Fog Computing Enabled Cost Effective Distributed Summarization of Surveillance Videos for Smart Cities

[1]Mansoor Nasir, [2]Khan Muhammad, [3]Jaime Lloret, [1]Muhammad Sajjad*

[1]Digital Image Processing Laboratory, Department of Computer Science, Islamia College Peshawar, Peshawar-25000, Pakistan

[2]Intelligent Media Laboratory, Digital Contents Research Institute, Sejong University, Seoul-143-747, Republic of Korea

[3]Instituto de Investigacion para la Gestion Integrada de Zonas Costeras, Universitat Politecnica de Valencia, Valenciana 46022, Spain

Abstract

Fog computing is emerging an attractive paradigm for both academics and industry alike. Fog computing holds potential for new breeds of services and user experience. However, fog computing is still nascent and requires strong groundwork to adopt as practically feasible, cost-effective, efficient and easily deployable alternate to currently ubiquitous cloud. Fog computing promises to introduce *cloud-like* services on local network while reducing the cost. In this paper, we present a novel resource efficient framework for distributed video summarization over a multi-region fog computing paradigm. The nodes of the fog network is based on resource constrained device Raspberry Pi. Surveillance videos captured by Raspberry Pi are distributed on different nodes and a summary is generated over the fog network which is periodically pushed to the cloud to reduce bandwidth consumption. Different realistic workload in the form of a surveillance videos are used to evaluate the proposed system. Experimental results suggest that even by using an extremely limited resource single board computer, the proposed framework has very little overhead with good scalability over off-the-shelf costly cloud solutions, validating its effectiveness for IoT-assisted smart cities.

Keywords

Fog Computing, Video Summarization, Green internet of things (IoT), Energy-Efficient Cloud Computing, Surveillance Videos, and Computational Efficiency

1. Introduction

Fog Computing is a recently introduced architecture and a paradigm in which the computing capabilities of a traditional cloud based network is shifted from a centralized data centers to local end-user devices and networks. Fog computing principally extends the cloud computing architecture to the edge of the network which enables an innovative variety of silent services and applications for end-users. Since the inception of IoT, the number of devices connected to the internet has jumped from millions to billions. Traditional cloud based centralized system cannot respond to all connected devices without degrading user experience. To manage challenges, presented by IoT infrastructure, fog computing is introduced which promises to effectively provide low latency, high mobility and wide spread of geographical coverage with compensation of huge number of nodes. Fog computing or Fogging is still nascent and gaining popularity due to its potential in wide range of applications including IoT based systems, real-time computing systems, energy aware computing applications, latency sensitive applications and mobile applications etc. [1].

Smart cities is not an abstract concept anymore as more and more tech giants like Alphabet is investing in first completely connected smart city project. Future smart city projects are essentially based on the IoT infrastructure and are aimed to be completely connected. As smart cities promises to provide comfort and a higher level of satisfaction to the citizens, one of the major issue that it can resolve is the public security [2]. Public security is a growing problem for cities worldwide. Future cities area aimed to be equipped with IoT based technology to provide facilities to police and emergency services to fight crime and make cities safer. Furthermore, future cities must ensure all these services with limited resources and lean more towards clean and renewable energy [3]. To enable challenging IoT applications with resource demanding and heterogeneous application requirements, researchers assesses, extends and improves state-of-the-art IoT communication technologies and protocols that are suitable for resource constrained devices [4]. These activities include the design of resource allocation protocols, improving resilience and robustness of communication with decreased energy consumption.

The underlying smart devices which makes up the whole IoT architecture are expected to be resource constrained. This constraint not only applies to memory and processing capabilities, but the low-power radio standards which could further constrain the network interfaces. To enable reliable IoT applications using small, low power, battery operated devices different design tradeoffs have to be considered, both in hardware, communication and software implementations. To cope with these diverging requirements, this research provided a proof of concept design to enable security over smart devices in smart cities using limited resources over IoT and fog infrastructure.

In this paper we propose a novel framework for distributed summarization of surveillance over a fog network. The fog network itself is based on multiple regions combined as clusters of Raspberry Pi. The video stream collected by surveillance cameras connected to the Raspberry Pi is summarized by the cluster itself without the use of any centralized server. Only the summarized content of the video is periodically sent to the cloud for long-term storage. The proposed method not only is cost effective, it also serves as a proof of concept for scalable, resilient and robust distributed fog network that can render streams of videos in parallel. In summary, the contributions of this work is as follows:

- We propose a novel fully distributed multi-region fog computing based framework for surveillance videos summarization without having to use a centralized cloud server. To the best of our knowledge, no such solutions exists as yet.
- The fog computing platform is built on low-cost, low-powered Raspberry Pi clusters, built on top of Apache Spark and Hadoop for speedy summarization of video streams.
- The proposed framework not only replaces the need for centralized server, but it also significantly reduced the bandwidth consumption of a centralized and cloud based framework.
- We designed and conducted series of experiments with real workload to evaluate the performance of the fog network based on these small low-powered devices.

The rest of the paper is organized as follows: In Section 2 we discuss the recent advancements in fog based systems and Internet of Things (IoT) in general. We also discuss some related literature for video summarization and Raspberry Pi. In Section 3 we discuss the proposed framework and methodology along with layout of the proposed method in detail. Section 4 presents the experimental results and performance evaluation of the proposed framework based on real workload. Section 5 concludes the paper with some insight and future directions.

