Journal of Parallel and Distributed Computing 126 (2019) 161–170

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

# Fog computing enabled cost-effective distributed summarization of surveillance videos for smart cities

Mansoor Nasir [a], Khan Muhammad [b], Jaime Lloret [c], Arun Kumar Sangaiah [d], Muhammad Sajjad [a,*]

[a] Digital Image Processing Laboratory, Department of Computer Science, Islamia College Peshawar, Peshawar 25000, Pakistan
[b] Department of Software, Sejong University, Seoul 143-747, Republic of Korea
[c] Instituto de Investigacion para la Gestion Integrada de Zonas Costeras, Universitat Politecnica de Valencia, Valenciana 46022, Spain
[d] School of Computing Science and Engineering, Vellore Institute of Technology, Vellore 632014, India

## HIGHLIGHTS

- Low-cost, low-powered Raspberry Pi cluster is build, using Apache Spark and Hadoop.
- Energy and bandwidth consumption is significantly reduced compare to Cloud solutions.
- Experimental evaluation of the fog network based on small low-powered devices.

## ARTICLE INFO

*Article history:*
Received 10 March 2018
Received in revised form 9 July 2018
Accepted 12 November 2018
Available online 29 November 2018

*Keywords:*
Fog computing
Video summarization
Internet of things (IoT)
Energy-efficient cloud computing
Surveillance videos
And computational efficiency

## 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 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.



## 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 initiation of IoT based applications, the number of devices connected to the internet has jumped from millions to billions and it is more likely to grow in the near future. Traditional cloud based centralized system cannot respond to all connected devices in real-time without degrading user experience. To manage challenges, presented by IoT infrastructure, Fog computing is introduced, that promises to effectively provide low latency, high mobility and wide spread of geographical coverage with compensation of huge number of nodes. Fog computing or sometimes also referred as *Fogging* is still nascent and gaining popularity due to its potential in wide range of applications including IoT based systems [15,25], real-time computing systems, energy aware computing applications [9,18], latency sensitive applications [9] and mobile applications etc. [4].

Smart cities are not an abstract concept anymore, as more and more tech giants like Alphabet is investing in first completely connected smart city project [10,14]. Future smart city projects are essentially based on the IoT infrastructure. As smart cities promise to provide comfort and a higher level of satisfaction to

* Corresponding author.
*E-mail addresses:* mansoornasir@icp.edu.pk (M. Nasir), khanmuhammad@sju.ac.kr (K. Muhammad), Muhammed.sajjad@icp.edu.pk (M. Sajjad).

https://doi.org/10.1016/j.jpdc.2018.11.004
0743-7315/

the citizens, one of the major issue that it can resolve is the public security [29,30]. Public security is a growing problem for cities worldwide therefore, 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 [28]. Furthermore, future cities must ensure all these services with limited resources and lean more towards clean and renewable energy [6]. To enable challenging IoT applications with resource-demanding heterogeneous application requirements, researchers assesses, extends and improves state-of-the-art IoT communication technologies and protocols that are suitable for resource constrained devices [19]. These activities include the design of resource allocation protocols, improving resilience and robustness of communication with decreased energy consumption.

The underlying smart devices which make 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 videos over the Fog network. The network itself is composed of 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, but 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 are as follows:

- We propose a novel fully distributed multi-region Fog computing enabled framework for surveillance videos summarization without having to use a centralized cloud server. To the best of our knowledge, no such solutions exists yet.
- The fog computing platform is built on low-cost, low-powered Raspberry Pi clusters, orchestrated with Apache Spark and Hadoop for distributed storage and speedy summarization of surveillance video streams.
- The proposed framework not only replaces the need for any centralized server, but it also significantly reduced the bandwidth consumption of a centralized and costly cloud-based solutions.
- 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 solutions 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 the 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 the 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 cause delay especially when the number of applications is in billions. In order to avoid such delays, the computation and storage capabilities of the traditional cloud-based system is brought down to the local network to exploit devices present on the edge of the network. This paper proposes a computation model to efficiently manage cloud resources for surveillance tasks allocation. Hossain et al. [16] proposed a model to optimizing the trade-off between average service waiting time and long-term service cost. The authors show that long-term service cost is inversely proportional to high and balanced utilization of cloud resources. A series of experiments show that the proposed approach proved to be very efficient for cloud resource management when handling the heterogeneous video surveillance tasks dynamically. 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 estimate the trajectory and estimation of missiles in the battlefield only by using mobile sensors [38]. Jaime Lloret et al. [21] 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 Quality of Service (Qos) in cloud mobile applications. García-Pineda et al. [12] proposed evaluation matrices for assessing quality of video streams in mobile cloud based applications. Taha et al. [34] 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, but is also useful in smaller networks, in this regard, [17] proposed Fog-computing based vehicular network to create a fog-enabled communication between vehicles. Secinti et al. [32] proposed a similar architecture that enables software defined networks to communicate in VANETs using Raspberry Pi as a computation platform. Benson et al. [3] proposed and developed SCALE: Safe community awareness and alerting system, which leverages the IoT architecture for its computation and scalability. The authors conclusively proved that the low powered, Raspberry Pi can be used as a computation platform for all home automation systems and with minimum effort without adding any extra cost. Not only is it useful in hardware based system, but this low cost devices is being used by [13] as portable switch for software-defined networks in a large scale network. Similarly [35] presented a testbed to simulate large scale wireless sensors network for research purposes. The authors argued that the proposed systems consume less power and has the ability to provide a resourceful testbed for cloud-based applications on software-defined networks.

