Journal of Network and Computer Applications 232 (2024) 104026

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

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

Research paper

Distributed Fog computing system for weapon detection and face recognition

Héctor Martinez *, Francisco J. Rodriguez-Lozano, Fernando León-García, Jose M. Palomares, Joaquín Olivares

Universidad de Córdoba, Department of Electronic and Computer Engineering, Córdoba, Spain

ARTICLE INFO

Keywords:
Weapon detection
Face recognition
Distributed computing
Fog computing
Deep Learning

ABSTRACT

Surveillance systems are very important to prevent situations where armed people appear. To minimize human supervision, there are algorithms based on artificial intelligence that perform a large part of the identification and detection tasks. These systems usually require large data processing servers. However, a high number of cameras causes congestion in the networks due to a large amount of data being sent. This work introduces a novel system for identifying individuals with weapons by leveraging Edge, Fog, and Cloud computing. The key advantages include minimizing the data transmitted to the Cloud and optimizing the computations performed within it. The main benefits of our proposal are the high and simple scalability, the immediacy of the detection, as well as the optimization of processes through distributed processing of high performance in the Fog layer. Moreover, the structure of this proposal is suitable for 5G camera networks, which require low latency and quick responses.

1. Introduction

Detection and surveillance systems have evolved as technology. However, despite the many improvements, in most cases, they remain non-automated. The number of cameras that are being installed is growing exponentially, while the volume of information that one person can monitor is limited. One of the main tasks of surveillance camera supervision is the identification of armed people.

The expansion of the Internet of Things (IoT) (Dhirani et al., 2017), Computer Vision, and, Deep Learning, have allowed the emergence of new automated systems (Mohanapriya and Mahesh, 2020), especially success in smart cities applications (Songhorabadi et al., 2023). However, this solution is partial, because, at the same time, the number of hardware devices that capture information that must be sent to the Cloud and processed also increases. In fact, optimizing communications in sensor networks aimed at detecting events is a subject of recent scientific interest (León-García et al., 2018, 2019; Cob-Parro et al., 2021). Identifying large data transfers in a computer network is very important for understanding and managing network traffic. However, current methods for measuring network traffic in real-time have problems because they struggle to accurately identify these big data transfers. This is because some data transfers are much larger than others, and the speed at which data is transferred can change quickly. As a result, existing methods often make mistakes and have limited accuracy, especially when they try to use only a small amount of memory to store information (Xiong et al., 2024).

Edge computing could allow data to be processed close to the end devices instead of sending raw data to be processed in the Cloud, reducing communications and Cloud computing costs (Tuli et al., 2023). However, Edge devices usually are restricted by limited resources and low-performance computing. So, to solve these limitations, Fog computing emerges as an interesting paradigm, providing network infrastructure with high-performance resources (Zolghadri et al., 2024). Energy consumption is also a challenge in camera sensor networks (Castillo-Secilla et al., 2010; SanMiguel and Cavallaro, 2017; Dao et al., 2017), so saving data communications and the number of transmissions is imperative (Azizi et al., 2022), furthermore when the network nodes are mobile (Ostrowski et al., 2023). Moreover, Fog computing allows distributed processing. This feature is also interesting because several nodes could process different algorithms for different purposes, for instance, to identify drones appearing in the scene (Taha and Shoufan, 2019), vehicles and their license plate (Olivares et al., 2010), people suffering fever (Rodriguez-Lozano et al., 2019), animals (Zhou et al., 2021), etc. Fog computing together with 5G networks propitiated a new technology called Open-RAN (ITU-T, 2018). Open-RAN technology (Liyanage et al., 2023) is a new trend that facilitates the inclusion of network infrastructure elements with processing capacity (Fog computing) for 5G and 6G Next Generation networks (iGillott, 2020; Deloitte, 2021).

