# Smart Cloud-Edge Video Surveillance System

Mahmoud G. Ismail
*Media Engineering and Technology*
*German University in Cairo*
Cairo, Egypt
mahmoud.gamal@alumni2020.guc.edu.eg

Fakhreldin H. Tarabay
*Media Engineering and Technology*
*German University in Cairo*
Cairo, Egypt
fakhreldin.tarabay@student.guc.edu.eg

Ramez El-Masry
*Media Engineering and Technology*
*German University in Cairo*
Cairo, Egypt
ramez.elmasry@student.guc.edu.eg

Mohamed Abd El Ghany
*Electronics Department*
*German University in Cairo*
Cairo, Egypt
*Integrated Electronic Systems Lab*
*TU Darmstadt*
Darmstadt, Germany
mohamed.abdel-ghany@guc.edu.eg

Mohammed A.-M. Salem
*Media Engineering and Technology*
*German University in Cairo*
Cairo, Egypt
mohammed.salem@guc.edu.eg

***Abstract*—As the world advances it becomes increasingly technology-dependent, bringing together infrastructure and technology to improve the quality of life for the citizens. Smart cities have become the future of urbanization. Since the priority of a city is to protect its citizens, a video surveillance system is required to ensure their safety. This paper proposes a multi-camera cloud-Edge surveillance system for smart cities and homes. Multiple units of Raspberry Pi act as the Edge Computing device that streams and summarizes the processed video footage. After summarizing the video to reduce its length and size, it sends the videos to the cloud (virtual machine). The cloud applies resource-intensive computer vision algorithms such as detecting motion, objects including humans, weapons, and fire. Furthermore, it manages the recorded surveillance videos, stores them in the database, and alerts the user if a threat occurs. The experimental results show that the time taken to perform these tasks was reduced by an average of 83% for the object detection models.**

***Index Terms*—Edge Computing, Video Summarization, Video Surveillance, Raspberry Pi, IoT, Machine Learning, Deep Learning, Object Detection.**

## I. Introduction

A smart city is a technologically modern urban area that uses electronic methods, voice activation methods, and sensors to collect specific data. The collected data is employed to manage assets, resources, and services efficiently. One such infrastructure is the security system, which is necessary to ensure safety and comfort for the cities' citizens. However, surveillance cameras record 24-hour live streams, which leads to a large amount of redundant data.

Conventional video surveillance systems consist of several cameras that act as video recorders, which record a high amount of surveillance footage and save that footage onto mass storage devices, footage that needs to be further analyzed by humans to detect and react to potential threats. However, for a camera operator to look into the camera footage every day for multiple cameras in a place full of people is impractical. Which makes it a costly process, and there is no guarantee that they won't miss any details. Ideally, we want to detect and react to threats in real-time or as soon as possible.

Another problem is that the extensive amount of data takes a lot of storage space, which proved to be very costly in terms of needed hardware. The final issue we are trying to address is that the current conventional security systems only act as eyes that record the crime or incident but lack the brain to analyze the footage. Consequently, these systems do not offer real-time security, so what is needed is a security system that takes advantage of recent advancements in technology to deliver security that offers real-time threat detection and alerting responsible parties.

## II. Literature Review

Video surveillance and video analysis constitute active areas of research. Currently, there is a wide range of video surveillance systems that have been implemented to address a wide array of problems.

[1] presented a heuristic architecture model for an on-board video surveillance system, which addressed the need for video surveillance in Transportation (Busses). The system consisted of a cloud server and Raspberry Pi boards as the Edge Nodes. [2] proposes a privacy-preserving real-time social distancing breach detection system, The system uses a combination of people detection and tracking algorithms to identify social distancing breaches. To ensure privacy they used a federated learning approach to allow computation on edge devices.

[3] developed a fog computing infrastructure, which uses the deep learning models to process the video feed generated by the surveillance cameras. [4] presents a framework to recognize abnormal activities of students during exams, such as copies of the answers from hidden sources or peeking at others' answers papers. The main features to identify the suspicious behaviors are Head, Iris, and Hand movements captured on the video surveillance cameras.

[5] proposed an Internet of Medical Things system that uses deep learning to detect diversified types of COVID-

19 symptoms and generate reports and alerts that can be used for medical decision support. [6] addressed the issue of network bandwidth and the need for quick response in medical applications by using Edge Computing. The proposed system incorporates background subtraction and deep convolution neural network algorithms on moving objects to detect and classify abnormal falling activity on the network's edge.