All IoT based applications must use some protocol to enable communication between heterogeneous devices. One of the most common protocol is the Message Queuing Telemetry Transport (MQTT) protocol. MQTT is used as signaling platform between all

devices connected in an IoT based application. Raspberry Pi is being used as a cheap yet efficient solution for IoT based applications. Raspberry Pi has quickly become one of the best-selling computers that 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 14 million units of this device [37]. In order to prove that even such a small and low powered device can perform intense tasks such as video summarization, we combine them in the form of a cluster. For this purpose, several Raspberry Pi devices are combined (5 in each region) to from an Apache Spark and Hadoop cluster. There are several Raspberry Pi based clusters being built by the research community to perform intense computation tasks. The first Raspberry Pi based cluster was built by Iridis-pi [7] and Glasgow Raspberry Pi Cloud [36]. The hardware construction of the nature of these projects is similar to the proposed work but there is a distinctive difference in the applications. In fact, there is no related literature on the topic where computer vision application has been evaluated on the Raspberry Pi based cluster. Iridis-pi was developed to educate students in understanding the data handling 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. [23] 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 purposed 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 non-virtualized, native execution environment. The authors of [31] proposed a cluster of seven nodes with focus on applications in big-data analysis. The proposed framework was designed in Hadoop and Apache and had more realistic applications on performance of the cluster. Mitchell et al. [22] created a cluster of 68 Raspberry Pi for students to evaluate the challenges in cluster based system. This cluster had one master node and 64 worker nodes, while one node was a *monitor* node and two nodes were set as storage units. The focus of the activity was to develop applications for the cluster and to provide benchmarking for the research community. Bellavista et al. [2] presented a technique for Fog enabled IoT applications to use Raspberry Pi as gateways for a large scale network. The authors claim that by using even this resource constraint device, the docker-based containerization of applications is feasible and can achieve high scalability.

As discussed in Section 1, 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 is playing an important role for smart security. 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 from these surveillance videos. Key frame extraction and video summarization are common approaches for video summarization. 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 [26] 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 [20] presented a summarization technique based on hierarchical clustering. In this technique the shots having similar features which 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. [27] present a summarization scheme based on shot-similarity graph. Wang et al. [39] 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 [24] 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 Fig. 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 network node as well as a surveillance camera because, Raspberry Pi can perform computation and can also captures videos, thereby enabling *embedded* vision. The centralized router is 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 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 the other region and also with the cloud via the centralized router.

In Fig. 2, the client and master nodes communicate with one another to initiate the summarization process of the surveillance videos. MQTT protocol is used to communicate with the broker and initializing the summary process. The clients initiate the process by subscribing to the *topic* 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 is periodically pushed to the cloud, in this way the clients receive messages without being told everytime. MQTT protocol is very simple to implement and has very limited overhead with even extremely resource constraint device i.e. Raspberry Pi.

The overall framework of the proposed method is presented in Fig. 3. Camera mounted 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 are mentioned in the next section. Once the video is summarized, the master node

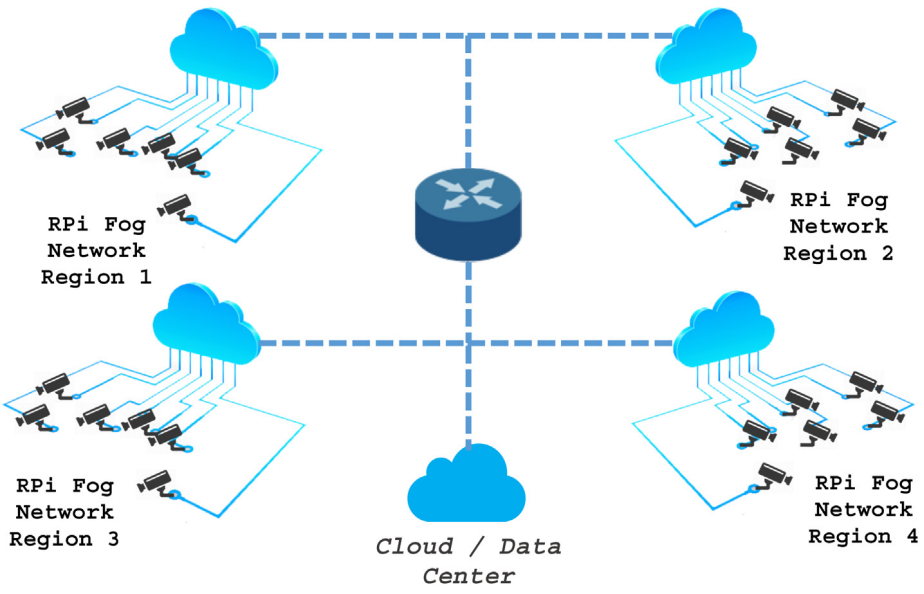


**Fig. 1.** Network layout for fog based surveillance network.

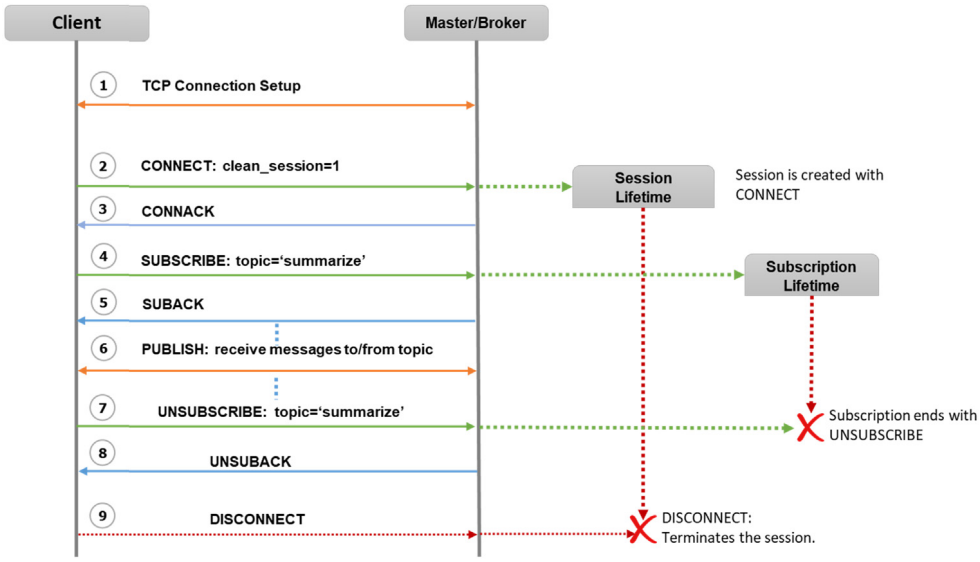


**Fig. 2.** Message flow between broker and client for initiating summarization process.

sends 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 significant bandwidth that would have been wasted, if it was to send the whole stream to the cloud.

