**Q&A**

**Requirement/Business Perspective**

1)    How did you arrive at this solution for the requirements?

Ans: We closely considered the requirements, experimented with the various techniques and evaluated the results. After some testing, we had ascertained that the current technique provided acceptable performance and we went with it.

2)      How will the business benefit from this system?

Ans: The demo asset provides invaluable value to the company and there is a need to protect it from tampering by unauthorized tampering and damage. Such consequence will lead to unnecessary downtime and lost opportunity to showcase it to prospects. This system provides the protection so as to avoid such downtime and loss.

3)      What are the main challenges you have faced in this project?

Ans: The main challenges is setting up the appropriate Anaconda environment to do the testing. Especially with Tensorflow library, some problems were encountered due mostly to version issues. Then there were issues with long training runs before we could make each round of tweaks to our algorithms.

4)      What are the skills you have learnt from your Masters course that you have applied to this project?

Ans: The main skill that we had learnt and applied are the Deep Learning/Machine Learning techniques.

**Intruder detection**

**(A) Intruder Detection by Background Subtraction: OpenCV**

Overview: Detect movement in webcam video through the Background Subtraction method. Any detected movement will be display in white against a black background.

Background Initialisation – compute an initial model of the background

Generate the foreground mask using the cv::BackgroundSubtractor object

Read the video stream images sequence using the cv::VideoCapture

Background Update – update model in order to adapt to possible changes in the scene

Each frame currently calculate the foreground mask and updating the background

Results of the updated video stream is displayed

Model = Gaussian mixture model

**(B) Intruders Alert: Twilio**

1.Communication API for SMS

2.It works in the background to bridge the gap between web-based applications and telephones

3.**Application-initiated event** - initiated via Twilio's public REST API

4.In our case, our python script send an HTTP request to Twilio’s API to send a text message from the application's phone number to a recipient phone number, with a message containing the alert

Which Concepts were Applied

**(A) Intruder Detection by Background Subtraction**

Application of Statistical Signal Classification by Expectation Maximisation

~ Pattern Recognition System, Dr Tian Jing, Statistical Signal Processing

Knowledge & Understanding - fundamentals of statistical signal processing, learning distribution from signal, and statistical classification

**(B) Intruders Alert: Twilio**

Application of Intelligent Software Agents (API)

~ Intuited from Cloud AI Workshop (Google Cloud), Sam Gu

Knowledge & Understanding - API: a software intermediary that allows two applications to talk to each other. In other words, an API is the messenger that delivers your request to the provider that you're requesting it from and then delivers the response back to you

Design & Built Considerations

1. Simple
2. Robust
3. Flexible

**(A) Intruder Detection by Background Subtraction**

Simple - only need OpenCV & Numpy package, about 20 lines of codes

Robust - minimal noises, unaffected by gradual illumination change, can account for shadow

Flexible - not required to perform prior mapping of "virtual" lines or detection areas with each deployment

**(B) Intruders Alert: Twilio**

Simple - API call to request service, only need twilio package, less than 10 lines of codes

Robust - twilio handles ever-changing telecom logic and carrier specific rules to ensure message reaches its destination

Flexible - can change the message content and recipient easily

**Face Recognition**

1. Face recognition speed?

0.2-0.6s based on CPU/GPU speed. (If further explanation required, most time is spent on face\_recognition, face\_cascade/MTCNN/classifier are using minimal time.) (face\_recognition may not be acting faster than facenet, however, its encoding performance is better than facenet.)

1. Accuracy and False Negative?

>99% accuracy with 0.4% false negative

1. Real time performance?

Largely accurate for person of interest. Lighting exposure on 1 side of face can cause false positive / false negative cases.

1. How many samples?

123 images. After Augmentation, ~100000 sample images.

1. How samples are augmented?
2. Change lighting by changing gamma to 0.8/1.2
3. Images are filtered using Laplacian of Gaussian
4. Images are filtered with Gaussian Blur
5. Images are zoomed in or zoomed out
6. Images are rotated right or left up to 15 degree
7. Images are flipped horizontally
8. How images are pre-processed?
9. Multiple faces are bounded using face cascade method
10. Each face captured is re-aligned using MTCNN
11. Each face is input to DLIB based face-recognition for face-encoding.
12. How the encoded images is being classified/trained?
13. Input to SVM/MLP/KNN/Logistic Regression
14. Ensembled using voting system (All 4 classifiers are with equal weight)
15. How long is the training?
16. Image pre-processing is using 6 hours for full sample set (3-5 minutes per image appended)
17. Classifier training costs 30 min
18. Does voting network helps in accuracy?