[7] proposed a video surveillance-based system for improving road safety, by detecting various traffic pre-events from traffic videos, such as speed violations, one-way traffic, overtaking, illegal parking, and wrong drop-off location of passengers. [8] adopts the collaborative Cloud-Edge architecture to analyze surveillance video and extract video keyframes for compressing video data at the edge.

## III. Smart Cloud-Edge Video Surveillance System

The system proposed in this paper is a smart cloud-based surveillance system enhanced with a video summarization algorithm that can work in real-time. As shown in Fig. 1, the system has three main sections, which are the Edge section, the central section, and the cloud section. The Edge section consists of multiple cameras connected to Raspberry Pis. The Raspberry Pi records the footage captured through the camera, summarizes them, then sends the recordings to the middle section consisting of a Main Raspberry Pi. The Main RP is responsible for compressing the received videos. Finally, the cloud is responsible for storing the recorded data and running resource-intensive algorithms such as humans, fire, and weapons detection. In case of any irregularity, the cloud will alert the users.

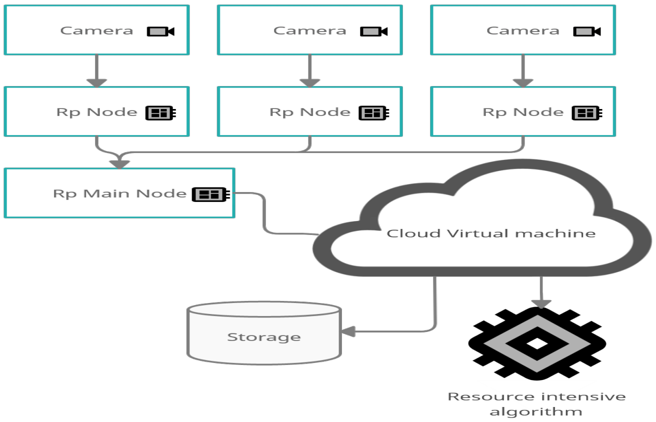

Fig. 1. Video Surveillance System Diagram

### A. Video Summarization using Deep-Learning

In this section, we provide an overview of our approach in video summarization using Deep Learning. As shown in Figure 2, the algorithm takes an input video from the Raspberry Pi, separates it into frames, and preprocesses it to remove noise. After that, the preprocessed frames are passed to the shot segmentation and image memorability score prediction.

*1) Shot Segmentation:* Feature extraction is a dimensionality reduction process where an initial set of the raw data is divided and reduced into more manageable groups, eliminating redundant data and making the processing stage faster and more accurate. In our implementation, MobileNetV2 [9] was responsible for this stage, where we divided the video into segments. As shown in Figure 3, the expansion layer acts as a decompressor that first restores the data to its complete form. After that, the depthwise layer performs the filtering process. Finally, the projection layer compresses the data.

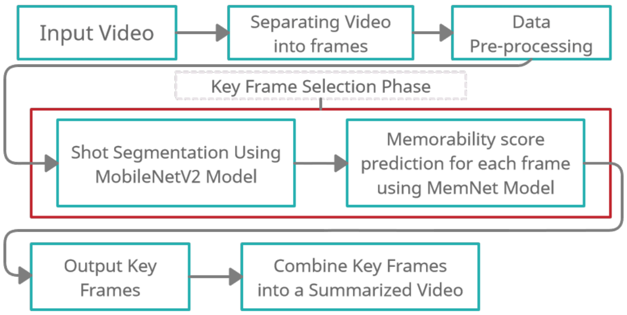
Input Video
Separating Video into frames
Data Pre-processing
Key Frame Selection Phase
Shot Segmentation Using MobileNetV2 Model
Memorability score prediction for each frame using MemNet Model
Output Key Frames
Combine Key Frames into a Summarized Video

Fig. 2. Flowchart for the Deep Learning Video Summarization.