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 need 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 [1]. Working of the MQTT protocol is presented in Fig. 4. MQTT works on even unreliable networks with some degree of assurance that the messages have 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 subscribed 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 h. Upon successful completion of the summary process, the processed summary video file is generated on master node and sent to the cloud periodically.

The summarization process is lightweight and is adapted from [8] 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 is quantized in the form of energy and is used as a criterion for *interesting event* detection. The process of event

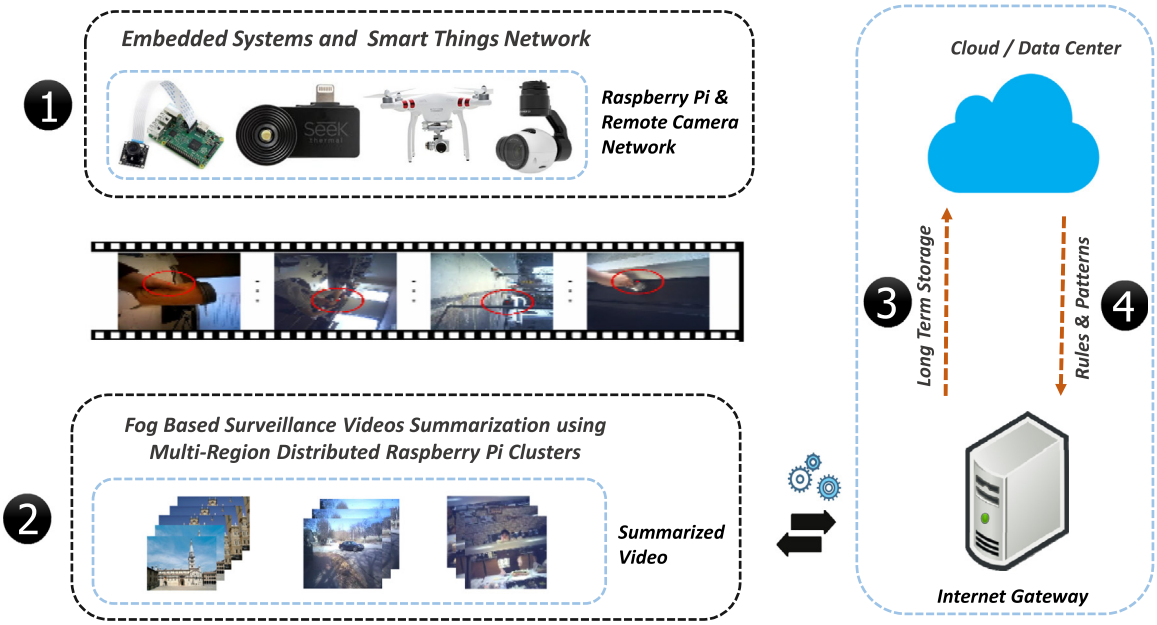


**Fig. 3.** Proposed framework for distributed summarization of surveillance videos.

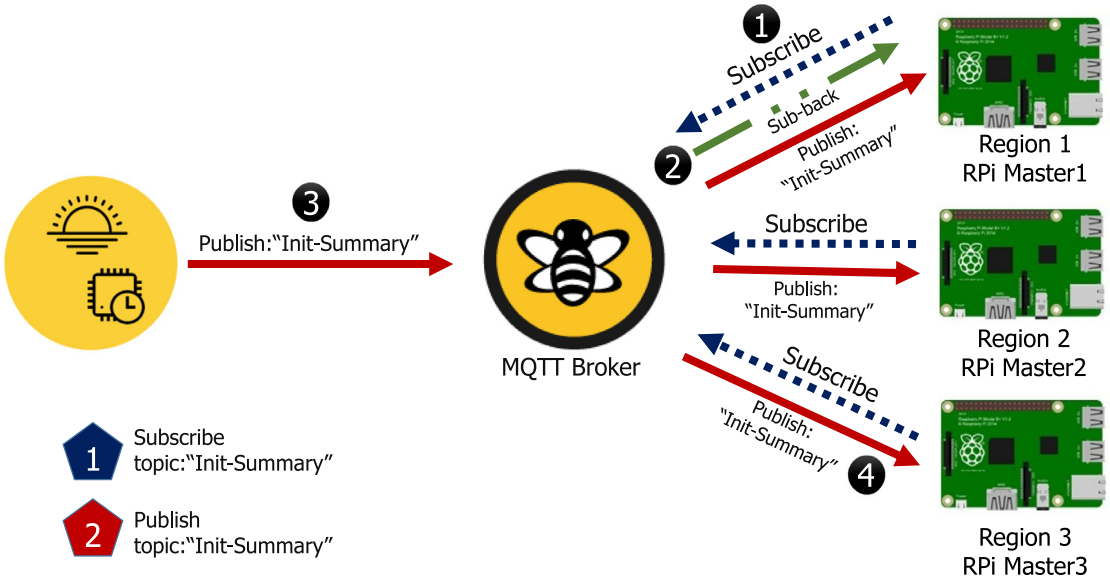


**Fig. 4.** MQTT framework for summary initializer module.

detection is divided into three steps. In the first step the absolute difference of two consecutive frames is calculated then, after finding the differences, 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 Eq. (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 is calculated and in step three, only the frames that crosses a fixed threshold is selected as keypoints which effectively represents the original video. The steps of the algorithms are presented in Table 1.

