Project : Face Mask Detection.

Visual Recognition : Mini Project

Hrithik Sharma (IMT2018029) hrithik.sharma@iiitb.ac.in
Prajwal Agarwal (IMT2018056) prajwal.agarwal@iiitb.ac.in
Sahaj Vaghasiya (IMT2018060) sahaj.kuman@iiitb.ac.in

1 Abstract

In this project, we experimented with a real world dataset given, and tried to explore how machine learning algorithms, typically person and face detection algorithms, can be used to find the features to detect face and face masks. We were expected to gain experience using a deep learning library keras. We implemented our own model to classify whether a person is wearing a mask or not, using Convolutional Neural Networks. We were expected to submit a report about the approach, methods we tried to implement and the algorithms used. This is the final report of our team for this project.

2 Introduction

Image recognition is the ability of software to identify objects, places, people, writing and actions in images. Computers can use machine vision technologies in combination with a camera and deep learning algorithms to achieve image recognition. Image recognition is used to perform a large number of machine-based visual tasks, such as labeling the content of images with meta-tags, performing image content search and guiding autonomous robots, self-driving cars and accident avoidance systems. One of the most important use case is person and face detection using CNN. Neural networks are really good in performing image classification. Models like AlexNet can classify objects in thousand classes with 99% accuracy. This project mainly deals with human face detection. We have tried to detect whether a person wears a mask or not in real time. We have used several deep learning algorithms like Viola jones algorithm for face detection and YOLO for person detection. This is followed by a classifier model build using multiple layers of convolutional neural networks.

3 Problem Statement

We are given a dataset consisting of images. Our goal is to detect person in the image given, detect the face of the person and detect whether the person is wearing a face mask or not. This has to be done using Neural networks in Python. We have to use the deep learning algorithms YOLO for person detection and Viola Jones for face detection. Then we need to build our own model using multiple layers of convolutional neural networks to classify images into binary classes.

4 About the Dataset

Dataset 1 (Provided with problem statement)
Dataset Size : 1500 images
Description : Images of people either wearing a white mask or without masks.
Number of Classes : 2 (People wearing a mask and People not wearing a mask)

Dataset 2 (Face Mask 12K Image Dataset)
Dataset Size : 12,000 images
Description : This dataset is available on Kaggle for training models. This dataset has images with people with or without masks. In this dataset, people are wearing coloured masks with some design as well. This dataset is used to generalize our model.

5 System Modules

1) **Human Detection (Using YOLO)**
YOLO Algorithm :
YOLO is a convolutional neural network (CNN) for doing object detection in real-time. The algorithm applies a single neural network to the full image, and then divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. YOLO learns generalizable representations of objects so that when trained on natural images and tested on artwork, the algorithm outperforms other top detection methods.

Implementation :
We have used the weights of pretrained YOLO model. These weights are used by DNN package from OpenCV library to detect human in the input image, and then outputs the coordinates of bounding box outlining all the humans in the image.

Real Time Accuracy :
Total Frames : 256
Total frames with correct face detection : 256
Accuracy : 100%

2) **Face detection (Using Viola Jones) :**
Viola Jones Algorithm :
This algorithm is an object recognition framework that allows detection of image features in real time. This algorithms using sliding window, i.e, a bounding box that is placed on the original gray-scaled image. This algorithm then tried to recognize the facial features(haar like features) in the bounding box. It has basically four major steps : Haar feature selection, Integral Image Creation, Adaboost Training, Cascading classifiers.

Implementation :
To implement face detection, we use a cascade classifier method present in the OpenCV package which takes in the haarcascade_frontalface_default.xml file, which is viola jones classifier. This method returns a face cascading object. For a given input image this classifier finds facial features and then recognizes all the human faces, and outputs the coordinates of the bounding box outlining all the faces.

Real Time Accuracy :
While real time testing, each frame contained only one face.
Total Frames : 256

Total frames with correct face detection : 198
Accuracy : 72.31%

Besides this we also used another method to detect face using DNN Frontal Face Detector in OpenCV.
DNN Frontal Face Detector :
It is a Caffe model which is based on the Single Shot-Multibox Detector (SSD) and uses ResNet-10 architecture as its backbone. It was introduced post OpenCV 3.3 in its deep neural network module. This detectors works very well in real time. It works well with occlusion, quick head movements, and can identify side faces as well. Moreover, it also gives the highest frame rate per second.