2. Related Work

Fog computing is intended to bring cloud based computation to a local network thereby reducing the computation cost and bandwidth of the network. Fog computing technology and related formal architectures are rare and this technology has not been widely accepted as standard for IoT based applications yet. But observing the trends in IoT based applications, we can safely predict that usability and applicability of fog paradigm in the near future for all types of applications. Typical IoT based applications connect to a single cloud which responds in real-time to all connected devices which can create delay. In order to avoid such delays the computation of the cloud is brought down to the localized network. L. Wan et al proposed an architecture to simulate a battlefield surveillance system with mobile cloud computing cognitive wireless network and a 5g link to estimates the trajectory and estimation of missiles in the battlefield only by using mobile sensors [5]. Mauri et al. [6] proposed a system that uses alternate channels of transmission to stream surveillance videos in rural environments. The proposed solution promises scalability, compression and practicability in real world situations without compromising the quality. Cloud based application uses virtualization which inherently create delays in real-time streaming applications over a mobile network that affects QoS in cloud mobile applications. García-Pineda et al. [7] proposed evaluation matrices for assessing quality of video streams in mobile cloud based applications. Taha et al. [8] proposed a technique for efficient handoff in 5G networks for better QoS to ensure better fog connectivity for IoT based applications.

With the advent of small, portable, low power, low cost, mini computers we can create a local Fog-based system to respond in real-time to the connected devices. Popularity of the fog-network is not only limited to large scale networks, [9] proposed fog-computing based vehicular network to create a fog-enabled communication between vehicles. Secinti et al. [10] proposed a similar architecture that enables software defined networks to communicate in VANETs using Raspberry Pi as platform. Benson et al. [11] proposed and developed Scale: Safe community awareness and alerting system which leverages the IoT architecture. The authors conclusively proved that the Raspberry Pi can be used as a computation platform for all home automation systems and with minimum efforts without adding any extra cost.

MQTT protocol is used as signaling platform between all devices. The system design is somewhat similar to our proposed architecture but the fundamental application is different. Raspberry Pi has quickly become one of the best-selling computers which stimulated various interesting projects across both industry and academia. This single board computer is so popular among hobbyist and academia alike that in the past five years the Raspberry Pi foundation has sold more than 12.5 million units of this device. In order to prove that even such small and low powered devices can perform intense tasks such as video summarization, we combine them to form clusters of these small devices. For this purpose several Raspberry Pi devices are combined (5 in each region) to from an Apache Spark and Hadoop cluster. The first Raspberry Pi based cluster platform is proposed by Iridis-pi [12] and Glasgow Raspberry Pi Cloud [13]. The hardware construction of the nature of these projects might be similar to our work but there is a distinctive difference between the applications of the cluster in our work. In fact, there is no related literature on the topic where computer vision application has been evaluated on the cluster. Iridis-pi was developed to educate students in understanding the data handing in high performance computing platforms. The Glasgow Pi Cloud project is mainly focused on virtualization technologies related to cloud computing. In spite of the popularity of these small low powered computers, there has been very limited study on performance of the systems on fog computing paradigm. Morabito et al. [14] run a series of experiments to test applications related to single node of Raspberry Pi, but these applications are only related to container-based technologies. The purpose work was to evaluate the Memory, Input/Output, Disk Input/Output, CPU and Network I/O. They concluded that the virtualization impact on the performance is negligible in comparison with native execution environment. Raspberry Pi

One of the most important goal of smart cities is to provide automated and enhanced security with minimum overhead. Any secure city needs to be equipped with thousands if not millions of smart surveillance cameras. With the development of the digital video processing technology, video surveillance has been playing an important role for security and management. Due to the high volume of videos, manually retrieving meaningful information from these videos is very time-consuming and impractical. It is necessary and important to allow the smart cameras to automatically extract the parts of interest form videos. Key frame extraction and video summarization are common approaches towards tackling this problem. This works by creating a brief version of the original video to effectively represent the original. The summarization structure is learned by a machine learning algorithm which is later on used to classify only portions of the video which can serve as *effective-representation* of the original video. Detecting interesting events in the surveillance videos is more common due to the advancements in artificial intelligence and machine learning techniques. In [15] the front and rear view of the pedestrian is detected using a novel wavelet coefficient technique. This technique was among the first to apply object detection in surveillance videos. The authors of [16] presented a summarization technique based on hierarchical clustering. In this technique the shots having similar features and are closely related in the time domain are combined. The authors make use of MPEG-7 visual descriptor to choose indices and to generate summary. The resulting summary contains the key-frames and preview of the original video which can be accessed in a non-linear fashion. Rasheed et al. [17] present a summarization scheme based on shot-similarity graph. Wang et al. [18] suggest that by analyzing object motion in video streams and by calculating the overall motion of the camera nodes, useful information can be extracted which represents the structure of the video. The approach can be used to create partial summaries of the original video. In [19] the authors presented a robust video summarization scheme that also make use of encryption techniques to securely transfer image data in IoT based systems. The authors use a novel technique to encrypt video frames and yet the memory consumption is low, which makes it more suitable for small, low-powered and resource constraint devices.

We propose a novel framework for a computer vision application based on realistic video footage captured by the Raspberry Pi camera. In addition we study the performance of the cluster in terms of computation cost as well as the resources consumed during the process, which gives a dynamic insight in the scalability of the fog computing based network.

3. Our Proposed Methodology

As discussed in the previous section, we propose a novel framework based on fog computing architecture for video summarization based on low powered computers. The overall layout of the network is presented in Figure 1.

The surveillance camera network is divided into multiple regions connected to a centralized router. Each region is composed of multiple surveillance cameras connected via a wired Ethernet cable. The camera itself serves both as node of the network as well as a surveillance camera because, Raspberry Pi can serve both as a computation platform and also as a camera module for the network, thereby enabling embedded vision. The centralized router is also connected to the internet gateway which is capable of sending summarized videos to cloud based on MQTT publishing. Each region of the fog network has a "master" node that servers as a virtualized server, whereas the rest of the nodes serve as "slaves" for the cluster. A job is divided and controlled by the *master* node within the region itself and it is also responsible for communication with the MQTT server for responses and cloud offloading. Master node of one region can also communicate with the master of other region and also with the cloud via the centralized router.

