Face Recognition

ISM 6251 – Data Science Programming – Project Paper

1. Ananya Shrivastava
2. Devansh Guddeti
3. Sai Srujan Reddy
4. Chandana Mushkam
5. Sharvari Bacha

Table of Contents

1. Introduction

The aim of face recognition is to recognize a previously detected object as a known or unknown face. The problems of face recognition and face detection are frequently conflated. Face Recognition, on the other hand, is the process of determining if a "face" belongs to a known or unknown person by using a database of faces to validate the input face.

Face detection entails dividing image windows into two classes: one that contains faces (tarnishing the background) It's challenging because, while there are some similarities amongst faces, they might differ significantly in terms of age, skin tone, and facial expression. Different lighting conditions, picture quality, and geometries, as well as the possibility of partial occlusion and disguise, exacerbate the situation. As a result, an ideal face detector would be able to detect the presence of any face on any background, in any lighting condition. The task of detecting faces can be divided into two steps.

The first step is a classification task, which accepts any image as input and returns a binary result of yes or no, indicating whether the image contains any faces. The second step is the face localization task, which aims to take an image as input and output the location of any face or faces within that image as some bounding box with (x, y, width, height).

2. Prior Work

A very good real-life example for face recognition is the first implementation of Iris scanner by Samsung. The came the Face-ID feature of Apple iPhones. This feature has seen a wide acceptance from the customers in the mobile market. Over the time this feature developed a recognition to be a gold standard for security. This feature is also being used for payments and other security features.

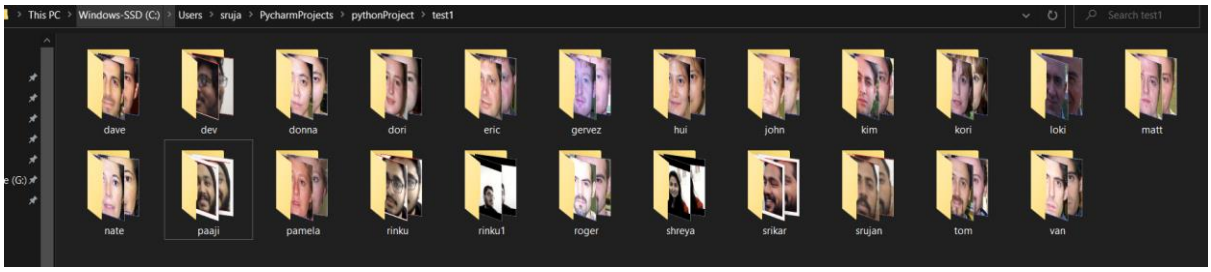
There are quite a few works done on Face Recognition, but each have their own methodology of dataset gathering, training, testing and validation. A few examples of works done include works based on Geometry and Template, based on appearance and model, statistical methods and neural networks. Prior works have been done using a wide variety of libraries like Keras, Face Recognition and several other kernel-based methods.

3. Dataset Overview

We started our work exploring about Face recognition using OpenCV. There were a lot of methods to approach this. The method that we followed was to create the dataset directly

by capturing our images from the webcam. But the background captured in the image was not letting us predict the face correctly. Hence, we added a feature to crop the face using cascade classifier in the dataset.

The dataset used for this project can be created as per the requirement. So if we want to predict the faces of say 10 persons, we run our code for them which opens up the camera and takes around 150 images of that person in various angles, crops the image with just the face of the person and saves in a folder (with the username given by us) in the working directory. So, the dataset looks like:



We can see the above screenshot with images saved in folders with the person's name. We use this directory for training the model and later predicting the testing image.

4. Data cleaning and pre-processing

- In our project we have created our own database using OpenCV and video capture function in OpenCV which takes 100 images in 10 seconds and stores in our local database.
- Then we have implemented cascade classifier which crops out only the face from the entire image and saves it in our local directory.
- All these cropped images obtained in our local directory are being used to train our model and predict the name of the user.

5. Challenges

- One challenge we faced at the start was with the dataset. Earlier we were working on a small dataset based on our captured images. We collected 20 images from 10 persons. Here, the prediction of the model was not so good. Later when we increased the size of the dataset (300+ images), we could get a better prediction.
- Another challenge we saw while testing, the face was not predicted correctly, because we were capturing the face along with the body of person and background. Here we

discovered that by cropping the face out we can avoid the background noise and when we used cascade classifier, we were successful in achieving a better prediction.

