PES University
Cloud Computing and Big Data

Status Report 1: CCBD PROJECT

MENTOR : D.R.HL PHALACHANDRA

PRIYA MOHATA	PES2UG19CS301
PRIYANSH JAIN	PES2UG19CS303
ARPIT KOGTA	PES2UG19CS065
SNEHIL JAIN	PES2UG19CS396

Table of contents

PESU CCBD

01

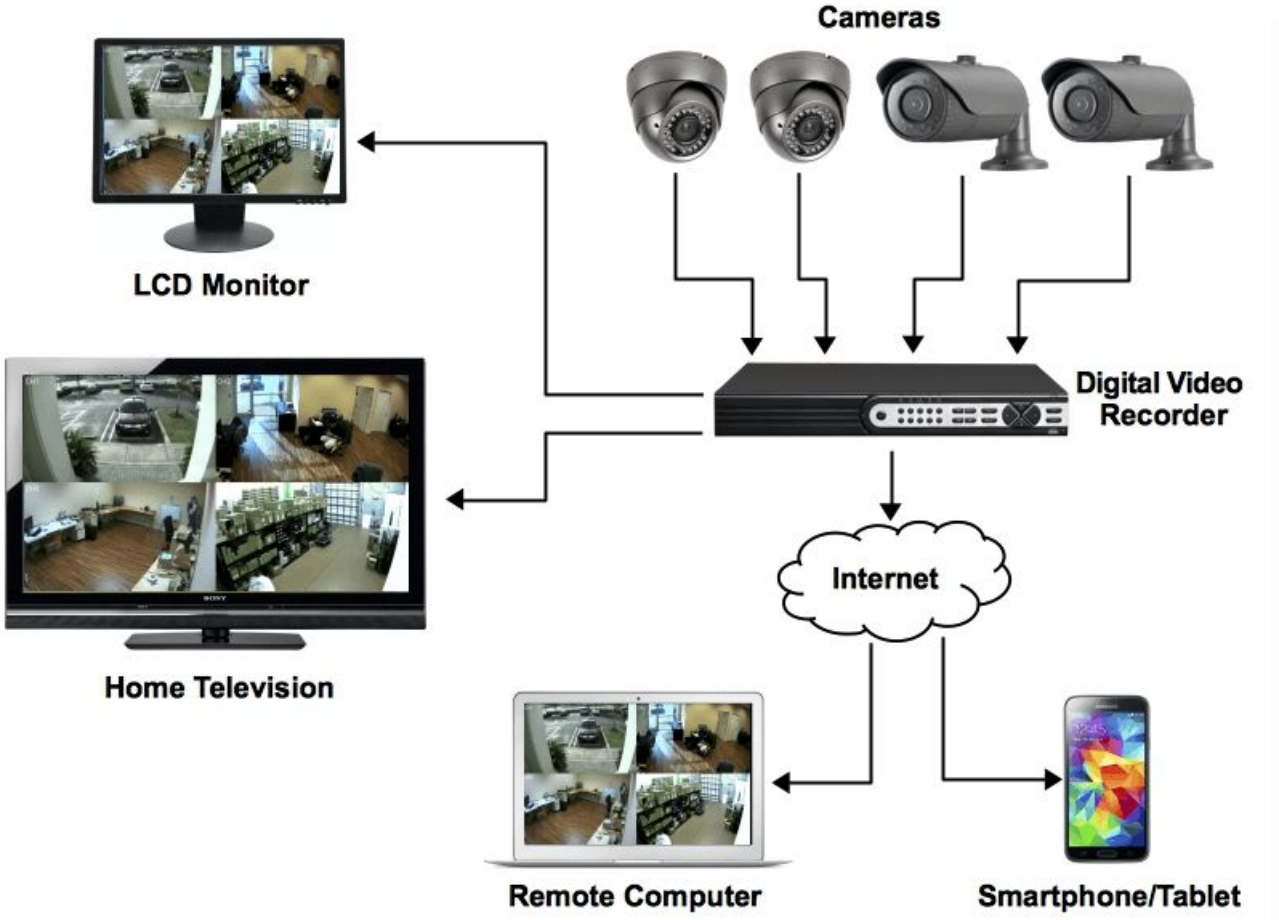
WORKING OF CCTV CAMERAS

WORKING :

- In any basic multi camera surveillance system there are four components:
    - Cameras
    - Monitor
    - Switcher
    - Memory Drive
- On a local CCTV network:
    - Cameras are connected with monitor for observation
    - By switcher we can switch from one camera to another

PESU CCBD

WORKING:

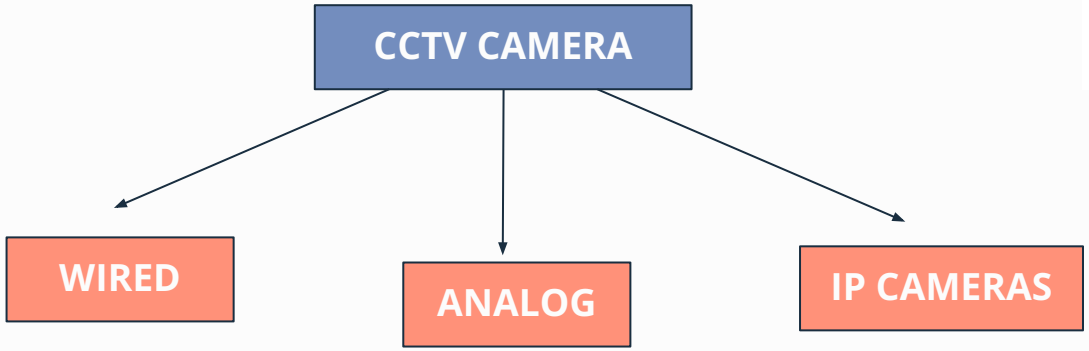

Cameras
LCD Monitor
Digital Video Recorder
Home Television
Internet
Remote Computer
Smartphone/Tablet

PESU CCBD

02

TYPES OF CCTV CAMERAS

THREE MAJOR CATEGORIES :



- They use cables to transmit footage and conduct video surveillance
- The signal can weaken when the transmission range exceeds 300 meters.
- Several cameras can be interconnected to a single monitor located in a security room.

- They have basic functionality and store the video onsite.

- IP cameras boast sharper, higher resolution images and more flexible features like remote zoom and repositioning
- Compresses the file to transmit over network
- We can access live videos on our personal devices.
- Costly

- Dome Camera
  i. Commonly used cctv , due to the shape one cant tell where the camera is looking
  ii. Has IR night vision

- Bullet Camera
  i. Used for long distance viewing
  ii. High quality images
  iii. IR night vision

- C-mount Camera
  i. Distance > 40ft
  ii. Detachable lenses
- Day/Night Camera
  i. Capable of operating in both normal and poorly lit environments
  ii. Record in gray scale and color as well.

- Thermal Image cameras
  i. Useful for detecting intruders
  ii. Good in low light

PESU CCBD

PESU CCBD

03

RESEARCH PAPER 1

Scalable architecture for an automated surveillance system using edge computing

Authors : Hwin Dol Park Ok-Gee Min Yong-Ju Lee

IMPORTANT POINTS FROM THE *ABSTRACT* OF THE RESEARCH PAPER

SUMMARY :

1) SURVEILLANCE SYSTEM - USED SENSORS
2) ANALYZED DATA - WAS SENT TO SERVER
3) THE PROJECT IDENTIFIED INTRUDERS AND ABNORMAL SITUATIONS
4) EDGE ACTS LIKE A MINI-ANALYSIS-SERVER

- Nowadays surveillance systems make use of sensors to improve intelligence.
- The analyzed data from multiple sensors are useful for video surveillance systems to detect abnormal situations.
- But due to the large number of sensors the complexity of the surveillance systems increase.
- To maintain these large numbers of sensors a system architecture is required.
- This architecture will face these **challenges**
    - **Scalability** - increase or decrease the power of a solution
    - **Network Bandwidth**
    - Process Resources on the server
    - **Reaction-time-** time taken by the server to respond
- In this paper they have proposed an architecture - which is to analyze **stream sensor data in real-time** to improve robustness (able to withstand adverse conditions and intelligence.
- The **core** part of the architecture is the **edge node** between server and sensors.
- The edge manages cameras and sensors like a small server.
- **Key points** of the architecture :
    - Each edge will maintain and manage its own sensors and cameras.
    - Edges won't upload unnecessary data to the server.
    - Collaboration between edge and server to improve processing.
    - Edge will reduce the reaction time between sensor and server.

IMPORTANT POINTS FROM THE *RELATED WORK* SECTION OF THE PAPER

SUMMARY :

INCREMENTAL VIEW OF DIFFERENT SYSTEMS WITH THEIR DISADVANTAGES
1)3GSS- DISADVANTAGE : WIRED , HUMAN OPERATORS.
2)WIRELESS SENSOR NETWORK - USED DETECTION SENSORS , DISADVANTAGE : THE PACKETS - HEAVY TRAFFIC ON NETWORK
3)MPEG - TO REDUCE BANDWIDTH MPEG WAS USED TO COMPRESS VIDEO.
4)AUTOMATED SYSTEMS WITHOUT HUMAN OPERATORS , IN THEIR PROJECT THEY USED EDGE TO PRE-PROCESS AND TRY TO REDUCE NETWORK BANDWIDTH.

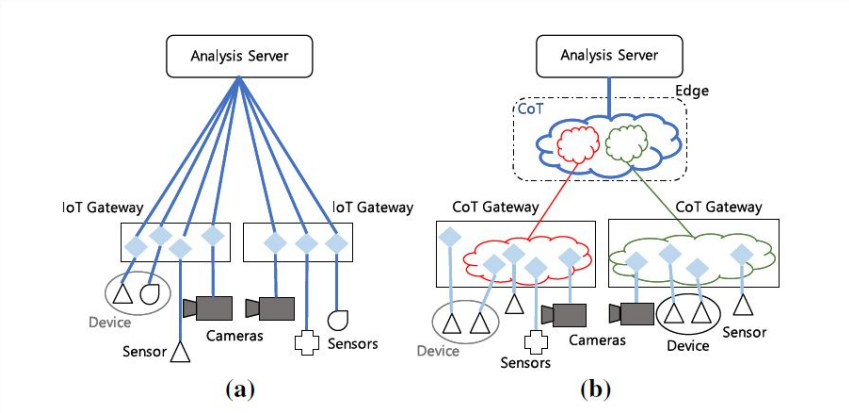
- The researchers have mentioned other works similar to the research topic in this section.
- Other Research Paper Works / Technologies :
    - Third generation surveillance systems- wired(disadvantage) : Showcasing scenes to the human operators and helping them to understand it well via video analysis , multi person tracking - in 3GSS - the number of surveillance cameras were limited.
    - Wireless Sensor Network - Used detection sensors - but the quality was same as the traditional surveillance cameras. The packets that reported the location of the sensors caused heavy traffic on the network(disadvantage).
    - To overcome the restriction MPEG used video compression techniques.
    - Other challenges in video streaming - packet loss
    - Techniques for wireless video streaming
        - Flexible Macroblock Ordering technique
        - Error resilient encoding
        - As the volume of data was increasing rapidly
        - Automated decision making software with multiple sensors instead of human operators.
        - Ex : IP Based intelligent surveillance system supported automated decision making using data fusion from multi-sensors
    - Next improvement was **grouping sensors for data fusion**.This improved scalability.
    - Next improvement was the use of edge computing , the edge would try to reduce the network bandwidth of video streaming.

IMPORTANT POINTS FROM THE *BACKGROUND* SECTION OF THE PAPER

SUMMARY :

PROBLEMS FACED BY THE SYSTEMS
1)HIGH QUALITY STREAMING - HIGHER BANDWIDTH.
2)THEY USED CoT SYSTEMS (CLOUD OF THINGS)
3)THESE SYSTEMS PREPROCESS AND SEND DATA TO ANALYSIS SERVER FOR FURTHER PROCESSING.

- Problems in IOT Surveillance systems
  - **Integrating the videos and the data** collected from the sensor and passing it to the analysis server - but high quality video streaming requires higher bandwidth and also there is processing limitation on the server.
  - In this paper the authors have proposed a CoT (cloud of thing) system.
  - CoT gateways are used - they are edge devices like mobiles , microcomputers etc.
  - These gateways process the data before sending it to the server.
  - The preprocessing solves problems of the limitations of the central analysis server.
  - The analysis server collects all CoT data and makes CoT's object.



IMPORTANT POINTS FROM THE *PROPOSED SYSTEM* SECTION OF THE PAPER

- The edge isn't just a simple bridge; instead it's like a mini-analysis-server.
- Edge can handle simple analysis algorithms.
- It can also control actuators alone as well , this reduces reaction time between the edge and sensor.(actuator is a part of a device or machine that helps it to achieve physical movements)
- Edge makes logical CoT groups from a large number of heterogeneous sensors and video streaming based on analysis algorithms.

IMPORTANT POINTS FROM THE *PROPOSED SYSTEM* SECTION OF THE PAPER

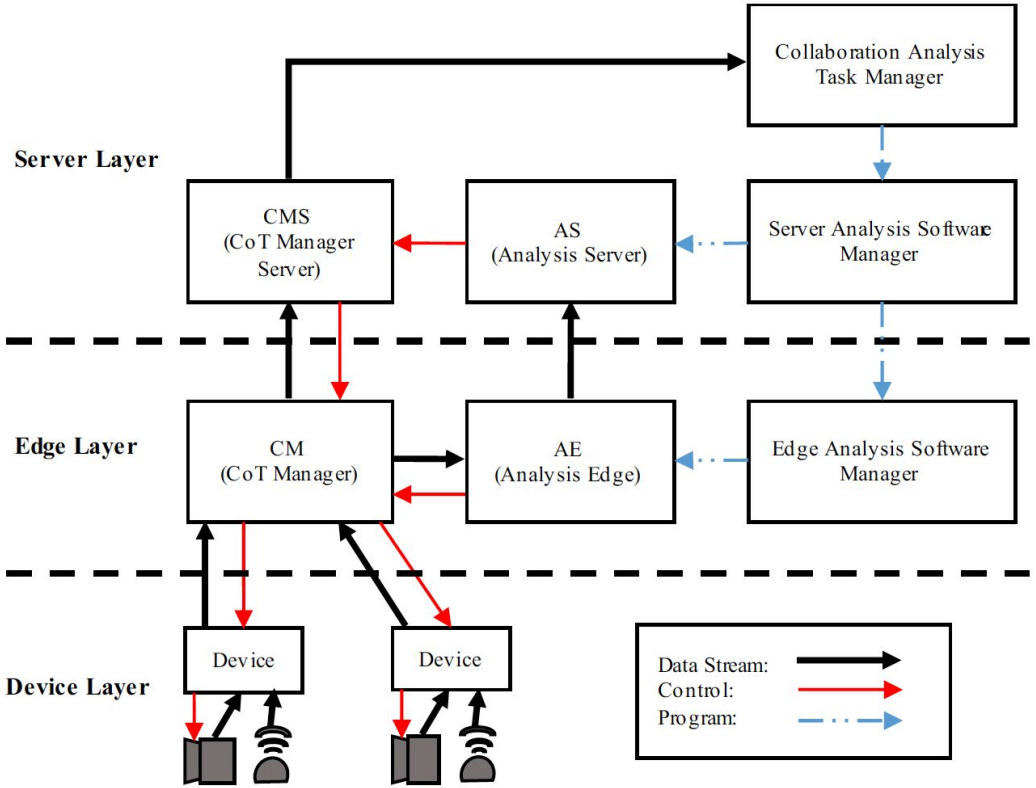
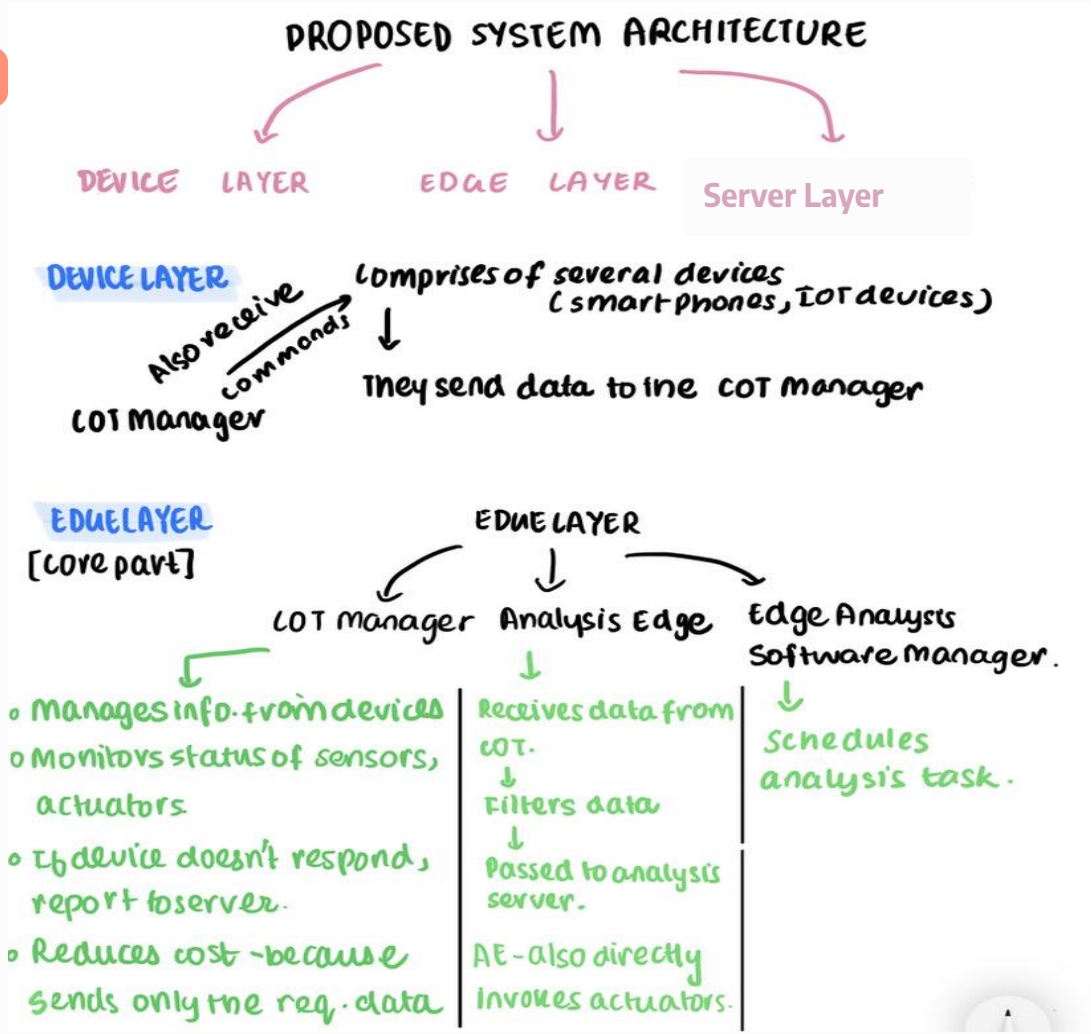


Fig. 2 The system architecture overview

IMPORTANT POINTS FROM THE *PROPOSED SYSTEM* SECTION OF THE PAPER



IMPORTANT POINTS FROM THE *PROPOSED SYSTEM* SECTION OF THE PAPER

SERVER LAYER

step-wise / orderly

- Collaboration Analysis Task Manager - Does administration of hierarchical analysis b/w server and edges
- Task Manager - Monitors analysis on analysis software
  - Monitors CPU usage and memory usage
  - Handles errors.
- Server software manager - checks if the software - identified by an unique ID is properly installed.
- COT — Cloud of Things
  ↓ sends data
  CMS — Cloud Manager System
  ↓
  Organizes data into groups
  ↓
  Does analysis on them
  ↓
  Transfers result to analysis server.