The overall framework of the proposed method is presented in Figure 3 (a). Smart camera based on Raspberry Pi captures the video stream, which is then distributed to all nodes in the same region and processed in parallel using Apache Spark and Hadoop based cluster, the specification and details of this step is mentioned in the next section. Once the video is summarized, the master node send this short version of the video to the cloud using the internet gateway. This technique not only summarizes the video stream into more meaningful chunks but also saves

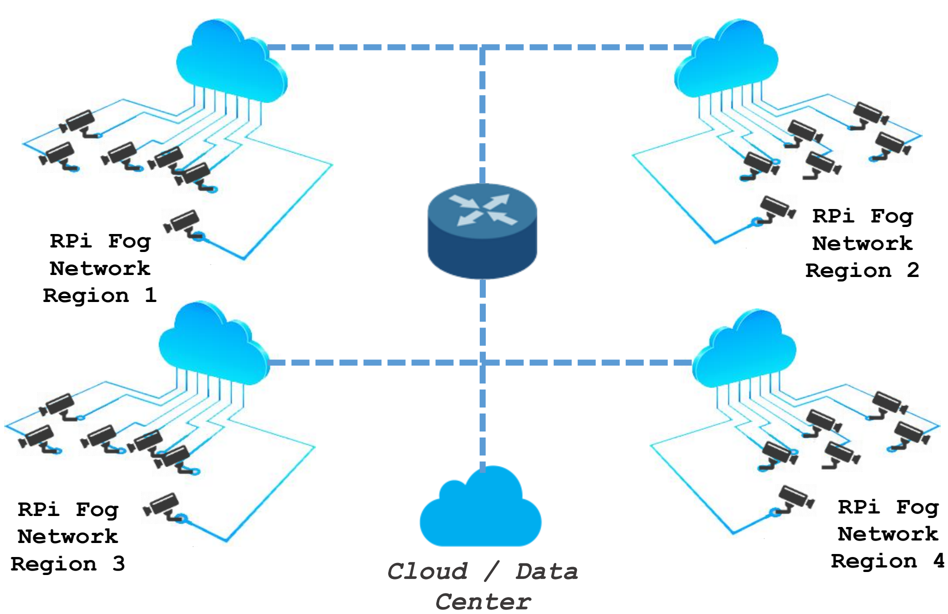

Figure 1: Network layout for fog based surveillance network.

significant bandwidth that would have been wasted, if we were to send the whole stream to the cloud.

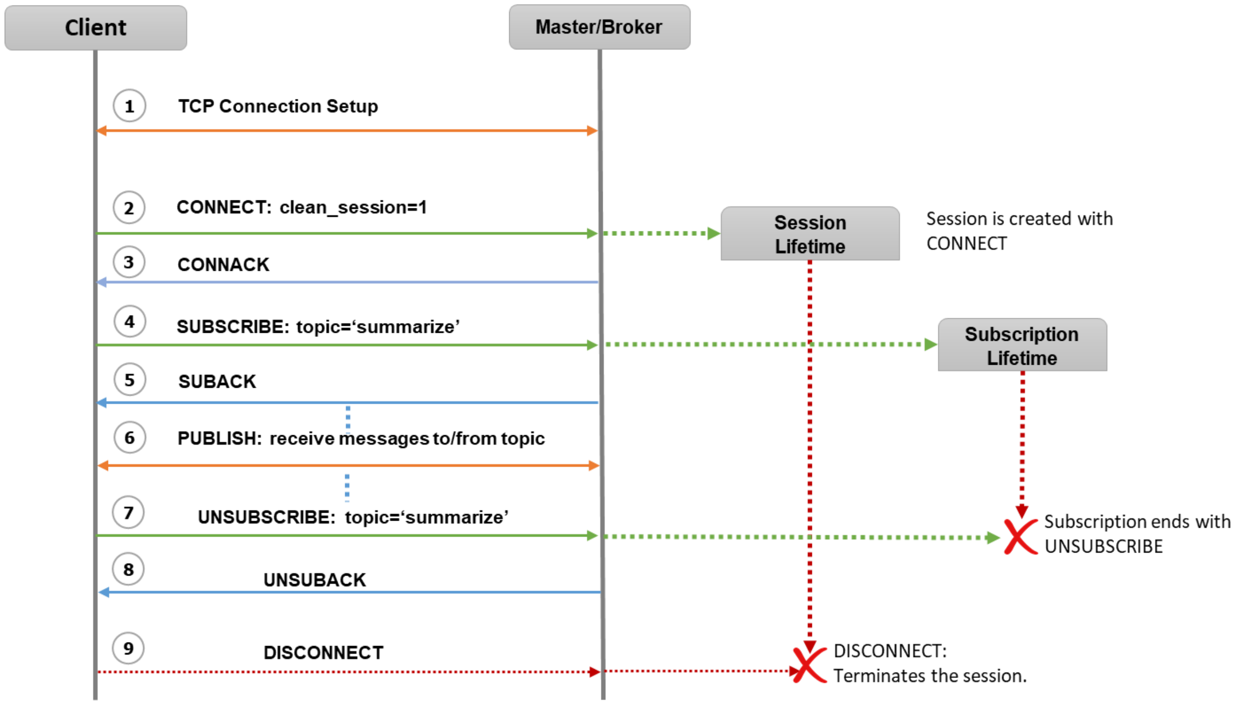

Figure 2: Message flow between broker and client for initiating summarization process

In Figure 2, the client and master communication with one another is presented for initiating the summarization process. They use MQTT protocol to communicate with the broker. The clients initiates the process by subscribing to the process of summarization using request with the name of the topic. A session with a pre-defined life-time is set for each client and then acknowledged accordingly. The clients subscribe to the same topics receive directives from the master node along with necessary data to initiate the process. The summarized content is then shared with the broker which then is transferred to cloud periodically. Using this approach all connected clients receive messages without having to be told time and again. MQTT protocol is very simple to implement and has very limited overhead with even extremely resource constraint device i.e. Raspberry Pi.

