EasyChair Preprint

№ 7441

Smart Attendance System Using Deep Learning

Atharva Amrapurkar, Pratik Parbat and Vandana Jagtap

February 8, 2022

Smart Attendance System using Deep Learning

Atharva Amrapurkar[1], Pratik Parbat[2], Vandana Jagtap[3]

#MIT World Peace University School of Mechanical Engineering, MIT World Peace University Pune India
[1]heyatharva9@gmail.com, [3]vandana.jagtap@mitwpu.edu.in, [2]prateekparbat@gmail.com

***Abstract*: Engagement is the key to the success of the intelligent user interface.Such interface requires to respond appropriately and also needs to recognize the level of engagement. This paper presents a Deep Learning model to improve face recognition by implementing and comparing various methods used for robust face detections where the subject would give attention to the interface for a few seconds or the subject does not have the attention of the interface or just passes by the interface to register an engagement. By combining some powerful deep learning tools like *Convolution Neural Network* with techniques like *Histogram of oriented Gradients*, we can outform the older techniques and with more accuracy and with an computational efficiency ,the model is trained by CNN inorder to precisely measure the embedding of an face (*128 measurement of each face*),the network was trained by Deep Learning and using any simple classifier and find the closest match with the measurements received from the model with the images present in the database.The end result will be the name of the student being recorded for the respective use.**

Keywords—Face Recognition , Deep learning , CNN and HOG

I. Introduction

The detection of human faces has been a difficult task since it was devised as an idea to collect information and perceive data in early 2000, where in just the detection of the feature set has been hard with the best and lit background. The next advancement was the develop robust feature sets that able to discriminate the human form of face feature sets cleanly and distinctly from each other irrespective of dark night or totally illuminated day time background, we are going to use Deep Learning and some powerful face feature sets which outperform other existing feature sets which which uses the incorporation of wavelets, where in we show that locally normalized *Histogram of Oriented Gradient (HOG)*[5][3] descriptors provide excellent performance with much more affordable computation required. The end result would be a name that would be recorded in the database for the respective purpose after matching a face in the database with a name associated with it. The purpose for the interface is to quickly and accurately distinguish between faces wherein the subject would be illuminated with different amounts of light and also with very different types of background, wherein he or she would interact with the system for a couple of seconds only. This gives rise to a need of a system which would detect and discriminate the face in just milliseconds while still being very accurate and precise.

II. Literature Review

A. *Automatic Recognition of Student Engagement using Deep learning and Facial Expression (Omid Mohamad Nezami, Mark Dras, Len Hamey, Deborah Richards, Stephen Wan, and Cecile Paris ´)*

This paper deals with engagement during learning via technology. Investigating engagement is vital for designing intelligent educational interfaces in different learning settings including educational games, massively open online courses *(MOOCs)[15]*, and intelligent tutoring systems (ITSs).

B. *Research on facial expression recognition based on Multimodal data fusion and neural network (Han Yi1,2 Wang Xubin2 Lu Zhengyu2*)*

In this paper, a neural network algorithm of facial expression recognition based on multimodal data fusion is proposed.

C. *Histograms of Oriented Gradients for Human Detection (Navneet Dalal and Bill Triggs)*

This paper studies the question of feature sets for robust visual object recognition, adopting linear SVM based human detection as a test case.

III. Proposed Methodology

We, as humans, are binded to recognize faces automatically and instantly. In fact, humans are great at recognizing faces and might end up seeing faces in everyday objects but computers are not capable of this kind of high level recognition. A pipeline is built here in order to solve steps of face recognition.

A. *Finding all the faces*

Face detection is the first step in the pipeline. From the last 10 years, face detection has been observed in almost every *android* or *IOS* [16] device. The perfect and most reliable method for face recognition will be *Histogram of Oriented Gradients*[5][3].

