celeb$_{r}ecognitionDocumentation$

Release main

Nov 09, 2021

Contents

Contents:

CHAPTER 1

Celebrity Recognition

Model to recognize celebrities using a face matching algorithm.

Model is based on a dataset of around 6000 images of 60 celebrities (100 each).

1.1 Basic working of the algorithm

- Face detection is done using MTCNN face detection model.
- Face encodings are created using VGGFace model in keras.
- Face matching is done using annoy library (spotify).

CHAPTER 2

Installation

- Run `pip install -r requirements.txt` to install all the dependencies (preferably in a virtual environment).

2.1 PyPI package

- To ensure you have all the required additional packages, run `pip install -r requirements.txt` first.
- To install pip package, run:

```
# pip release version
pip install celeb-detector
# also install additional dependencies with this (if not installed via␣
↪requirements.txt file)
pip install annoy keras-vggface keras-applications
# Directly from repo
pip install git+https://github.com/shobhit9618/celeb_recognition.git
```

- If you are using conda on linux or ubuntu, you can use the following commands to create and use a new environment called celeb-detector:

```
conda env create shobhit9618/celeb-detector
conda activate celeb-detector
```

This will install all the required dependencies. To ensure you are using the latest version of the package, also run (inside the environment):

```
pip install --upgrade celeb-detector
```

CHAPTER 3

Using pip pakcage

- For using my model for predictions, use the following lines of code after installation:

```
import celeb_detector # on running for the first time, this will
download vggface model
img_path = 'sample_image.jpg' # this supports both local path and web url like␣
↪https://sample/sample_image_url.jpg
celeb_detector.celeb_recognition(img_path) # on running for the first time, 2␣
↪files (celeb_mapping.json and celeb_index_60.ann) will be downloaded to the␣
↪home directory
```

  This returns a list of dictionaries, each dictionary contains bbox coordinates, celeb name and confidence for each face detected in the image (celeb name will be unknown if no matching face detected).

- For using your own custom model, also provide path to json and ann files as shown below:

```
import celeb_detector
img_path = 'sample_image.jpg'
ann_path = 'sample_index.ann'
celeb_map = 'sample_mapping.json'
celeb_detector.celeb_recognition(img_path, ann_path, celeb_map)
```

- For creating your own model (refer next section for more details on usage) and run as follows:

```
import celeb_detector
folder_path = 'celeb_images'
celeb_detector.create_celeb_model(folder_path)
```

CHAPTER 4

Create your own celeb model

- Create a dataset of celebs in the following directory structure: A root folder (say `celeb_images`), inside this should be the folders corresponding to each of the celebs inside which would be the individual pics of the celebs.
- Each folder name will be considered as the corresponding celeb name for the model (WARNING: Do not provide any special characters or spaces in the names).
- Make sure each image has only 1 face (of the desired celebrity), if there are multiple faces, only the first detected face will be considered.
- Provide path to the dataset folder (for example, `celeb_images` folder) in the create_celeb_model.py file.
- Run create_celeb_model.py file.
- Upon successful completion of the code, we get `celeb_mapping.json` (for storing indexes vs celeb names), `celeb_index.ann` (ann file for searching encodings) and `celeb_name_encoding.pkl` files (for storing encodings vs indexes for each celeb). (WARNING: You need to provide paths for storing each of these files, default is to store in the current directory)

CHAPTER 5

Predictions

5.1 Model predictions in jupyter

- Provide paths to `celeb_mapping.json` and `celeb_index.ann` files in celeb_recognition.ipynb file. If you want to try my model, ignore this step.
- Run all the cells in the celeb_recognition.ipynb file, the final cell will provide widgets for uploading images and making predictions (this will also download the necessary model files).
- NOTE: celeb_recognition.ipynb is a standalone file and does not require any other files from the repo for running.

5.2 Model predictions in python

- Provide paths to `celeb_mapping.json` and `celeb_index.ann` files in celeb_recognition.py and celeb_utils/celeb_utils.py files. If you want to try my model, ignore this step.
- Run celeb_recognition.py file, provide path to image in the file.
- Output includes a list of the identified faces, bounding boxes and the predicted celeb name (unknown if not found).
- It also displays the output with bounding boxes.

