Guess Who App
Documentation

by Marinos Galiatsatos

CONTENTS

App's Folder Map	page 1
Small Description	page 2
How to run the app's scripts	page 3

APP's FOLDER MAP

GUESS_WHO_APP
- EXTRAS
  - capture_video.py
- EYEGLASSES
  - negative glasses
  - positive glasses
  - test glasses
  - eyeglass_libsvm.py
  - eyeperceptron.xml
  - testeyes.py
- HAIR
  - BLACK
    - negative black
    - positive black
    - test black
  - BLONDE
    - negative blonde
    - negative blonde
    - test blonde
  - BROWN
    - negative brown
    - positive brown
    - test brown
  - hair_libsvm.py
  - test_hairlibsvm.py
- HATS
  - negative hats
  - positive hats
  - test hats
  - hatperceptron_t2.xml
  - hatsvm.py

    - test_hats.py
- SHIRTS
    - BLACK
        - negative black
        - positive black
        - test black
    - BLUE
        - negative blue
        - positive blue
        - test blue
    - GREEN
        - negative green
        - positive green
        - test green
    - RED
        - negative red
        - positive red
        - test red
    - WHITE
        - negative white
        - positive white
        - test white
    - RGB HSV SVMs
    - RGB S SVMs
    - RGB SVMs
    - shirt_libsvm.py
    - testshirtsvm.py
- TRAINED_ALGORITHMS
- collect_all.py
- detection.py
- Guess Who App – Description.pdf
- Guess Who App – Documentation.pdf

SMALL DESCRIPTION

This App provides useful tools for:

1. Person's shirt and hair color detection
2. Decision if a Person wears or not eyeglasses and hat

The App uses libsvm and perceptron machine learning algorithms for classification.

It can detect a red, green, blue, white and black shirt and brown, blonde and black hair.
Also, it can detect lots of types of eyeglasses and a specific type of hat.
(for more details take a look at: Guess Who App – Description file)

HOW TO RUN THE APP'S SCRIPTS

Requirements:

1. Download and Install opencv-2.4.9: http://www.samontab.com/web/2014/06/installing-opencv-2-4-9-in-ubuntu-14-04-lts/
2. Install GSL: 'sudo apt-get install libgsl0ldbl' without quotes
3. Download and install mlpy 3.5.0: http://sourceforge.net/projects/mlpy/files/

collect_all.py

- Description:

  The python script collect_all.py can collect all the data that we want, in order to detect the shirt's and hair' color and to decide, if someone wears or not eyeglasses and hat.
  We have created Region of Interests (ROIs) in order to retrieve data from certain regions of the video. The regions are represented by colorful rectangles so we can easily check their positions.
  The Person must be at approximately 2 meters from the camera. It will be useful if there is something white behind him, so we will not have to deal with wrong detected faces or wrong colors. If all the rectangles are created correctly then an 'OK!' message will appear on the screen.
  From every region we collect data from each frame of the video. The frames that we need are 100. The user has the ability to decide whether to start collecting the data. Also, he can collect data in real time or to collect data from a prerecorded video.
  We divide every ROI in 6 smaller ROIs, which they have the same size, in order to collect more features.

  Shirt ROI: For shirt classification we collect Histogram values with 10 bins from every sub shirt roi with the getRGBHistogram() and getHSVHistogram() functions. We have thought 2 ways in order to decide which way is better for color classification. The 1st way is to get from each sub shirt roi, only the RGB Histogram values. The 2nd way is to take the RGB and HSV Histogram values.

  The colors that we classify are red, green, blue, white and black. The files for each way are 3: a test file, a positive training file (with 1s at the last column) and a negative training file (with -1s at the last column). The files are csv.

  The feature vector for RGB and RGB HSV values of one frame has length 360 features and takes one row in the csv file.

  Finally, we save the trained classifiers in an xml file, in order to use them for detection.

  Hair ROI: For hair classification we collect Histogram values with 10 bins from every sub hair roi with the getRGBhistogram() function. The colors that we classify are brown, black and blonde. The files that are being created are 3: a test file, a positive train file (with 1s at the last column) and a negative training file (with -1s at the last column). The files are csv.
  Important note: due to the fact that we did not have many training data, we just provide the code for training and classification, alongside with the neccesary functions.