* Corresponding author.
E-mail addresses: el2mapeh@uco.es (H. Martinez), fj.rodriguez@uco.es (F.J. Rodriguez-Lozano), fernando.leon@uco.es (F. León-García), jmpalomares@uco.es (J.M. Palomares), olivares@uco.es (J. Olivares).

https://doi.org/10.1016/j.jnca.2024.104026
Received 11 March 2024; Received in revised form 24 July 2024; Accepted 6 September 2024
Available online 11 September 2024

The main motivation of this work is to present a new surveillance system for weapon detection and face recognition, based on the efficient use of computational resources in the Fog layer. The main benefits are optimizing communication bandwidth and saving transmission energy, avoiding data and computing saturation in the Cloud layer.

This paper is organized as follows: Section 2 presents the state of the art of the computing paradigm and surveillance systems. Section 3 shows how is designed the new surveillance system. Results are presented in Section 4, followed by discussion in Section 5. Finally, the main contributions and future works are presented in Section 6.

2. Background

In this section, we present the foundations of two different technologies covered in this paper: the computing paradigm for the IoT and proposals for weapons detection systems.

2.1. The IoT computing paradigm

Fog computing (FC) has become a linchpin for Fifth Generation (5G) networks (Meng et al., 2020). FC and 5G are updating the data transmission using new technological approaches that are intelligently processing data to provide enhanced communications. Fog computing being an extension of cloud computing plays an important role along with 5G networks (Khalid et al., 2020). Fog computing aims to improve the latency and response time by providing the computer and storage facilities near to the end-user connected with 5G networks (Buyya and Srirama, 2019).

- Edge computing (Dao et al., 2020) is based on the principle of making the processing as closer to the data generators as possible. Wireless sensor networks (WSN) and most IoT devices suffer from low-performance computing with strict restrictions due to the low energy available. Typically, Edge computing performs pre-processing tasks and adapts data for Fog or Cloud layers.
- Fog computing (Zolghadri et al., 2024) provides cloud services and high-performance computing using network infrastructure nodes, allowing distributed computing and low latency response to the edge devices.
- Cloud computing (Bhowmik, 2017) is the most popular computing paradigm for IoT applications based on artificial intelligence technology during the last decades. The cloud provides several services to connected clients. Besides, it supplies a computing infrastructure composed of large servers equipped for high data storage and high-performance computing.

2.2. Detection and classification algorithms

In order to provide a new surveillance system that is able to detect weapons and recognize the faces of the weapon holders for the identification of those persons, we will review the image processing techniques for detection and classification. The most popular algorithms used for detection and classification could be divided into two categories (Warsi et al., 2020): Computer Vision Algorithms and Deep Learning Algorithms.

The Computer Vision approaches are based on understanding the internal structure of the images. Current trends in Computer Vision Algorithms for feature detection are mainly designed using color segmentation, interest points, shapes, and edge detectors. Active Appearance Models are based on matching the shape and appearance of objects to a new image using a statistical model (Cootes et al., 2001). Harris corner detector algorithm is commonly used in extracting corners of an image (Harris and Stephens, 1988) as a previous step for further identification stages. Different elements in an image are usually conformed with different colors. Therefore, color-based segmentation is another technique widely used to make the first step of a detection method. This type of segmentation can be obtained using the well-known and efficient k-means algorithm (Tiwari and Verma, 2015). However, color-based segmentation (Sasikaladevi and Mangai, 2018) could fail if the object shares the same color with its environment. Other techniques rely on detecting the outline of the elements in the images. For this purpose, edge detectors are used with a large variety of methods. However, edge detector algorithms could be strongly affected by variations in the light conditions. Thus, low light scenes could lead to undetect many contours of objects. To overcome this problem, the LIP-Canny algorithm (Palomar et al., 2010) was proposed. It is an edge detection algorithm based on a robust mathematical model closer to the human vision system, which is invariant to illumination variations. However, it increases the computational cost.

In conclusion, most of these detection methods depend on the quality and angle of the image. Images with noise and occlusion could be badly detected. It has been proved that it is hard to detect the corners, shapes, and edges in occluded images (Sánchez et al., 2018).