The algorithm works by processing an input video $\mathcal{d}$. The absolute difference $\alpha_0$ of two consecutive frames are calculated and stored in candidate list $\mathcal{C}_F$. 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 Eq. (1). The energy $\varepsilon(f)$ of each frame is calculated and stored in candidate list $\mathcal{C}_I$. The frame is considered as interesting only, if it crosses a fixed threshold $t_0$. The list $\mathcal{Y}$ is the ratio of both candidates' lists and event is considered as interesting only if the ratio crosses a fixed threshold $t_0$. These thresholds are selected after experimentation and vary widely, depending on type and condition of the camera calibration. In our case the threshold was fixed and no changes were made to the default setting of the Raspberry Pi camera. A simple clustering technique K-Nearest Neighbor is applied to the list $\mathcal{Y}$ to obtain a summarized video content that effectively represents the original video.

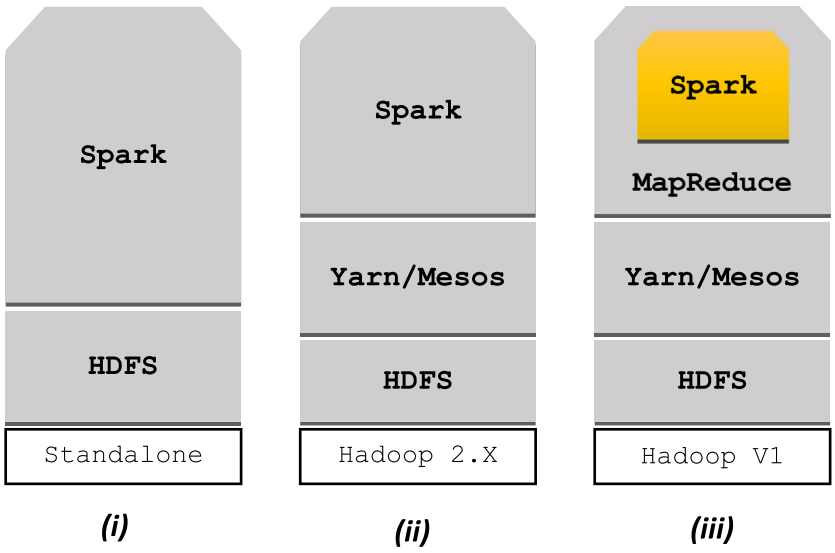


**Fig. 5.** Overview of the Hadoop's Distributed Filesystem.

**Table 1**
Algorithm for surveillance video summarization over the distributed Fog nodes.

**Input:** Surveillance video ɗ.

for f := 1 to len(ɗ)
  $\alpha_0 = abs(f_i - f_{(i-1)})$
  $Ç_F$ appends ($\alpha_0$)

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

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

## 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 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, 1.2 GHz processor, Video Core IV GPU, and an SD card slot along with 1GB RAM and 100MBps ethernet connection. The GPU of the Raspberry Pi is capable of video playback using H.264, which can play Blu-ray quality videos at 40MBits/s. The cluster of each region is composed of five nodes working as *slave* under one *master* node. Each of the connected node is installed with Apache Spark and Hadoop's Distributed File System (HDFS) [5,33]. 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 the *lite* version [11] of the Raspbian operating system to conserve more memory and processing power.

Apache Spark [40] 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 one submitted by 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 Fig. 5. The working of a general-purpose Spark cluster is usually in four 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 and generate a summarized video file.

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. Fig. 6 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 Slave Node3 *namenode* and multiple slave nodes called *datanode*. The master node acts as a server and it manages the access of shared files between different connected slaves. Besides this, the master nodes handle distribution of data between slaves and consolidating the output as well. Each block of file is divided and distributed across the cluster, the default size of each block is 64 MB and each of which is replicated across three nodes by default. We designed experiments to evaluate the proposed framework on different sizes of data and on single and multiple clusters. The file sizes are categorized in different problem sizes given in Table 2. 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.

In the first series of experiments we evaluate the Fog enabled Spark–Hadoop cluster for CPU usage, cluster throughput and network performance. Fig. 7 shows the complete evaluation of the system. At the first glance we observe somewhat predictable results and identify identical patterns for clusters, as it tends 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 compared to the only four available on a single