- One other challenge we faced was with implementing the cascade classifier. We couldn't implement the code at the first go as it took a few minutes for each image. On optimizing it further we could achieve cropping out the faces out of the dataset quicker.

6. Models

a) Classification Model

Object Detection using **Haar** feature-based **cascade** classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It's a machine-learning approach in which a cascade function is trained using a large number of positive and negative images. After that, it's used to detect objects in other images. To train the classifier, the algorithm requires a large number of positive images (images of faces) and negative images (images without faces). Then we need to extract features from it. It is still used with **OpenCV** .One of the primary benefits of Haar cascades is that they are just so fast — it's hard to beat their speed. The disadvantage of Haar cascades is that they are prone to false-positive detections, require parameter tuning when used for inference/detection, and are simply not as accurate as today's more "modern" algorithms.

b) CNN Deep Learning Model

A Convolutional neural network (CNN) is a neural network that has one or more convolutional layers that are primarily utilized for image processing, classification, segmentation and also for other auto correlated data.

In CNN deep learning model, each input image will pass through a sequence of convolution layers with filters (Kernals), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1.

Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data.

When the images are too huge, the pooling layers section would lower the number of parameters. Spatial pooling can be of different types:

- Max Pooling
- Average Pooling
- Sum Pooling

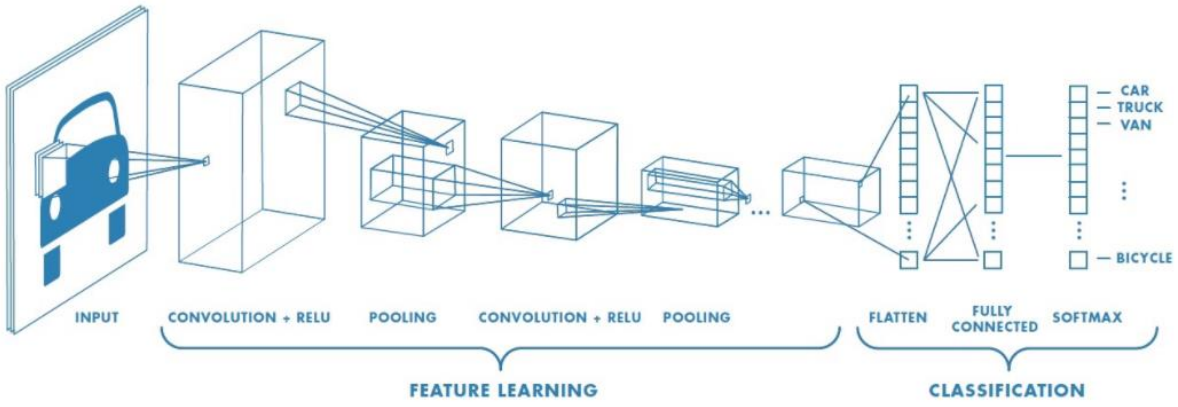


Figure 2 : Neural network with many convolutional layers

In our model, we do a categorical classification. We create a look up table and map each face with an ID. Later this is used to predict the test image with this ID and show the name of the person by mapping the ID. The model will recognize the ID and mapping will help us to display the corresponding name to that ID.