- First step would be to convert the image into black and white so that we cannot include color data to find faces .
- Then the algorithms will look at each single pixel in the image one by one. For each pixel it looks at the pixels that surround it directly.
- The major goal is to figure out how dark the current pixel is compared to the pixel that

surrounds it. Then it will draw an arrow showing in which direction the image is getting darker. The arrows mentioned here are called gradients and it shows the flow from light to dark across the image

- To lose up the task it is recommended to break up the image into small squares of 16 x 16 pixels each. With these images being made we will count up how many gradients points in each major direction
- In order to find faces, we need to find the part of the image that looks the most similar to the HOG pattern that was extracted from a bunch of other training faces.

B. *Posing and Projecting Faces*

Detecting faces from still images will be done with this model but our next problem would be detecting the face which is turned in different directions. Faces turned into different directions look totally different for a computer

- To encounter this issue we need to try to wrap each picture so that the eyes and lips are always in the sample place in the image.
- **Face landmark estimation** is used to encounter this. The approach used here was invented by *Vahid Kazemi and Josephine*[1].
- The algorithm comes up with 68 specific points that exist on every face, a machine learning algorithm will be trained to be able to find these 68 specific points on any face.
- After getting to know about the positions on every face, we rotate, scale and shear the image in order to get the eyes and mouth as centered as possible. A basic image transformation like rotation or scale is done that preserves the parallel lines; this method is known as *affine transformations*[2].

C. *Encoding Faces*

The easiest way to recognize a face would be to directly compare the unknown face we found with all the pictures in the database, but there's a huge problem with that approach. It is really difficult and time consuming for a model to loop through all the photos in the database.

- To simplify things, it is needed to extract a few basic measurements from each face which will help to measure our unknown face the same way we find the known face with closest measurements.
- Measuring things seems obvious to humans but it doesn't make sense to machines which look at individual pixels in an image.
- Deep learning plays really important role in figuring out which parts of faces *are important to measure than humans*[3]
- The algorithm finds the measurements and then tweaks the neural network slightly so that it makes sure the measurements generated are slightly closer.
- Repeating this step millions of times for images of many people, the neural network learns to generate 128 measurements of each person.
- The *term embedding*[4] is used in machine learning which *reduces the complicated raw data like pictures into list of numbers.*[5]
- Encoding the image involves training of a convolutional neural network which requires a lot of power and expensive video cards like the *Nvidia Tesla*[6][13] which takes about 24 hours of continuous training which is done with the help of *OpenFace*[7]. They have published several *trained networks*[8] which can be directly used.

D. *Finding the person's name from the encoding*

- This method comprises the process of finding the person from our database of known people who has closer measurements to the given image.
- The simple machine learning classification algorithm *SVM classifier*[9] is used in order to get it done.
- The classifier is trained in a way that it can take the measurements from a new image and tell which person in the database is the closest match. The returned result will be the name of the person.

E. Adding the record into .csv file

- Whenever the model detects the new face the os library included in the program helps it to add the record into the predefined .csv file.
- The time library included in the program helps it to track the exact system time when it detects the new image.

IV. RESULTS AND DISCUSSIONS

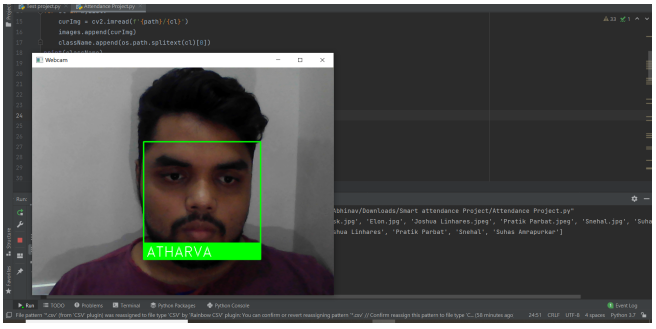

Figure 1 : The image of the face

The face is recognized only with a single test image in the database, and the image is named with the name saved for sample test image.