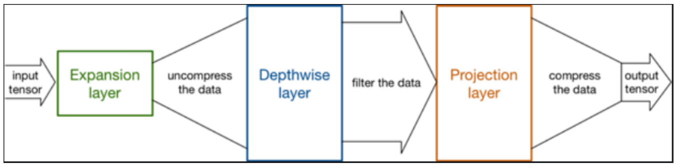

Fig. 3. MobileNetV2 Filter Design. Source [10]

*2) Key Frame Extraction:* In Key Frame Extraction, the MemNet model was used to score each frame's memorability. A higher memorability decides the frames considered in the final summarized output video. The model classifies different images with the same level of memorability under the same label. Furthermore, the model was retrained on LaMem data set [11] which is a large data set of 60,000 images specifically made for this model.

### B. Video Summarization using Mixture of Gaussian (MoG)

In this section, we explain the video summarization using Mixture of Gaussian. The algorithms had three main phases starting from Frame Pre-processing, followed by background modeling and subtraction, and finally, post-processing.

*1) Frame Pre-processing:* In this phase, the input data are cleaned from noise and other unneeded features depending on the application. In our video summarization algorithm, a simple Gaussian filter followed by a gray-scale transformation was enough as pre-processing. The Gaussian filter removed noise while the gray-scale transformation enhanced the performance of the background modeling algorithm and reduced the required processing time.

*2) Background Modelling and Subtraction:* Background subtraction is a way of eliminating the background from an image. The Mixture of Gaussian(MoG) algorithm performed this task. the MoG algorithm was proposed in [12] and later improved in [13]. the advantage of this algorithm is that it uses multiple Gaussian distributions to classify the image into background and foreground. Furthermore, the number of

Gaussian distributions is adaptive depending on the scenes captured, providing better adaptability to varying scenes due to illumination changes and other factors.

*3) Post-processing:* The image obtained initially will require some processing steps to enhance its detection. The post-processing methods used consisted of Thresholding and gaussian blur. After that, if the motion area exceeds 0.05% of the frame dimensions, the frame is determined to contain motion. The frames with motion are saved, while those without motion are discarded.

### *C. Cloud-based surveillance system*

This section is a general overview of the cloud-based surveillance system and its components. The system runs on a virtual machine and consists of a ReactJS website, two servers with different technologies (Flask, NodeJS) to handle the main algorithm and the website, a MongoDB database, and other cloud services.

*1) Cloud computing and capabilities:* A Microsoft AZURE virtual machine handles the resource-intensive system algorithm. A virtual machine provides security, convenience, accessibility, security, and ease of use. The created machine had virtual CPUs, a large RAM, and an Nvidia GPU, which is crucial since hardware acceleration is only applicable on Nvidia GPUs. The other cloud services used are Google's Gmail and google drive APIs. The Gmail API allowed sending email alerts to the users. Finally, Google Drive API allows users to upload relevant recorded videos or images taken by the system to their drive.

*2) Object detection:* To implement object detection, Yolov4 [14] is used because it is the most suitable for real-time applications. We trained multiple custom models with YoloV4 and YoloV4-tiny to detect Humans, guns, rifles, and Fire. YoloV4 had higher accuracy but required more processing time, while the YoloV4-tiny had lower accuracy and shorter processing time.

*3) User Interface:* The users' access to the system was through a specifically created Website. The system was accessible from any PC with an internet connection. The users can add or delete camera connections, watch live streams, enable or disable features, and view recorded footage. Finally, A database stored information such as credentials / Email and Settings / Camera connections.

## IV. RESULTS

### *A. Hardware Specification*

These tests were done on a laptop with the following specs: Processor: Intel(R) Core(TM) i7-7500 CPU @2.70GHZ 2.90GHz. Ram: 16GB. GPU: NVIDIA GeForce 940MX 4GB. Operating System: Windows 10.

### *B. Datasets*

*1) The Virat Dataset:* Virat Dataset [15] is a large-scale video data-set designed to assess the performance of visual event recognition algorithms, with a focus on continuous event recognition in outdoor regions. The data-set consists of many outdoor scenes with actions occurring naturally by non-actors in continuously captured videos. We chose two videos from the data-set, which were VIRAT_S_000001(V1) and VIRAT_S_000002(V2). Due to their relatively higher duration and relevance to the task of surveillance video summarization. Those videos showed a parking lot with people talking together with cars moving in and out of the parking lot.

*2) Private Dataset:* The private data-set used was obtained from a local grocery store. This data-set contained the records of four cameras over one week.