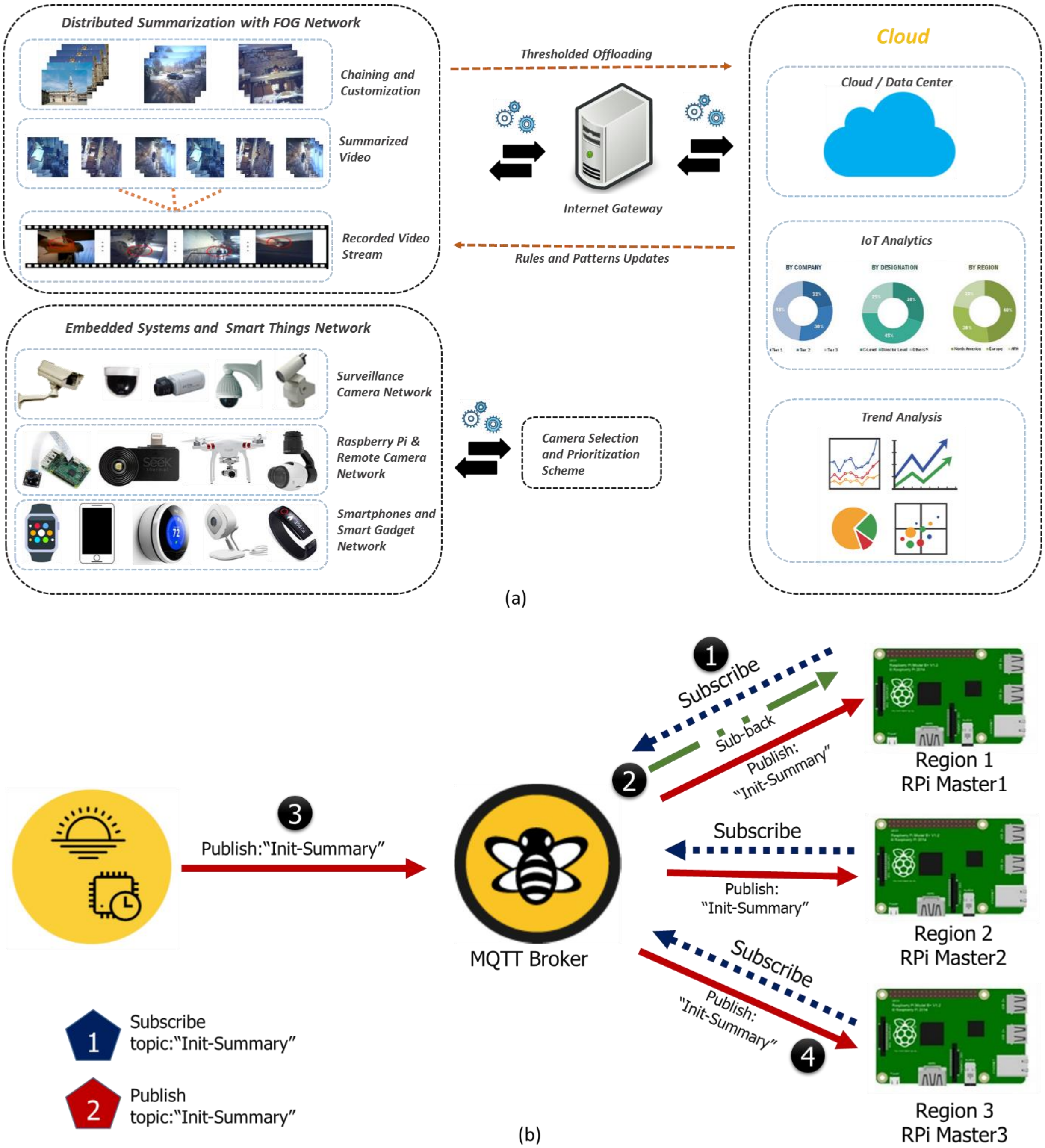

Figure 3 (a): Proposed framework for distributed summarization of surveillance videos (b) MQTT framework for summary initializer module.

The proposed framework is based on multiple regions of surveillance cameras working together in the form of a cluster to summarize surveillance videos in order to conserve energy, reduce computational cost and also reduce the bandwidth. All these regions in the fog network is controlled via a master node that servers as a server for all other nodes in the same region. These master nodes communicate with each other via MQTT protocol. MQTT stands for Message Queue Telemetry Transport and it is a lightweight message queue protocol specifically designed for low bandwidth networks where small data packets needs to be send across very high latency links. MQTT provide simplicity and it serves as an accepted communication protocol for almost all internet-of-things projects [20]. Working of the MQTT protocol is presented in Figure 3 (b). MQTT works on even unreliable networks with some degree of assurance that the messages has been delivered successfully. In our case MQTT is used to pass messages between master nodes of each region. In the first step, all the master nodes subscribe to the topic "init-summary" and listens to this topic. In the second step the MQTT message broker send confirmation to each subscribing node that subscription has been successful and they are now ready to receive messages related to the topic subscribe earlier. In the third step the message broker publishes the topic "init-summary" indicating that all master nodes may start the summarization process. This step in our case is *time-triggered* meaning that it is initiated only once per 12 hours. Upon successful completion of the summary process, the processed file is generated on master node and sent to the cloud periodically.

The summarization process is lightweight and is adapted from [21] where the algorithm assume that the surveillance cameras are still and the background is static i.e. not moving and pointed to the fixed location and at a fixed viewing angle. The redundancy between frames are quantized in the form of energy and is used as a criteria for *interesting event* detection. The process of event detection is divided into three steps. In the first step the absolute difference of two consecutive frames is calculated. After finding the differences between frames, the values of each frame is quantized in the form of energy, that is later on used for event detection. Energy of each frame is calculated using equation 1.

$$Energy\ (f) = \frac{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1} x(i,j)}{N\ .\ M} \tag{1}$$

The energy of all frames are calculated and in step three, only the frames that crosses a fixed threshold is selected as keypoints which effectively represents the whole video. The steps to the algorithms are presented in Table 1.

Table 1 Proposed algorithm for surveillance video summarization.