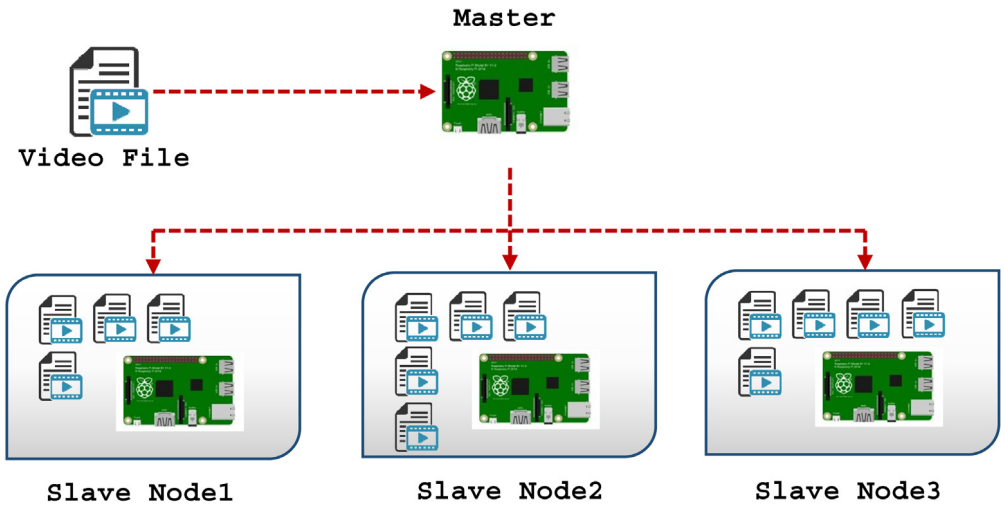


**Fig. 6.** File Distribution over Hadoop & Apache Spark.

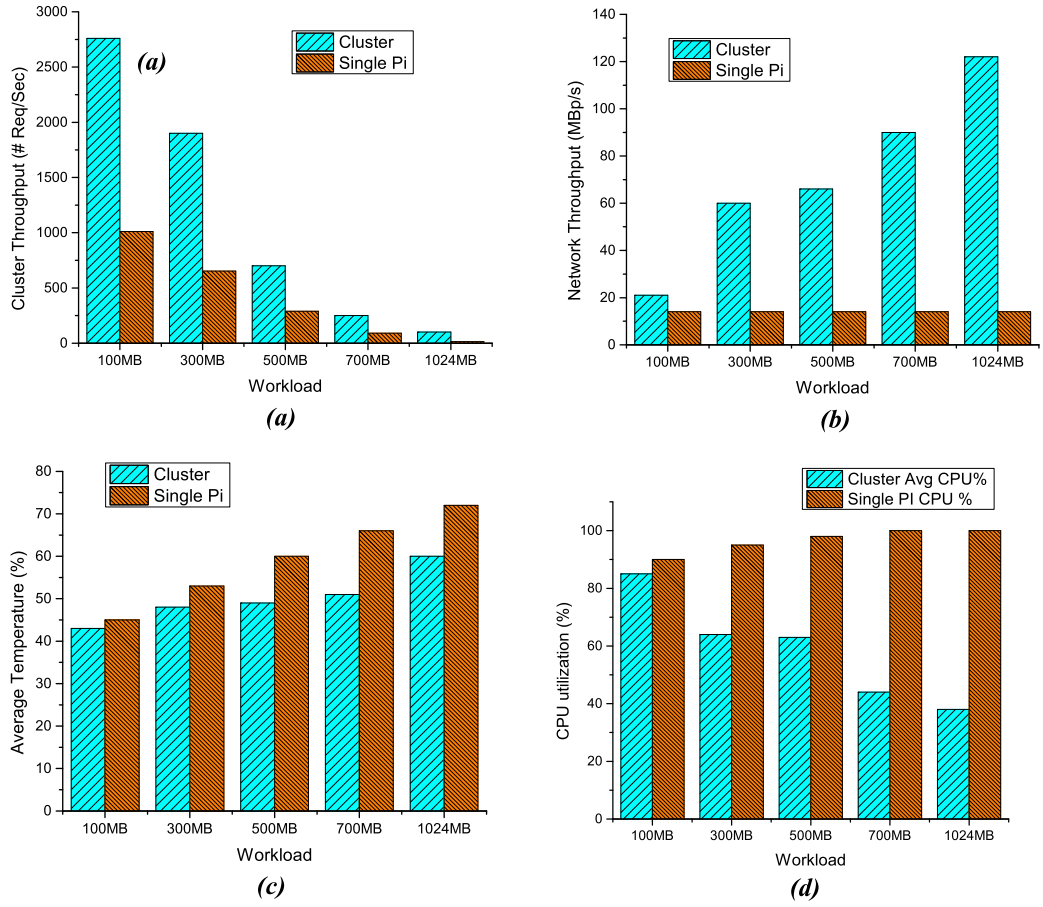


**Fig. 7.** Performance evaluation of the proposed architecture using (a) Cluster throughput vs Single System (b) Network Thoughput (c) Temperature Analysis (d) CPU Utilization.

Raspberry Pi. Fig. 7(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 Raspberry Pi significantly increases even with an installed air coolant. The network throughput Fig. 7(b) for the same cluster is very high for clusters as multiple video files are shared across the cluster for parallel processing. Whereas in this case the single Raspberry Pi does not have any overhead as it shares the data between its own cores and only communicate with the *namenode* at a constant 17 kb/s.

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 5 VDC fan connected to the GPIO pins of the Raspberry Pi. Fig. 7(c) shows that even then it is approaching very high temperatures and is pushing this small device to its limits. However, the average temperature of the rest of the cluster remains relatively low even

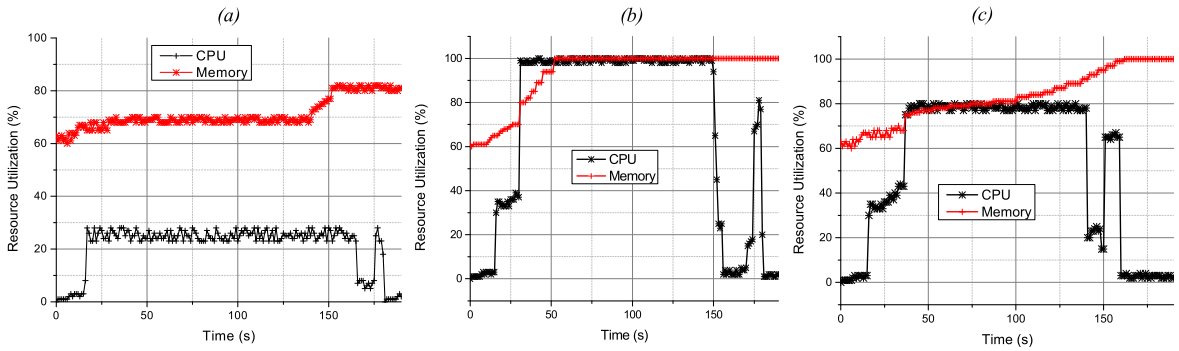


**Fig. 8.** (a) Small Video File (b) Medium Video File (c) Large File CPU and Memory usage of the Raspberry Pi cluster video summarization.