Implementation :
To implement face detection, we first load the DNN model's layers and weights using readNetFromCaffe method of the DNN module from OpenCV package. Then we run the network on the input image to detect faces and get coordinates of the bounding box outlinging all the faces in the image.

Real Time accuracy :
While real time testing, each frame contained only one face.
Total Frames : 256
Total frames with correct face detection : 256
Accuracy : 100%

3) **Mask Detection :**

Implementation of Classifier :

a) We have used 5 CNN layers and 2 Fully Connected layers for our classifier with batch normalization.
b) For all the CNN layers we have used "ReLU" as the activation function and have used max pooling after every two CNN layers.
c) For the last FCC layers, we have used softmax activation function to get the probabilities of the binary classes.

6 Problems and Issues

Issues with Dataset :
All the masks in the dataset have same color , i.e., white with almost no design. Most often masks are available in different colors, shapes as well as with some printed design. The dataset was not at all diverse to provide a variety of test cases needed for training.
Solution : We used another dataset of about 12,000 images from Kaggle. This dataset was very diverse and helped in increasing our accuracy of our model alot.

Issues with Face Detection :
Face detection is done using Viola Jones algorithm. Although this algorithm is very strong, but still this algorithm was not able to detect faces when their was a little movement in the frame or lighting was a bit less. This was affecting the accuracy of the whole algorithm. Since goal of our model is to detect whether a person is wearing a mask or not, it becomes difficult for a standard face detection algorithm to detect a face with mask, since the mask hides some important facial features.
Solution : We used another face detector DNN from OpenCV package. This algorithm is far more accurate than Viola Jones algorithm.

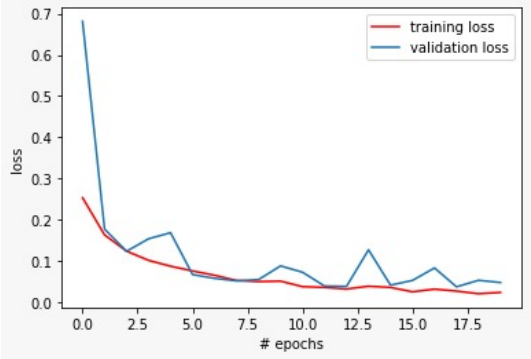
7 Result

Classifier Model Accuracy : 99.03 %

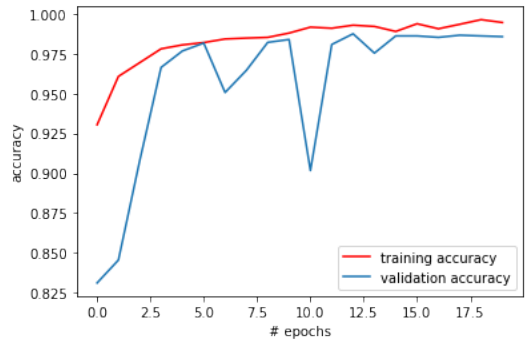
Real time accuracy (Based on submitted Demo Videos) :

1) **Person Detection Accuracy :** 100%
2) **Face Detection Accuracy :** 100%.
3) **Classification Accuracy :** 100%

Below graph represents the training loss when we trained the model on whole datasets : dataset1, dataset2.



Below graph represents the training accuracy when the model was trained and validated on the whole dataset.



Notes :

1) The real time accuracy might be different based on the environment such as lighting, face angle, sudden movements etc.
2) To view the output open the FaceClassifier.ipynb file in jupyter notebook in the local system.

Instructions to Run :
Keep all the files in same directory.

1) To create the classifier, run the FaceClassifier.ipynb file in jupyter notebook. This will create a model FaceClassifier.hdf5.
2) Download the YOLO config files and weights.

3) Download the weight file from : Link
4) To download config file run $ git clone https://github.com/pjreddie/darknet
5) Download the given file from github : Link
6) Download this file from from github : Link
7) For live detection, run ViolaJones.py on terminal using $ python3 ViolaJones.py or $ python3 DnnDetector.py

8 Conclusion

In this problem, we have tried to solve the problem in bottom up fashion. We have tried to generalize the model using additional datasets, and tried to use improve the face detection using additional methods available in OpenCV package. Overall this project proved to be a good learning task for us. We learnt how algorithms like YOLO and Viola Jones work in detail. We learnt how deep learning is used in detecting humans and human faces. This project can be further developed. Currently this project requires that the image has complete frontal face, but this can be improved for side face also.

References

1) Wikipedia Viola Jones
2) DNN OpenCV
3) YOLO Medium Article
4) Dataset
5) Understanding Viola Jones Algorithm