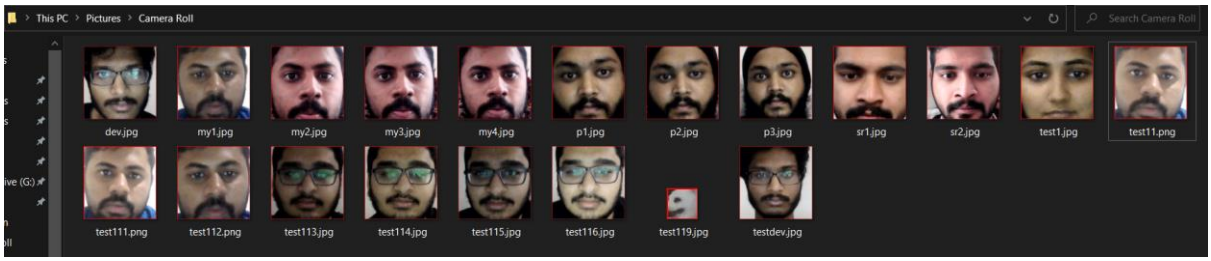
```
Mapping of Face and its ID

 {0: 'dave', 1: 'dev', 2: 'donna', 3: 'dori', 4: 'eric', 5: 'gervez', 6: 'hui', 7: 'john', 8: 'kim', 9: 'kori', 10: 'loki', 11:
'matt', 12: 'nate', 13: 'paaji', 14: 'pamela', 15: 'rinku', 16: 'rinku1', 17: 'roger', 18: 'shreya', 19: 'srikar', 20: 'sruja
n', 21: 'tom', 22: 'van'}

 Number of Output Neurons:  23
```

Then we trained our model with a Sequential Multi-Layer CNN Deep Learning Model with ‘relu’ and ‘softmax’ activation. We are using a 3 matrix of size (64X64) pixels representing red, green and blue components of pixels.

Later we move to the prediction. Here we do a prediction of the image by passing it through an image path. Then the cascade classifier crops the face out of this image, and this is used for predicting the face from the dataset. A sample of test images is shown below:



Here, the model predicts the face by matching the number tag identified with the name mapped to that tag and displays that name in the output.

7. Conclusion and Future work

In this experimental investigation, the system was evaluated under extremely rigorous conditions, and it is expected that real-world performance will be significantly more precise. The fully automated face detection and recognition system lacked the robustness required to achieve high recognition accuracy. However, if some sort of further processing, such as an eye detection technique, was implemented to further normalise the segmented face image, performance will increase to levels comparable to the manual face detection and recognition system. Implementing an eye detection technique would be a minor extension to the implemented system and may not require a huge deal of additional investigation. There are better techniques such as iris or retina recognition and face recognition using the thermal spectrum for user access and user verification applications since these need a very high degree of accuracy. The implemented fully automated face detection and recognition system (with an eye detection system) can be further used for simple surveillance applications such as ATM user security.

In future, we will extend this project in a way that the names would be visible if we show a face to the camera. Also, we are thinking to add the iris/retina recognition to understand and dig deeper into Neural Networks.

8. Acknowledgement

We would like to express our sincere gratitude to our Professor Dr Balaji Padmanabhan for being so supportive and helpful. His concepts taught in the class has only enabled us to complete this project successfully. He explained the Data science topics so beautifully that they are captured and fixed in our memory. In the coming future, we would be privileged to extend this project and contribute more to Machine Learning and Artificial Intelligence. We would also like to thank Arindam Ray for his amazing insights on our project. As a team, we are grateful to each member of the team and our peers who directly or indirectly helped us in the successful completion of this project.

9. References

- https://bhashkarkunal.medium.com/face-recognition-real-time-webcam-face-recognition-system-using-deep-learning-algorithm-and-98cf8254def7
- https://machinelearningmastery.com/how-to-perform-face-recognition-with-vggface2-convolutional-neural-network-in-keras/
- https://link.springer.com/chapter/10.1007/978-981-15-6707-0_48
- https://towardsdatascience.com/facial-recognition-using-deep-learning-a74e9059a150
- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6068932/
- https://www.tandfonline.com/doi/full/10.1080/21642583.2020.1836526
- https://thinkingneuron.com/face-recognition-using-deep-learning-cnn-in-python/
- https://machinelearningmastery.com/how-to-perform-face-detection-with-classical-and-deep-learning-methods-in-python-with-keras/
- https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/
- https://www.codeproject.com/Articles/5275261/Custom-AI-Face-Recognition-With-Keras-and-CNN
- https://www.mygreatlearning.com/blog/face-recognition/
- https://www.pyimagesearch.com/2021/04/19/face-detection-with-dlib-hog-and-cnn/
- https://towardsdatascience.com/drowsiness-detection-using-convolutional-neural-networks-face-recognition-and-tensorflow-56cdfc8315ad
- https://towardsdatascience.com/how-to-create-real-time-face-detector-ff0e1f81925f