PESU CCBD

IMPORTANT POINTS FROM THE *EXPERIMENTAL VIDEO SURVEILLANCE SYSTEM* SECTION OF THE PAPER

- The project a home situation
- Dynamic CoT configuration has two parts :
    - CoT Manager Server (CMS)
    - CoT Manager (CM)
- CMS monitors the device status ie the sensors and actuators
- CMS can also request the devices for specific data if needed to send to the analysis edge
- All the sensors are grouped by CM
- Managing the groups is the function of the CM
- CM collects data from them and sends it to AE (Analysis Edge)
- AE analyses data collected from the sensors
- Processed data is sent to AS (Analysis Server)

IMPORTANT POINTS FROM THE *ALGORITHMIC DESCRIPTION* SECTION OF THE PAPER

SUMMARY : (ALGORITHMS USED)

1) HAAR FEATURE BASED CASCADE CLASSIFIER IN OPENCV
2)EIGENFACES ALGORITHM
3)ABNORMAL SITUATION DETECTION

- In the paper - they have the intrusion alert system .
- For the **intrusion alert system** they have used PIR sensor to detect the intruder
- Edge will start live streaming of the video
- If the security switch is turned on - ultrasonic sensor will measure the distance from the intruder.
- When the intruder is detected - the led and buzzer will start blinking.
- When the intruder is detected - edge increases streaming quality
- The edge tries to detect an intruder's face in streaming video using **Haar feature-based cascade classifiers in OpenCV.**
- Instead of sending the entire video , the edge will only send the face of the intruder to the analysis server.
- After the server finds the image - the server tries to determine if the face - is of a family member of not. // faces are recognized by Eigenfaces algorithm
- The edge finds the nearest neighbour of the image in the database.
- Abnormal Situation Detection - when the sensor detects fire and gas - the edge will increase the streaming quality - from low to high
- The edge tries to detect faces in the bedrooms , if it finds any faces sends the alert to the server

IMPORTANT POINTS FROM THE *ALGORITHMIC DESCRIPTION* SECTION OF THE PAPER

SUMMARY : (ALGORITHMS USED)

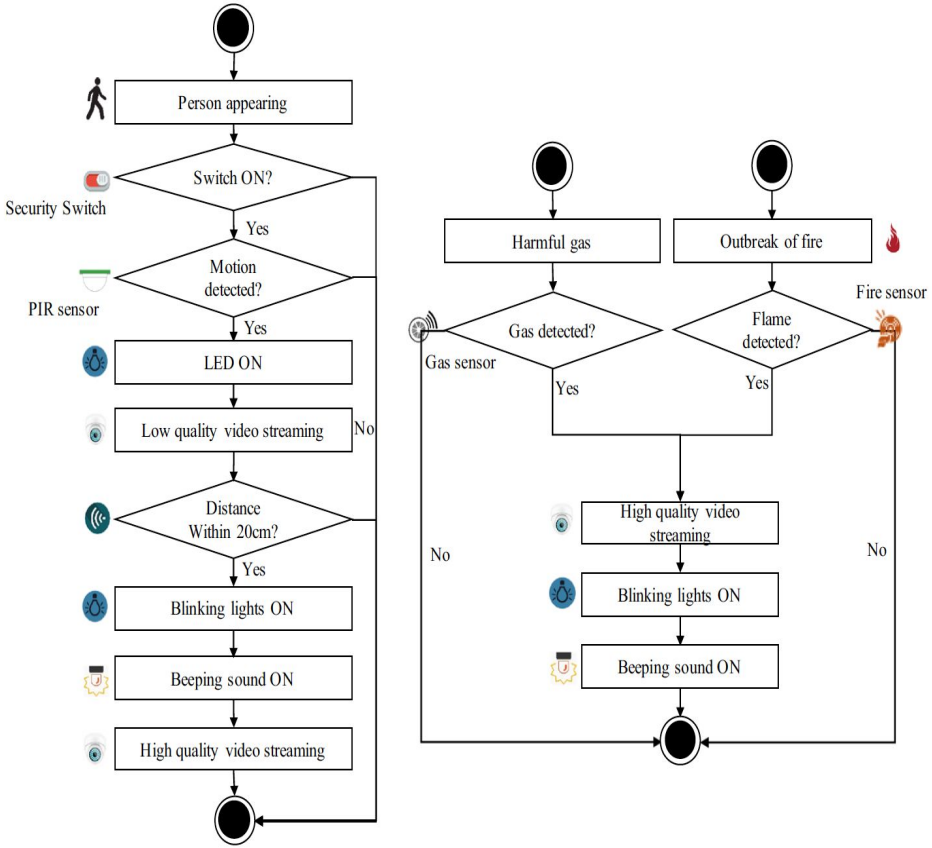
1) HAAR FEATURE BASED CASCADE CLASSIFIER IN OPENCV

REF :

MEDIUM ARTICLE - FACE DETECTION

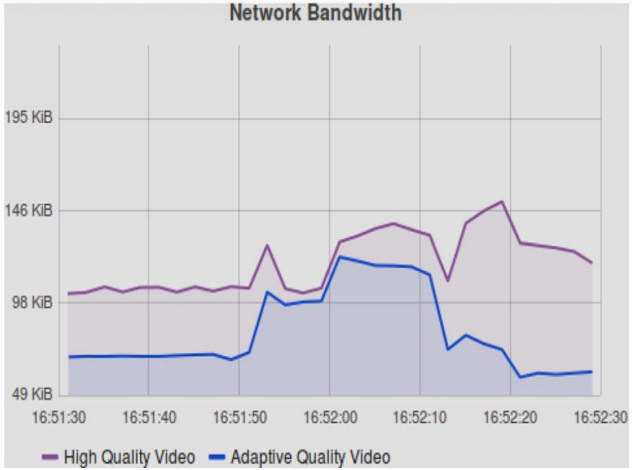
- The features - for example are that the region around the eye is darker , than the region of the cheek.
- Another example - region of the eye is darker than region of the nose
- So if the first feature does not match , that region is rejected , if it matches move onto the next feature and so on.

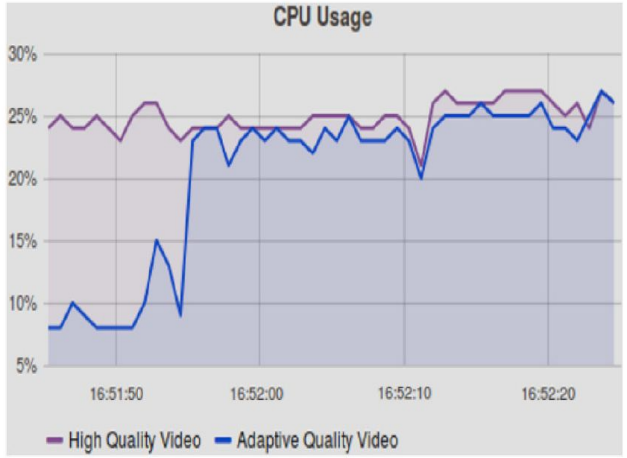
IMPORTANT POINTS FROM THE *ALGORITHMIC DESCRIPTION* SECTION OF THE PAPER (FLOWCHART)



IMPORTANT POINTS FROM THE *ADVANTAGES* SECTION OF THE PAPER

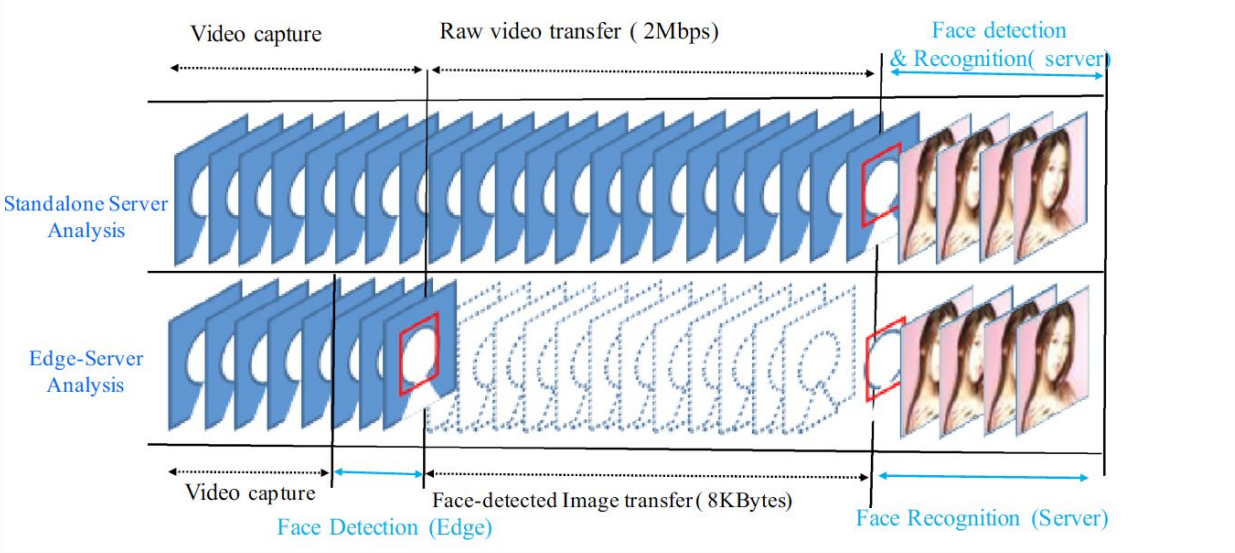
- Streams high quality video only when there is a intruder / abnormal situation
- When used NetHogs (**a realtime network monitoring tool**) - it can identify the network bandwidth , the bandwidth usage was drawn by Grafana **(allows for visualization ie graphs etc)**





- The dramatic increase of CPU usage is detected when there is an intruder / abnormal situation , the same goes with network bandwidth.

IMPORTANT POINTS FROM THE *ADVANTAGES* SECTION OF THE PAPER



- The above image shows two different steps
  - The first half shows how the system would work without edge computing - the entire load would be on the analysis server
  - The second half shows how the system would work with edge computing - so here the edge would detect the face and then send it to the server - reducing the load of the server.

IMPORTANT POINTS FROM THE *CONCLUSION* SECTION OF THE PAPER

- Architecture was designed on real-time data
- Edge computing was used to reduce load on network and server
- They were able to reduce CPU usage by 50%.
- They were able to reduce Network Bandwidth from 15 MB to 80KB

04

Research Paper 2

VU: Edge Computing-Enabled Video Usefulness Detection and Its Application in Large-Scale Video Surveillance Systems

http://weisongshi.org/papers/sun19-vu.pdf

IEEE INTERNET OF THINGS JOURNAL, VOL. 7, NO. 2, FEBRUARY 2020

Hui Sun , Weisong Shi , Fellow, IEEE, Xu Liang, and Ying Yu

Important Points from the Abstract

1. Video Surveillance systems involving thousands of cameras really important for public safety.
2. When a failure encountered in such a system,the maintenance teams spend substantial amount of time locating and identifying failure,resulting in a slower online response.
3. Moreover video data having potential failures consume unnecessary bandwidth and waste lot of storage in the cloud.
4. *Discussed in the article*: an edge computing enabled video usefulness (i.e., VU) model for large-scale video surveillance systems and its application in early failure detection and bandwidth improvement
5. Prime goals as per the article:

(a) Exploring feasibility of a comprehensive VU model and to determine VU values in a real application.

(b) Reducing Mean Time to Detection (MTTD) via edge-computing enabled fast online detection approaches.

(c) Relieving network bandwidth of large scale video surveillance systems.

The Problem Statement

1) What is the failure status in the large scale video surveillance systems?
2) How can a failure in the video data be detected on the fly to avoid incurring the cost of storing useless video data?
3) How can these failure detection approaches be deployed using the edge computing model?

Some Observations:

- Video data with an occlusion failure exhibits satisfactory QoS metrics in the network with-out failures (e.g., delay and packet loss),however is useless for video analytics.

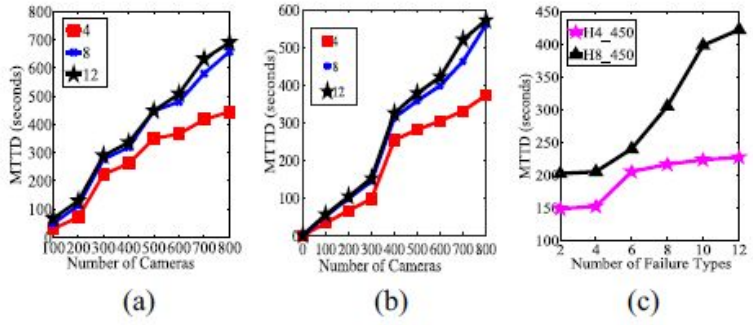


Fig. 2. MTTDs of 4, 8, and 12 failures for cameras with (a) H8-hop and (b) H4-hop network distances. Fig. 2(c) presents the MTTDs of 12 failures for 450 cameras with the H4-hop (H4_450 for example) and H8-hop network distances.

The notation H4 indicates that the number of hops from the cloud to a camera satisfies $1 \leq hops \leq 4$, while the notation H8 indicates $4 < hops \leq 8$ is satisfied.

We conclude that the massive number of cameras and a long surveillance time render the detection of a failure in the large-scale video surveillance system scarcely possible as its scale is increased in addition to the fact that the cloud has a lot of video data with failures with burdens the network and increases storage usage on the cloud.

Fig. 1. Failures in a video stream. Normal video versus videos with four types of failures: 1) normal (a) versus occlusion failure (b); 2) normal (c) versus tilt failure (d); 3) normal (e) versus blurring failure (f); and 4) normal (g) versus brightness failure (h).

The VU Model

Objectives of VU Model

1) Analyzing video data from an edge camera and a data package from a network node (router or switch) in a fast online fashion
2) Detecting a failure efficiently via intelligent methods on the fly
3) Sending prompt alarm messages of a failure to the operations and maintenance team and helping them recover it as fast as possible.

Goals to be Achieved

1. Reducing MTTD via early detection of failure
2. Relieving burden of data transmission in the network

Contributions as per the article authors

Main Contributions in the Article

1. Identifying three failure domains in city surveillance system-edge failure domain,end-user failure domain and cloud failure domain
2. An edge-computing enabled VU model which filters useless data in nodes to avoid network congestion and reduce cloud storage usage
3. Implementing a fast online edge computing enabled failure detection aimed at detecting a failure at each node using edge computing, thus improving MTTD.
4. Devising a Failure Aware and Dynamic Adjustment(FADA) scheduling strategy for dynamically reordering the frequencies of failure detection approaches to tradeoff the consumption of computing resources and the performance of MTTD in the edge computing environment.

FAILURES AND FAILURE DOMAINS IN VIDEO SURVEILLANCE SYSTEMS

- Video data is characterized in terms of three qualities: 1) satisfactory QoS* (or QoE); 2) poor QoE*; and 3) useless content.
- we define three failure domains,namely, the F_{edge} domain, the Cloud domain, and the F_{user} domain in a video surveillance system.
- 24 types of failures in the three failure domains identified.

QoS and QoE as per *Section-VII* of the research paper(Quality of Video Service)

★ QoS is defined as the collective effect of service performances, which
determine the degree of satisfaction of a user with the service as per certain QoS metrics typically defined in terms of network metrics such as error rate, bit rate, throughput, transmission delay, round trip time, and loss ratio. It is a necessary but insufficient parameter for video surveillance purposes.

★ QoE considers quality of video surveillance from user's perspective.

VIDEO USEFULNESS DEFINITION

The amount of useful video data is the size of failure-free data in the video stream.

In a video surveillance system, let V represents the amount of video data which is captured from a camera in the time interval [1,NT], where the times are expressed in seconds.

Le nt of video data from the eth camera in the ith unit time.

$$V_e = \sum_{i=1}^{i=N_T} (d_i).$$

U(j) indicates the usefulness value of the video data in the jth node (e.g., an edge camera and a router in the network) per unit time in a video surveillance system.
Mathematically,

Then the total amount of useful video data captured by the eth camera in the cloud is given by:

$$U(j) = \begin{cases} u, & \Delta v > 0 \\ 0, & \Delta v \leq 0 \end{cases}$$

where j ∈ [1,Mnode] and v refers to the usefulness boundary of the video data which depends on the cases

$$V_e^u = \sum_{i=1}^{i=N_T} (D_i) = \sum_{i=1}^{i=N_T} \left(d_i \times \prod_{j=1}^{j=M_{\text{node}}} U(j) \right)$$

NOTE:the VU value is sent to end-users for a further decision. The absolute value of v, which is denoted as |v|, represents a boundary value of VU for the video data. It divides the VU into various degrees.

Failure Detection Framework

- This framework preprocesses video streams or network packets for early detection of failure in each edge node along the path between a camera and a cloud server.
- A software stack is deployed on an edge computing unit which preprocesses the video streams before it is uploaded to the cloud.
- Three types of frameworks to detect failures for VU by using different computing resources, i.e., edgeVU, cloudVU, and userVU frameworks.

A. EdgeVU

❏ we employ a lightweight real-time operating system as an edge computing-oriented operating System

❏ the edgeVU captures video data from a camera using the Capturing Module and executes failure detection approaches to determine the VU in various ways

B. CloudVU and UserVU

❏ The two frameworks have the same structure and detect failure video data by using local computing resources in a cloud server (e.g., data center) or a mobile phone.

Failure Detection Approaches and FADA Scheduling

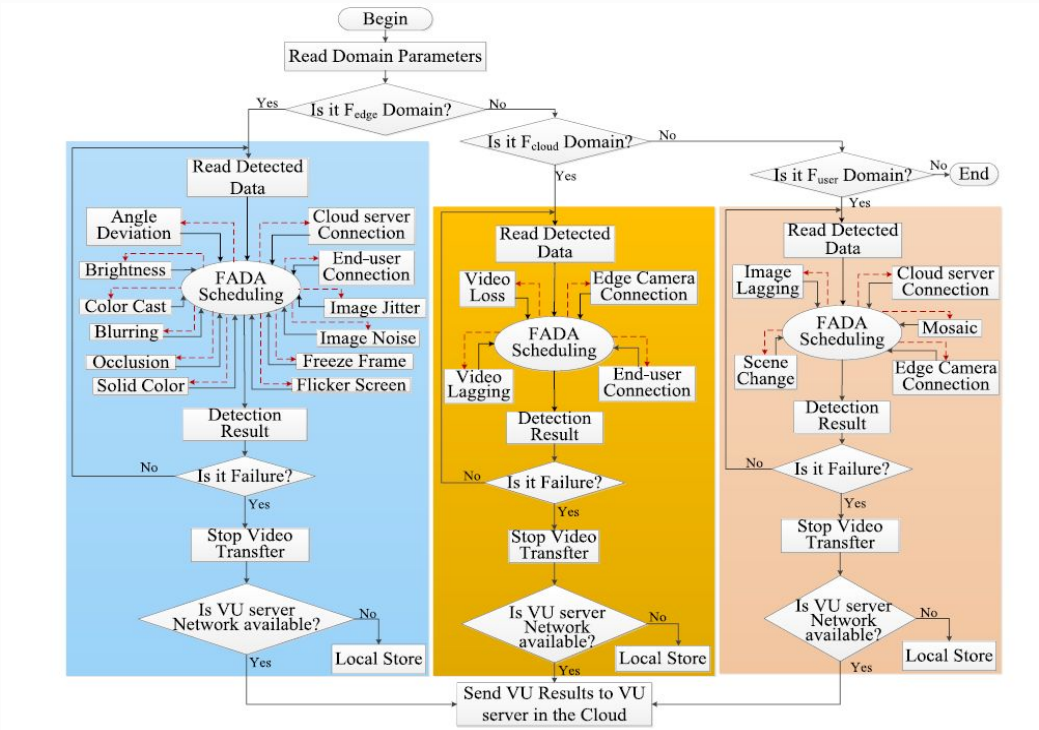


Fig. 4. Workflow of failure detection approaches using FADA in edge computing-enabled VU systems. The *black solid line* and *the red dashed line* represent the data stream and the control stream, respectively, in each failure

NOTE: In all the connection detection approaches in all domains,disconnection/problematic connections are detected by using feedback from **TCP/IP** data packets.

Failure Detection Approaches and FADA Scheduling

FADA Scheduling

- a scheduling scheme that dynamically adjusts the frequencies of failure detection approaches to balance the MTTD and the frequency.
- the FADA scheduling scheme can realize two goals:
  1) finding a failure fast online in a tradeoff frequency in the edge model with the limited resources
  2) improving the MTTD and the frequency for failure detection via the FADA scheduling strategy

Algorithm 1 FADA Scheduling

```
procedure FREQUENCY
    T_D ←the time interval for each failure detection in the edge, cloud and user domain.
    sum ← the times of consecutive failure
    sum1 ← the times of consecutive normal
    tag ← the flag of the approach
    Sa ←the sequence of the approaches (F_1, F_2, F_3, ..., F_max)
    Subscript max: the number of detection approaches in an domain.
    while true do
        Runing Sa
        tag ←the flag of F_1
        if F_j prompt camera failure then
            OutPut("Video Surveillance failure")
            Waiting for repair
            sum1 ←0
            if tag == the flag of F_j then
                sum++
            else
                sum←0
                tag←the flag of F_j
            end if
            if sum > 3 then
                T_D ← T_D/2
                set the F_j as the first approach
            end if
        else
            sum ←0
            sum1++
            if sum1 > 3 then
                T_D ← T_D×2
            end if
        end if
    end while
end procedure
```

Accuracy Of The VU MODEL

- It detects and classify the detection results into two categories, i.e., normal and failure.
- The VU model detects categories approximately and the classification results are listed as follows.