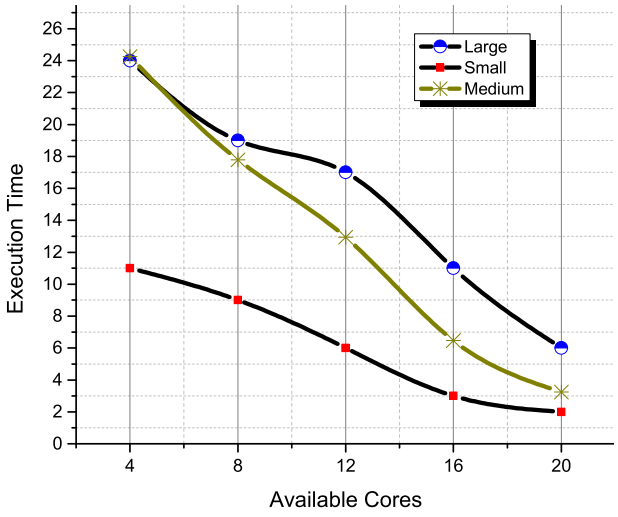


**Fig. 9.** Execution time of the summarization process on all available cores.

**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 |

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 Fig. 7(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.

In order to further evaluate the performance of the system we divide the video file to (i) Small 1 GB (ii) Medium 3 GB and (iii) Large 6 GB file respectively. Fig. 8 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 Hadoop and Apache Spark based clusters 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 is constant swapping in the memory and smaller jobs have to wait even if completed earlier.

The number of available cores in Raspberry Pi 3 is four, and we have combined 5 of them in one region of the Fog network. The total number of available cores are 20 at a given time, thereby limiting the computation to only 20 cores. Fig. 9 shows the total execution time of each file on each available core. The execution time is aggregated and presented in minutes. As we can observe the large files take much more time even though if the number of cores is fully available. On the contrary the smaller and medium jobs require less time and is more efficient if all cores are available.

In addition to the previous analysis, the network transmission of the fog network is depicted in Fig. 10. 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 and generate a single summarized video file.

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

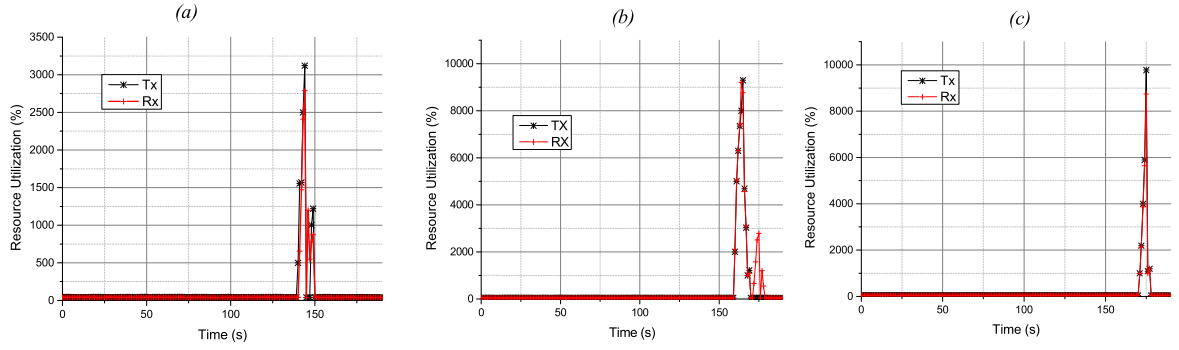


**Fig. 10.** (a) Small Video File (b) Medium Video File (c) Large File Network usage of the Raspberry Pi cluster for video summarization.

embedded vision are either based on single-system, or had not considered the powerful features that Fog computing offers.

## 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 can not only replace a costly cloud solution, but also holds potential for scalable for enabled applications without adding any significant cost. In the future, we will explore video summarization algorithms that are specifically build for resource constraint devices. A thorough review of different CNN based *nano* architectures and its application in video summarization schemes as well as its applicability on single board computer is also under consideration. 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 tasks such as deep learning models and deep learning enabled activity recognition for surveillance videos.

## References


[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, Internet of things: a survey on enabling technologies, protocols, and applications, IEEE Commun. Surv. Tutor. 17 (2015) 2347–2376.

[2] P. Bellavista, A. Zanni, Feasibility of fog computing deployment based on docker containerization over raspberrypi, in: Proceedings of the 18th International Conference on Distributed Computing and Networking, 2017, p. 16.

[3] K. Benson, C. Fracchia, G. Wang, Q. Zhu, S. Almomen, J. Cohn, et al., Scale: safe community awareness and alerting leveraging the internet of things, IEEE Commun. Mag. 53 (2015) 27–34.

[4] F. Bonomi, R. Milito, P. Natarajan, J. Zhu, Fog computing: a platform for internet of things and analytics, in: Big Data and Internet of Things: A Roadmap for Smart Environments, Springer, 2014, pp. 169–186.

[5] D. Borthakur, HDFS architecture guide, Hadoop Apache Project, 53, 2008.

[6] C.C. Byers, Architectural imperatives for fog computing: use cases, requirements, and architectural techniques for fog-enabled iot networks, IEEE Commun. Mag. 55 (2017) 14–20.

[7] S.J. Cox, J.T. Cox, R.P. Boardman, S.J. Johnston, M. Scott, N.S. O'brien, Iridis-pi:.a. low cost, Iridis-pi: a low-cost compact demonstration cluster, Cluster Comput. 17 (2014) 349–358.

