Pulkit Agarwal
October 3, 2021

# Facial Detection in Amazon Ring

## Acknowledgements

I would like to thank Ekin Gurgen, Masters student at ETH Zurich, for mentoring me throughout this project.

## I. Motivation

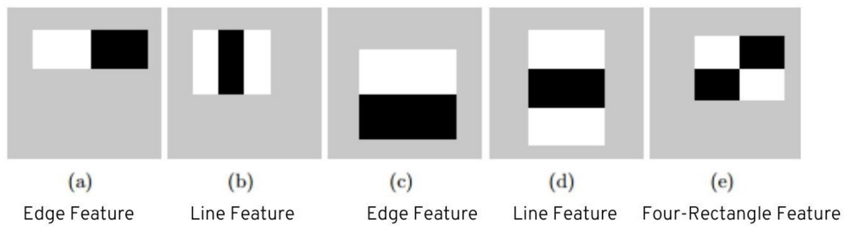
As new technological breakthroughs are made everyday, our lives are becoming more convenient. With current features such as speech recognition, our lives are becoming more hands-free. A good example of this is the Amazon RING. From anywhere, RING customers can see who is at their door and even talk to them through the RING app. While this makes RING customers' lives significantly easier, and not to mention safer, there is a large space for improvements to this system. In order to make our doorbells more capable, I looked towards implementing a facial recognition system in the Amazon RING.

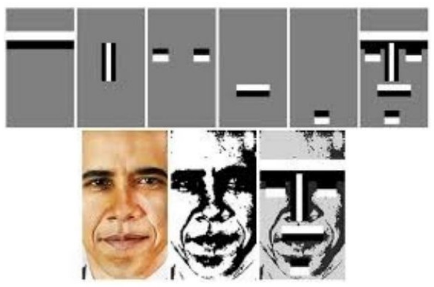
## II. Background

### Template Matching Methods

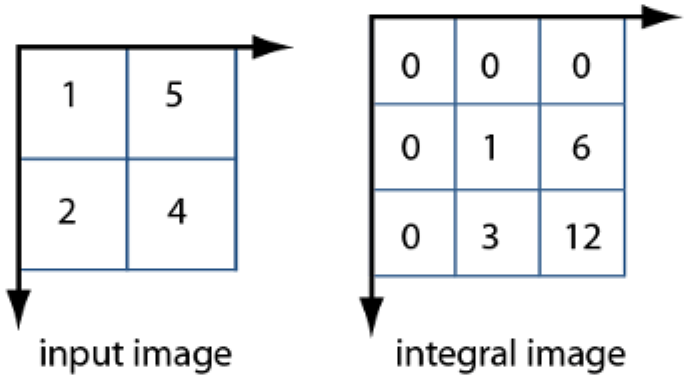
The Viola Jones Face Detection method is a robust face detection that can perform face detection in real time. The Viola Jones Face Detection algorithm is a version of a supervised learning method, which means it must be trained before it can accurately detect objects, or in this case faces. Although the Viola Jones method is only capable of performing face detection, it is still relevant to this project because face detection can be seen as the first step to facial recognition. The method is composed of 4 parts: Haar features, integral images, Adaboost, and cascade classifiers. In order to detect faces, the Viola Jones method uses Haar features.

Haar features are many common features that are in faces that can easily be detected, such as the bridge of the nose, the mouth, the eyes, etc. In order to detect these haar features in an image, the Viola Jones method uses edge detection, which scans the image for a known pattern of bright pixels next to dark pixels. The method starts with a 24 x 24 window to detect these features, and after scanning through the whole image, the method increases the window size and repeats the process until the window cannot be scaled anymore. When scanning through the image, the method takes the sum of the pixels under the white space - the sum of the pixels under the black place. If this results in a high value, the method knows the haar feature is found because it will only match if the pixels under the white region are light and the pixels under the black region are dark.  This method is called the sliding window method. Through this method of detecting haar features, the Viola Jones method can precisely detect many common features from a given face.