❏ **True Positive (TP):** A failure video data item is correctly detected as a failure √.
❏ **False Negative (FN):** A failure video data item is mistakenly detected as a normal ×.
❏ **True Negative (TN):** A normal video data item is correctly detected as a normal √.
❏ **False Positive (FP):** A normal video data item is mistakenly detected as a failure ×

- Four types of accuracy ratios (i.e., **efficiency, precision, recall, and specificity**) are employed to evaluate the accuracy degree of the detection approaches in the VU model

Experimental Setup

A. Evaluation Dataset:-

- Data coming from 4232 online cameras
- There are ten types of failures (see Table I) that can occur in the da[illegible]

TABLE I
FAILURES AND DISTRIBUTION IN THE EVALUATION

Symbol	A	B	C	D	E	F	G	H	I	J
Definition	Angle Deviation	Blurring	Brightness	Color Cast	Freeze Frame	Solid Color	Image Jitter	Occlusion	Noise	Flicker
#Failure	60	60	60	60	60	60	60	60	60	60
#Normal	100	100	100	100	100	100	100	100	100	100

Ten types of failures are considered in this study. #Failure and #Normal represent the numbers of failure and normal data items, respectively, for a failure-detection-approach test at each of four resolutions (*e.g.*, 1920×1080).

- Failure Detection:
  - for video => failure is detected by preprocessing video clips, such as angle, freeze frame, jitter, occlusion, or flicker.
  - For image => image is preprocessed to analyze whether a blur, brightness, color cast, solid color, or noise failure occurs.

- Data items size: 1.7 MB to 9.6 MB for video clips and 2.4 KB to 1.5 MB for images.

Experimental Setup

B. Experimental Environment:-

1) basic evaluation

- 160 test video data items

2) scalability evaluation

- 4000 data items

1. we build an edge computing-enabled camera platform using Raspberry Pi V3 (i.e., RPi V3). RPi V3 is equipped with a 1.2 GHz, 64-bit quad-core ARMv8 CPU.
2. test dataset is stored in a device on the RPi V3 testbed.
3. RPi testbed is wired to the cloud
4. This cloud platform is emulated by six processors and 8 GB of memory in a Dell PowerEdge R730 server.

transfer time is relatively small in the edge mode

Experimental Setup-Performance Evaluation

MTTD of VU Model Evaluation

The article authors conducted two tests to evaluate the MTTDs of failure detection approaches:
Type-1: The failure detection approaches are all performed in the edge model.
Type-2: The failure detection approaches are all performed after uploading the test data items in the cloud model.
Without loss of generality, we use the *image-based brightness* and the *video clip-based flicker* detection approaches to evaluate the MTTDs of VU at the four-type resolutions. The MTTD is the sum of the transfer time and
the execution time.

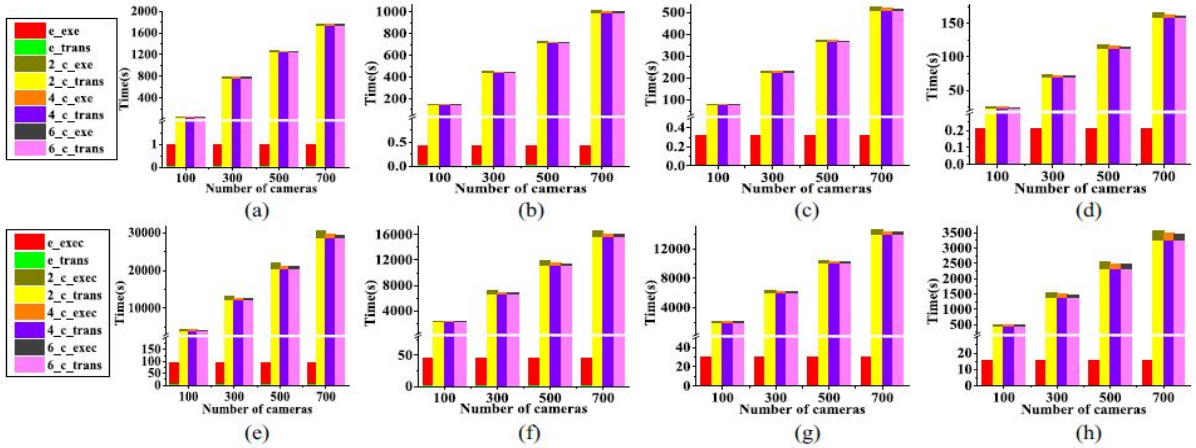


Fig. 7. MTTDs of failures C (abnormal brightness) and J (flicker screen) in the edge and the cloud models with four resolutions. The result is specified as (Failure type : Resolution) in this figure. For example, (C : 1920 × 1080) denotes the MTTD of the abnormal brightness failure detection under a resolution of 1920 × 1080. The *c* and *e* in *x_c_x* and *e_x* represent the cloud and edge models, respectively. In the cloud model, the MTTD consists of the transfer time from a camera to the cloud (such as *2_c_trans*) and the execution time (e.g., *2_c_exec*) of failure detection. In the edge model, MTTD consists of the execution time (i.e., *e_exec*) and transfer time (such as *e_trans*); however, the latter is much smaller than the former. In the cloud model, we configure two-processor, four-processor, and six-processor modes and use them to study the MTTDs of failure detection approaches in a cloud server. Notation *4_c_exe* denotes that the testbed is configured with 4 processors for failure detection in the cloud model. (a) C : 1920 × 1080. (b) C : 1280 × 720. (c) C : 720 × 576. (d) C : 352 × 288. (e) J : 1920 × 1080. (f) J : 1280 × 720. (g) J : 720 × 576. (h) J : 352 × 288.

Experimental Setup- Scalability Evaluation and Storage Usage Improvement

Scalability Evaluation

In a real-world video surveillance system, the commercial failure detection system, which is a cloud model, is deployed by Uniview Company at Anhui University.

We compare the accuracy between the VU and Uniview platforms.

- (TN+TP) in the VU-based detection system is larger than that in the Uniview platform.
- The longest MTTD for each detection approach is smaller than that in Uniview platform.

Storage Usage Improvement

From the experimental results, we observe two phenomena:

1) The Growth Trend of the Video Data Volume in the Cloud: In the VU-based framework, the growth rate of the storage usage of the data volume in the cloud can vary over time; hence, this framework is considered an elastic storage framework.
2) The Total Amount of Video Data in the Cloud: The VU based total video storage usage costs are substantially reduced, which decreases the storage usage of failure data and improves the utilization of video storage space.

Experimental Setup- Bandwidth Usage Improvement

	Normal Case		Failure Case	
	noVU	VU	noVU	VU
352×288	1.042 MB/s	0.998 MB/s	0.825 MB/s	0.519 MB/s
720×576	1.058 MB/s	0.984 MB/s	1.184 MB/s	0.472 MB/s
1280×720	0.962 MB/s	0.893 MB/s	1.142 MB/s	0.424 MB/s
1920×1080	0.953 MB/s	0.854 MB/s	0.859 MB/s	0.477 MB/s

- We select two types of video streams (i.e., failure and normal) with
- four resolutions (such as 352 × 288, 720 × 576, 1280 × 576, and 1920 × 1080) from the dataset the length of each video stream being around 10 min.
- At the edge node, Raspberry Pi3 in Section VI-B2 are employed to capture video streams, perform VU-based failure detection, and push video data to the cloud. The default network bandwidth on RPi3-based platform is 1000 Mb/s and TCP protocol is used in the data transmission.
- The data volume transferred as well as the processing and transfer time over the network gives the average bandwidth.
- In the noVU-based framework, the bandwidth for video data with normal or failure presents high occupation, such as among 0.8 MB/s ~ 1.2 MB/s. This reason is that the entire video data is transmitted to the cloud without data processing at the edge, which costs bandwidth in the network.
- The failure detection module in the VU model lengthens the procedure of data processing at an edge node; and then, the average bandwidth cost is around 0.1 MB/s.
- However, the bandwidth of video streams is improved in the VU model as the failure video data is discarded thus improving bandwidth.

FADA Scheduling Scheme in the VU Model

TABLE III
EVALUATION ON FADA SCHEDULING SCHEME

	Energy Cost (Joule)		MTTD (seconds)		Frequency	
Data Item	FADA	noFADA	FADA	noFADA	FADA	noFADA
fvc_1	2196	1728	13.61	18.74	43	25
fvc_2	2016	1404	12.01	12.71	43	27
fvc_3	2052	1584	5.88	6.77	64	42
nvc	5112	6588	174.54	162.77	8	11
mvc	7848	8244	42.49	63.26	52	36

FADA refers to the tested results that are obtained using the FADA scheduling scheme in the VU model and no-FADA represents the sequential order for failure detection approaches without the FADA scheduling scheme.

We configure the test video items with various sizes and failure sequences for the following reasons: 1) solid color, color cast, and blurring are detected by using video clips while failure detection approaches employ images to detect freeze frame and flicker and 2) the scheduling order of failure detection approaches in FADA is determined by the sequence of failure in video streams. Then, the failure time interval in the subsequent procedure of failure detection will be influenced as well.

Related Work

1) quality of video services

2) edge computing in video-analytics application

- Video surveillance systems are impacted by distortion, data is lost in transmission.

  These impact QoS/QoE.
- QoS and QoE are applied to measure the quality of video services.
  - **QoS = degree of satisfaction of a user with the service**
    - Metrics : error rate, bit rate, throughput, transmission delay, round trip time, and loss ratio
  - **QoE = quality of video services from the user perspective.**
    - evaluations mainly include three types of methods, i.e., subjective, objective, and hybrid.
    - It uses algorithms to measure the quality of video services
    - hybrid approach : combines the subjective and objective,
- Video quality assessment model:
  - full-reference (FR) | reduced-reference (RR) | no-reference (NR)
    - FR : compare original images (high quality) to the candidate images
    - RR : use partial features information
    - NR : use information from the pixel domain
  - These are performed for video quality assessment.

Related Work

1) quality of video services
2) edge computing in video-analytics application

Need for Edge Computing!

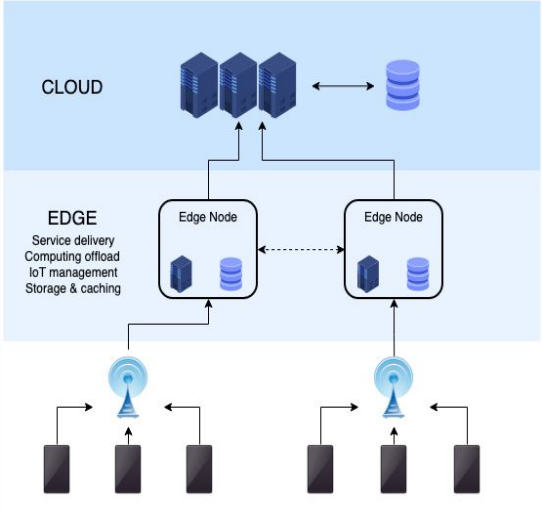
1. Inability of the computing resources that are available in the centralized cloud to keep up with the explosively growing computational requirements of massive video data from edge-based cameras.

2. Longer user-perceived delay that is caused by the data movement between edge cameras and the cloud.

3. Privacy and security concerns of data owner.

4. Energy constraints of the edge devices.

These issues in the centralized big data processing era have pushed the horizon of a new computing paradigm, namely, **edge computing**, which calls for processing data at the edge of the network



Related Work

1) quality of video services
2) edge computing in video-analytics application

- **Edge computing paradigm** enables edge nodes to perform image/video processing, which improves the performance of failure detection and satisfies real-time demands for large-scale video surveillance systems.

- **VU Model :-**
    - combines video/image processing + edge computing, employs edge computing to preprocess video streams and to detect failures in the video on edge nodes.
    - reduces the bandwidth of video transmission to the cloud and optimizes video storage in the cloud.

PES University
Cloud Computing and Big Data

CCBD
STATUS REPORT 2

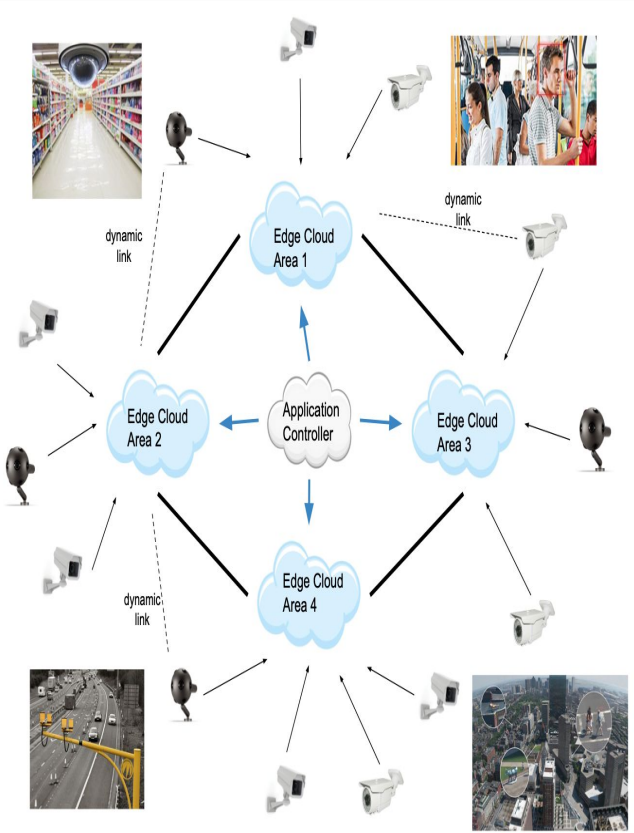
MENTOR : D.R.HL PHALACHANDRA

PRIYA MOHATA	PES2UG19CS301
PRIYANSH JAIN	PES2UG19CS303
ARPIT KOGTA	PES2UG19CS065
SNEHIL JAIN	PES2UG19CS396

01

Research Paper

Elastic urban video surveillance system using edge computing
Presented at : **Smart IOT Conference 2017**
Source of Website : ACM
Authors : Jianyu Wang , Jianli Pan , Flavio Esposito
They are students of University of Missouri-St. Louis and St.Louis University



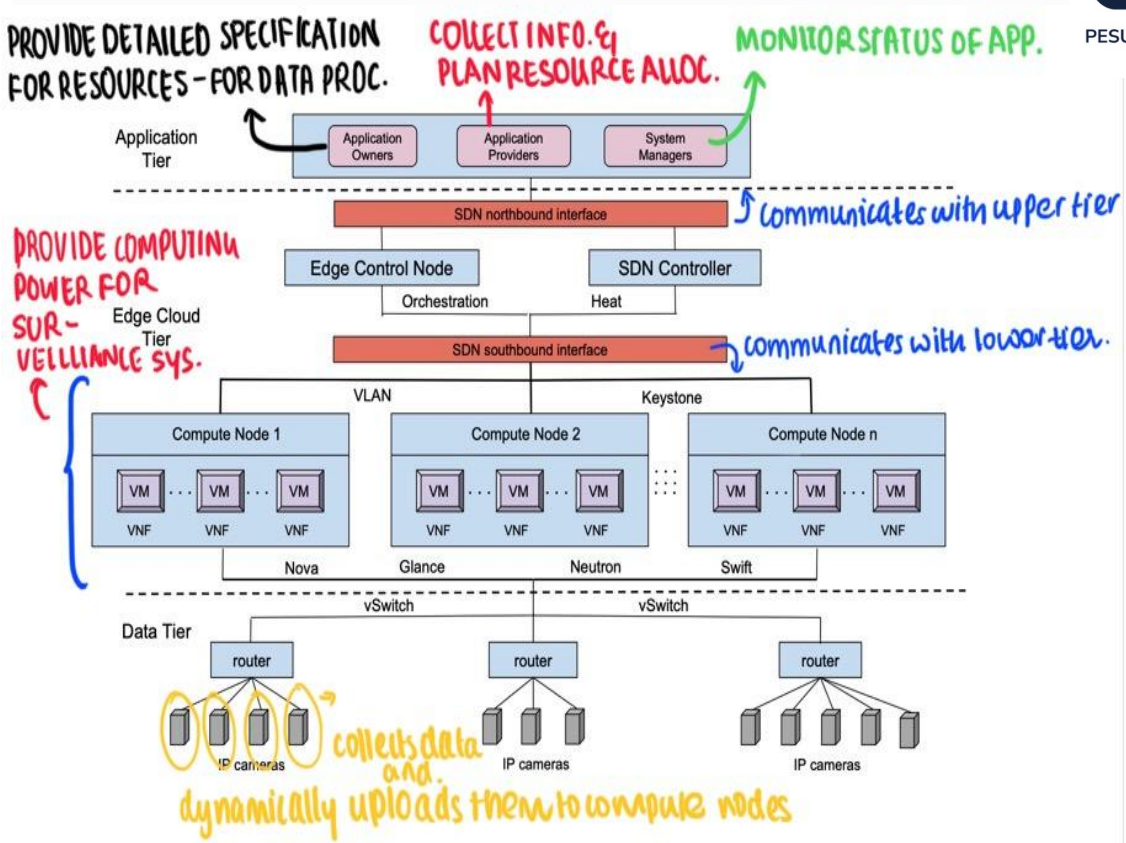


Figure 2: System Architecture.

WHAT PROBLEMS DID THEY TARGET :

1.Network Congestion
2.Lack of computing power for real-time analysis.

SIX BASIC STEPS IN WORKING OF THEIR SYSTEM

1.Collect Application Requests
2.Translate Requirements
3.Configure the Resources
4.Launch the VMs
5.Route Data Flow
6.Capture the video

HOW WAS THEIR SYSTEM WAS DIFFERENT

1.Virtualized computing service
2.Flexible data flow control
3.Dynamic Resource Allocation
4.Elastic Surveillance Modes - two modes normal monitoring and emergency mode

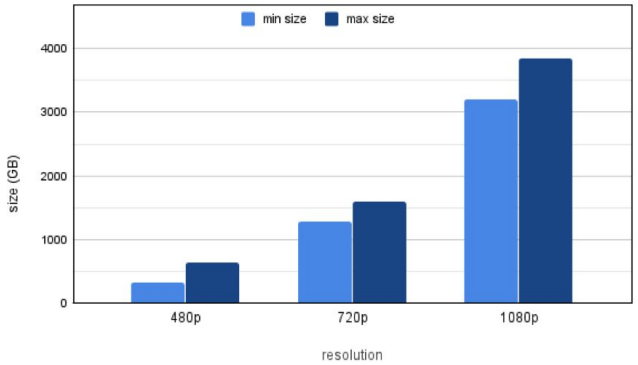
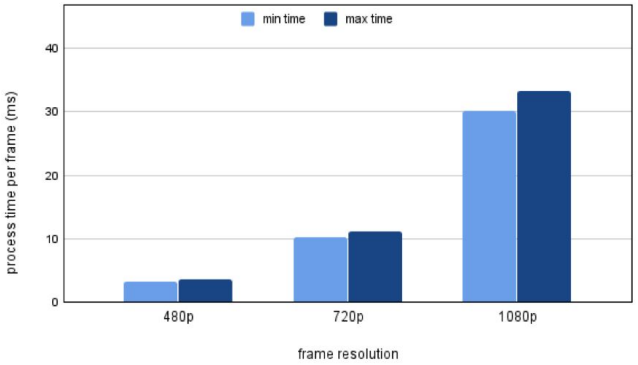


Figure 4: Data flow size per second.



SERVICE USED FOR IMPLEMENTING PROTOTYPE : A prototype for evaluating practical environment was implemented on a lightweight OpenStack platform and VMs serves as the unit of computing

1.These diagrams showcase the data flow size and the process time per frame taken by their prototype
2.For data flow , they consider the video flow rate as 16 frames per second and consider 2,00,000 cameras. (CITY LIKE SITUATION)
3.For data processing they assumed facial recognition would be used and was implemented using Haar Cascade.
4.Their experimental result proved that the architecture was rapid, responsive, flexible, scalable and easily configurable.

02

Research Paper

Smart Video Surveillance System Based on Edge Computing

Presented at : **Open Source Submitted to Sensors**

Source of Website : MDPI

Authors : Antonio Carlos Cob-Parro , Cristina Losada-Gutiérrez , Marta Marrón-Romera , Alfredo Gardel-Vicente * and Ignacio Bravo-Muñoz

Department of Electronics, University of Alcalá, Alcalá de Henares, 28801 Madrid, Spain;

Abstract

- This Paper presents a smart video surveillance system executing AI algorithms in low power consumption embedded devices

- The discussion of results shows the usefulness of deploying this smart camera node throughout a distributed surveillance system.

Problem Statement and Evolution of Surveillance system

Old systems -

- Did not process video , operator's analysis required

Modern Video processing

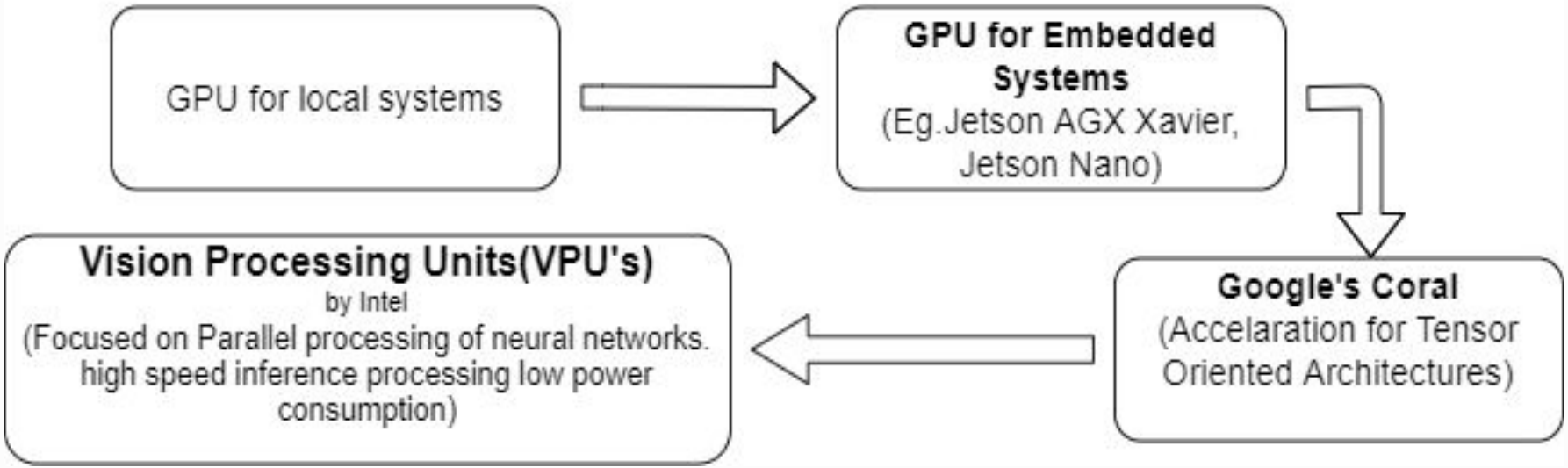
- Process the video information retrieved from camera network.

Complex Detection Algorithms

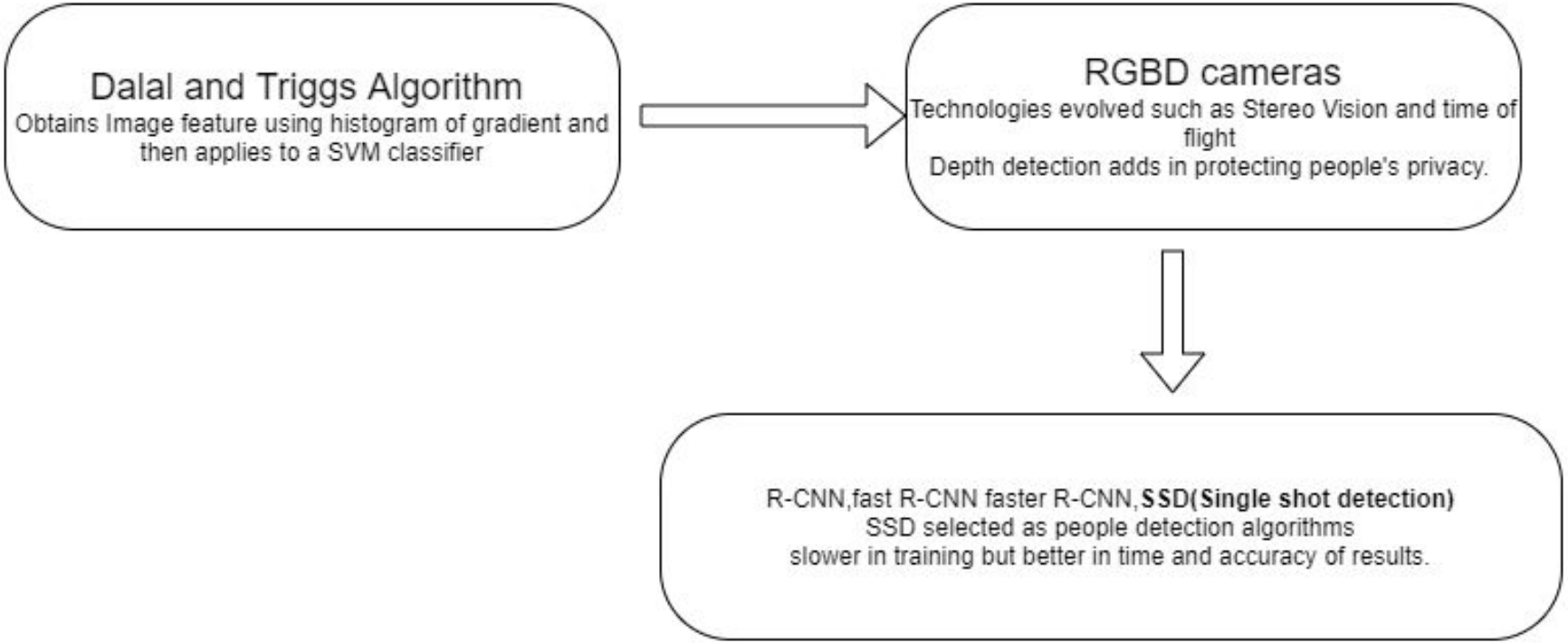
- Deep Neural Network Algorithms
    - Require high computing power and high end hardware systems
    - GPU's
    - High energy consumption d

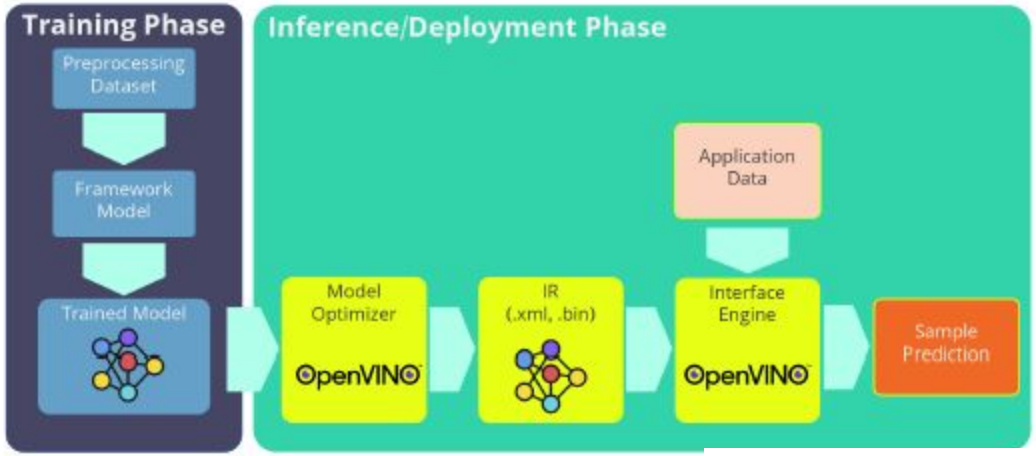
Difficult to deploy on edge of distributed smart camera nodes.

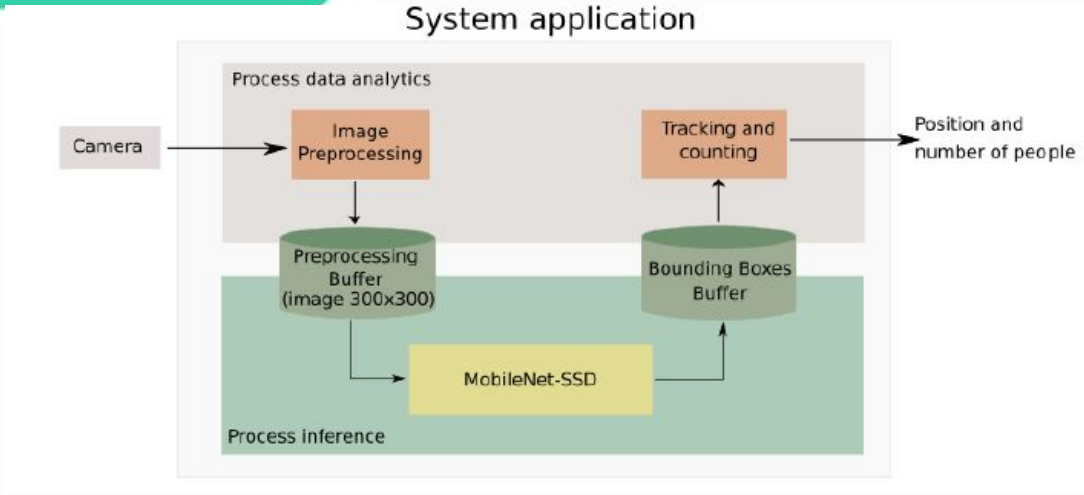
Evolution behind bringing AI enabled systems to function on Edge



Evolution of Algorithms for detection of people




Training Phase
Preprocessing Dataset
Framework Model
Trained Model
Inference/Deployment Phase
Application Data
Model Optimizer
OpenVINO
IR (.xml, .bin)
Interface Engine
OpenVINO
Sample Prediction

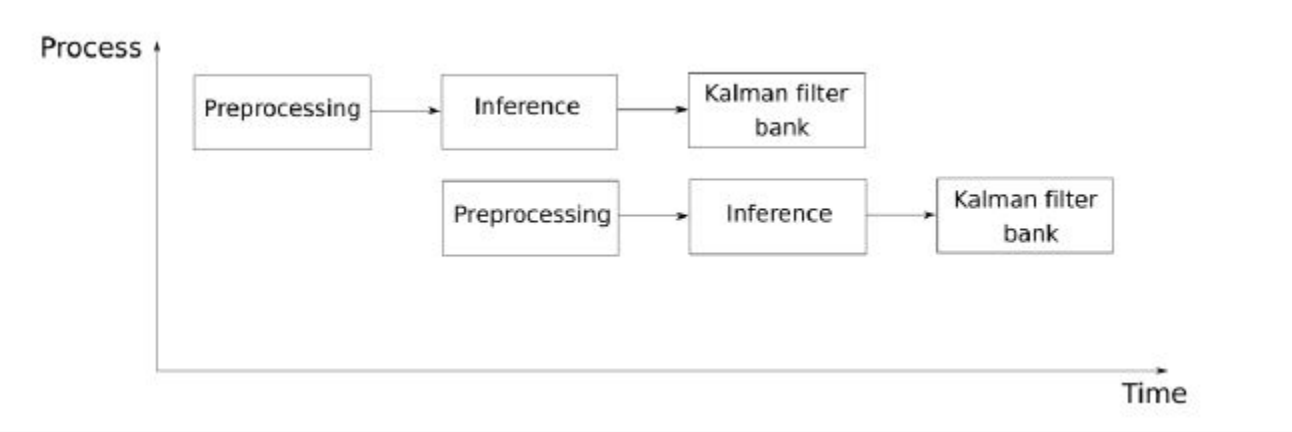

System application
Process data analytics
Camera
Image Preprocessing
Tracking and counting
Position and number of people
Preprocessing Buffer (image 300x300)
Bounding Boxes Buffer
MobileNet-SSD
Process inference

Mobile SSD and Kalman Filter

- **MobileNet SSD architecture:**
  - Uses Single shot detection for object detection
  - Efficient in terms of type of convolution layers that allows use of fewer resources
  - 35 convolutional layers responsible for the feature extraction (5 of those applying SSD)

- **Kalman Filter Bank(People Tracking)**
  - Tracker creation
  - Object association
  - Kalman filter iteration
  - Filter elimination

Pipeline Operations

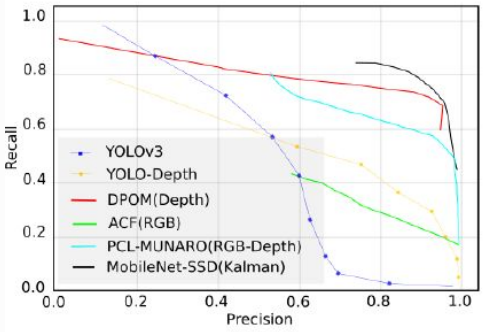


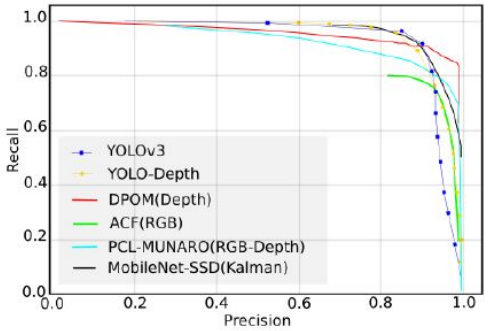
Allows processing of more than one video stream at the same time,

Solution to problem

- Using new **vision processing units (VPUs)** hardware modules. The paper describes both the hardware
- A **Mobile-Net-SSD Architecture** for task of tracking and detecting people.
- This is further attached to a bank of **Kalman filters.**
- Making a **pipelined architecture** capable of processing upto 12 video streams simultaneously.
- Model optimiser to achieve acceleration in the inference of the system.

Results





Methods	EPFL-LAB		EPFL-CORRIDOR	
	Precision	Recall	Precision	Recall
MobileNet-SSD+Kalman	87.82	88.14	81.3	80.6
ACF	83.8	86.4	66.3	40.3
DPOM	98.5	85.4	96.3	70.9
PCL-MUNARO	88.61	82.36	92	56
YOLOv3	89.7	90.3	58.6	59.8
YOLO-depth	89.8	88.6	78.4	47.9

Inference Performance on UpSquared2	t_{CPU}(ms)	t_{VPU}(ms)
MobileNet-SSD	13.93	8.71
YOLO-v3	47.54	30.07

03

Research Paper

Distributed and Efficient Object Detection in Edge Computing: Challenges and Solutions

Authors : Ju Ren; Yundi Guo; Deyu Zhang; Qingqing Liu; Yaoxue Zhang

Abstract

- With the explosion of mobile devices and the IOT,it becomes more promising to undertake real-time data processing at the edge of the network in a distributed way.
- The amount of data generated by IoT devices has dramatically increased, posing great challenges on cloud computing in terms of response delay and communication burden.
- How to integrate the edge knowledge in the cloud and make the cloud interact with edge servers to update their locally trained models is another challenge in edge computing based object detection.
- In the given article,an edge computing based object detection architecture to achieve distributed and efficient object detection via wireless communications for real-time surveillance applications is proposed.

Edge Computing Based Object Detection: Architecture and Benefits

Distributed responsibilities of various components in the architecture:

1. Media Data Compression in the End Device Layer
2. Distributed and Timely Object Detection in the Edge Layer
3. Edge Model Integration and Parameters Updating in the Cloud Layer

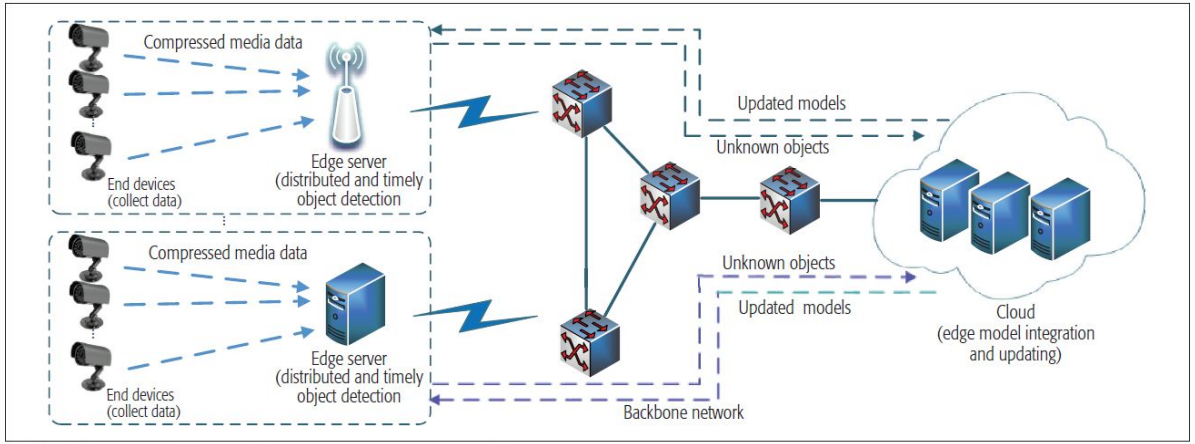


FIGURE 1. Edge computing based object detection architecture.

Key Challenges

Trade-off Between Compression Ratio and Detection Accuracy-

- Lossy compression techniques like JPEG significantly reduce image detection accuracy.
- When the channel is in deep fading, the delay caused by data transmission becomes the key obstacle for real-time applications, especially for the case that the RoI takes a large portion of the image.

Inference Model Integration and Updating

- The local model is trained to detect several specific objects that constitute a detectable object pool. To obtain the global knowledge, the cloud server integrates the local models into a much larger global model
- Although we can put the global model at the edge server to enable the detection of all objects, the small-scale edge servers may not be able to afford it due to the computation capability.

Preliminary Implementation

The binarized normed gradients (BING) model proposed is a data-driven object proposal generation algorithm, that can find latent object bounding-boxes at 300 fps on a lightweight terminal.
Edge servers detect images by adopting R-CNN model locally and implementing it in 3 steps.

1. Uses CNN which extracts compressed image as input and gives feature maps.
2. RPN used to obtain set of rectangular object proposals.
3. the softmax layer and the bounding box regression layer serve as the output layers of the Faster R-CNN.

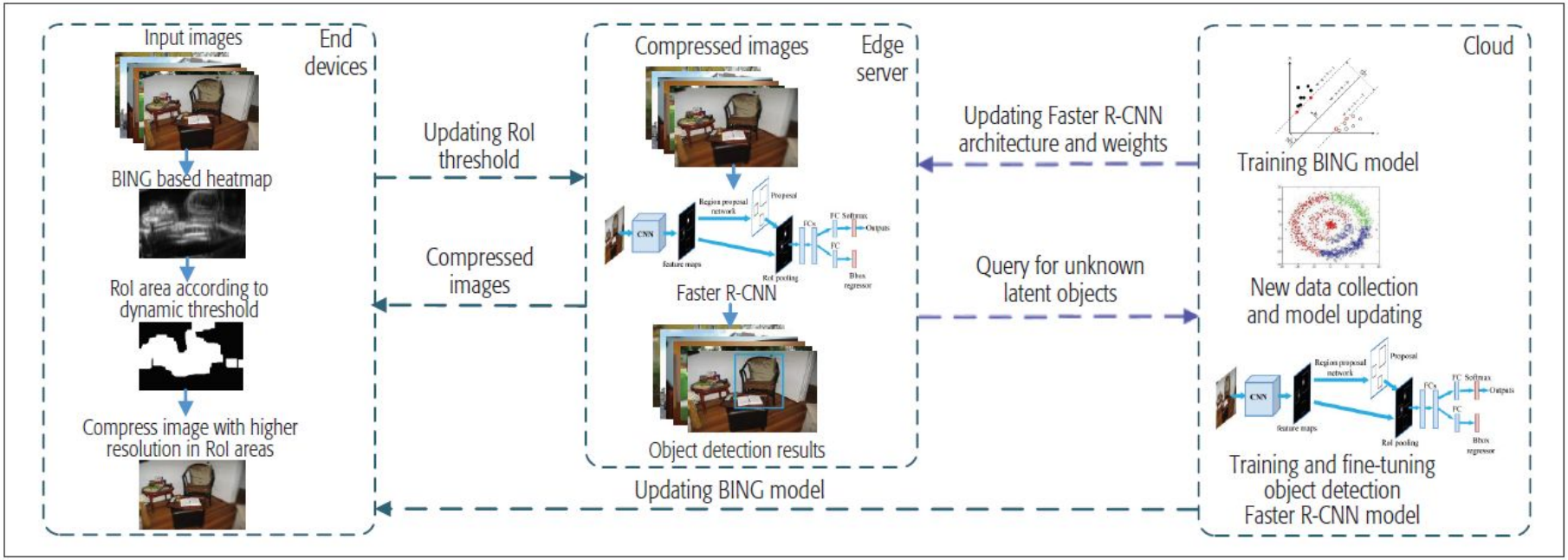


FIGURE 2. Implementation overview.

Interactions b/w the edge,cloud and the end devices

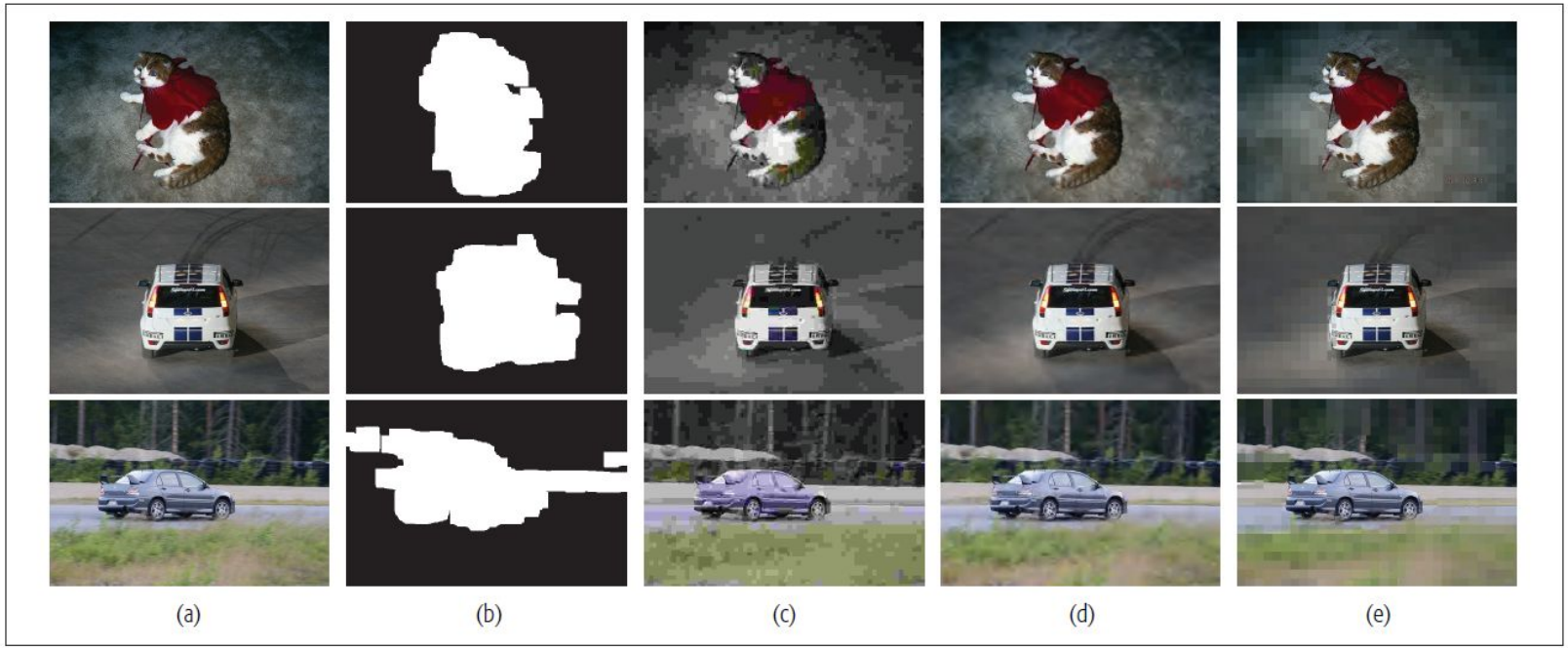


FIGURE 3. Comparison of different image compression methods: a) original images; b) RoI area; c) JPEG; d) JPEG2000; d) RoI based.

04

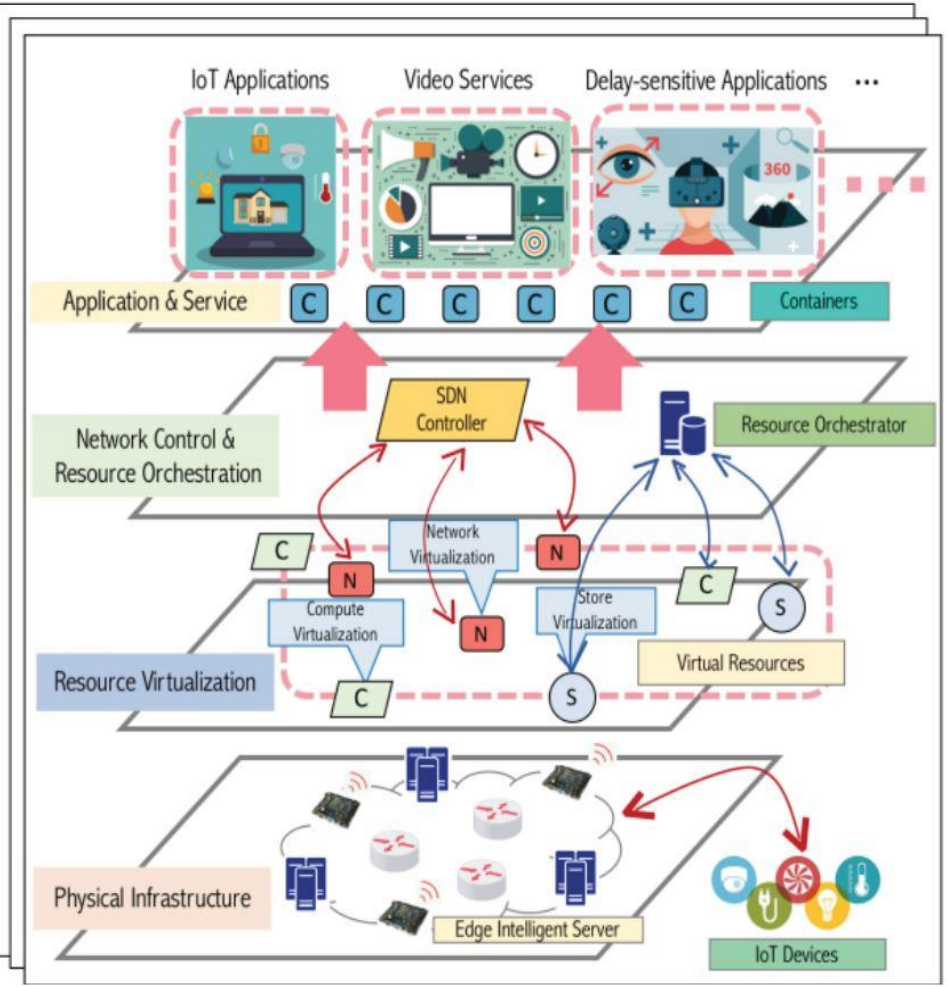
Research Paper

A Lightweight Edge Computing PLatform Integrating Video Services
Presented at : **Beijing Laboratory of Advanced Information Networks**
School of Information
Source of Website : ACM
Authors : Jian Wang, Yihong Hu, Hongxing Li, Guochu Shou
They are students of Beijing University of Posts and Telecommunications

Abstract

- IOT is involved in different domains in our daily life - healthcare, transportation, agriculture, vehicles.

- **Edge computing**, represents a new trend as a feasible supplementary solution under the cloud architecture.
- **Edge video monitoring** enables videos captured by the camera to be stored and analyzed locally, reducing network traffic on the core network.
- **Edge video analysis** enables large computational tasks such as target detection, tracking and recognition to be processed on the edge computing platform, reducing the pressure of cloud.
- **Real-time video interaction** offers the ability to match calculations and rendering to AR applications, reducing response delay and allowing users to obtain real- time and smooth AR interactive experiences.



Lightweight edge computing platform

design is characterized by
1) Low latency.
2) Efficient bandwidth utilization.
3) Lightweight virtualization.
4) Cloud-network fusion.
5) Centralized control.
6) Flexibility and Scalability.

Physical infrastructure layer contains EISs (Edge Intelligence Server), EIS has CPUs and memories needed for computing and storage and have Ethernet for providing network connections.
Resource virtualization layer contains virtual resources extracted from physical layer.
Network control layer layer mainly implements the management of virtual resources.
Application service layer. consists of containers, which are isolated but can communicate with each other through the container engine and scheduled by the container.

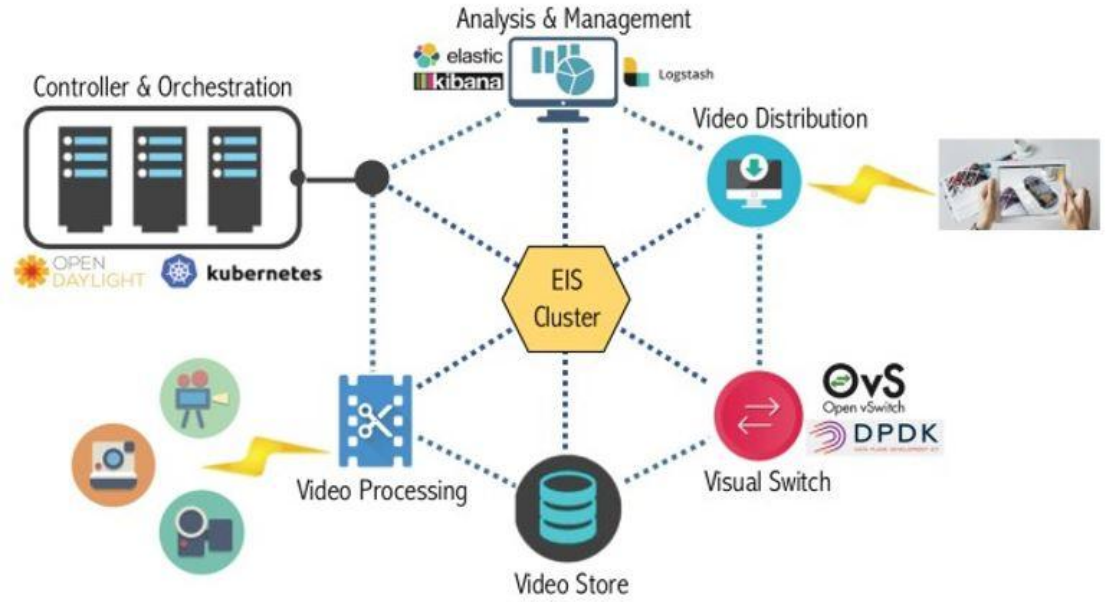
Video services scheme

- video services based on the lightweight edge computing mainly composed of edge video analysis, edge video storage, and edge video distribution services.

DIVIDED INTO SIX CATEGORIES.

1. **Virtual switch module:** It is done by OVS. It gives EIS data exchange capabilities to enable nodes to possess the function of network switches.

2. **Edge computing module:** It serves th edge cameras and off- loads computing tasks to the edge.Large-scale video analysis processing tasks are divided in multiple small computing tasks distributed among multiple EISs for computing, and then the results are summarized



3. **Edge video delivery module:** platform provides users with nearby distributed video delivery services and supports low- delay AR video delivery. Through the load balancing mechanism, service requests of users are handled, distributed video services are provided and request waiting time of users is reduced.

4. **Control and orchestrator module:** SDN controller, is used on edge computing platforms to manage virtual network resources., uses Kubernetes.

5. **Visual analysis and management module:** It realize log collection, transmission, storage, analysis, and monitoring. It also provides visualization and management of video analysis, video storage, and video distribution.

6. **Edge storage module:** provides storage for video processing, analysis, and post-cut output while providing edge caching capabilities for video distribution services

Performance evaluation

1. Figure 3 shows the test result of RTT for network packet transmission: EIS node performs much better than other nodes
2. Figure 4 shows the test result of the request response time.

3. Figure 6 shows the results of service response time for different deployments

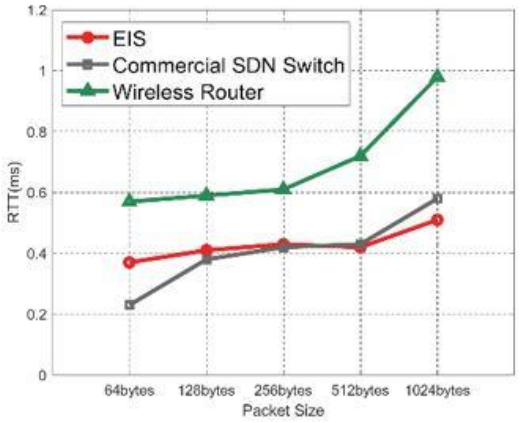
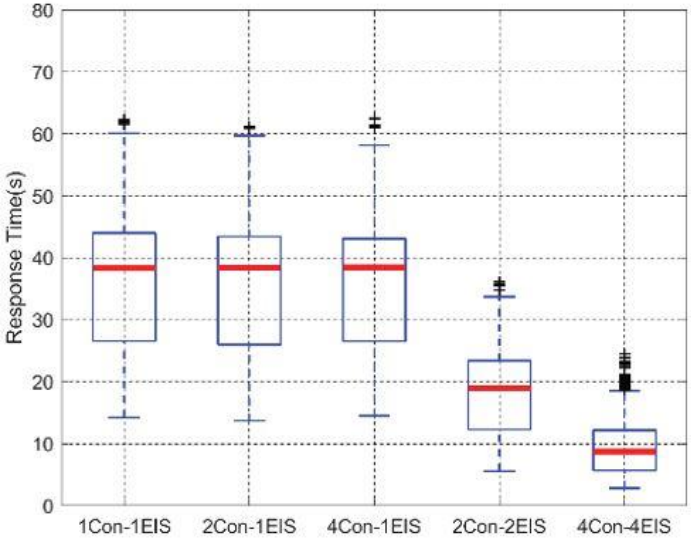


Figure 3 Comparison of forward delay



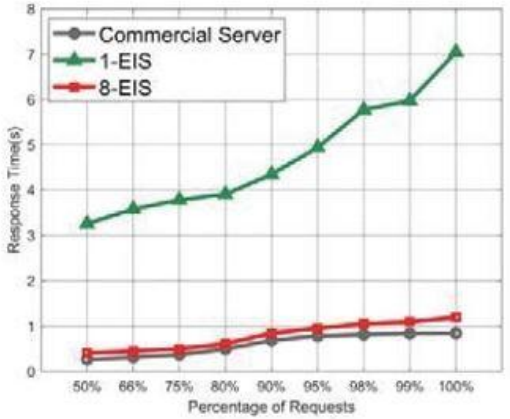


Figure 4 Response time. EIS vs. commercial server

Video services

Table 1 Video type for test.

Definition	Resolution	Bitrate/kbps	Length/s
Short Video	640*360	289	91
480P Video	856*480	840	321
720P Video	1280*720	1542	321
1080P Video	1920*1080	2980	321
2K Video	2560*1440	19442	119
4K Video	3840*2160	31563	221

- the EIS platform has higher MOS values than remote servers and is similar to edge commercial server
- The result shows that the EIS platform can provide cost- effective service and stable QoE of users.

Conclusions

EIS implemented three functions: **- computing** - **storage,** and **- networking**

- experimental results show that the EIS forward delay can meet the standards of **commercial SDN switches,**
- the processing time of requests of EIS clusters can meet the requirements of **commercial servers**
- results show that our solution provide distributed and low-latency services for video applications, hence effectively reducing service response time and **improving QoE of users.**

PES University
Cloud Computing and Big Data

CCBD
STATUS REPORT 3

MENTOR : D.R.HL PHALACHANDRA

PRIYA MOHATA	PES2UG19CS301
PRIYANSH JAIN	PES2UG19CS303
ARPIT KOGTA	PES2UG19CS065
SNEHIL JAIN	PES2UG19CS396

01

Research Paper

Smart Surveillance as an Edge Network Service: from Harr-Cascade, SVM to a Lightweight CNN

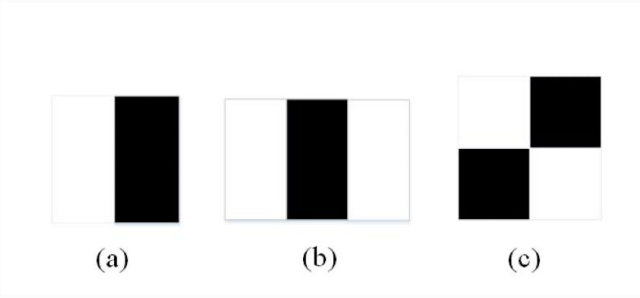
Published in**: 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)**

Source of Website **: IEEE**

Authors **: Seyed Yahya Nikouei, Yu Chen, Sejun Song, Ronghua Xu, Baek-Young Choi, Timothy R. Faughnan**

They are students of **New York State University.**

What have they proposed in the paper ?
To devolve more intelligence to the edge to significantly improve many smart tasks such as fast object detection and tracking.



Algorithm they are using : Lightweight CNN (L-CNN) algorithm.It can process an average of 1.79 and up to 2.06 frames per second (FPS) on the selected edge device

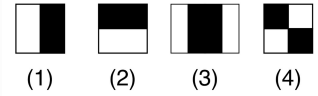
Main points from the paper
1. Lightweight CNN was used to enable real time object detection.
2.Paper discusses disadvantages of Haar Casacade and SVM [Support Vector Machine]

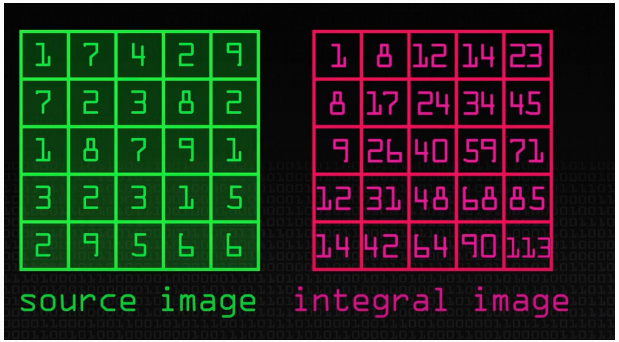
Disadvantages of Haar Cascade :
1)If all different **angles** are considered in an image to train the model , for a **small image** also **thousands** of features would be produced.
2)The computation of Haar Cascade is that's why **expensive**.
3)But once the computation is done ,However, once the training is finished, a feature set is ready and only the selected features are stored for future classification. Thus, the computational complexity of the **overall algorithm is small in execuution phase.**
4)From their observation , Haar-Cascade object classification will not perform well if the training set does not contain all possible angles
5)Also **doesnt perform well if the object is far from the camera**.
6)However **haar cascade is used for low power and real-time applications because of its fast detection.**

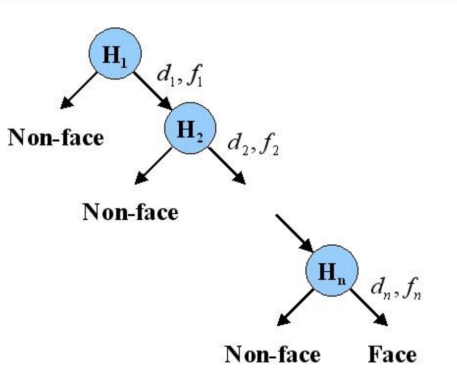
REF :
MEDIUM ARTICLE - FACE DETECTION

Haar Cascade Classifier :

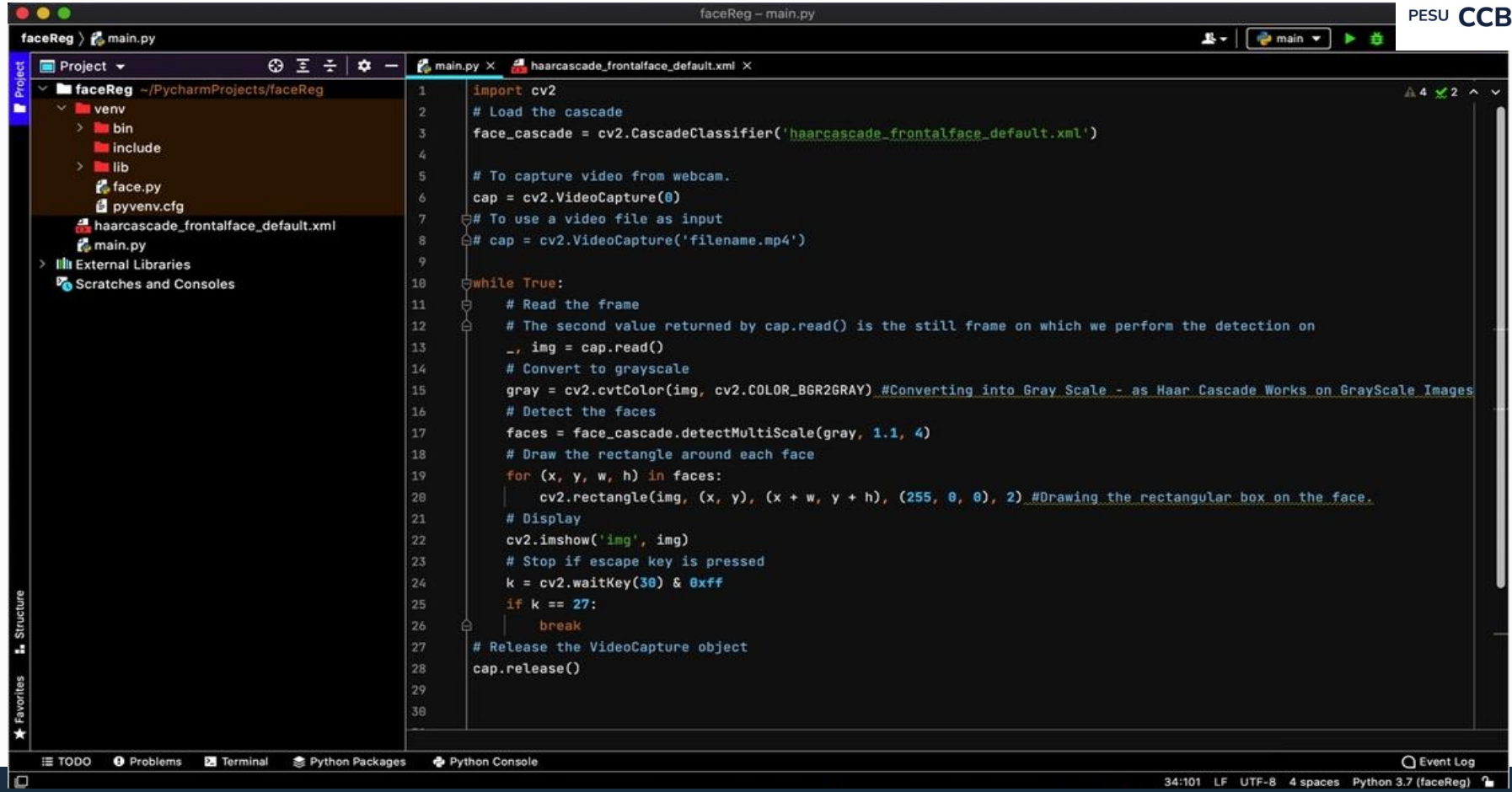
- Uses feature based classification
- Image is first converted into grayscale
- Here the classifier looks for Haar-features
- Value is found.
- Value=∑(pixels in black region) - ∑(pixels in white region)
- If the value > threshold then move onto the next feature , do this until all the features are passed.
- In each image , there would be hundreds or thousands of Haar features , calculating all of them for every region of the image would be inefficient.
- The classifier states if one Haar feature does not match for a particular region move onto the next region.
- In order to optimize calculations they used, Integral image optimization.



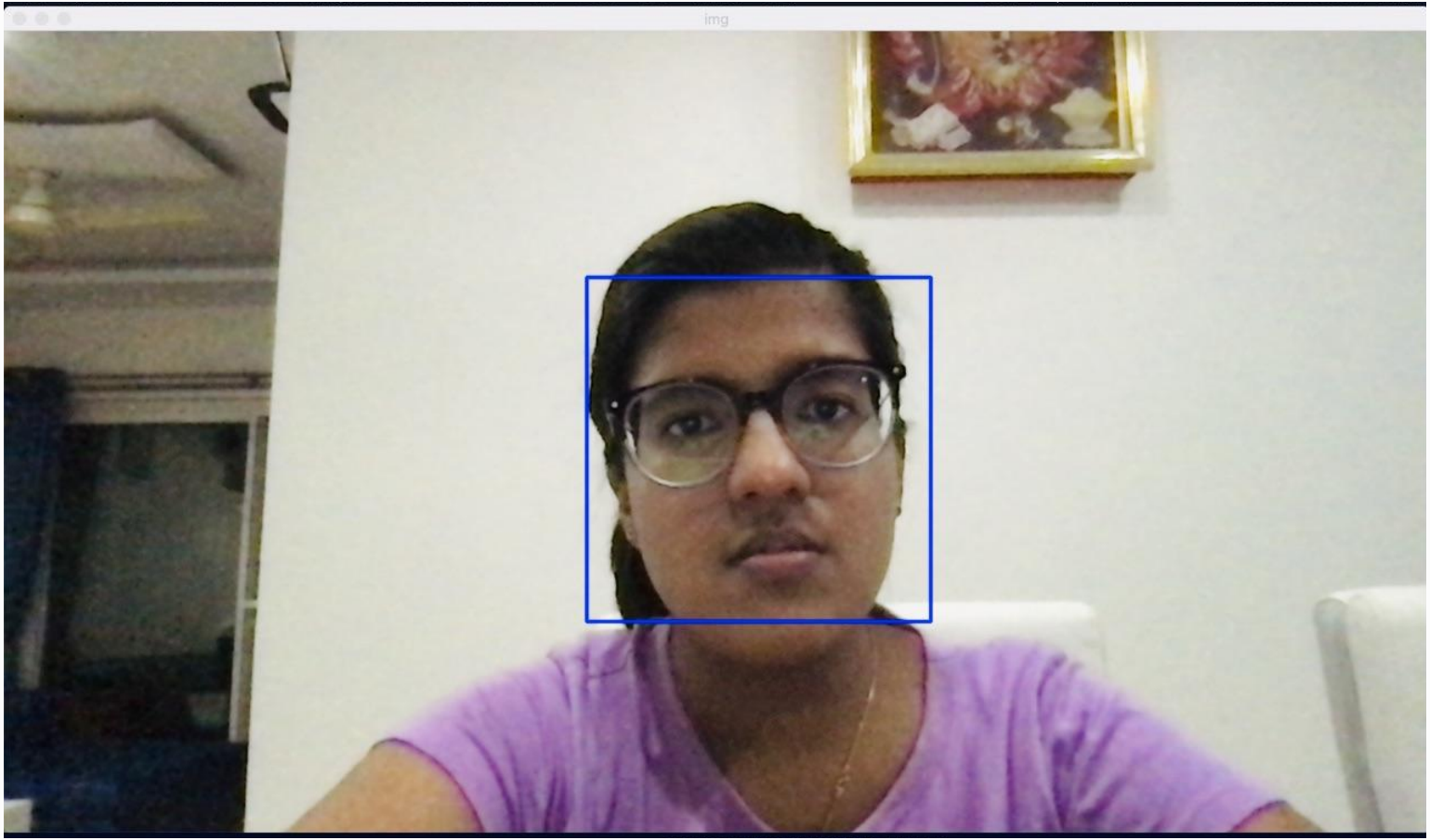




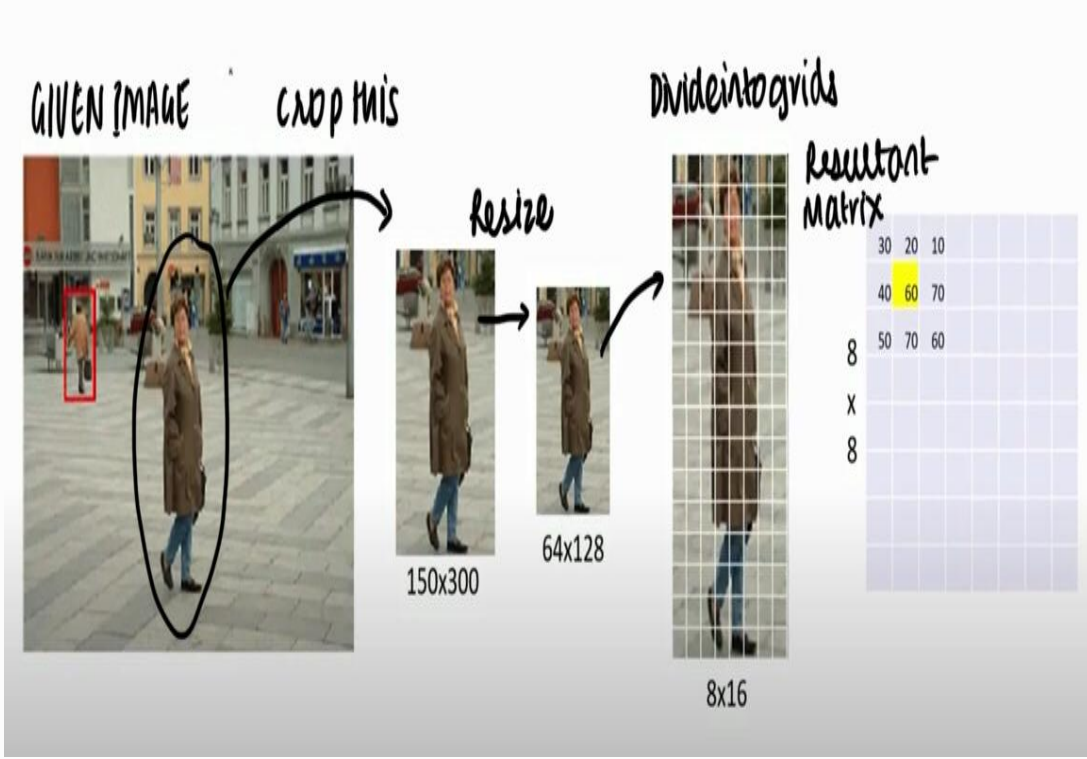
HAAR CASCADE IMPLEMENTATION

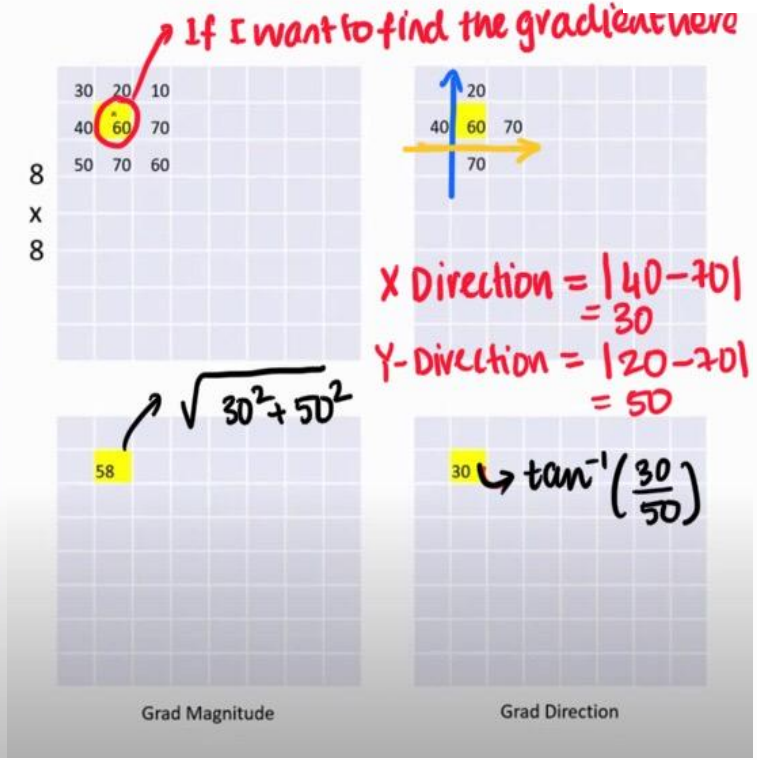


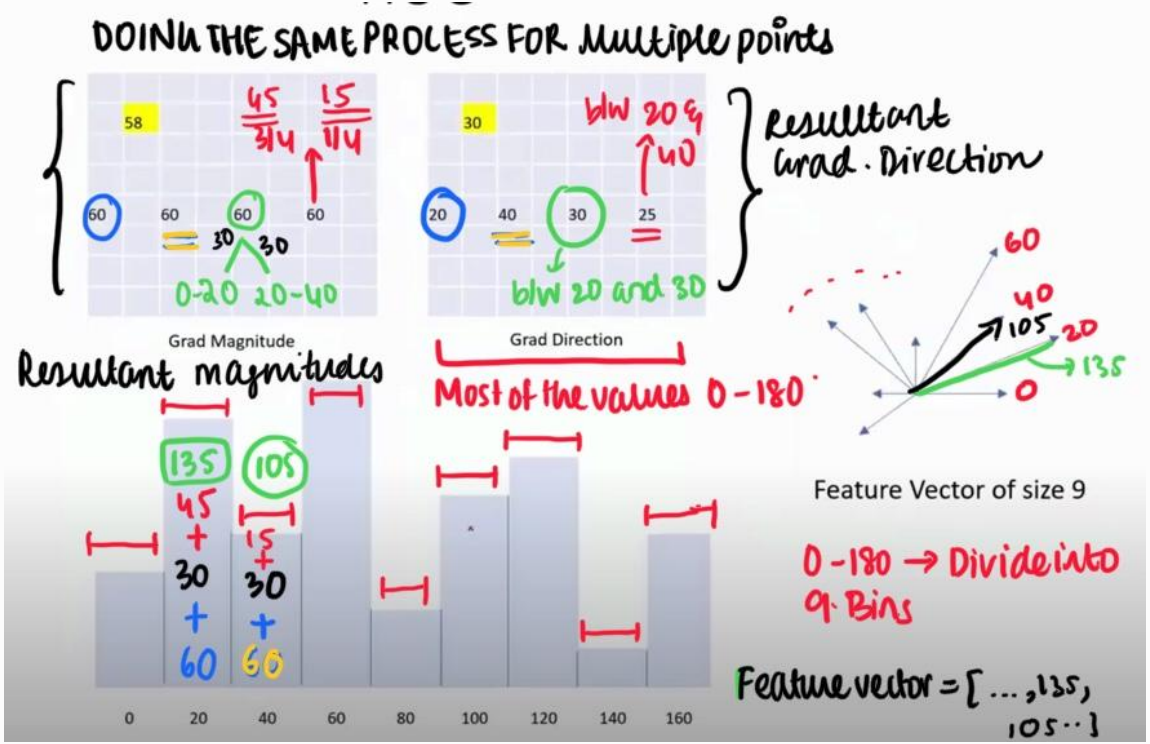
HAAR CASCADE IMPLEMENTATION

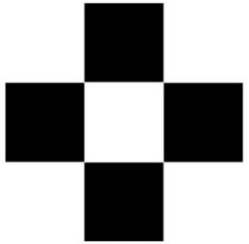


HOG Feature Vector Calculation

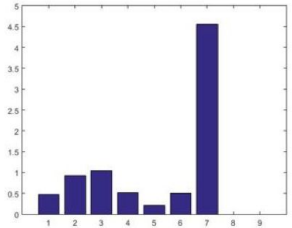





DOING THE SAME PROCESS FOR Multiple points
58
45
3/4
15
1/4
60
60
60
60
30
30
0-20 20-40
30
blw 20 & 40
20
40
30
25
blw 20 and 30
Resulltant Grad. Direction
Grad Magnitude
Grad Direction
Resultant magnitudes
Most of the values 0-180
60
40
105
20
135
0
135
105
45
+
30
+
60
15
+
30
+
60
0
20
40
60
80
100
120
140
160
Feature Vector of size 9
0-180 → Divide into 9 Bins
Feature vector = [... ,135, 105 ...]



(a)



(b)

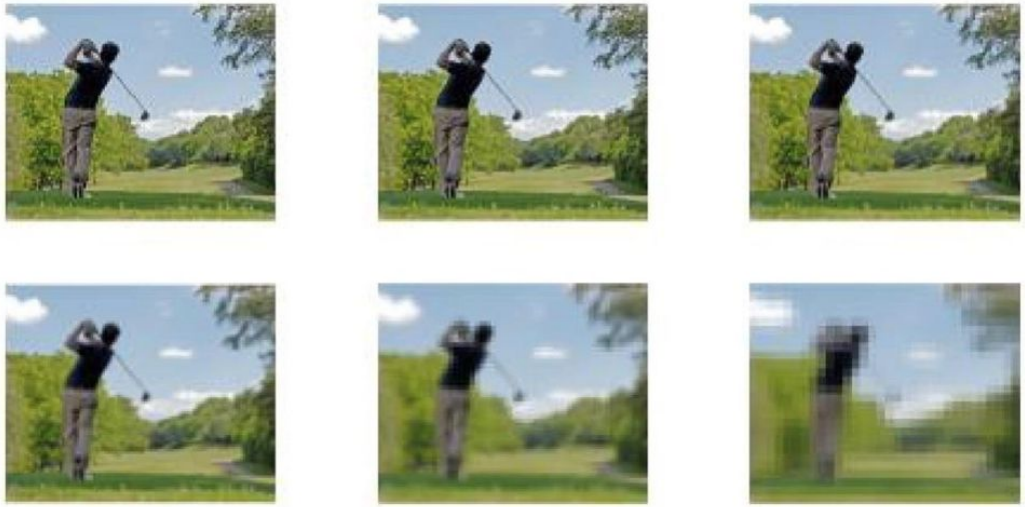


Image1 - HIGH RESOLUTION

Image2 - Eliminating some pixels

Image6 - Until all the pixels are blurred

for every step the SVM (Suuport Vector Machine) , classifies object , as a result there might be multiple copies of the same object. (DISADVANTAGE)

In an attempt to capture all details with different distances from camera location, usually a pyramid of the image is employed

The histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection

HOG FEATURE EXTRACTION

PESU CCBD

Spyder (Python 3.7)

/Users/priyamohata/.spyder-py3

Editor - /Users/priyamohata/.spyder-py3/temp.py

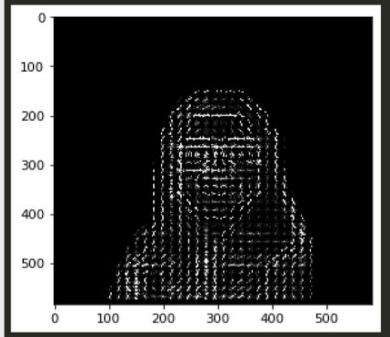
temp.py

```
1 from skimage.io import imread, imshow  #imread - input , imshow - to show the image
2 from skimage.feature import hog # To import the hog feature extraction
3 from skimage import exposure
4
5 img = imread('/Users/priyamohata/PycharmProjects/faceReg2/img.png')  # Read the image
6 imshow(img)
7
8 MC = True  # For color-images
9
10 hogfv, hog_image = hog(img,
11                        orientations=9,  # 9 - point histogram
12                        pixels_per_cell=(16, 16),
13                        cells_per_block=(2, 2),
14                        visualize=True,  # To see the plot
15                        multichannel=MC)
16
17 #Setting the exposure features to show the features properly
18 hog_image_rescaled = exposure.rescale_intensity(hog_image, in_range=(0, 5))
19 imshow(hog_image_rescaled)
20
21
22
```

IPython console

Console 1/A

```
In [7]: runfile('/Users/priyamohata/.spyder-
py3/temp.py', wdir='/Users/
priyamohata/.spyder-py3')
```

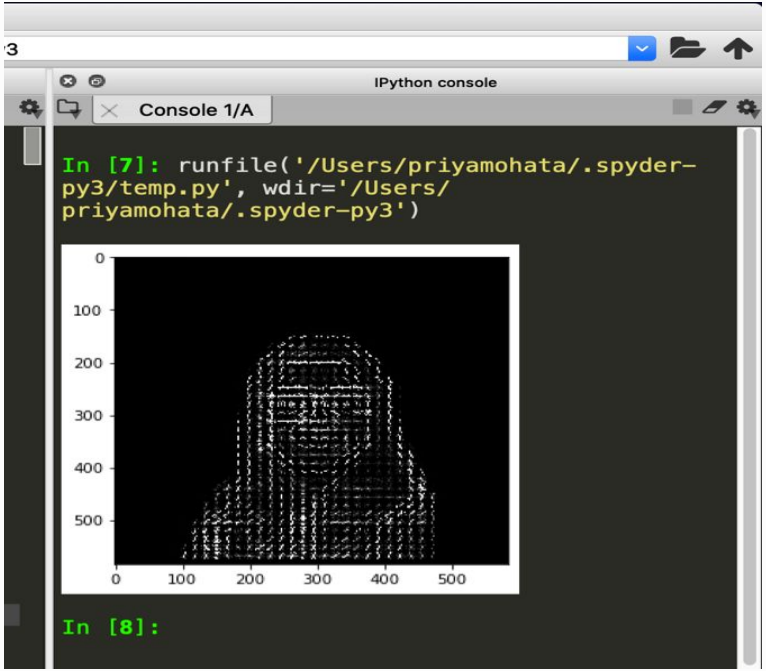



```
In [8]:
```

Run current cell (Ctrl+Enter) [Use #%% to create cells] Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 22 Column: 1 Memory: 72 %

ORIGINAL IMAGE

FEATURE EXTRACTED PLOT

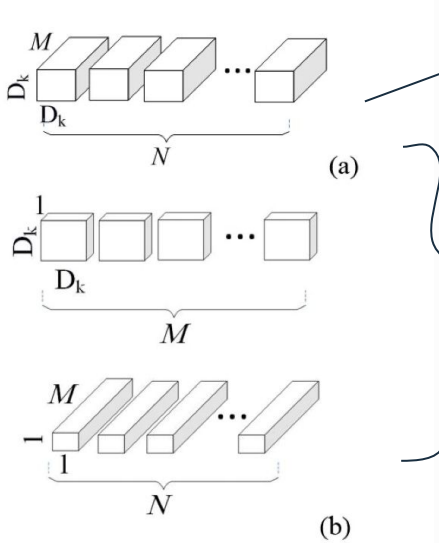


LIGHTWEIGHT CNN

1) CNN'S are used for object detection,however, it is considered as a **challenging task to fit the CNNs into the network edge devices due to the very restrict constraints on resources**

2)Therefore lightweight CNN is designed so that its suitable to edge devices.

3)L-CNN architecture Depthwise Separable Convolution is employed to reduce the computational cost of the CNN itself



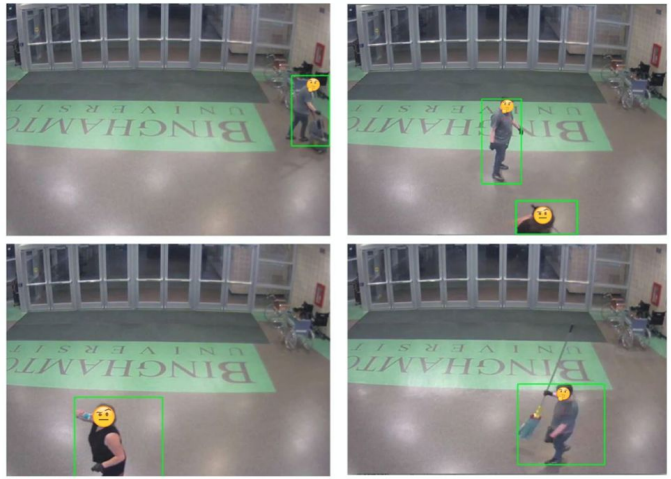
Traditional CNN

Depthwise CNN+Pairwise CNN

1)The proposed L-CNN network architecture has 23 layers considering depthwise and pointwise convolutions as separate layers

2)The resultant neural network classifier identifies the objects within the proposed window, and adds the label for output bounding boxes at the end of the network

3)The first convolutional layer of the L-CNN architecture is a conventional convolution, but in the rest of the network depthwise along with pointwise convolutions are used



RESULTS :

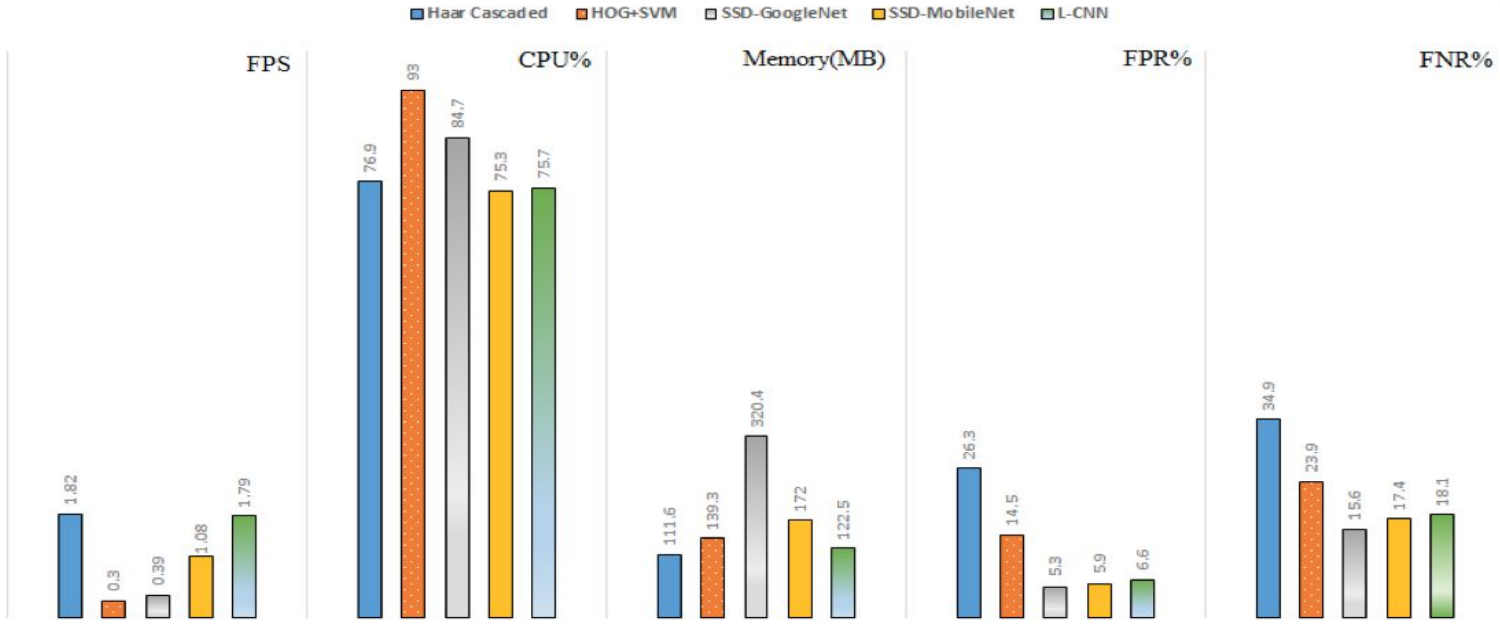


Fig. 8. Performance in FPS, CPU, Memory Utility, Average False Positive Rate (FPR%) and Average False Negative Rate (FNR%)

RESULTS :

Fig. 9. (a)Haar Cascaded. (b)HOG+SVM. (c)SSD-GoogleNet. (d)L-CNN.

Research Paper 2:
Enhancing Face Identification Using Local Binary Patterns and K-Nearest Neighbours

Idelette Laure Kambi Beli * ID and Chunsheng Guo
School of Communication Engineering, Hangzhou Dianzi University,
Xiasha Higher Education Zone,
Hangzhou 310018, China
(Published in the Journal of Imaging)

What was basically proposed?

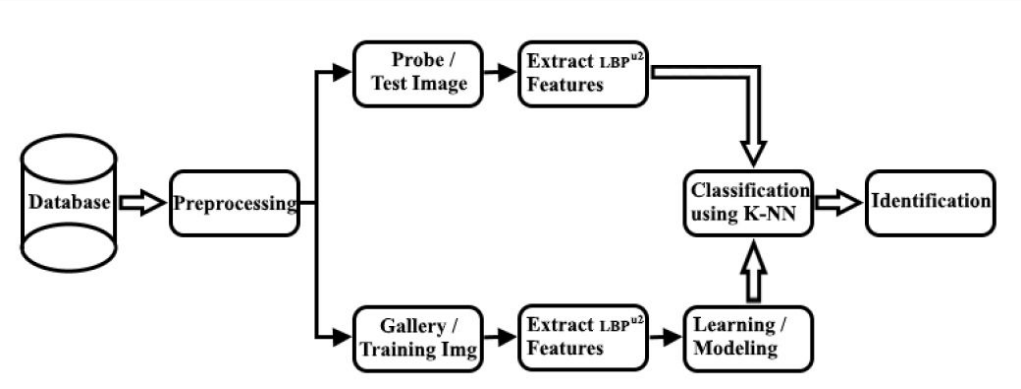
Combining two algorithms-*Local Binary Patterns* for facial feature extractions and *K-NN* for image classifications to provide an approach that contributes to resolve face identification issues with large variations of parameters such as pose, illumination, and expression.

The Approach

Face image divided into grid of small non-overlapping regions, where the global LBP histogram of a particular face image is obtained by combining histogram sequence of each non-overlapping region; explicitly, the global features are collected in single vector and therefore classified using the k-nearest neighbor algorithm.

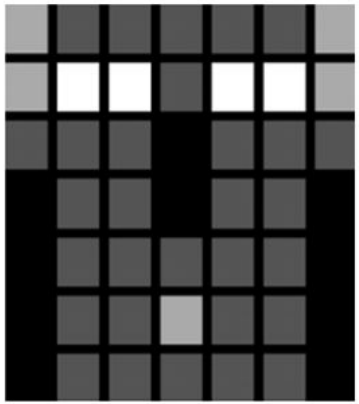
Conclusion

The simulation results indicate that the LBP features and K-NN classifier form a strong base for facial identification on unconstrained environment databases (85.71% on LFW dataset).



Ref:PyImageSeach Article by Adrian Rosebrock

- Given a face in the dataset,the first step of the algorithm is to divide the face into 7×7 equally sized cells:
- for each of these cells, we compute a **Local Binary Pattern** histogram.
- by computing a histogram for each of the cells, we actually are able to encode a level of spatial information such as the eyes, nose, mouth, etc.



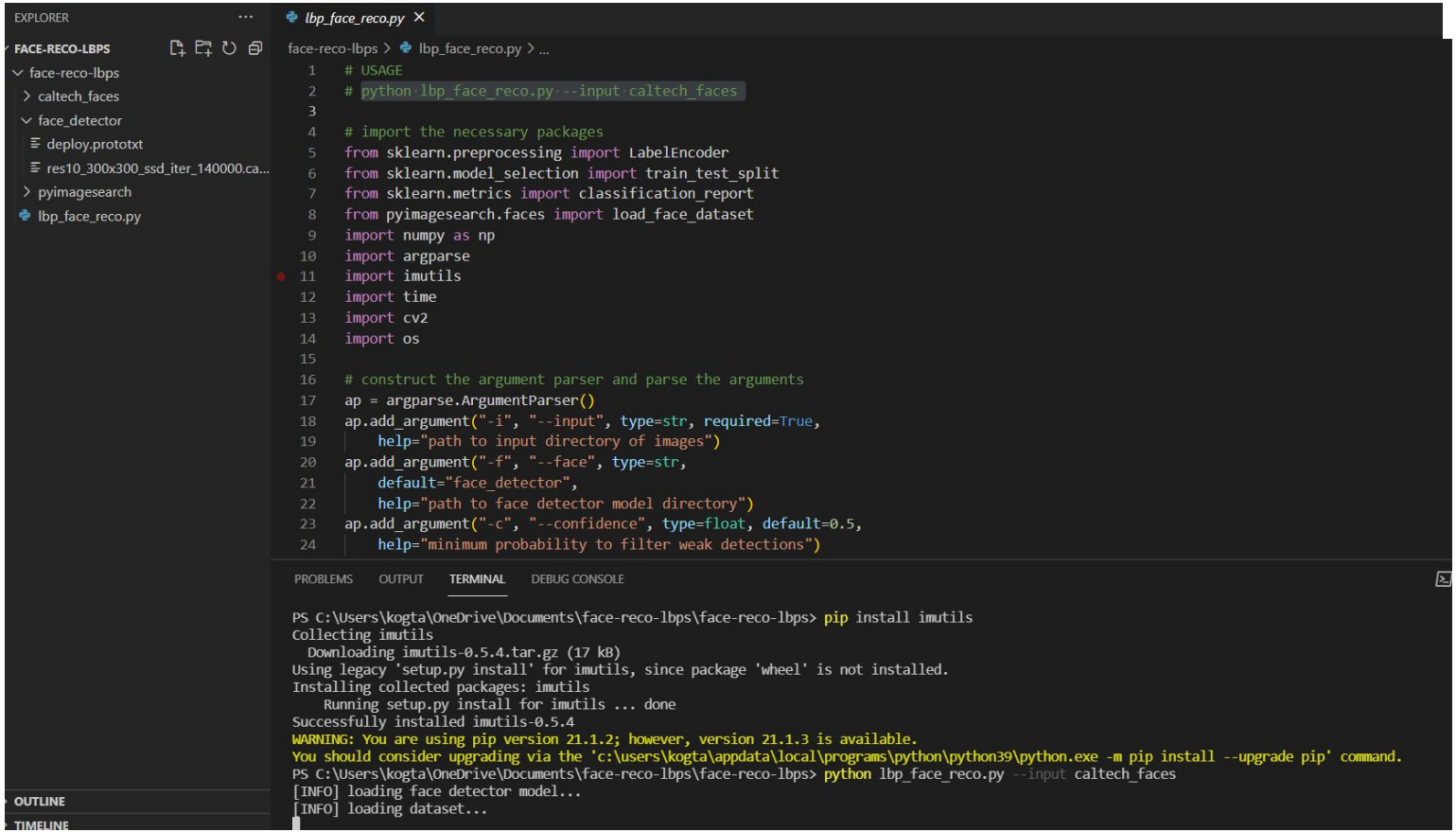
LBP histograms for white cells (such as the eyes) are weighed 4x,for light gray cells (mouth and ears) 2x for dark grey cells(inner cheeks and forehead) 1x and for nose and outer cheek 0x.
the weighted 7×7 LBP histograms are concatenated together to form the final feature vector

Performing face recognition is done using the χ^2 distance and a nearest neighbor classifier. So in summary:

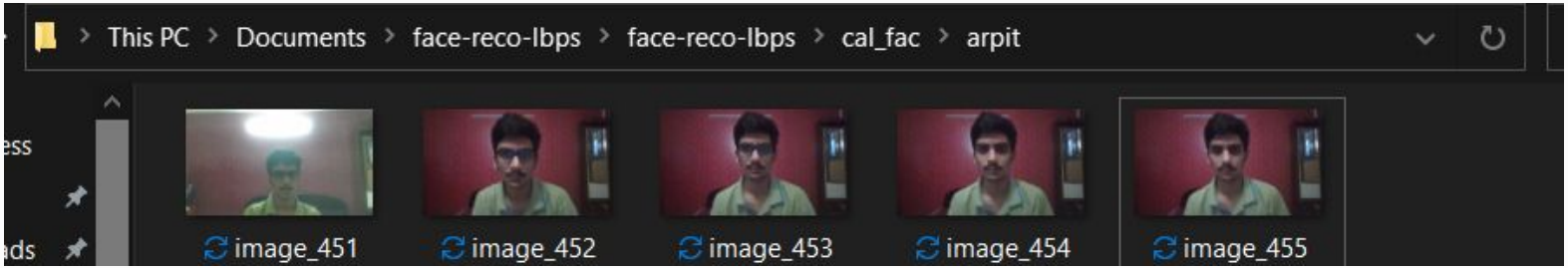
χ^2

- A face is presented to the system
- LBPs are extracted, weighted, and concatenated in the same manner as the training data
- k-NN (with *k=1*) is performed with the distance to find the closest face in the training data.
- The name of the person associated with the face with the smallest distance is chosen as the final classification

LBP Implementation



LBP Implementation-The CALTECH faces dataset(450 images from 27 different people)




```
[INFO] prediction: mae, actual: mae, confidence: 160.92
[INFO] prediction: abraham, actual: abraham, confidence: 171.67
[INFO] prediction: allen, actual: allen, confidence: 172.83
[INFO] prediction: allen, actual: allen, confidence: 176.73
[INFO] prediction: abraham, actual: abraham, confidence: 163.09
[INFO] prediction: arpit, actual: arpit, confidence: 0.00
```

Face detection model

The `face_detector` directory contains our **OpenCV deep learning-based face detector**. This detector is both fast and accurate, capable of running in real-time without a GPU.

`faces.py`, lives in the `pyimagesearch` module. This file contains two functions:

1. `detect_faces`: Applies our face detector to a given image, returning the bounding box coordinates of the face(s)
2. `load_face_dataset`: Loops over all images in `caltech_faces` and applies the `detect_faces` function to each

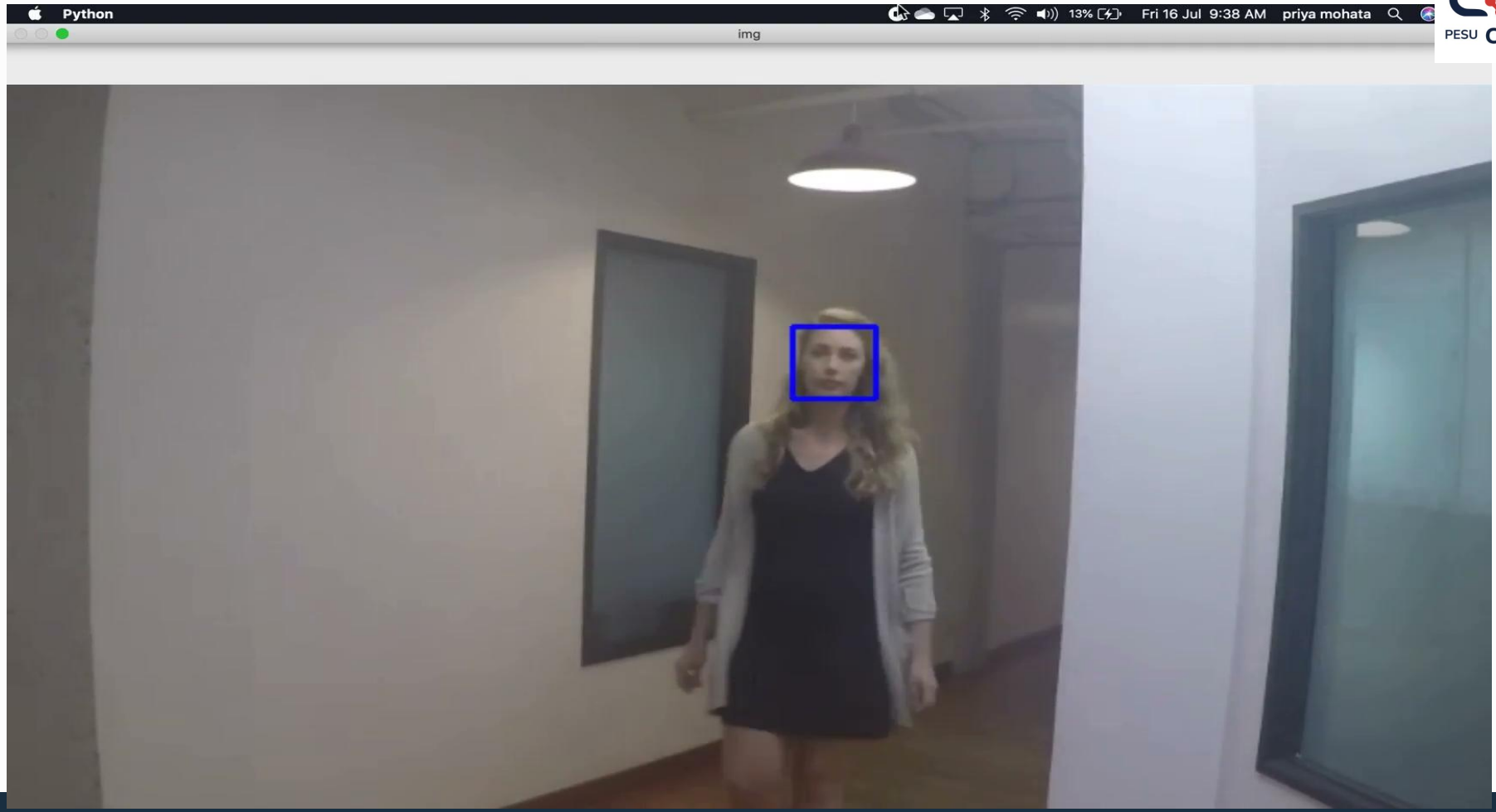
PES University
Cloud Computing and Big Data

CCBD
STATUS REPORT 4

MENTOR : D.R.HL PHALACHANDRA

PRIYA MOHATA	PES2UG19CS301
PRIYANSH JAIN	PES2UG19CS303
ARPIT KOGTA	PES2UG19CS065
SNEHIL JAIN	PES2UG19CS396

- Haar Cascade can be used to detect the faces at the edge , since its performance and ability to detect faces is good.
- Haar cascade can be used to extract the faces at the edge and then we can send the images to the server for further processing and tracking if required.



PESU CCBD

HAVE PATIENCE,
THERE IS A BEAUTIFUL PLAN FOR
YOU.

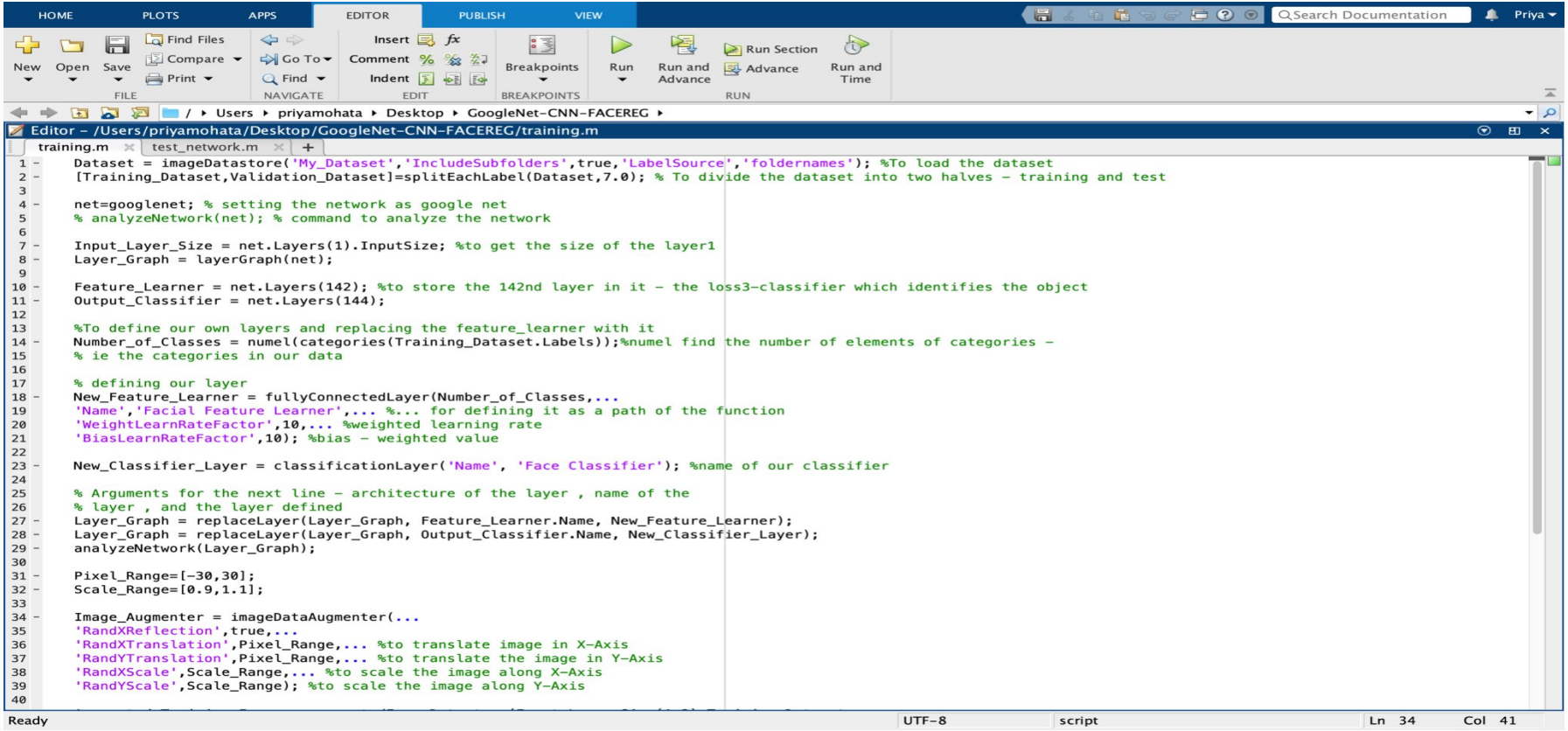
GoogleNet CNN

- The CNN has 144 layers
- The dataset comprises of images of Einstein , Elon Musk , Stephen Hawking and Tom Cruise.
- We extract the 142nd layer here and replace it, because in this layer , the training of the network is done with the images , for example if we want to detect cars then we will place car images here so our network trains on that.
- The image must be of 224*224 pixels.
- The images must have three layer format - R,G,B
- The tuning of the images is done in the code.
- The 142nd layer is replaced with the layer defined in the code , that recogonises images.
- After defining the layer , we train the network with 6 epoch training cycles.
- The accuracy achieved was 81.82%.