Input: Surveillance video đ.

for f := 1 to len(đ)
    α_0 = abs($f_i - f_{(i-1)}$)
    $Ç_F$ appends (α_0)

for i:=0 to N-1
    for j:=0 to M-1
        α_1 = abs($x_i - x_j$)
        $\varepsilon(f) = \alpha_1/N*M$
        $Ç_I$ = d (x_i , x_{i+1}) < $Ł_0$
        Ɏ = $Ç_I$ /$Ç_F$ > $Ł_1$

Output: Ɏ containing list of interesting events representing the key-frames.

The algorithms works by processing an input video ***đ.*** The absolute difference α_0 of two consecutive frames are calculated and stored in candidate list $Ç_F$. This list contains all the key-frames that has some difference with the previous frames. Similarly the whole video is processed to extract interesting frames based on equation 1. The energy $\varepsilon(f)$ of each frame is calculated and stored in candidate list $Ç_I$. The frame is considered as interesting only, if it crosses a fixed threshold $Ł_0$. The list *Ɏ* is the ratio of both candidates' lists and event is considered as interesting only if the ratio crosses a fixed threshold $Ł_0$. These thresholds are selected after experimentation and vary widely, depending on type and condition of the camera calibration. In our case the threshold were fixed and no changes were made to the default setting of the RPi camera. A simple clustering technique KNN is applied to the list Ɏ to obtain a summarized video content that effectively represents the whole video.

4. Experimental Results

In this section we evaluate the performance of the proposed framework by comparing the results of a single node with the Spark based cluster. The system is either tested on a single node of Raspberry Pi 3 Model B which has a Broadcom BCM2837 system on a chip which includes an ARM Cortex-A53, 1.2GHz processor, Video Core IV GPU, and an SD card slot along with 1GB

RAM and 100MBps Ethernet connection. The cluster of each fog region is composed of 5 nodes working as slaves under one *master* node. Each of the connected node is installed with Apache Spark and Hadoop's HDFS. The reason that we chose Spark is because of its popularity in big-data analytics and the fact that its performance is very good on small embedded devices like Raspberry Pi. On each node of every region of fog network, we installed *Lite* version of the Raspbian operating system to conserve more memory and processing power.

Apache Spark is used as a general-purpose clustering system which can be used to work as a traditional *Extract*, *Transform*, and *Load* for feeding data to the warehouses, or it can be used to perform interactive analysis for online pattern matching etc., There are three different ways in which Apache Spark can be used for clustering: 1) Standalone Mode (i): in which Spark and HDFS directly communicate with each other and optionally MapReduce can submit jobs in the same cluster; 2) Hadoop Yarn (ii): In this mode, the Spark executes over a Hadoop container manager distributed across the cluster; 3) SIMR (iii) or Spark in MapReduce, where Spark can execute its own jobs along with the MapReduce. In our experiment we used Standalone deployment of the Apache Spark in which both HDFS and Apache Spark are the part of the cluster. All types of Spark deployment are illustrated in Figure 4 (a). The working of a general purpose Spark cluster is usually in for steps; 1) the candidate data (video file in our case) is distributed across all working nodes; 2) the mapper functions process the data upon reception; 3) the aggregation of comparable patterns is performed by shuffling process and lastly 4) to produce a consolidated output the reducer combines all the processed data.

HDFS is Hadoop's distributed file system designed to run on ordinary hardware without the use of high end expensive hardware. HDFS is specifically designed for huge datasets processing. HDFS works by replicating the data into smaller portions and distributing and replicating it across multiple nodes. The reason that the data is replicated is because it makes the system fault tolerant in case of failure of nodes. Figure 4 (b) shows the working of a typical Hadoop Distributed File System over a video file distributed among different Raspberry Pi based nodes. Typically the HDFS based clusters are composed of server nodes called *namenode* and multiple slave nodes called *datanode*. We designed experiments to evaluate the proposed framework on