The program is made in such a way that it shows the Names of images in the database in list format and after encoding

the test images tells the interpreter that it is done with the encoding. The program captures the images from the camera, reduces its size to ¼ th the size of the actual image and sends it to the algorithm. The HOG algorithm then Finds the small 16 X 16 pixels from the image and draws the gradients, through which the face is detected. The Face landmark estimation then helps the rotating face image to make it in the way that it will follow affine transformations which will help the program to encode the Face. The predefined API of Openface then runs the CNN algorithm to encode the image which finds the 128 projections on the face and compares the locations with the locations of the images from the database, hence the image with more matches is considered as the comparing image and hence the face gets recognized. Once the face gets recognized with the image from the database, the program will use the name of the image from the database which will help the system to know which person is currently in front of the camera. After successfully completing this step the program will use *Python OS Library* to write the attendance record into a predefined *CSV* file and use the *Python Time Library* to add the system time when it captures the image.

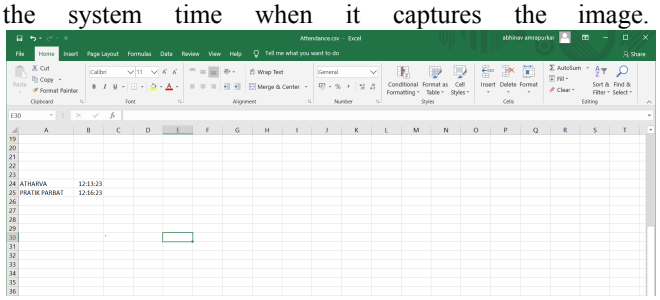

Figure 3: Record for the faces detected as attendance.

Comparative Analysis

Comparing Dlib HOG with Dlib CNN we can see how HOG calls 1st call loads fastly and as fast as CNN when the images are small, while it is observed that it runs 2.7 times slower than CNN and is not accurate enough. CNN turned out to be more accurate, robust and stable and can successfully handle multiple executions.

Conclusion

There are many face features which can be implemented with various feature sets but they come at computational cost. HOG is the most robust technique with minimum computation cost and a very reliable way to face detection whereas the result has to be calculated in milliseconds. The network trained by Deep Learning where it becomes easier to find a match in a large database where in the database would have hundreds of faces , even with some similar faces where other counterparts would detect them as the same faces. The system would be easy to implement and would work in any harsh environment where the other existing techniques wouldn't even detect a face in such an environment . The linear SVM classifier is able to classify the image within a second but there is plenty of room for optimization. The CNN network can be trained more rigorously in order to differentiate smallest differences in the faces. This can be implemented to record attendance where in the system doesn't need an active attention once configured.

References

[1] *Automatic Recognition of Student Engagement using Deep learning and Facial Expression (Omid Mohamad Nezami 1 , 2 ( ), Mark Dras 1 , Len Hamey 1 , Deborah Richards 1 Stephen Wan 2 , and Cecile Paris ′ 2)*

[2] *Research on facial expression recognition based on Multimodal data fusion and neural network (Han Yi1,2 Wang Xubin2 Lu Zhengyu2*)*

[3] Histograms of Oriented Gradients for Human Detection (Navneet *Dalal and Bill Triggs)*

[4] *Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning by Adam Geitgey*

[5] Face detection with dlib (HOG and CNN) by Adrian Rosebrock

[6] OpenFace Free and open source face recognition with deep neural networks.

[7] Viola–Jones object detection framework

[8] see the forest for the trees by Wikidictionary

[9] One Millisecond Face Alignment with an Ensemble of Regression Trees (Vahid Kazemi and Josephine Sullivan KTH, Royal Institute of Technology Computer Vision and Active Perception Lab Teknikringen 14, Stockholm, Sweden)

[10] Affine transformation wikipedia

[11] Machine Learning is Fun! Part 3: Deep Learning and Convolutional Neural Networks by Adam Geitgey

[12] FaceNet: A Unified Embedding for Face Recognition and Clustering by Florian Schroff, Dmitry Kalenichenko and James Philbin

[13] High performance Computing by Nvidia

[14] Support Vector Machine Wikipedia

[15] Massive Open Online Courses (MOOCs) by Mooc.org

[16] IOS by Apple