[8] U. Damnjanovic, V. Fernandez, E. Izquierdo, J.M. Martinez, Event detection and clustering for surveillance video summarization, in: Image Analysis for Multimedia Interactive Services, 2008. WIAMIS'08. Ninth International Workshop on, 2008, pp. 63–66.

[9] A.V. Dastjerdi, R. Buyya, Fog computing: helping the internet of things realize its potential, Computer 49 (2016) 112–116.

[10] M. Elshenawy, B. Abdulhai, M. El-Darieby, Towards a service-oriented cyber–physical systems of systems for smart city mobility applications, Future Gener. Comput. Syst. 79 (2018) 575–587.

[11] R.P. Foundation, Raspbian, 2018. Available from: https://www.raspberrypi.org/downloads/raspbian/. (18 May 2018).

[12] M. García-Pineda, S. Felici-Castell, J. Segura-García, Adaptive SDN-based architecture using QoE metrics in live video streaming on Cloud Mobile Media, in: Software Defined Systems (SDS), 2017 Fourth International Conference on, 2017, pp. 100–105.

[13] V. Gupta, K. Kaur, S. Kaur, Developing small size low-cost software-defined networking switch using raspberry Pi, Singapore, 2018, pp. 147–152.

[14] G. Han, L. Liu, W. Zhang, S. Chan, A hierarchical jammed-area mapping service for ubiquitous communication in smart communities, IEEE Commun. Mag. 56 (2018) 92–98.

[15] G. Han, L. Zhou, H. Wang, W. Zhang, S. Chan, A source location protection protocol based on dynamic routing in wsns for the social internet of things, Future Gener. Comput. Syst. (2017).

[16] M.A. Hossain, B. Song, Efficient resource management for cloud-enabled video surveillance over next generation network, Mobile Netw. Appl. 21 (2016) 806–821.

[17] C. Huang, R. Lu, K.-K.R. Choo, Vehicular fog computing: architecture, use case, and security and forensic challenges, IEEE Commun. Mag. 55 (2017) 105–111.

[18] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, R.S. Tucker, Fog computing may help to save energy in cloud computing, IEEE J. Sel. Areas Commun. 34 (2016) 1728–1739.

[19] R. Khan, S.U. Khan, R. Zaheer, S. Khan, Future internet: the internet of things architecture, possible applications and key challenges, in: Frontiers of Information Technology (FIT), 2012 10th International Conference on, 2012, pp. 257–260.

[20] J.-H. Lee, G.-G. Lee, W.-Y. Kim, Automatic video summarizing tool using mpeg-7 descriptors for personal video recorder, IEEE Trans. Consum. Electron. 49 (2003) 742–749.

[21] J. Lloret, P.V. Mauri, J.M. Jimenez, J.R. Diaz, 802.11 g WLANs design for rural environments video-surveillance, in: Digital Telecommunications, 2006. ICDT'06. International Conference on, 2006, pp. 23–23.

[22] J.P. Mitchell, A.R. Young, J. Sangid, K.A. Deuso, P.J. Eckhart, T. Naderi, et al., Performance, management, and monitoring of 68 node raspberry pi 3 education cluster: big orange bramble (BOB), 2017.

[23] R. Morabito, A performance evaluation of container technologies on Internet of Things devices, in: Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on, 2016, pp. 999–1000.

[24] K. Muhammad, R. Hamza, J. Ahmad, J. Lloret, H.H.G. Wang, S.W. Baik, Secure surveillance framework for iot systems using probabilistic image encryption, IEEE Trans. Ind. Inform. (2018).

[25] M. Mukherjee, L. Shu, D. Wang, Survey of fog computing: fundamental, network applications, and research challenges, IEEE Commun. Surv. Tutor. (2018).

[26] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, T. Poggio, Pedestrian detection using wavelet templates, in: Computer Vision and Pattern Recognition, 1997. Proceedings. 1997 IEEE Computer Society Conference on, 1997, pp. 193–199.

[27] Z. Rasheed, M. Shah, Detection and representation of scenes in videos, IEEE Trans. Multimedia 7 (2005) 1097–1105.

[28] M. Sajjad, S. Khan, T. Hussain, K. Muhammad, A.K. Sangaiah, A. Castiglione, et al., CNN-based anti-spoofing two-tier multi-factor authentication system, Pattern Recognit. Lett. (2018).

[29] M. Sajjad, M. Nasir, K. Muhammad, S. Khan, Z. Jan, A.K. Sangaiah, et al., Raspberry pi assisted face recognition framework for enhanced law-enforcement services in smart cities, Future Gener. Comput. Syst. (2017).

[30] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, A. Oliveira, Smart cities and the future internet: towards cooperation frameworks for open innovation, Future Internet (2011) 431–446.

[31] N. Schot, Feasibility of raspberry pi 2 based micro data centers in big data applications, in: Proceedings of the 23th University of Twente Student Conference on IT, Enschede, The Netherlands, 2015.

[32] G. Secinti, B. Canberk, T.Q. Duong, L. Shu, Software defined architecture for vanet: a testbed implementation with wireless access management, IEEE Commun. Mag. 55 (2017) 135–141.

[33] K. Shvachko, H. Kuang, S. Radia, R. Chansler, The hadoop distributed file system, in Mass storage systems and technologies (MSST), in: 2010 IEEE 26th symposium on, 2010, pp. 1–10.

[34] M. Taha, L. Parra, L. Garcia, J. Lloret, An Intelligent handover process algorithm in 5G networks: The use case of mobile cameras for environmental surveillance, in: Communications Workshops (ICC Workshops), 2017 IEEE International Conference on, 2017, pp. 840–844.