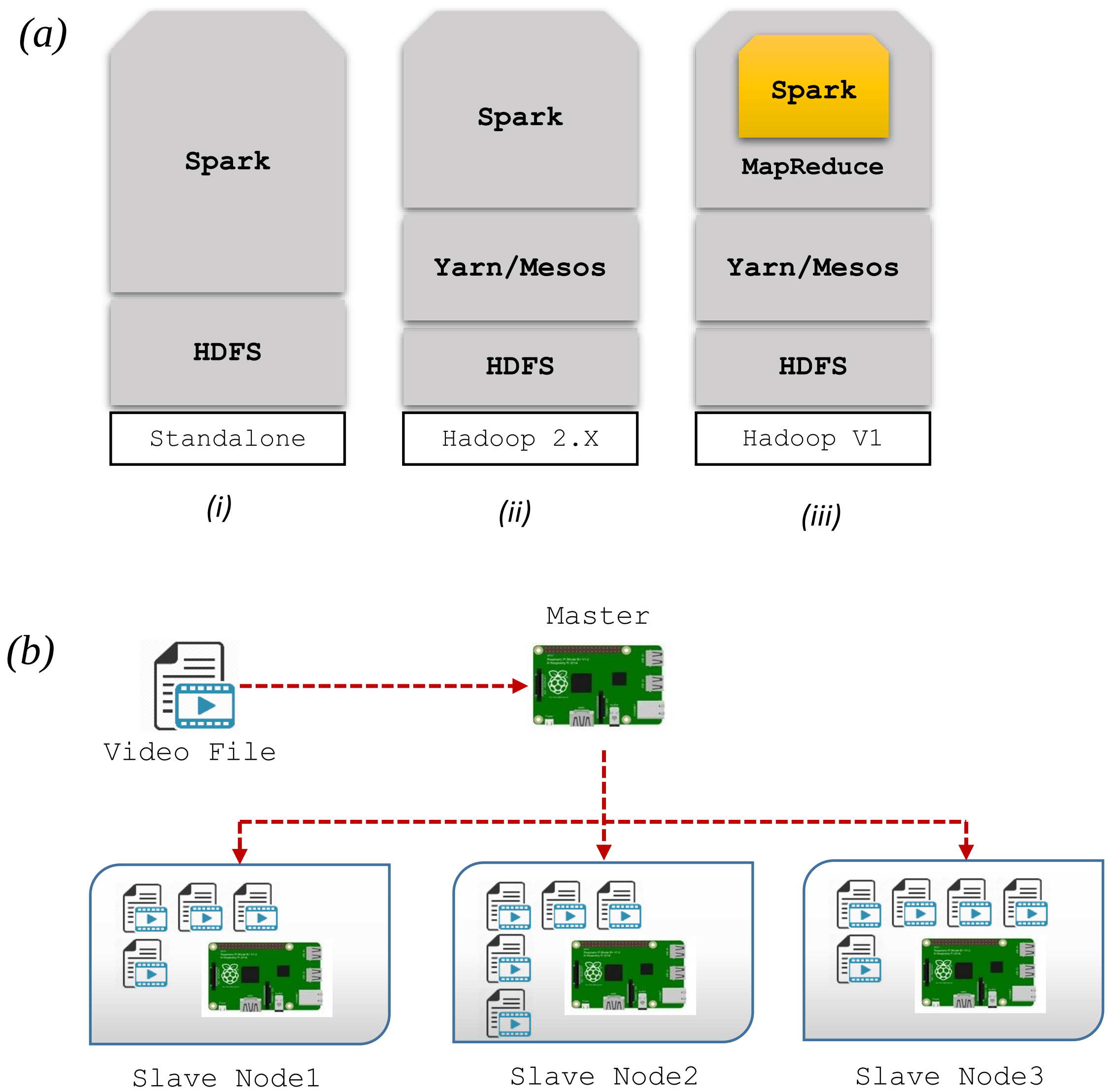

Figure 4 (a): Types of Apache Spark Deployments (b) File Distribution over Hadoop & Apache Spark different sizes of data and on single and multiple clusters. The file sizes are categorized in different problem sizes given in Table 1. In a real-world situation, the problem size may vary widely and the actual results may too, but by dividing these problems into sub categories can give us in depth performance evaluation.

Table 2: Problem sizes for cluster performance evaluation

File Size	Problem Size	Type
100 MB	Small	Video File
300 MB	Medium	Video File
500 MB	Medium	Video File
700 MB	Large	Video File
1024 MB	Large	Video File

In the first series of experiments we evaluate the fog-based Spark-Hadoop cluster for CPU usage, cluster throughput and network performance. Figure 5 shows the complete evaluation of the system. At the first glance we observe somewhat predictable results and identify identical patterns for clusters, as it tend to be more robust and efficient then a single system. For small workloads the performance of the cluster is very high as there are many available cores compare to the only 4 available on a single Raspberry Pi. Figure 5 (a) shows that the req/s is 2700 for cluster where it is at 100 for single Pi. For larger jobs the single system not only significantly degrades but the temperature of the Pi significantly increases even with an install air coolant. The network throughput Figure 5 (b) for the same cluster is very high for clusters as multiple files are shared across the cluster for parallel computation. Where as in this case the single Pi does not have any overhead as it shares the data between its own cores and only communicate with the namenode at 17kb/s constant.

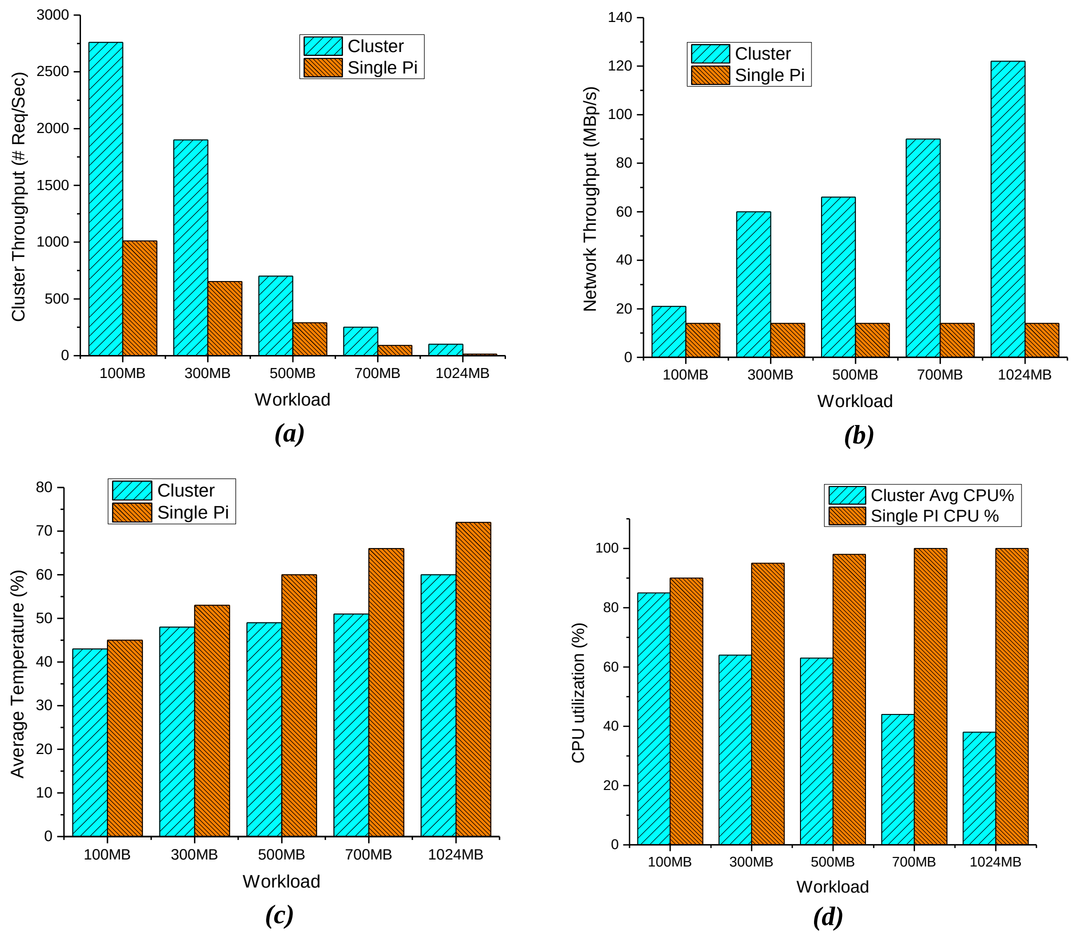

Figure 5: Performance evaluation (a) Cluster throughput vs Single System (b) Network Thoughput (c) Temperature Analysis (d) CPU Utilization