Eyeglasses ROI: For eyeglasses detection we collect values from Canny Edges for each sub eye roi. We have 2 values 'wears' and 'doesn't wear'. The files that are being created are 3: a test file, a positive train file (with 1s at the last column) and a negative training file (with -1s at the last column). The files are csv.

Hat ROI: For hat detection we collect values from Canny Edges and FloodFill. The type of the hat that we try to classify are being shown in the this picture. We chose that type because it is easier to detect than other types of hats like jockeys. We have 2 values 'wears' and 'doesn't wear'. The files that are being created are 3: a test file, a positive train file (with 1s at the last column) and a negative training file (with -1s at the last column). The files are csv.

- Run the script:
  The script takes 9 arguments.
  - **--pn** : this argument takes as input the name of the person that we want to record for training our classifiers. (i.e. --pn marinos)
  - **--gla** : this argument takes as input 'true' if the person wears eyeglasses and 'false' if the person does not wear. The input values are without quotes. (i.e. --gla true)
  - **--hc** : this argument takes as input the hair' color of the person. (i.e. --hc brown)
  - **--sc** : this argument takes as input the shirt's color of the person. (i.e. --sc red)
  - **--hat** : this argument takes as input 'true' if the person wears a hat or 'false' if the person does not wear a hat. The input values are without quotes. (i.e. --hat false)
  - **--q** : this argument takes as input 'true' if the person wants to collect data from a prerecorded video and 'false' if he wants to detect someone in real time. The input values are without quotes. (i.e. --q false)
    - **--v** : this arguments is used if the person has chosen 'true' in the –q argument. If he has chosen 'false' then you can skip it. (i.e. --v marinos_video.avi)
  - **--path** : this argument takes as input the path to the opencv folder. (i.e. --path /home/username/opencv2.4.9/)
  - **--c** : this argument takes as input 0,1,2,... values, which correspond to the order of webcams that you have in your pc. For example if you have 2 webcams the first one will have the value 0 and the other 1. If you have only one webcam then you choose 0. (i.e. --c 0)

  A complete example:
  ```
  $ python collect_all.py --pn marinos --gla false --hc brown --sc black --hat true --q false
  --path /home/marinos/opencv2.4.9/ --c 0
  ```

detection.py

- Description:
  The python script detection.py can detect the shirt's and hair' color of a person and also it can detect if someone wears or not eyeglasses and hat.
  The person must be at approximately 2 meters from the webcam. It creates all the necessary ROIs, like the collect_all.py script, but when the detection has finished then it deletes all the files that were created during the detection.
  Important Note: Due to the lack of hair training samples the hair' color is disabled. If you train the hair color libsvm that we provide, then you can uncommented the code and run the script.

Also, we have provided functions that can collect the color of the hat, but again due to the lack of samples we have disabled it.
The user can detect a person in real time and also can detect someone through a prerecorded video.
So to sum up in the end, the scripts detects: the shirt's color, if someone wears or not eyeglasses and if someone wears or not hat.

- <u>Run the script:</u>
  The script takes 5 arguments:
  - **--q** : this argument takes as input 'true' if the user want to detect someone from a prerecorded video and 'false' if the user want to detect someone real time. The input values are without quotes. (i.e. --q true)
    - **--v** : this arguments is used if the person has chosen 'true' in the –q argument. If he has chosen 'false' then you can skip it. (i.e. --v marinos_video.avi)
  - **--path** : this argument takes as input the path to the opencv folder. (i.e. --path /home/username/opencv2.4.9/)
  - **--apppath** : this argument takes as input the path to the App folder. (i.e. --apppath /home/username/documents/GUESS_WHO_APP/)
  - **--c** : this argument takes as input 0,1,2,... values, which correspond to the order of webcams that you have in your pc. For example if you have 2 webcams the first one will have the value 0 and the other 1. If you have only one webcam then you choose 0. (i.e. --c 0)

  A complete example:
  $ python detection.py --q false --path /home/marinos/opencv2.4.9/ --path /home/username/documents/GUESS_WHO_APP/ --c 0

shirt_libsvm.py

- <u>Description:</u>
  In this script we train our shirts' color libsvms. We provide the positive and the negative csv files for every color. The files are preloaded and the user must only enter the path to the app. If someone wants to change the training files, then he must write his files into the code.
  The classifiers are saved in xml file.