[35] A.N. Toosi, J. Son, R. Buyya, CLOUDS-Pi: a low-cost raspberry-pi based testbed for software-defined-networking in clouds, 2017.

[36] F.P. Tso, D.R. White, S. Jouet, J. Singer, D.P. Pezaros, The glasgow raspberry pi cloud: A scale model for cloud computing infrastructures, in: Distributed Computing Systems Workshops (ICDCSW), 2013 IEEE 33rd International Conference on, 2013, pp. 108–112.

[37] L. Tung, Raspberry Pi: 14 million sold, 10 million made in the UK, 2018. Available from: https://www.zdnet.com/article/14-million-raspberry-pis-sold-10-million-made-in-the-uk/ (14 January 2018).

[38] L. Wan, G. Han, L. Shu, N. Feng, C. Zhu, J. Lloret, Distributed parameter estimation for mobile wireless sensor network based on cloud computing in battlefield surveillance system, IEEE Access 3 (2015) 1729–1739.

[39] Y. Wang, T. Zhang, D. Tretter, P. Wu, Real time motion analysis toward semantic understanding of video content, in: Visual Communications and Image Processing 2005, 2006, 596027.

[40] M. Zaharia, R.S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, et al., Apache spark: a unified engine for big data processing, Commun. ACM 59 (2016) 56–65.


**Mansoor Nasir** received his M.S. degree in Computer Science from Institute of Management Sciences, Peshawar, Pakistan. He is currently pursuing Ph.D. course from Islamia College, Peshawar, Pakistan. His research interest include, Network Security, Distributed Systems and Trusted Computing.

**Khan Muhammad** (S'16-M'18) received the Ph.D. degree in Digital Contents from Sejong University, South Korea. He is currently working as a researcher at Intelligent Media Laboratory (IM Lab), Sejong University, Seoul. His research interests include medical image analysis (brain MRI, diagnostic hysteroscopy and wireless capsule endoscopy), information security (steganography, encryption, watermarking and image hashing), video summarization, computer vision, fire/smoke scene analysis, and video surveillance. He has published over 50 papers in peer-reviewed international journals and conferences in these research areas with target venues as IEEE COMMAG, TII, TIE, TSMC-Systems, Access, TSC, Elsevier INS, Neurocomputing, PRL, FGCS, COMCOM, COMIND, JPDC, PMC, BSPC, CAEE, Springer NCAA, MTAP, JOMS, and RTIP, etc. He is also serving as a professional reviewer for over 40 well-reputed journals and conferences.

**Jaime Lloret** received his B.Sc.+M.Sc. in Physics in 1997, his B.Sc.+M.Sc. in electronic Engineering in 2003 and his Ph.D. in telecommunication engineering (Dr. Ing.) in 2006. He is currently Associate Professor in the Polytechnic University of Valencia. He is the Chair of the Integrated Management Coastal Research Institute (IGIC) and he is the head of the "Active and collaborative techniques and use of technologic resources in the education (EITACURTE)" Innovation Group. He is the director of the University Diploma "Redes y Comunicaciones de Ordenadores". He has been Internet Technical Committee chair (IEEE Communications Society and Internet society) for the term 2013–2015. He has authored 22 book chapters and has more than 450 research papers published in national and international conferences, international journals (more than 200 with ISI Thomson JCR). He has been the co-editor of 40 conference proceedings and guest editor of several international books and journals. He is editor-in-chief of the "Ad Hoc and Sensor Wireless Networks" (with ISI Thomson Impact Factor), the international journal "Networks Protocols and Algorithms", and the International Journal of Multimedia Communications. Moreover, he is Associate Editor-in-Chief of "Sensors" in the Section sensor Networks, he is advisory board member of the "International Journal of Distributed Sensor Networks" (both with ISI Thomson Impact factor), and he is IARIA Journals Board Chair (8 Journals). Moreover, he is (or has been) associate editor of 46 international journals (16 of them with ISI Thomson Impact Factor). He has been involved in more than 450 Program committees of international conferences, and more than 150 organization and steering committees. He has led many local, regional, national and European projects. He is currently the chair of the Working Group of the Standard IEEE 1907.1. He has been general chair (or co-chair) of 40 International workshops and conferences. He is IEEE Senior, ACM Senior and IARIA Fellow.

**Arun Kumar Sangaiah** has received his Doctor of Philosophy (PhD) degree in Computer Science and Engineering from the VIT University, Vellore, India. He is presently working as an Associate Professor in School of Computer Science and Engineering, VIT University, India. His area of interest includes software engineering, computational intelligence, wireless networks, bio-informatics, and embedded systems. He has authored more than 100 publications in different journals and conference of national and international repute. His current research work includes global software development, wireless ad hoc and sensor networks, machine learning, cognitive networks and advances in mobile computing and communications. Also, he was registered a one Indian patent in the area of Computational Intelligence. Besides, Prof. Sangaiah is responsible for Editorial Board Member/Associate Editor of various international journals.

**Muhammad Sajjad** received his Master degree from Department of Computer Science, College of Signals, National University of Sciences and Technology, Rawalpindi, Pakistan. He received his Ph.D. degree in Digital Contents from Sejong University, Seoul, Republic of Korea. He is now working as an assistant professor at Department of Computer Science, Islamia College Peshawar, Pakistan. He is also head of "Digital Image Processing Laboratory (DIP Lab)" at Islamia College Peshawar, Pakistan, where students are working on research projects such social data analysis, medical image analysis, multi-modal data mining and summarization, image/video prioritization and ranking, Fog computing, Internet of Things, virtual reality, and image/video retrieval under his supervision. His primary research interests include computer vision, image understanding, pattern recognition, and robot vision and multimedia applications, with current emphasis on raspberry-pi and deep learning-based bioinformatics, video scene understanding, activity analysis, Fog computing, Internet of Things, and real-time tracking.