacd356d86.jpg
0fa9910a-809b-428a-b042-40639e2c9a2b.jpg
01-bojack-horseman-netflix-2019-billboard-1548-compressed.jpg
1_475f644c1a_640.jpg
1_hc5X5FRqn5jOyRESrKonlg.jpeg
1_JF6hsJEZbxVHE4KvqYJLow-1210x642.png
1B6OjG9cFp-CCrb2D_DYmBA-1024x682.png
1LO2ZQMiaBX5kTittbWjKkn3a6W.jpg
3-33.jpg
4db1e9a41301458ecbabc7f7f430d8e6.jpg
4nIGNp0CWUnBklv19XzjXp8Hwiv.jpg
7nMD1peg3XhXrNCdLave6N7lxgl.jpg
9eab85877a78b2c032c6bb01fa90e5c3fa-mainimage-2.1x.rsocial.w12...
9eab85877a78b2c032c6bb01fa90e5c3fa-mainimage-2.2x.rsquare.w5...
9f35da83aef28fc7404a00a75c8641c9.jpg
12-bojack.w700.h700.jpg
28bbb22aebb3893aafb2371681cc3c20.jpg
53c3e3e162d2165cdbfcc384f0c0474df421caa2_hq.jpg
58a7f4cd9ed576094432be875eb84219.jpg
91c4ea944868d0efdd48f47d04e65eeabb-26-bojack-horseman.rsqu...
104-1044932_bojack-horseman-sad-png-download-best-bojack-h...
198a09f9beb9f740a85738ce4e982aa3.jpg

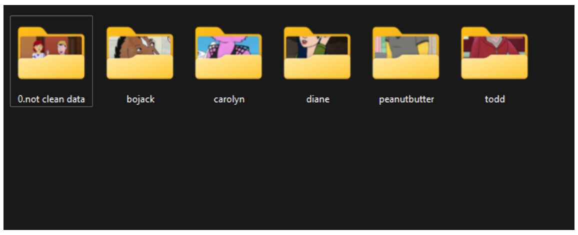
2. Split the data into a training and test set:

Impressibly this part is one of the parts that takes more time, cleaning the data is actually quite easy but even taken a little time for one image, it results is a long time for the big amount of images

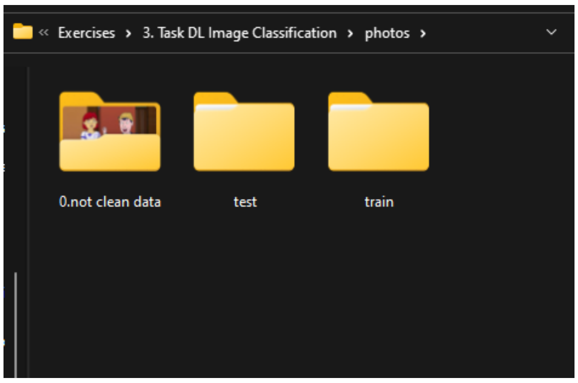
So, a little cleaning of the data and the result is:


Exercises › 3. Task DL Image Classification › photos › bojack
Search bojack

Now all the folder only have the images for one character:



And the last part was split the images for the training(90 images) and the testing(20 images):



3. Design a CNN network with some regularisation options:

Now the fun part comes:

Starting with the most important, the libraries.

In [ ]:
```
# !pip install tensorflow
# !pip install scipy

# importing the Keras libraries and packages
from tensorflow import keras #we need keras from tensorflow
from keras.models import Sequential # to initialise our neural network model as a s
from keras.layers import Conv2D # to perform the convolution operation i.e the firs
from keras.layers import MaxPooling2D # used for the pooling operation
from keras.layers import Flatten # used for Flattening. Flattening is the process o
from keras.layers import Dense # used to perform the full connection of the neural
from keras.layers import Dropout # used to prevent overfitting
```

In [ ]:
```
# initialising the CNN
```

```
model = Sequential()

#50 filters of 3x3 each. Taking images of 64x64 pixels with 3 stands(RGB) with a Re
model.add(Conv2D(50, (3, 3), input_shape = (64, 64, 3), activation = 'relu'))

#Reducing the size of the image with a MaxPooling
model.add(MaxPooling2D(pool_size = (2, 2)))

#random elimination of some conections between layers => 0.2 is 20% of the conectio
model.add(Dropout(0.2))

#adding a second Convolution (repeating previous steps):
model.add(Conv2D(50, (3, 3), input_shape = (64, 64, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Dropout(0.2))

#and why not adding another Convolution (repeating previous steps):
model.add(Conv2D(50, (3, 3), input_shape = (64, 64, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Dropout(0.2))

#converting pooled images into a continuous vector
model.add(Flatten())

#function to add a fully connected layer. The units are the number of nodes that sh
model.add(Dense(activation="relu", units=5))

# compiling the CNN
model.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accurac
```