- <u>Run the script:</u>
  This script takes 1 argument.
  - --apppath : this argument takes as input the path to the App Folder. (i.e –apppath /home/username/documents/GUESS_WHO_APP/)

A complete example:
$ python shirt_libsvm.py --apppath /home/username/documents/GUESS_WHO_APP/

testshirtsvm.py

- <u>Description:</u>
  This script takes as input a test set and shows the results for every trained shirts' color

libsvm. It can show the result from the 2 ways that we have previously mentioned (RGB, RGBHSV). The results are presented with many details, so the user can decide which way is more suitable for him.

- <u>Run the script:</u>
  This script takes 3 arguments.
  - **--test1** : this argument takes as input the name of test file with RGB type of color values. (i.e. --test1 test_red_color.csv)
  - **--test2** : this argument takes as inout the name of the test file with RGB HSV type of color values. (i.e. --test2 test_red_color.csv)
  - **--apppath** : this argument takes as input the path to the App Folder. (i.e –apppath /home/username/documents/GUESS_WHO_APP/)

hatsvm.py

- <u>Description:</u>
  This script we train a Perceptron machine learning algorithm for hat detection. With the help of Canny Edges and FloodFill we feed it with the positive and negative values. If the user want to change the training files, then he has to write their names in the code. After training the trained Perceptron is saved in a xml file.

- <u>Run the script:</u>
  This script takes 1 argument.
  - **--apppath** : this argument takes as input the path to the App Folder. (i.e –apppath /home/username/documents/GUESS_WHO_APP/)

test_hats.py

- <u>Description:</u>
  This script detects if a person wears or not a hat. We load the trained Perceptron and we give it the test set that we want to test. The possible results could be: wears, maybe wears, doesn't wear and maybe doesn't wear.

- <u>Run the script:</u>
  This script takes 2 arguments.
  - **--test** : this argument takes as input the test csv file that we want to test with Hat Perceptron. (i.e. --test hat_test.csv)
  - **--apppath** : this argument takes as input the path to the App Folder. (i.e –apppath /home/username/documents/GUESS_WHO_APP/)

hair_libsvm.py

- <u>Description:</u>
  This script trains 3 libsvms algorithms for brown, blonde and black hair color. The trained libsvms are saved in xml file.
  Important note: Most of the code is commented because of the lack of training data. If you

can find more data you can pass your own files.

- Run the script:
  This script takes 1 argument.
  - **--apppath** : this argument takes as input the path to the App Folder. (i.e –apppath /home/username/documents/GUESS_WHO_APP/)

test_hairlibsvm.py

- Description:
  This script takes as input a test hair file and it can detect a person's hair' color. The possible 3 answers are brown, blonde and black hair.

- Run the script:
  This script takes 2 arguments.
  - **--test** : this argument takes as input the test csv file that we want to test with the 3 libsvms. (i.e. --test hat_test.csv)
  - **--apppath** : this argument takes as input the path to the App Folder. (i.e –apppath /home/username/documents/GUESS_WHO_APP/)

eyeglasses_svm.py

- Description:
  This script takes trains an Perceptron algorithm in order to detect if someone wears or not eyeglasses. The files for training are written inside the code, so if the user wants to retrain the Perceptron with his own data, then he has to pass the files' names in the code.

- Run the script:
  This script takes 1 argument.
  - **--apppath** : this argument takes as input the path to the App Folder. (i.e –apppath /home/username/documents/GUESS_WHO_APP/)

testeyes.py

- Description:
  This script takes as input a test eye file and it can detect if someone wears or not a hat. It loads the previously trained Hat Perceptron.

- Run the script:
  This script takes 2 arguments.
  - **--test** : this argument takes as input the test csv file that we want to test.
    (i.e. --test hat_test.csv)
  - **--apppath** : this argument takes as input the path to the App Folder. (i.e –apppath /home/username/documents/GUESS_WHO_APP/)

capture_video.py

- Description:
  This script starts recording a video until the user presses the 'ESC' key.

- Run the script:
  This script takes 1 argument.
  - **--v** : this argument takes as input the preferable name of the video that the user wants to give, without the postfix .avi or something else. (i.e --v marinos_video)

Home

End of Documentation