Deep Learning (DL) Algorithms are also used for detection and classification. These algorithms are based on deep Neural Networks (NN) (Zhao et al., 2019), such as, Convolutional Neural Networks (CNN) (O'Shea and Nash, 2015), Faster R-CNN (Ren et al., 2017), and You Only Look Once (YOLO) (Redmon et al., 2016). These algorithms learn the features during a training process. They require a lot of data for training. Moreover, they are even able to detect occluded objects if a huge amount of data is provided. In the case of CNN, Faster R-CNN, and YOLO, the data needs to be labeled before the training, this is supervised learning. Thus, the learning process for the detection is made by making the computer find internal inferences without providing specific procedures for obtaining the desired features of the objects. This is the main difference between these techniques based on DL and those techniques based on Computer Vision. For the learning process, two important datasets are used: Imagenet (Fei-Fei et al., 2004) and Terahertz Human Dataset (Zhang et al., 2018). These datasets present 26,505 images with the human body as the main reference. However, several authors used other different customized datasets.

We will briefly describe the main DL techniques in the following paragraphs.

- CNN (O'Shea and Nash, 2015) consists of an architecture composed of an input layer, several hidden layers, and one output layer. In any feed-forward neural network, any middle layers are called hidden ones, because their inputs and outputs are masked by the activation function and final convolution. In a convolutional neural network, the hidden layers include some layers that perform convolutions. Although CNNs were invented in the 1980s, their breakthrough in the 2000s required fast implementations on graphics processing units (GPUs). To check the accuracy of the network, error functions are used. If the error rate is too high, then it will pass back to the first layer and update the weights where required. A technique known as backpropagation is used to calculate the weights that need to be updated. The process is repeated until the required level of accuracy is achieved. Asrith et al. (2018) trained their CNN by using very low-resolution images to detect and recognize human faces and weapons in real-time. However, their results and discussion are more focused on face detection and do not reflect any information about weapon detection. Gelana and Yadav (2019) proposed a method to decrease the complexity and rise the accuracy by using edge information as a feature to identify firearms.
- CNN OverFeat (Sermanet et al., 2014) are based on a sliding-window approach, where the classifier is trained on the center image first. Afterward, the classifier is applied to every location of the target image. The main drawback of using any sliding-window approach is its slowness. Therefore, it is not appropriate for real-time object detection.

- R-CNN (Girshick et al., 2014) extracts 2000 regions warped into squares and forwarded to a CNN. This neural network generates a 4096-dimensional feature vector. Then, it performs a regression algorithm to find the bounding boxes of the classified object. This method requires huge computation power. Thus, it is not appropriate for real-time object detection.
- Fast R-CNN (Girshick, 2015) the same approach as R-CNN. But here, it does not calculate the regions. Instead, it feeds the image directly to CNN and generates feature maps. From this convolutional feature map, the region proposals are identified and warped into squares, and using the ROI (region of interest) pooling layer, the shape changes to a fixed size, which is forwarded to a fully connected layer. A softmax layer is used to predict the class and bounding box from the ROI.
- Faster R-CNN (Ren et al., 2017) is a full CNN consisting of two networks. The first network provides the region proposals and the second one uses the proposed regions to detect and classify the objects. The bounding boxes (BB) list, the label of each BB, and the probability of each BB are obtained when an image is processed. The images are always represented as height, width, and depth, and they are known as tensors. The tensor (image) is then passed through a convolutional model which, is pre-trained until it reaches an intermediate layer and produces a feature map. Afterward, it goes to Region Proposal Network. The Region Proposal Network (RPN) uses the features to propose the regions which may have some objects. Anchors are used by RPN to solve the variable-length problem. They are boxes of fixed size placed on the image having multiple rations and sizes and will be used for predicting the location of objects. Once a candidate list of objects with locations is obtained, objects are classified.
- YOLO (Redmon et al., 2016): The algorithm “only looks once” requires the image or the video to pass through the neural network exactly only once to make the predictions. A single CNN simultaneously predicts multiple bounding boxes and class probabilities for those boxes. This basically means that it recognizes where the object is, marking the position with a BB. Moreover, at the same time, YOLO uses class probabilities to determine what the object is. The next step to calculate the BB implements two key post-processing steps: Intersect over Union (IoU), and Non-maximum suppression (NMS).