Now we just check some parameters to know how the whole model looks like

In [ ]:
```
print('Input:')
print(model.input_shape)
print('Output:')
print(model.output_shape)
print('')
print('Summary:')
print(model.summary())
```

```
Input:
(None, 64, 64, 3)
Output:
(None, 5)

Summary:
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_11 (Conv2D)          (None, 62, 62, 50)        1400      
                                                                 
 max_pooling2d_11 (MaxPoolin  (None, 31, 31, 50)       0         
 g2D)                                                            
                                                                 
 dropout_11 (Dropout)        (None, 31, 31, 50)        0         
                                                                 
 conv2d_12 (Conv2D)          (None, 29, 29, 50)        22550     
                                                                 
 max_pooling2d_12 (MaxPoolin  (None, 14, 14, 50)       0         
 g2D)                                                            
                                                                 
 dropout_12 (Dropout)        (None, 14, 14, 50)        0         
                                                                 
 conv2d_13 (Conv2D)          (None, 12, 12, 50)        22550     
                                                                 
 max_pooling2d_13 (MaxPoolin  (None, 6, 6, 50)         0         
 g2D)                                                            
                                                                 
 dropout_13 (Dropout)        (None, 6, 6, 50)          0         
                                                                 
 flatten_7 (Flatten)         (None, 1800)              0         
                                                                 
 dense_13 (Dense)            (None, 5)                 9005      
                                                                 
=================================================================
Total params: 55,505
Trainable params: 55,505
Non-trainable params: 0
_________________________________________________________________
None
```

4. Train your model, but don't overfit (plot the training and validation/test error)

Now lets load the data for the training and test:

In [ ]:
```python
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True,
                                   rotation_range = 90)

test_datagen = ImageDataGenerator(rescale = 1./255)

training_set = train_datagen.flow_from_directory('./photos/train',
                                                 target_size = (64, 64),
                                                 batch_size = 32,
                                                 class_mode = 'binary')

test_set = test_datagen.flow_from_directory('./photos/test',
                                            target_size = (64, 64),
                                            batch_size = 32,
                                            class_mode = 'binary')
```

```
Found 446 images belonging to 5 classes.
Found 95 images belonging to 5 classes.
```

Now lets train the model with the 90 images for each character. So we use a number of training images that is used during every step (step_per_epoch) and the the number of steps (epochs).

In [ ]:

```
model.fit(training_set,
                    steps_per_epoch = 12,
                    epochs = 6,
                    validation_data = test_set)
```

```
Epoch 1/6
12/12 [==============================] - 3s 258ms/step - loss: -15.0896 - accurac
y: 0.2225 - val_loss: -15.0887 - val_accuracy: 0.2632
Epoch 2/6
12/12 [==============================] - 3s 232ms/step - loss: -14.9299 - accurac
y: 0.2277 - val_loss: -15.0887 - val_accuracy: 0.2632
Epoch 3/6
12/12 [==============================] - 3s 232ms/step - loss: -15.3291 - accurac
y: 0.2461 - val_loss: -15.0887 - val_accuracy: 0.2632
Epoch 4/6
12/12 [==============================] - 3s 232ms/step - loss: -15.3690 - accurac
y: 0.2016 - val_loss: -15.0887 - val_accuracy: 0.2632
Epoch 5/6
12/12 [==============================] - 3s 223ms/step - loss: -15.7682 - accurac
y: 0.2277 - val_loss: -15.0887 - val_accuracy: 0.2632
Epoch 6/6
12/12 [==============================] - 3s 229ms/step - loss: -15.8880 - accurac
y: 0.2277 - val_loss: -15.0887 - val_accuracy: 0.2632
```

Out[ ]:

```
<keras.callbacks.History at 0x25a85943070>
```

Yeap.. the accuracy is not good at all. I'm still trying to understand why... So I am stuck here for now.

In [ ]:

```
model.save_weights('./saved_models/modelcifar-10.h5')
```

In [ ]:

```
import numpy as np
from tensorflow.keras.preprocessing import image

# test_image = image.load_img("./photos/single_image/BoJack-Horseman-1.jpg", target
test_image = image.load_img("./photos/single_image/BoJack-Horseman.png", target_siz
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis = 0)
result = model.predict(test_image)

print(result)
```

```
1/1 [==============================] - 0s 93ms/step
[[3651.7058 4181.6455 3961.5227 4307.317  4332.819 ]]
```

5. Compare your model's performance to Google's Teachable Machine, using the same training dataset

Still in progress...

Some websites that helps me to make this program:

-https://www.tensorflow.org/api_docs/python/tf/keras/Sequential

-https://stackoverflow.com/questions/72383347/how-to-fix-it-attributeerror-module-keras-preprocessing-image-has-no-attribu

-https://stackoverflow.com/questions/70961988/layer-count-mismatch-when-loading-weights-from-file-model-expected-106-layers

-https://www.tensorflow.org/tutorials/images/data_augmentation

-https://stackoverflow.com/questions/59864408/tensorflowyour-input-ran-out-of-data

-https://towardsdatascience.com/data-augmentation-compilation-with-python-and-opencv-b76b1cd500e0

-https://neptune.ai/blog/data-augmentation-in-python

-https://www.kaggle.com/code/prateek0x/multiclass-image-classification-using-keras

-https://stackoverflow.com/questions/67960945/keras-and-tensorflow-on-python-valueerror-input-data-in-numpyarrayiterator

-https://pub.towardsai.net/multiclass-image-classification-hands-on-with-keras-and-tensoflow-e1cf434f3467

-https://www.analyticsvidhya.com/blog/2020/10/create-image-classification-model-python-keras/

-https://www.geeksforgeeks.org/python-image-classification-using-keras/#:~:text=Image%20classification%20is%20a%20method,of%20the%20model%20using%2

-https://keras.io/api/preprocessing/image/