Facial recognition

Data collection

• We collected our own data by taking frontal facial images of ourselves and celebrities

Information about facial recognition read from

• https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/

• https://towardsdatascience.com/real-time-face-recognition-an-end-to-end-project-b738bb0f7348

• https://www.youtube.com/watch?v=jG3bu0tjFbk

• http://scipy-lectures.org/advanced/image\_processing/

• https://www.youtube.com/watch?v=PmZ29Vta7Vc

• https://coding-robin.de/2013/07/22/train-your-own-opencv-haar-classifier.html

• https://www.promptcloud.com/blog/how-to-scrape-instagram-data-using-python/

Tools used:

• Requirements for imports are OpenCV, Pillow, Pickle, Os, numpy glib, requests

• We used OpenCV features to help with this project

• We used “haarcascade\_frontalface\_default.xml” to help detect faces in the webcam

• We used faces-train.py to train the algorithm and to create our own yml file

To make the facial recognition more accurate,

• We converted colored images to grey to reduce color bias

• We used gradient vectors to to avoid lighting bias

Extra features

• Detects a face and if that face is trained, will show the Instagram stats of that person

• It shows, Name, Username, Followers, Following, and Posts

Folders Outline

- data set

-names of insta account, and photos

- encode\_faces.py

- encodingss.pickle

- Insta\_Info\_scraper.py

- users.txt

- main.py

History:

- we started studying how we are going to tackle the problem

- we tried pertained algorithms to start with. this took a lot of time

- we installed the packages and it was failing most of the time.

-when we finally made our first version to work, it was an image detection.

- started with simple face detection on images, draws boxes on faces using HAAR cascades

- show the results…

- moved to face detection on webcam and draws boxes on live feed using HAAR cascades

- show the results

- moved to facial recognition on webcam using HAAR cascades, can classify faces not too accurately

-we decided to gray out the image while also using gradients to eliminate bias, it was a little bit more accurate. (sample of gradient pictures) just x or y, vs both x and y

- show the results

-we decided that we will automatically capture and save thousands of images from the webcam feed, to automatically retrain a HAAR cascade and make it more accurate. this will take hours.

-the problem is, it took so much computing power that we lowered it down to only hundreds. even then, it still too much images to process. and that we would need to upgrade our hardware. we decided not to follow through this feature.

- we implemented another feature with is web scraping. it takes the instagram details of the person identified

-show results

- there were problems with identification since it wasn’t very accurate.

- we switched to HOG instead of HAAR.

- it is more accurate but very slow. also, when there were unknown faces, the system would crash.

- we fixed the bugs, and made it faster (moention how it was made faster)

-show the results

- we added another feature and put it in the dash plotly app as a host

- fixed bugs (won’t show in the app, crashes in the app, unknown faces crash)

- added image capture

Applications

Law enforcement - recognition of criminals, finding missing children

Example: instead of instagram, there will be a database with all the information of people. and their IDs such as social security numbers. The application can pull out information from this database based on the face detected. It will identify who the person is. theoretically, with better and better algorithms and training, the machine should recognize and classify faces better than the average human.