Yes. In certain case, one of the classifier may predict wrongly. With voting system, the accuracy improves. It is rare to have all classifiers predict wrongly.

1. Which model is better?

KNN/SVM are performing better. However, MLP/Logistic Regression can still covers images which are predicted wrongly on KNN/SVM.

1. Why use face cascade?

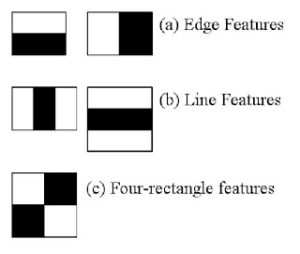
CV2 has face cascade implemented which can capture multiple faces with relatively high accuracy with faster speed.

1. What is Face cascade?

<https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html>

Haar Cascade has 4 stages:

1. Haar Feature Selection (Not Haar Wavelet)
2. Creating Integral Images (Allows multiple Haar features to be determined across the image)
3. Adaboost Training (Boosting method, ensemble on weaker convolutional classifier using Haar Feature as kernel. Pre-trained using positive and negative images)
4. Cascading Classifiers (Stages of Adaboost classfier. Determine the strongest image region than resemble the face)



1. Why using MTCNN?

Giving position of face landmark in fast speed. It can re-align the faces so the eyes are in horizontal axis. It reduces the rotation of images in the augmentation process. Hence, MTCNN alignment is used during both training and testing stage.

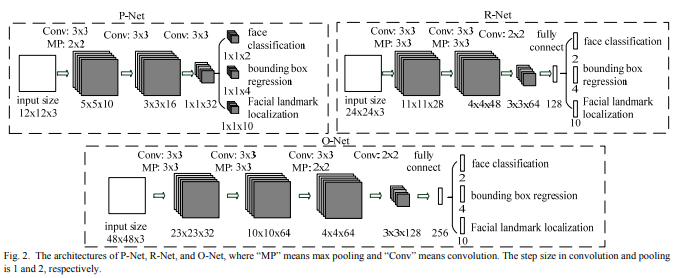
MTCNN realignment is faster than DLIB. Hence using MTCNN at image pre-processing.

1. What is MTCNN?

MTCNN or Multi-Task Cascaded Convolutional Neural Network is used for face landmark detection and face boundary detection.

MTCNN has 3 stages of convolutional neural networks namely, P-Net (Proposal), R-Net (Refine) and O-Net (Output)

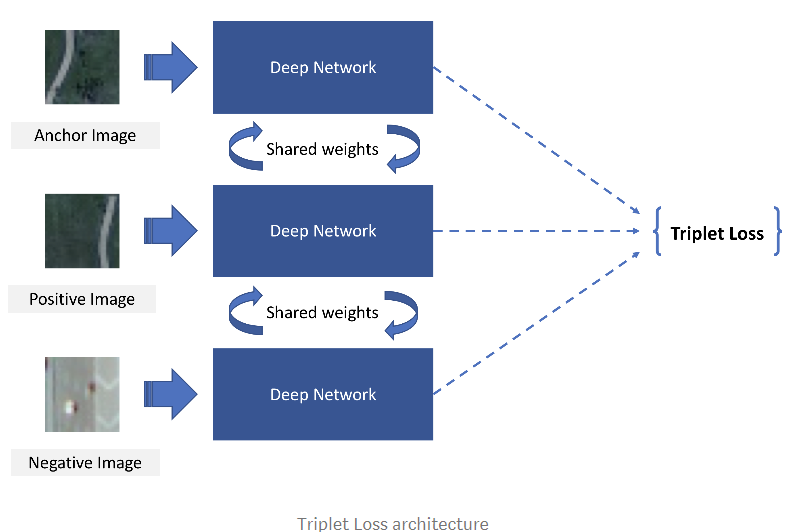
Best bounding coordinate and 5 face landmark coordinates are selected.



1. Why using face\_recognition library?

Face\_recognition is using triplet loss in the shared input network. With Positive image, target image and negative images are used to train the network, classification and encoding can be done with high accuracy. Its performance is higher than the facenet.

Moreover, 128-bit code is generated from each image, it is easier for SVM/KNN/MLP/Logistic regression training. Resnet which is using larger code, it is harder for classifier training.



1. What is face\_recognition?

It is a DLIB based library created by Ageitey. Face detection is done using HoG to extract the features while the encoding is done thru triplet loss pre-trained convolutional network.