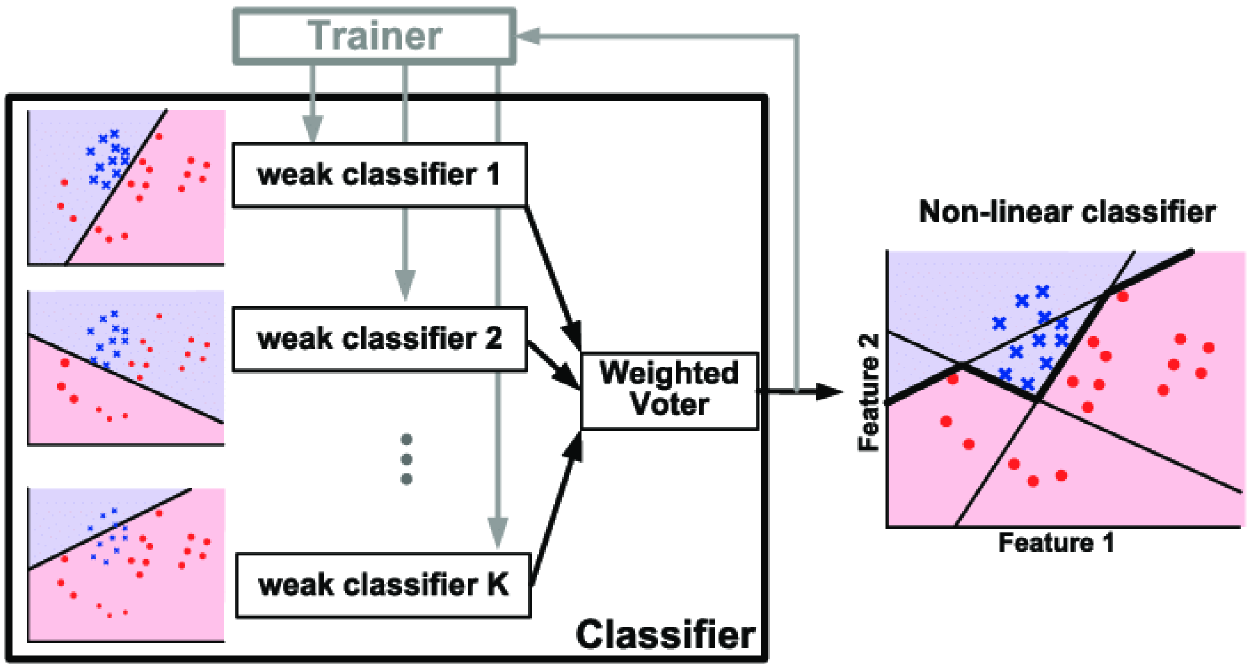


In order to robustly detect these haar features in an image, the Viola Jones method converts the image to an integral image. An integral image converts each pixel value to a value of the sum of all the pixels to the top and left of the current pixel.

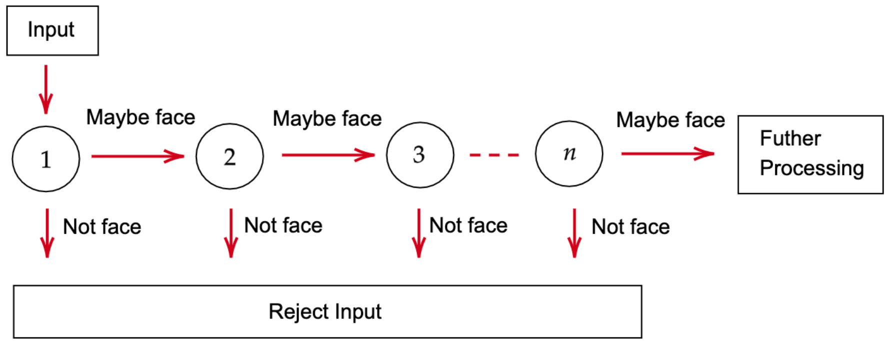


After converting an image to an integral image, haar features can be found very robustly. In order to find the value of any black or white region, instead of getting the sum of all the pixels, the value of the region can be found by subtracting the diagonal from the top right to the bottom left from the diagonal from the bottom left to the top right of the region.

In order to detect faces in real time, the Viola Jones method uses a machine learning algorithm called Adaboost. While there are over 160,000 haar features that can be detected in a face, it is given that all the features are not relevant. In order to create a strong classifier, a method that can accurately detect a face consistently, Adaboost creates a linear combination of weak classifiers attached to a given weight. A weak classifier is any haar feature that can correctly detect a face more than fifty percent of the time.



Lastly, in order to be robust, the Viola Jones method uses an idea called cascading. Essentially, cascading is using the Adaboost algorithm to quickly discard obvious negatives, photos without faces. By running an image through many weak classifiers, an image can be discarded immediately if the initial weak classifiers are not able to detect haar features in the image. This makes the Viola Jones method robust because instead of completely processing each image, it focuses on removing negatives initially and only processing possible positives.



## Neural Network Based Methods

Although template matching based methods are effective, they still result in many false positives (labeling a non-face as a face). Additionally, in template based matching methods, the angle of the face is very crucial, and any non-frontal facing faces will not be detected. Hence comes neural network based methods. Neural networks are a deep learning method used for many different machine learning problems. Neural networks are far more accurate than template matching based methods. How are neural networks so accurate? They use a combination of thousands of basic single function nodes. A neural network consists of many layers of nodes, where each node passes on a single value onto the next layer based on its input. Each node assigns a weight to it's input value, and then passes on the input value with the added weight to the next layer of nodes. At first, these nodes are assigned random weights. However, as the model is trained by giving it positive and negative values, the neural network is able to change the weights in order to start correctly labelling positives. Essentially, the neural network is able to recognize a pattern between all the faces, just like we humans do, and then use that pattern to identify faces in the future. Although this method may seem very simple, it is a very complex method that has many applications. For this project, a convolution neural network was used, a neural network which is commonly used for image and video processing.

# III. Development

## Tools

For this project, I used Python version 3.6.13. The framework I conducted experiments on were the Amazon RING doorbell, the YALE Door Lock, and the Amazon Alexa. In order to manipulate these frameworks, I had to use many unofficial API's. The API's used in this project are referenced below in the references section.

## Facial Recognition Model

While creating the facial detection model for this project, I encountered many road bumps. Initially, I tried to use OpenCV's built-in face detection system. Although the system was functional, it had many limitations. For starters, it resulted in many false positives. Considering the end goal of this project was to deal with unlocking a door, security was a top priority, and therefore false positives were something I aimed to minimize. Additionally, the model was only able to detect front facing faces. For this project, this was not ideal. Since the RING camera is located at the arm level in most houses, the camera would be viewing the person from a lower angle, which means this model would not be able to detect the faces as well if integrated into a ring camera. Because of this, I decided to try a deep learning model. While a deep learning model is very accurate, it also has some downsides. For starters, my macbook was not strong enough to run a deep learning algorithm, let alone train it. However, in the deep learning model, it was easier to go onto the next step: facial recognition. Since the depe learning model already took a "blob" from each frame, a region-of-interest where the model thinks there is a face, facial recognition could be done by comparing this "blob" to the user's database. Because of the combination of all these reasons, I decided to use a pre-trained deep learning model. This allowed me to take advantage of the accuracy of a deep learning model while avoiding the long time it takes to train a deep learning model.

## Integration

After creating the facial recognition system, it was time to integrate the model into the Amazon RING. A significant problem I faced during this project was finding official API's for the framework I wanted to manipulate. Since there were no official API's, I had to rely on unofficial API's. These unofficial API's often had many bugs, which took many hours to fix in my code. Additionally, since they were unofficial API's the methods they used to access the framework were very complicated. This resulted in many issues, like Amazon temporarily blacklisting my RING account because I accessed our RING too many times. However, despite these problems, after many hours, the project is able to conduct facial recognition in Amazon's RING Doorbell now.

### Limitations

Because of the reliance of unofficial API's there are many limitations to this project. Initially, the project was supposed to be able to run in real time, however, the unofficial API I used for the Amazon ring takes a minimum of 35 seconds to download videos from the RING doorbell. Despite numerous tries of reducing video quality or changing other settings, the API was not able to download the video faster. With the addition of having to conduct facial recognition on the frames after downloading the video, it came out to be able to produce results 37 seconds after the doorbell was rang. The main reason for this was due to the long process of downloading the video. In order for the video to be downloaded, the RING doorbell had to first upload it to the cloud, from which the API could then access it and download it to the laptop, from which finally facial recognition could be done. This limitation could easily be fixed with an official API and then the system would be able to run in real time (below 5 seconds). Additionally, there were some technological limitations. Because I am working on a macbook, I cannot run high tech facial recognition systems such as the one apple uses for their iphones. Therefore, there are still some false recognitions in the model. This seems to happen especially with South Asain faces. Considering the model was not trained with South Asain faces, this is expected. This limitation could be fixed by training a model with a more diverse training set. To go even further, a complex method like Apple's facial recognition method, a method that created 128-D encondings of faces, could be used to have utmost accuracy.

## IV. Conclusions

The main takeaway I had from this project was the necessity to weigh the costs and benefits for each method used in a program. I often faced situations where I had to choose sacrifice accuracy for efficiency or the other way around. Many times, when using an API, I had to weigh the different methods an API offered me in order to find the most robust method. When dealing with problems such as running in real time, accurate face detections and recognitions, and technologically possible methods, I had to make many tough choices when deciding how to write my program. Essentially, I learned that I had to make sacrifices even in coding because I simply could not have everything. This was a fairly new idea to me, because in the past I had never had to deal with making sacrifices in coding. Having worked with complex programs for the first time, I have started to understand the numerous decisions that go into coding. This has vastly changed my perspective on the devices I use everyday, and it has even made me appreciate the millions of complex functions I use everyday and expect to happen in a matter of milliseconds.

## V. Future Work

Currently, the program is able to recognize and announce a face on a laptop after 37 seconds of the doorbell being pressed. In order to increase the efficacy of this program, the main task would be to make it run in real time. This would require an official API, or some other method which is able to get videos straight from the RING doorbell without the unnecessary uploads. After this, the program could be implemented in the YALE Door Lock and the Amazon Alexa. This way, Alexa can announce when someone is at the door, including the person's name. Additionally, the YALE Door Lock could be manipulated to unlock once a VIP face is recognized. If all of this is completed. To go even further, a method that creates 128-d encondings could be added to the program, in order to maximize the program's efficiency. Obviously these improvements would take a lot of time, money, and high tech technology, but it would be a revolutionizing addition to home security, just like how Face-ID was for the Iphone.

## VI. References

1. RING Doorbell API: https://github.com/tchellomello/python-ring-doorbell
2. Viola Jones Method: https://www.mygreatlearning.com/blog/viola-jones-algorithm/
3. Neural Networks Explained: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414
4. Deep Learning Lecture: https://www.youtube.com/watch?v=FBggC-XVF4M