The advantage of this data-set is that each surveillance video had a duration ranging from 1 hour 50 minutes to 2 hours 20 minutes. Furthermore, all videos had a size of 0.99 GB, set by the DVR, which allows the summarization results to be more noticeable, unlike the short duration and smaller size of other public data-sets data-set. The data-set will be made public after processing it and blurring the faces of the people shown on the videos to preserve their privacy.

*a) 100H Test Results:* In this test case, we choose random videos from the private dataset to assemble a 100 hours surveillance record. as shown in Table. I, the tough summarization reduced the video duration to 2.4% and video size by 0.7% of the original.

TABLE I
PRIVATE DATASET 100H SUMMARIZATION RESULT

| | **100H Test Results** | |
|---|---|---|
| | ***Original*** | ***Tough Summarized*** |
| Duration | 100:01:46 | 2:26:25 |
| Size | 41,656 MB | 310 MB |
| Color | RGB | GrayScale |
| No. Frames | 4,321,278 | 105,430 |

### *C. Summarization Algorithm Results*

*1) Mixture Of Gaussian:* The results of summarizing the V1 video are shown in Table II. The summarized version of the video is 7.05% of the duration of the original, while its size is 1.10% of the original. On the other hand, the tough summarized video has a duration and size of 3.5% and 0.11% of the original video respectively.

TABLE II
V1 SUMMARIZATION RESULTS

| | **V1** | | |
|---|---|---|---|
| | ***Original*** | ***Summarized*** | ***Tough Summarized*** |
| Duration | 0:11:29 | 0:01:33 | 0:00:46 |
| Size | 1388 MB | 15.3 MB | 1.46 MB |
| Color | RGB | RGB | Gray Scale |
| No. Frames | 20,655 | 2,810 | 1,405 |

As for the summarization results of the V2 video presented in Table III. The summarized version of the video is 18.22% of the original's duration, while its size is 1.5% of the original. The tough summarized video has a length and size of 9.11% and 0.15% of the original video respectively.

*2) DL Summarization:* As shown in Table IV using the Deep Learning Summarization on the two videos mentioned before can decrease the total video length for video V1 to 4.2% and the video size to 0.25% of the original. While the length of video V2 got reduced to 15.91%, and its size to 1.11%.

TABLE III
V2 Summarization Results

| | V2 | | |
|---|---|---|---|
| | *Original* | *Summarized* | *Tough Summarized* |
| Duration | 0:05:02 | 0:00:55 | 0:00:27 |
| Size | 612 MB | 9.16 MB | 0.9 MB |
| Color | RGB | RGB | Gray Scale |
| No. Frames | 9,075 | 1,654 | 827 |

TABLE IV
Results of DL-Summarization on VIRAT dataset.

| **Video Properties** | **VIRAT Dataset** | | | |
|---|---|---|---|---|
| | **V1** | **SUMM1** | **V2** | **SUMM2** |
| Fps | 30 | 30 | 30 | 30 |
| Color Format | YUV | YUV | YUV | YUV |
| Video Format | MP4 | MP4 | MP4 | MP4 |
| Video size | 1388MB | 3.52MB | 612MB | 6.81MB |
| Video duration | 11:29 | 00:29 | 05:02 | 00:49 |
| Number of frames | 20655 | 854 | 9075 | 1444 |

### *D. Summarization effect on Object Detection*

Table.V shows the details of the original video, followed by the results of running object detection on it and the summaries. Each row contains the processing time for the YoloV4-tiny model/processing time for the yolov4 model.

The MoG summarization process reduced the time required for object detection by 89.1% on average in the case of tough summarization on the YoloV4 model and 90.5% for the YoloV4-tiny model. On the other hand, the normal MoG summarization reduced the processing time by 77.2% for the YoloV4 model and 73.03% for the YoloV4-tiny model. Finally, the DL approach reduced the required processing time by 90.04% for the YoloV4 model and 87.7% for the YoloV4-tiny model.