5.3 Sample image output

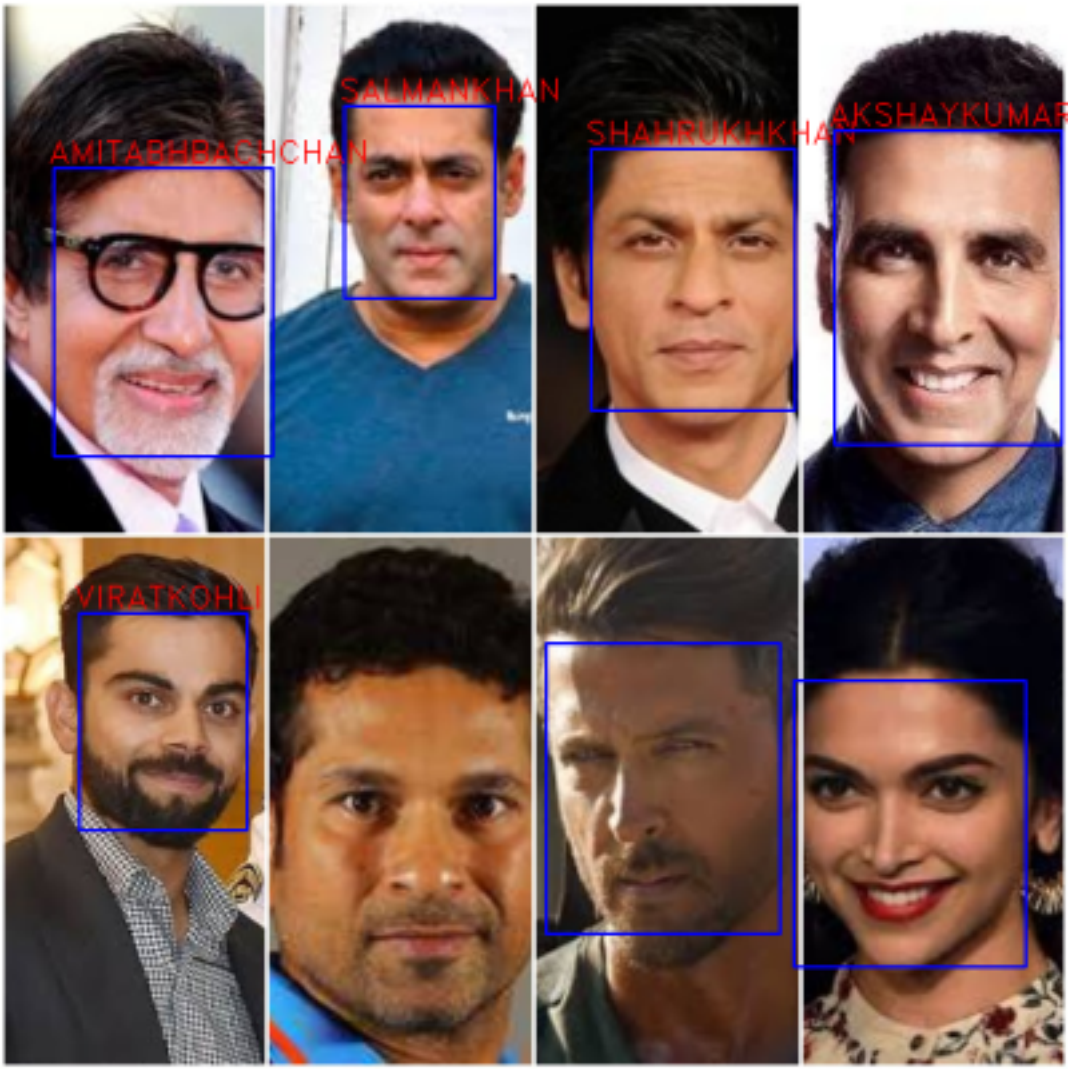
AMITABHBACHCHAN
SALMANKHAN
SHAHRUKHKHAN
AKSHAYKUMAR
VIRATKOHLI

Fig. 1: Image

CHAPTER 6

Additional ways to use

6.1 Binder

You can run a binder application by clicking the following link:

You can also launch a voila binder application (which only has widgets for image upload and celeb prediction) by clicking here.

6.2 Google Colab

To open and run celeb_recognition.ipynb file in google colab, click the following link: