Building a Face Recognition System with Email Alerts

Project Overview
The project aims to create a facial recognition system out of a Raspberry Pi 4 that can recognize visitors and send email alerts upon their arrival. Using the capabilities of Python, OpenCV for real-time computer vision, and SendGrid for email communication, the system integrates machine learning algorithms to recognize and manipulate faces.

Requirements
- Python 3.6 or newer
- A webcam (built-in or external) -We used an external vlog-style webcam
- A SendGrid account for API communication?

Software Requirements
- Git: To clone facial recognition packets
- OpenCV: For real-time computer vision
- Imutils: For image processing helper function
- Face-recognition - : To recognize and manipulate faces
- Python-dotenv: To manage environment variables
- Dlib: to run face recognition package

Key Steps

1. **Use Git to clone the necessary packets to run Python packages**
   a. git clone https://github.com/loopDelicious/facial-recognition.git

2. **Download Dlib to allow Python Packages to run**
   a. **Check for updates and download programs for faster download speeds**
   ```
   $ sudo apt-get update
   $ sudo apt-get install build-essential cmake
   $ sudo apt-get install libopenblas-dev liblapack-dev libatlas-base-dev
   $ sudo apt-get install libx11-dev libgtk-3-dev
   ```

   b. **Create a Python environment and install numpy and dlib**
   ```
   Source venv/bin/activate
   $ pip install numpy
   $ pip install dlib
   ```

3. **Set up Python Virtual Environment and download required packages**
   a. **Set up Python environment**
   ```
   Source venv/bin/activate
   ```
   b. **Change into Folder “Facial-recognition”**
   ```
   cd Facial-recognition
   ```
   c. **Install python packages**
   ```
   pip install opencv-python imutils face-recognition sendgrid python-dotenv
   ```

4. **Make a folder under the dataset with the desired name**
   Add faces with different angles Axel/Roge
5. **Create a custom face recognition dataset**
   a. **Create Python Environment**
      Source venv/bin/activate
   b. **Run the Script and press space to take pictures**
      python headshots.py <Input Your Folder Name>
   c. Puts all pictures into your folder

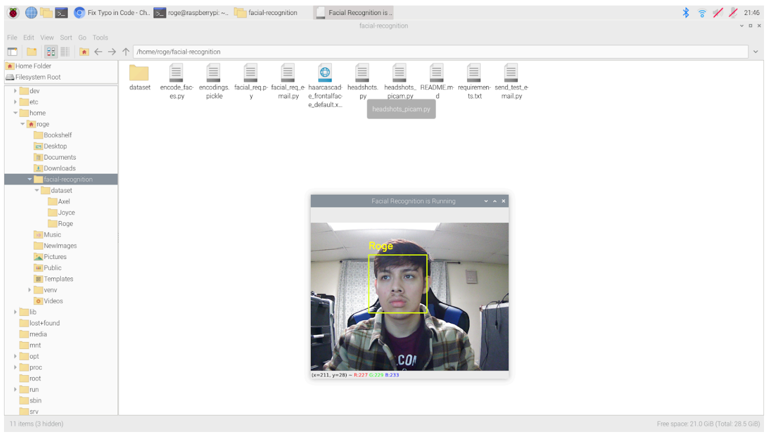
6. **Encode the faces using a deep learning-base model**
   a. python encode_faces.py
   b. Train the model by using more than 10 pictures

7. **Test Facial Recognition model**
   a. (venv) $ python facial_req.py
   b. Hit Q to quit

8. **Set up SendGrid email notifications**
   a. Open Text editor and input
      i. SENDGRID_API_KEY=<your-sendgrid-api-key>
      ii. SENDGRID_EMAIL=<your-verified-sender-email>
      iii. RECIPIENT_EMAIL=<your-recipient>
   b. Save the File as .env under the facial-recognition folder

9. **Add Email Notifications to face recognition**
   a. **Run script to run webcam and test email system**
      python facial_req_email.py

Conclusion
This tutorial provides a hands-on approach to building a face recognition system with practical applications. The integration of OpenCV for face recognition and SendGrid for email alerts creates a powerful system capable of enhancing security and providing real-time notifications. The combination of Python, OpenCV, and SendGrid into a Raspberry Pi exemplifies the seamless integration of technology to address real-world scenarios, making it a valuable solution for intrusion detection.


Fix Typo in Code - Ch..
roge@raspberrypi: ~..
facial-recognition
Facial Recognition is ..
21:46
File Edit View Sort Go Tools
/home/roge/facial-recognition
Home Folder
Filesystem Root
dev
etc
home
roge
Bookshelf
Desktop
Documents
Downloads
facial-recognition
dataset
Axel
Joyce
Roge
Music
dataset
encode_faces.py
encodings.pickle
facial_req.py
facial_req_email.py
haarcascade_frontalface_default.x...
headshots.py
headshots_picam.py
README.md
requirements.txt
send_test_email.py
Facial Recognition is Running
Roge