TABLE V
Summarization effects on Object Detection.

| | **V1 Processing Time**<br>V4 / V4-Tiny | **V2 Processing Time**<br>(V4 / V4-Tiny) |
|---|---|---|
| **Without Summ** | **33.9m / 119.8m** | **12.29m / 52.2m** |
| **MoG Tough Summ** | **2.49m / 11.11m** | **1.43m / 6.51m** |
| **MoG Summ** | **6.7Mins / 23.45Mins** | **4.2m / 13.56m** |
| **DL Summ** | **1.15Mins / 4.57Mins** | **2.6m / 8.4m** |

### *E. Comparison*

As shown in Tables II - IV, the results of the DL algorithm are much better for summarizing a pre-recorded video, but that comes at the cost of needing more time to run the algorithm, which is about 4-5 times the required time for the MoG approach. On the other hand, although the Mixture of Gaussian algorithm has a lower summarization percentage and effect on the object detection processing time, it compensates by having a shorter processing time for the pre-recorded videos. Furthermore, it has the advantage of running in real-time at 15FPS on Raspberry Pi, which is why in our opinion is more compatible with this application.

## V. Conclusion

In conclusion, we propose a video surveillance system that utilizes both the cloud and edge components. The Raspberry Pis would record the footage, summarize and compress it on the edge of the network. The virtual machine will run resource-intensive tasks such as humans, guns, rifles, and fire detection on the summarized videos on the cloud. On average, the time to perform these tasks got reduced by 83.15% for the YoloV4 object detection model and 83.765% for the yoloV4-tiny object detection model.

Replacing Raspberry pi with other devices specialized for edge computing such as Intel Movidius USB, Nvidia Jetson, and Google Coral should enhance the performance of the edge nodes. Using a virtual machine with higher processing power is another viable enhancement, allowing the object detection process to run in real-time on the stream received from the Edge Node instead of waiting for the complete video to be uploaded.

## References

[1] I. Quintana-Ramirez, L. Sequeira and J. Ruiz-Mas, ”An Edge-Cloud Approach for Video Surveillance in Public Transport Vehicles,” in IEEE Latin America Transactions, vol. 19, no. 10, pp. 1763-1771, Oct. 2021, doi: 10.1109/TLA.2021.9477277.

[2] Sugianto, Nehemia, et al. ”Privacy-preserving AI-enabled video surveillance for social distancing: Responsible design and deployment for public spaces.” Information Technology & People (2021).

[3] Natesha, B. V., & Guddeti, R. M. R. (2021). Fog-based video surveillance system for smart city applications. In Evolution in Computational Intelligence (pp. 747-754). Springer, Singapore.

[4] Alairaji, Roa'A. M., Ibtisam A. Aljazaery, and Haider TH ALRikabi. ”Abnormal Behavior Detection of Students in the Examination Hall from Surveillance Videos.” Advanced Computational Paradigms and Hybrid Intelligent Computing. Springer, Singapore, 2022. 113-125..

[5] Rahman, Md Abdur, and M. Shamim Hossain. ”An Internet-of-Medical-Things-Enabled Edge Computing Framework for Tackling COVID-19.” IEEE Internet of Things Journal 8.21 (2021): 15847-15854.

[6] Rajavel, R., Ravichandran, S.K., Harimoorthy, K. et al. IoT-based smart healthcare video surveillance system using edge computing. J Ambient Intell Human Comput (2021). https://doi.org/10.1007/s12652-021-03157-1

[7] Pramanik, Anima, Sobhan Sarkar, and J. Maiti. ”A real-time video surveillance system for traffic pre-events detection.” Accident Analysis & Prevention 154 (2021): 106019.

[8] Hou, B., & Zhang, J. (2021, July). Real-time Surveillance Video Salient Object Detection Using Collaborative Cloud-Edge Deep Reinforcement Learning. In 2021 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[9] Sandler, Mark, et al. ”Mobilenetv2: Inverted residuals and linear bottlenecks.” Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[10] M.Hollemans, Mobilenet version 2, April 2018.

[11] Khosla, Aditya, et al. ”Understanding and predicting image memorability at a large scale.” Proceedings of the IEEE international conference on computer vision. 2015.

[12] Zivkovic, Zoran. ”Improved adaptive Gaussian mixture model for background subtraction.” Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.. Vol. 2. IEEE, 2004.

[13] Zivkovic, Z., & Van Der Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern recognition letters, 27(7), 773-780.

[14] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection. 2020. arXiv: 2004.10934 [cs.CV].

[15] S. Oh et al., ”A large-scale benchmark dataset for event recognition in surveillance video,” CVPR 2011, 2011, pp. 3153-3160, doi: 10.1109/CVPR.2011.5995586.