The number of epochs is a hyperparameter that defines the number times that the learning algorithm will work through the entire training dataset. One epoch means that each sample in the training dataset has had an opportunity to update the internal model parameters.

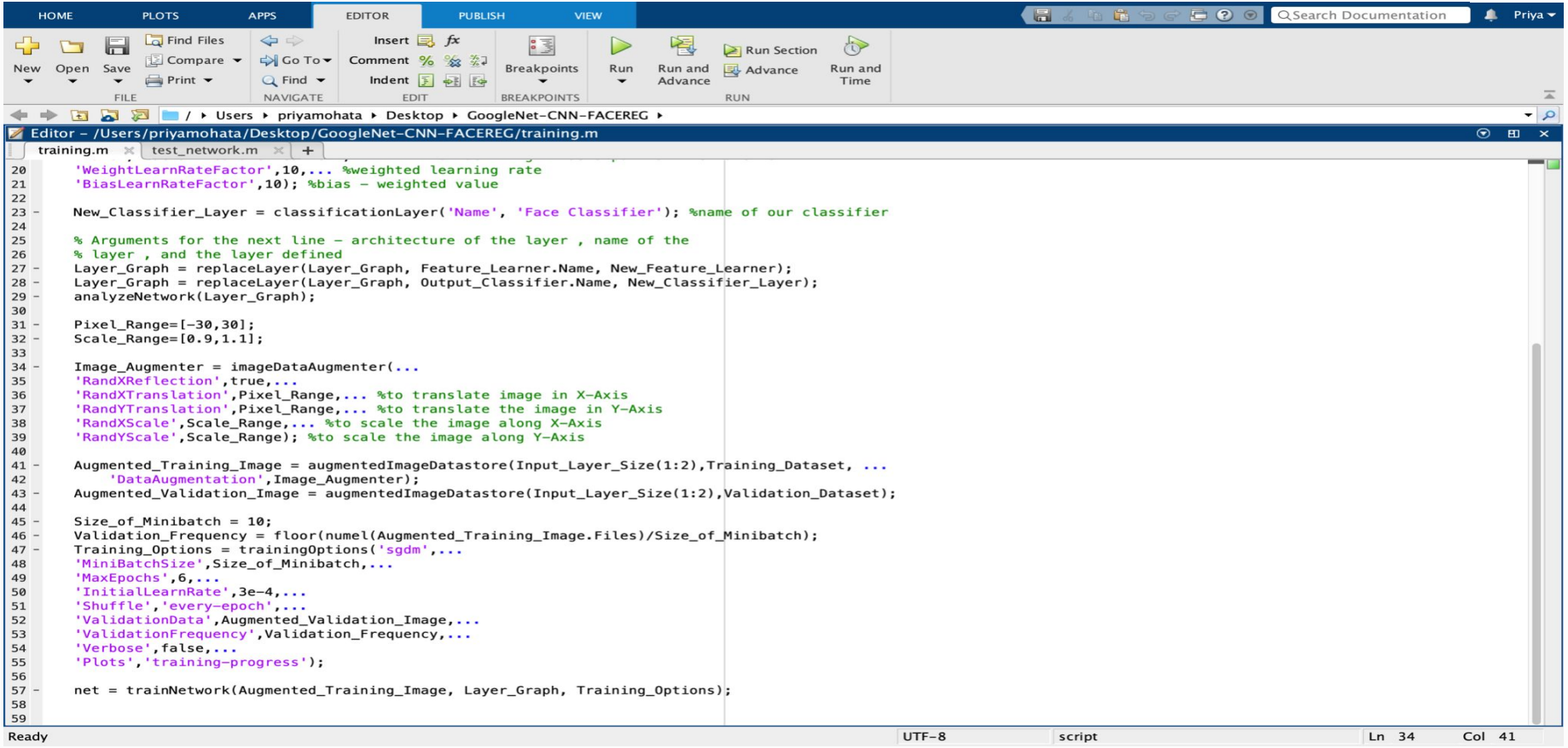
GoogleNet CNN

PESU CCBD

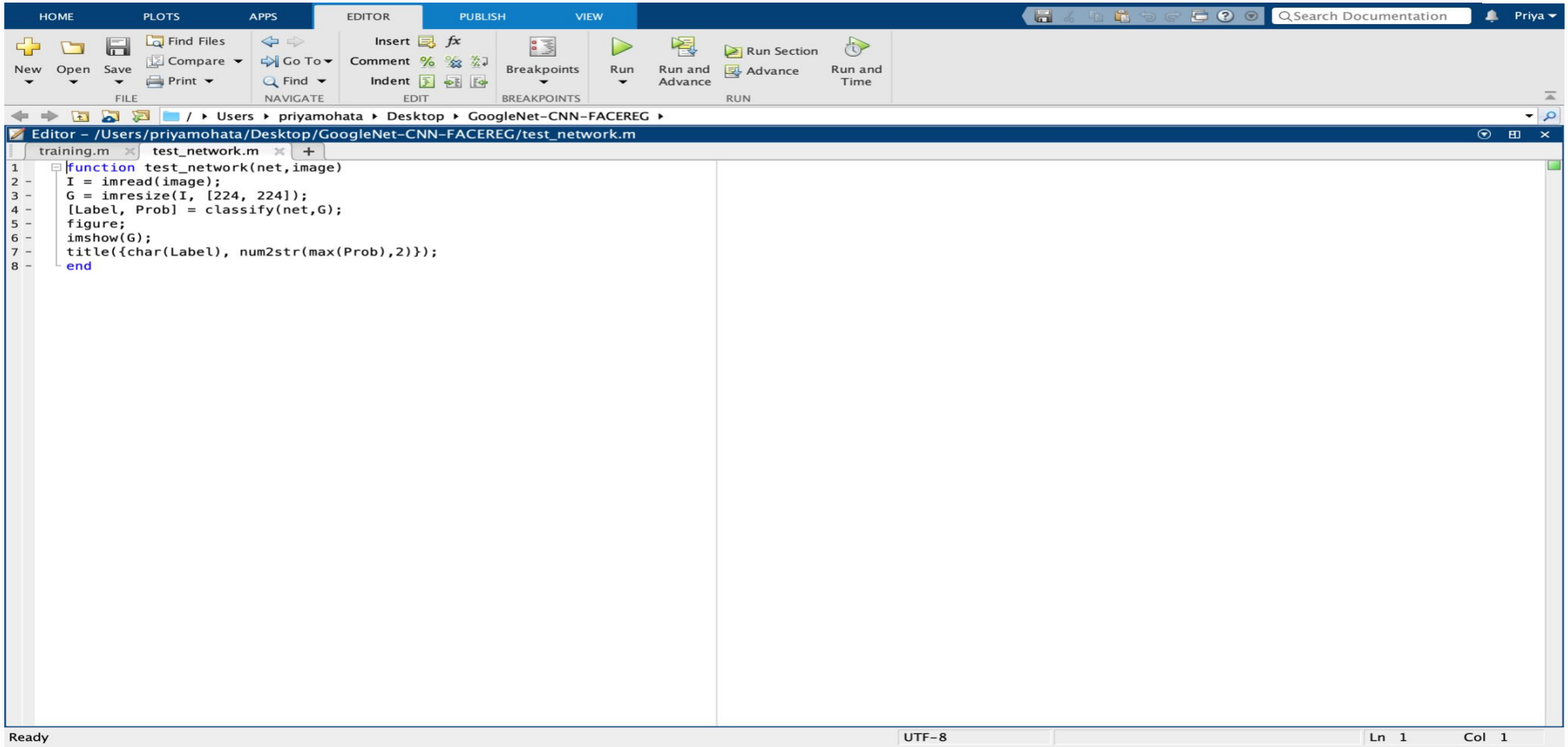


GoogleNet CNN

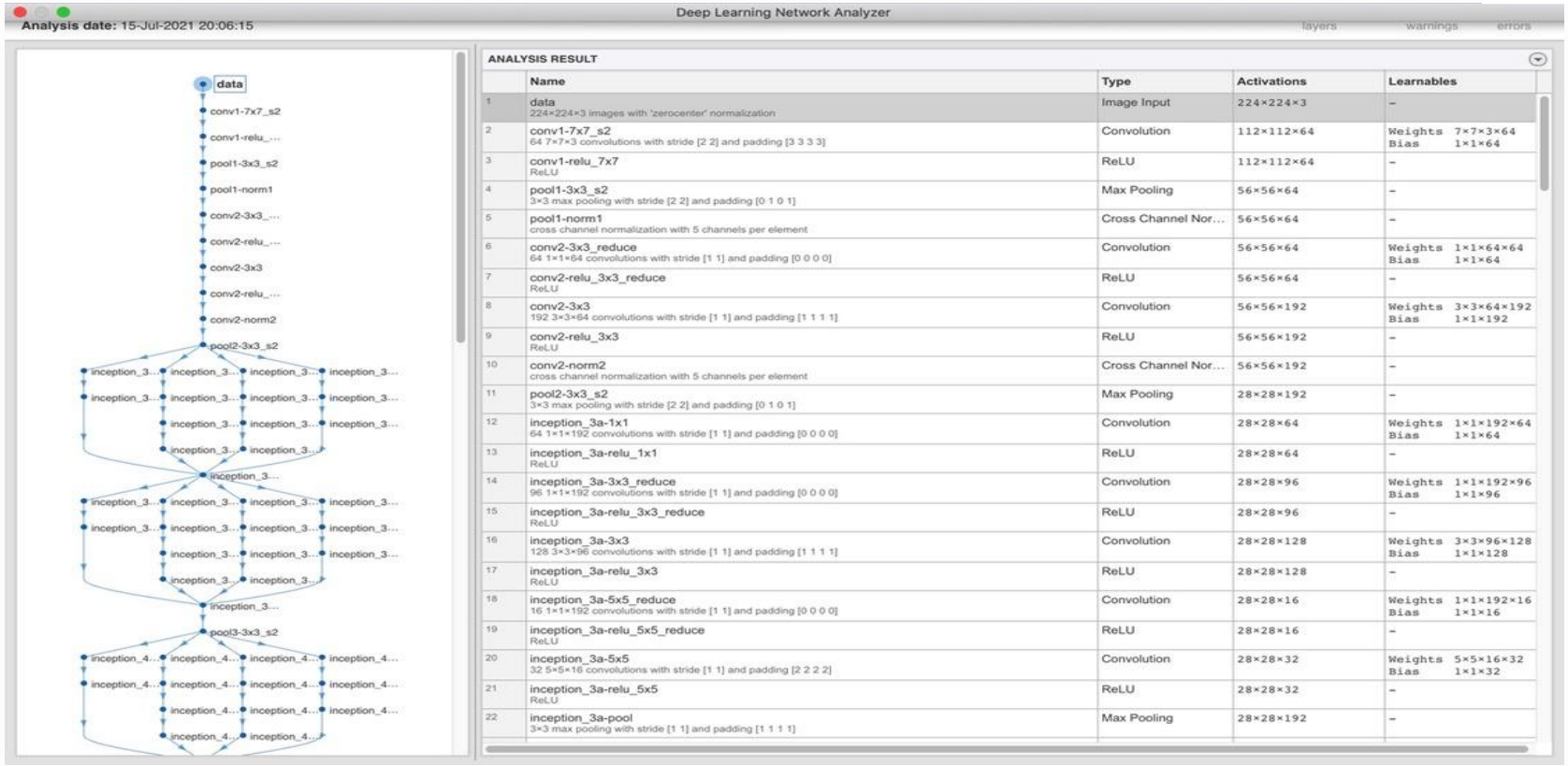
PESU CCBD



GoogleNet CNN



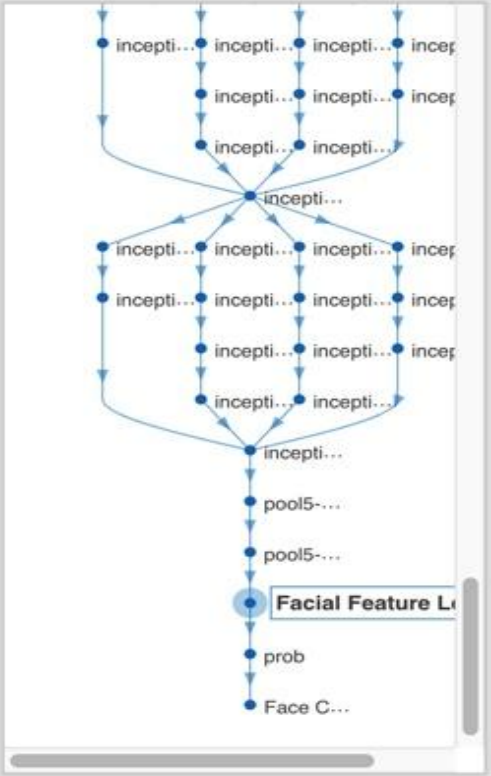
GoogleNet CNN



Layer_Graph

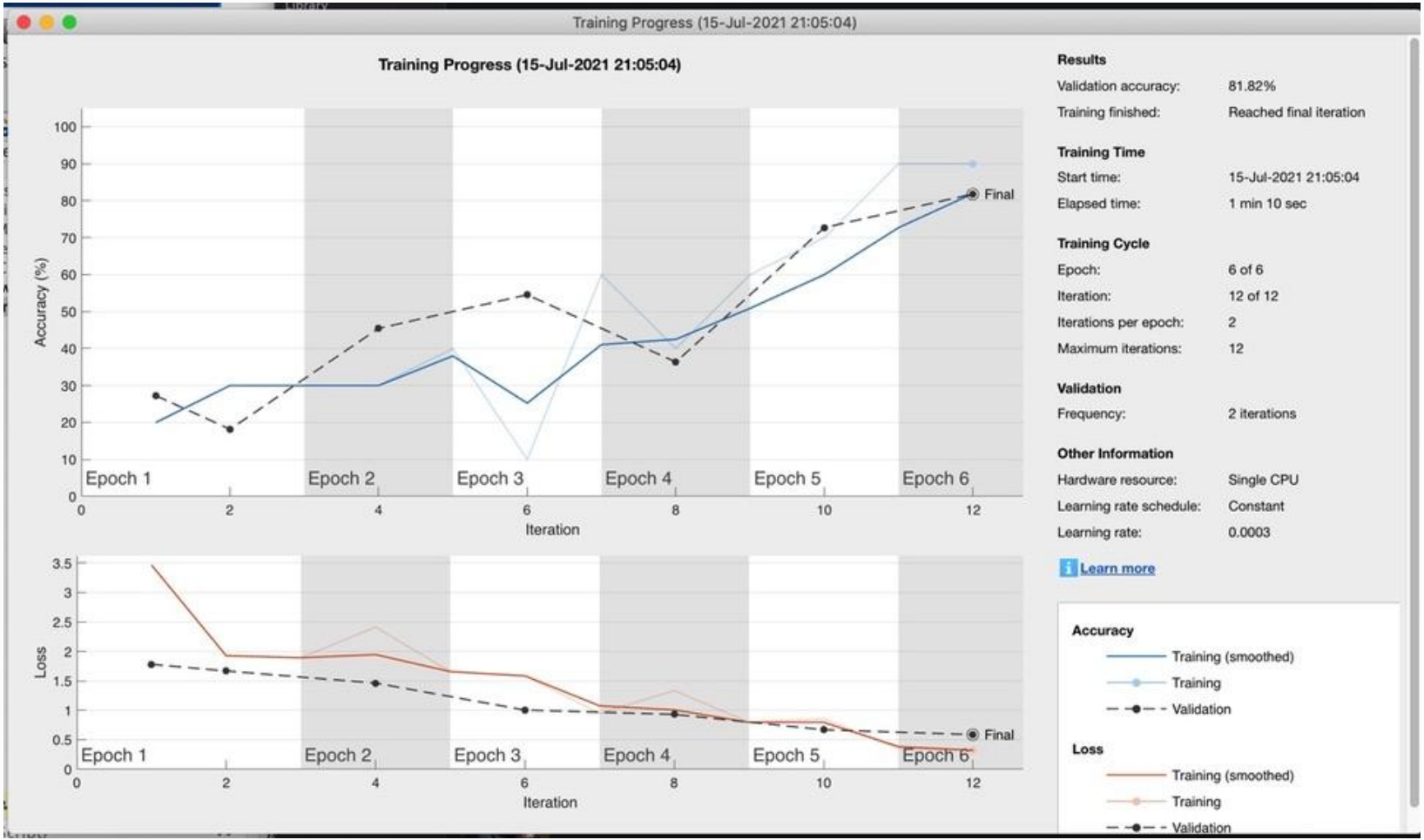
Analysis date: 15-Jul-2021 20:25:27

144 layers | 0 warnings | 0 errors

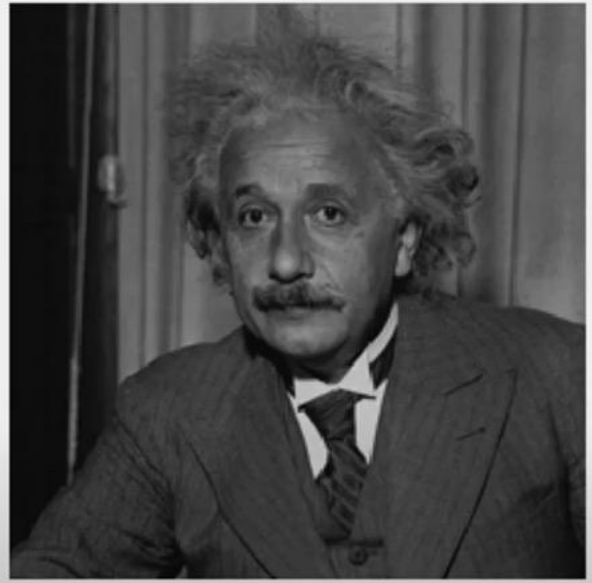


ANALYSIS RESULT

	Name	Type	Activations	Learnables
	ReLU			
134	inception_5b-5x5 128 5×5×48 convolutio...	Convolution	7×7×128	Weights 5×5×48×128 Bias 1×1×128
135	inception_5b-relu_5... ReLU	ReLU	7×7×128	-
136	inception_5b-pool 3×3 max pooling with st...	Max Pooling	7×7×832	-
137	inception_5b-pool_... 128 1×1×832 convoluti...	Convolution	7×7×128	Weigh... 1×1×832×1... Bias 1×1×128
138	inception_5b-relu_p... ReLU	ReLU	7×7×128	-
139	inception_5b-output Depth concatenation of ...	Depth concatenation	7×7×1024	-
140	pool5-7x7_s1 Global average pooling	Global Average Po...	1×1×1024	-
141	pool5-drop_7x7_s1 40% dropout	Dropout	1×1×1024	-
142	Facial Feature Lear... 4 fully connected layer	Fully Connected	1×1×4	Weights 4×1024 Bias 4×1
143	prob softmax	Softmax	1×1×4	-
144	Face Classifier crossentropyex	Classification Output	1×1×4	-


Training Progress (15-Jul-2021 21:05:04)
Training Progress (15-Jul-2021 21:05:04)
Accuracy (%)
Iteration
Epoch 1
Epoch 2
Epoch 3
Epoch 4
Epoch 5
Epoch 6
Final
Loss
Iteration
Epoch 1
Epoch 2
Epoch 3
Epoch 4
Epoch 5
Epoch 6
Final
Results
Validation accuracy: 81.82%
Training finished: Reached final iteration
Training Time
Start time: 15-Jul-2021 21:05:04
Elapsed time: 1 min 10 sec
Training Cycle
Epoch: 6 of 6
Iteration: 12 of 12
Iterations per epoch: 2
Maximum iterations: 12
Validation
Frequency: 2 iterations
Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.0003
Learn more
Accuracy
Training (smoothed)
Training
Validation
Loss
Training (smoothed)
Training
Validation

Einstein
0.98



THANK YOU !