Signature Verification

An Individual Project

COMP 3710 – Applied Artificial Intelligence

Thompson Rivers University

Hryhorii Pertaia

T00531749

Table of Contents

Introduction **2**

Background **3**

Project Details **3**

Results **6**

References: **6**

Introduction

According to recent studies, check fraud alone costs banks $ 900 million a year, with 22% of all fraudulent checks related to signature fraud. Obviously, more than 27.5 billion (according to a 2010 study of the Federal Reserve's payments) conducted annually in the United States, visually comparing signatures with manual effort on hundreds of millions of verified daily checks, is not practical.

Handwritten signatures are very important in our social and legal life for verification and authentication. A signature may be accepted only if it is from the intended person. The likelihood that two signatures of the same person will be the same is very small. Many properties of a signature may differ, even if two signatures are made by the same person. Thus, fake detection becomes a difficult task. Despite the fact that in recent years there have been many publications in the field of verification of handwritten signatures, the task of creating highly effective tools is still open for new technologies.

Background

For my final project, I chose to work on the Signature Verification Dataset. It seems to me quite interesting, because there is a way to apply machine learning to it so that it can learn whether any signature is a real signature.

The dataset that I chose contains both genuine and fraud signatures of Dutch users. All the data were extracted from ICDAR 2011 signature dataset.

In this dataset, I expect to find patterns through which the system would be able to identify whether the signatures are genuine or fraud.

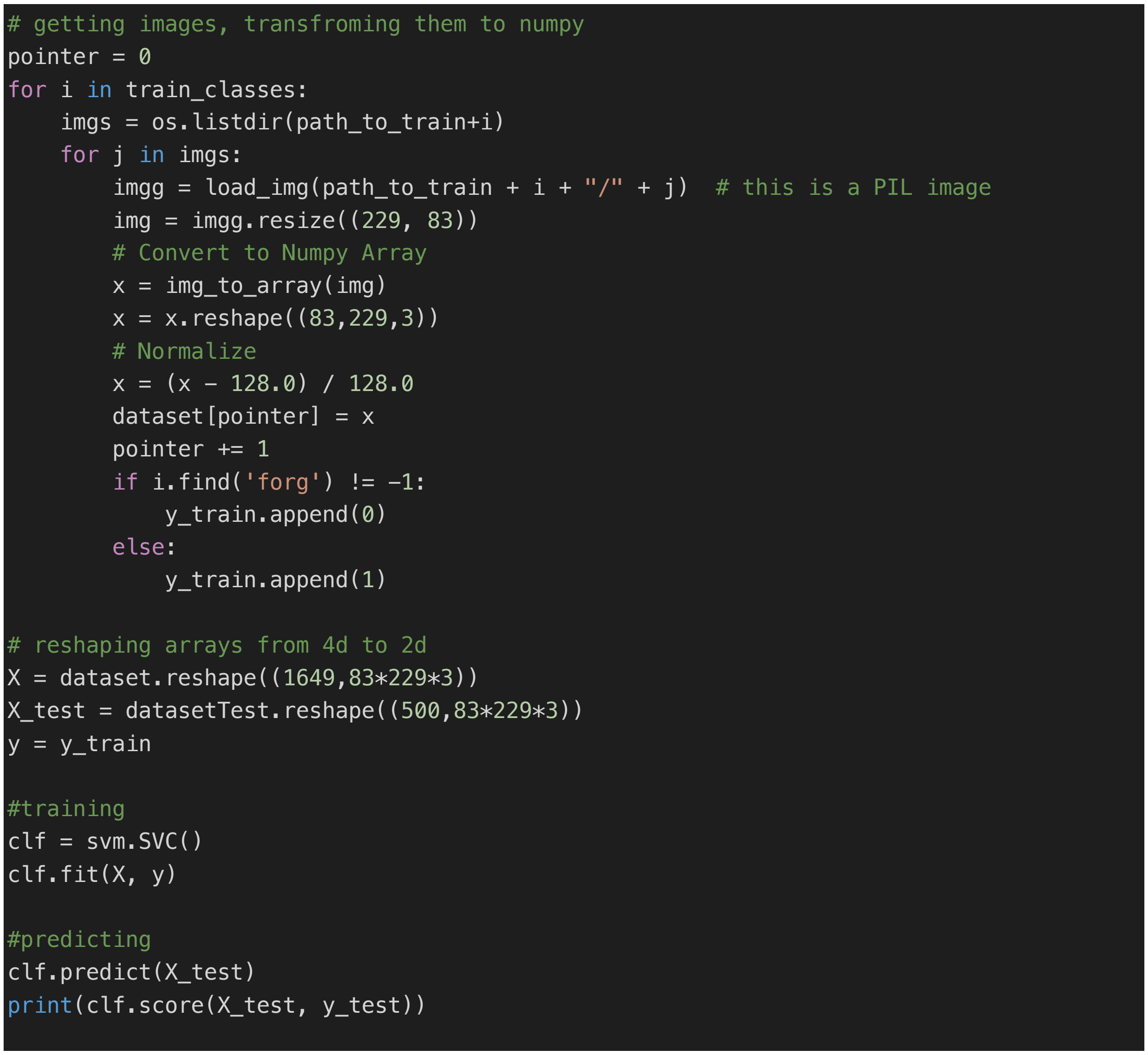
For my project, I will use classification machine learning technique to predict the class value for given signature. The output will show whether signature is real or fake.

Project Details

Test class:

```
# importing necessary packages
import numpy as np
import pandas as pd
import os
import torch
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn import svm
from PIL import Image
from sklearn.model_selection import train_test_split
from keras.preprocessing.image import ImageDataGenerator, array_to_img, img_to_array,
load_img
torch.manual_seed(0)


# defining paths
path_to_train = "./sign_data/sign_data/train/"
path_to_test = "./sign_data/sign_data/test/"
train_classes = os.listdir(path_to_train)
test_classes = os.listdir(path_to_test)
```

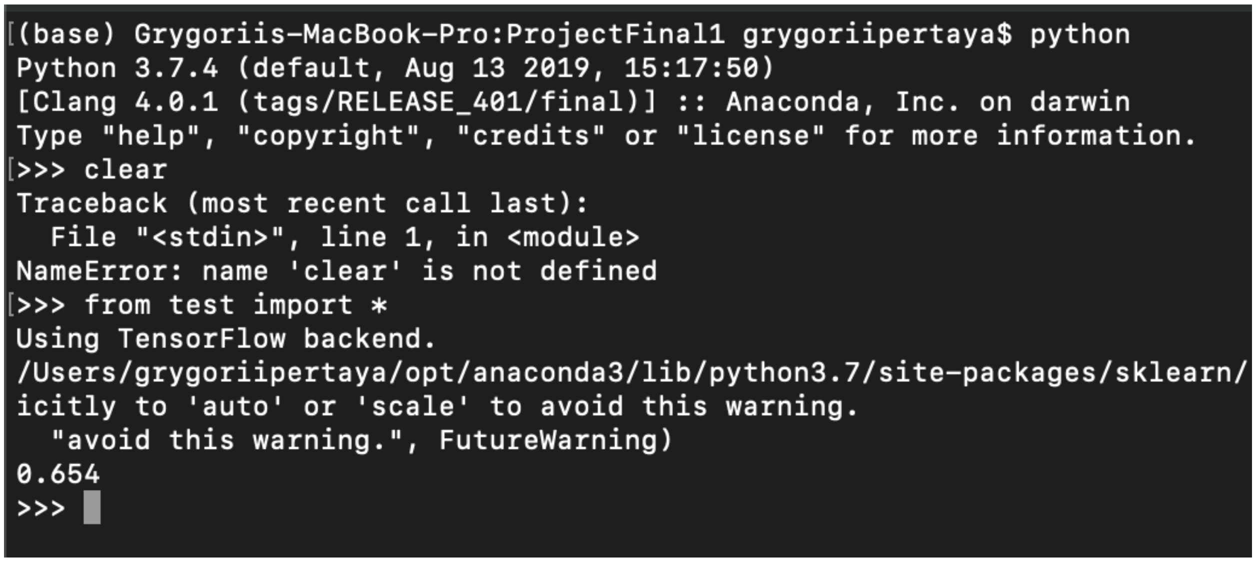
```python
# defining y sets for test and train
y_train = []
y_test = []

#test - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

# calculating number of test images
value_of_img1 = 0
for ii in test_classes:
    if os.path.isdir(path_to_test+ii) == 1:
        value_of_img1 += len(os.listdir(path_to_test+ii))

# creating dataset for test
datasetTest = np.ndarray(shape=(500,83,229,3),
                     dtype=np.float32)

# getting images, transfroming them to numpy
pointer1 = 0
for i in test_classes:
    if os.path.isdir(path_to_test+i):
        imgs = os.listdir(path_to_test+i)
    for j in imgs:
        if os.path.isdir(path_to_test+i) == 1 and j.find("DS") == -1:
            imgg = load_img(path_to_test + i + "/" + j)  # this is a PIL image
            img = imgg.resize((229, 83))
            # Convert to Numpy Array
            x = img_to_array(img)
            x = x.reshape((83,229,3))
            # Normalize
            x = (x - 128.0) / 128.0
            datasetTest[pointer1] = x
            pointer1 += 1
            if i.find('forg') != -1:
                y_test.append(0)
            else:
                y_test.append(1)

#train - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

# calculating number of train images
value_of_img = 0
for ii in train_classes:
    value_of_img += len(os.listdir(path_to_train+ii))

# creating dataset for train
dataset = np.ndarray(shape=(value_of_img,83,229,3),
                     dtype=np.float32)
```



```python
# getting images, transfroming them to numpy
pointer = 0
for i in train_classes:
    imgs = os.listdir(path_to_train+i)
    for j in imgs:
        imgg = load_img(path_to_train + i + "/" + j)  # this is a PIL image
        img = imgg.resize((229, 83))
        # Convert to Numpy Array
        x = img_to_array(img)
        x = x.reshape((83,229,3))
        # Normalize
        x = (x - 128.0) / 128.0
        dataset[pointer] = x
        pointer += 1
        if i.find('forg') != -1:
            y_train.append(0)
        else:
            y_train.append(1)

# reshaping arrays from 4d to 2d
X = dataset.reshape((1649,83*229*3))
X_test = datasetTest.reshape((500,83*229*3))
y = y_train

#training
clf = svm.SVC()
clf.fit(X, y)

#predicting
clf.predict(X_test)
print(clf.score(X_test, y_test))
```

Results

Was able to achieve 65% accuracy. Screenshot attached below.



```
[(base) Grygoriis-MacBook-Pro:ProjectFinal1 grygoriipertaya$ python
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
[>>> clear
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'clear' is not defined
[>>> from test import *
Using TensorFlow backend.
/Users/grygoriipertaya/opt/anaconda3/lib/python3.7/site-packages/sklearn/
icitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
0.654
>>> 
```

References:

Osborn, A.: Questioned Documents. Nelson Hall Pub. (1929)Google Scholar

Robertson, E.W.: Fundamentals of Document Examination. Nelson-Hall (1991)Google Scholar

Bradford, R.R., Bradford, R.: Introduction to Handwriting Examination and Identification. Nelson-Hall (1992)Google Scholar

Hilton, O.: Scientific Examination of Questioned Documents. CRC Press, Boca Raton (1993)Google Scholar

Huber, R., Headrick, A.: Handwriting Identification: Facts and Fundamentals. CRC Press, Boca Raton (1999)CrossRefGoogle Scholar

Slyter, S.A.: Forensic Signature Examination. Charles C. Thomas Pub. (1995)Google Scholar

Mitchell, T.M.: Machine Learning. McGraw-Hill, New York (1997)zbMATHGoogle Scholar

Srihari, S.N., Xu, A., Kalera, M.K.: Learning strategies and classification methods for off-line signature verification. In: Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition(IWHR), pp. 161–166. IEEE Computer Society Press, Los Alamitos (2004)CrossRefGoogle Scholar

Winston, P.: Learning structural descriptions from examples. In: Winston, P. (ed.) The Psychology of Computer Vision, pp. 157–210. McGraw-Hill, New York (1975)Google Scholar

Leclerc, F., Plamondon, R.: Automatic signature verification: the state of the art, 1989-1993. International Journal of Pattern Recognition and Artificial Intelligence 8, 643–660 (1994)CrossRefGoogle Scholar

Guo, J.K., Doermann, D., Rosenfield, A.: Local correspondences for detecting random forgeries. In: Proceedings of the International Conference on Document Analysis and Recognition, pp. 319–323 (1997)Google Scholar

Plamondon, R., Srihari, S.N.: On-line and off-line handwriting recognition: A comprehensive survey. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 63–84 (2000)CrossRefGoogle Scholar