The average temperature of the single system is very high even for medium job sizes. All the nodes in the cluster are equipped with a 5v DC fan connected to the GPIO pins of the Raspberry Pi. Even then it is approaching very high temperatures and could damage the Pi, Figure 5 (c). However, the average temperature of the rest of the cluster remains relatively low even for larger jobs. It suggests that the cluster can be expanded without having to worry about the health of each node in the cluster. The average CPU utilization Figure 5 (d) of the cluster is significantly low because of the job distribution among multiple systems. On the contrary the CPU utilization of the single Pi is significantly high even though the throughput is very low.

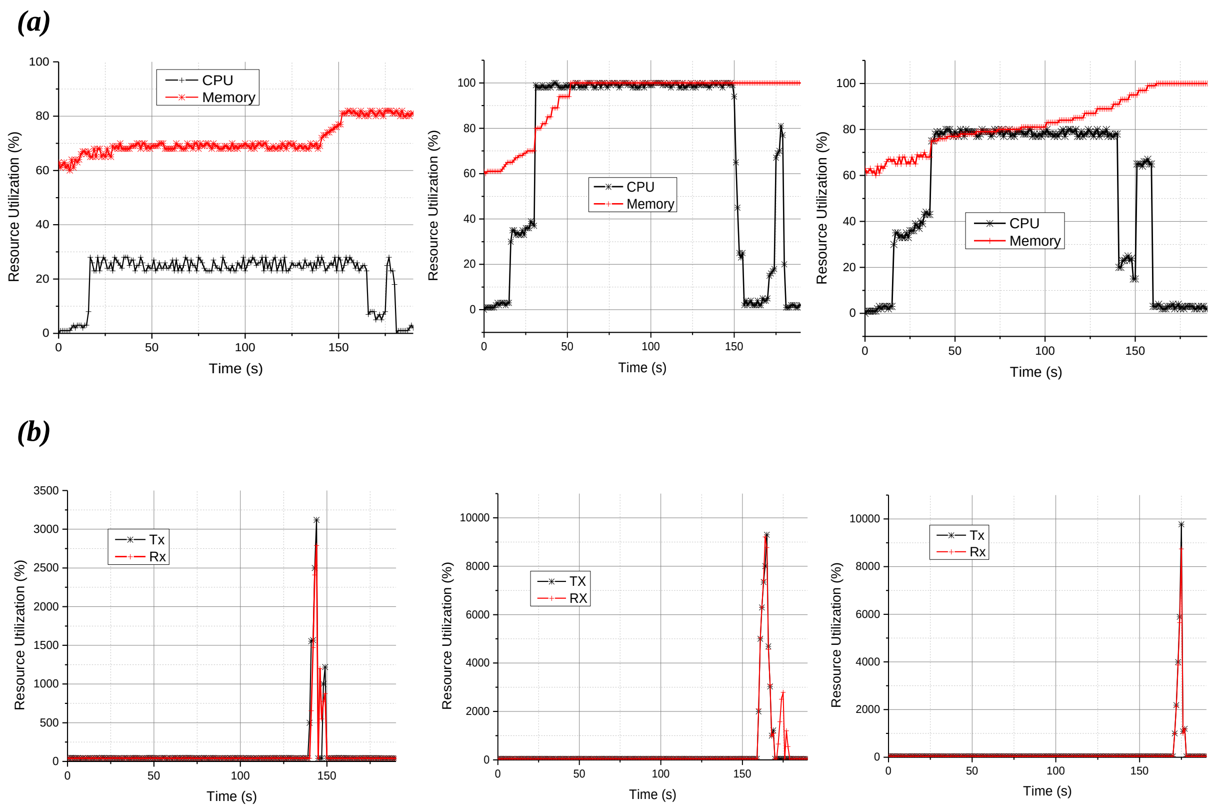

Figure 6 (a) CPU and Memory usage of the Raspberry Pi cluster for video summarization (b) Network usage of the Raspberry Pi cluster for video summarization.

In order to further evaluate the performance of the system we divide the file sizes to 1) Small 1 GB 2) Medium 3 GB and 3) Large 6 GB file respectively. Figure 6 (a) shows the resources utilization of the system in terms of memory and CPU usage. As evident from these results, the memory utilization of the cluster is very high even for small jobs. This is because the Spark based cluster are memory greedy and the RAM available in the Raspberry Pi is sparse. The high CPU and memory utilization observed in the large jobs implies that there are constant swapping in the memory and smaller jobs have to wait even if completed earlier.

In addition to the previous analysis, the network transmission of the fog network is depicted in Figure 6 (b). Smaller jobs reach the peak value of 3.2 Mbps while medium and large jobs reached to 9 and 10 mbps respectively. The network traffic is significantly high at the end of each job because of the shuffling process, where every node is sharing results with their neighbors to consolidate results.

This architecture serves as a proof of concept design for future cities and become valuable asset for all future law enforcement services. The fact that small, low powered, single-board tiny computers can be combined to become a valuable cluster is more natural and can serve as a platform for many computer vision applications. The currently available techniques which utilizes the embedded vision are either single-system based, or had not considered the powerful features of fog computing.

5. Conclusion

In this paper we proposed a novel fog computing enabled distributed video summarization scheme for surveillance videos based on small, low-powered, cheap, single board computer known as Raspberry Pi. The fog network is composed of multiple regions of nodes, based on these single board computers. We conclusively proved that this low power computer not only replace a costly cloud solution, but also holds potential for scalability without the added significant cost. In the future, we will explore video summarization algorithms that are specifically build for resource constraint devices. Furthermore we will look into more robust fog architectures that not only benefits the scalability of the network but also provide robust response for intense computer vision based tasks such as deep learning models and deep learning based activity recognition.

References

1 Bonomi, F., Milito, R., Natarajan, P., and Zhu, J.: ‘Fog computing: A platform for internet of things and analytics’: ‘Big Data and Internet of Things: A Roadmap for Smart Environments’ (Springer, 2014), pp. 169-186

2 Schaffers, H., Komninos, N., Pallot, M., Trousse, B., Nilsson, M., and Oliveira, A.: ‘Smart cities and the future internet: Towards cooperation frameworks for open innovation’, The future internet, 2011, pp. 431-446

3 Byers, C.C.: ‘Architectural imperatives for fog computing: Use cases, requirements, and architectural techniques for FOG-enabled IoT networks’, IEEE Communications Magazine, 2017, 55, (8), pp. 14-20

4 Khan, R., Khan, S.U., Zaheer, R., and Khan, S.: ‘Future internet: the internet of things architecture, possible applications and key challenges’, in Editor (Ed.)^(Eds.): ‘Book

Future internet: the internet of things architecture, possible applications and key challenges' (IEEE, 2012, edn.), pp. 257-260

5 Wan, L., Han, G., Shu, L., Feng, N., Zhu, C., and Lloret, J.: 'Distributed parameter estimation for mobile wireless sensor network based on cloud computing in battlefield surveillance system', IEEE Access, 2015, 3, pp. 1729-1739

6 Lloret, J., P.V., Jimenez, J.M., and Diaz, J.R.: '802.11 g WLANs design for rural environments video-surveillance', in Editor (Ed.)^(Eds.): 'Book 802.11 g WLANs design for rural environments video-surveillance' (IEEE, 2006, edn.), pp. 23-23

7 García-Pineda, M., Felici-Castell, S., and Segura-García, J.: 'Adaptive SDN-based architecture using QoE metrics in live video streaming on Cloud Mobile Media', in Editor (Ed.)^(Eds.): 'Book Adaptive SDN-based architecture using QoE metrics in live video streaming on Cloud Mobile Media' (IEEE, 2017, edn.), pp. 100-105

8 Taha, M., Parra, L., Garcia, L., and Lloret, J.: 'An Intelligent handover process algorithm in 5G networks: The use case of mobile cameras for environmental surveillance', in Editor (Ed.)^(Eds.): 'Book An Intelligent handover process algorithm in 5G networks: The use case of mobile cameras for environmental surveillance' (IEEE, 2017, edn.), pp. 840-844

9 Huang, C., Lu, R., and Choo, K.-K.R.: 'Vehicular fog computing: architecture, use case, and security and forensic challenges', IEEE Communications Magazine, 2017, 55, (11), pp. 105-111

10 Secinti, G., Canberk, B., Duong, T.Q., and Shu, L.: 'Software defined architecture for VANET: a testbed implementation with wireless access management', IEEE Communications Magazine, 2017, 55, (7), pp. 135-141

11 Benson, K., Fracchia, C., Wang, G., Zhu, Q., Almomen, S., Cohn, J., D'arcy, L., Hoffman, D., Makai, M., and Stamatakis, J.: 'Scale: Safe community awareness and alerting leveraging the internet of things', IEEE Communications Magazine, 2015, 53, (12), pp. 27-34

12 Cox, S.J., Cox, J.T., Boardman, R.P., Johnston, S.J., Scott, M., and O'brien, N.S.: 'Iridis-pi: a low-cost, compact demonstration cluster', Cluster Computing, 2014, 17, (2), pp. 349-358

13 Tso, F.P., White, D.R., Jouet, S., Singer, J., and Pezaros, D.P.: 'The glasgow raspberry pi cloud: A scale model for cloud computing infrastructures', in Editor (Ed.)^(Eds.): 'Book The glasgow raspberry pi cloud: A scale model for cloud computing infrastructures' (IEEE, 2013, edn.), pp. 108-112

14 Morabito, R.: 'A performance evaluation of container technologies on Internet of Things devices', in Editor (Ed.)^(Eds.): 'Book A performance evaluation of container technologies on Internet of Things devices' (IEEE, 2016, edn.), pp. 999-1000

15 Oren, M., Papageorgiou, C., Sinha, P., Osuna, E., and Poggio, T.: 'Pedestrian detection using wavelet templates', in Editor (Ed.)^(Eds.): 'Book Pedestrian detection using wavelet templates' (IEEE, 1997, edn.), pp. 193-199

16 Lee, J.-H., Lee, G.-G., and Kim, W.-Y.: 'Automatic video summarizing tool using MPEG-7 descriptors for personal video recorder', IEEE Transactions on Consumer Electronics, 2003, 49, (3), pp. 742-749

17 Rasheed, Z., and Shah, M.: 'Detection and representation of scenes in videos', IEEE transactions on Multimedia, 2005, 7, (6), pp. 1097-1105

18 Wang, Y., Zhang, T., Tretter, D., and Wu, P.: 'Real time motion analysis toward semantic understanding of video content', in Editor (Ed.)^(Eds.): 'Book Real time motion analysis toward semantic understanding of video content' (International Society for Optics and Photonics, 2006, edn.), pp. 596027

19 Muhammad, K., Hamza, R., Ahmad, J., Lloret, J., Wang, H.H.G., and Baik, S.W.: 'Secure Surveillance Framework for IoT systems using Probabilistic Image Encryption', IEEE Transactions on Industrial Informatics, 2018

20 Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., and Ayyash, M.: 'Internet of things: A survey on enabling technologies, protocols, and applications', IEEE Communications Surveys & Tutorials, 2015, 17, (4), pp. 2347-2376

21 Damnjanovic, U., Fernandez, V., Izquierdo, E., and Martinez, J.M.: 'Event detection and clustering for surveillance video summarization', in Editor (Ed.)^(Eds.): 'Book Event detection and clustering for surveillance video summarization' (IEEE, 2008, edn.), pp. 63-66