All these techniques produce several BBs that contain the detected instances of any of the desired objects. Thus, further processing may be applied to obtain richer information based on those BBs in later stages.

2.3. Smart weapon and people detection systems

Computer Vision algorithms require precise techniques to detect some specific features on the desired objects to be properly detected. On the other hand, DL algorithms are suitable for the extraction and detection of any object, as long as enough data is provided to the neural networks to learn those objects. However, it is possible to specify some enhancements to the extraction procedure to reduce the processing time and to rise the accuracy with both approaches. Therefore, there are some efforts made on the detection of weapons that should be remarked on. Most recent algorithms for handgun detection and knife detection could be divided into two categories (Warsi et al., 2020): Deep Learning Algorithms and Computer Vision Algorithms.

- Olmos et al. (2018) implemented a DL algorithm for automatic alarm when a handgun is detected. Extending this work, Castillo et al. (2019) used CNN to automatically detect cold metallic weapons in a surveillance video. However, these works are limited to gun-type weapons, and the datasets are trained using cameras when people are frontal and well-focused, images are clear, and obtained in a short distance, so, this is not a realistic camera deployment for detecting threats.
- Narejo et al. (2021) use YOLO v3 (Redmon et al., 2016), however, they divide the processing flow in the same equipment into three stages corresponding to the pre-processing of the images in which the detection of objects is carried out, their analysis, and the decision-making to send alerts of those cases when weapons are detected.
- Ashraf et al. (2022) use YOLO-v5s to speed up all the processing of detecting pistols by learning from 3.000 positive images and 12.000 negative ones. They are able to obtain large recall values (99%), while reducing the processing time from 0.19 s per frame using the Faster R-CNN, used as the baseline comparison, to 0.01 s per frame using their proposed mechanism based on YOLO-v5s. However, the processing is centralized in a single quite powerful computer (a 2.8 GHz Intel Core i7 processor MacBook Pro with 16 GB RAM and an Intel IRIS Pro GPU with 1536MB).
- Bhatti et al. (2021) carry out a weapon segmentation study using images from CCTV surveillance cameras. They analyze the operation of different deep learning neural networks concluding that YOLO is the one that yields the best results, being possible to work with real-time restrictions.
- Nakib et al. (2018) presented a system for detecting guns, rifles, and knives, as we do. They use Rectified Linear Unit (ReLU), Convolutional Layer, Fully connected layer, and dropout function of CNN to reach a result for the detection. They realize centralized computation without data optimization.

In conclusion, most works published in reputable scientific journals and conferences are dedicated only to gun detection, however, we detect rifles, guns, and knives. To the best of our knowledge, they do not include face detection and recognition after the arm detection, as we do. Arm detection is usually realized in Cloud/Server, and sometimes in Edge, without taking advantage of Fog possibilities, as stated in the following recent surveys (Yadav et al., 2023; Santos et al., 2024; Debnath and Bhowmik, 2021).

2.4. Centralized and decentralized network infrastructures

Current trends in the network infrastructures are the pathway leaded by wireless global communications, such as 4G and 5G. The 5G communications network is the fifth generation of wireless broadband technology. 5G enables ubiquitous communication transparent to the user. It promises significant improvements over current 4G/LTE standards: Extremely high data transfer rates of up to 10 Gbps; Very low latency; High robustness; Very low power consumption; and, Significant processing of more network subscribers. 5G wireless technologies will increase bandwidth, improve QoS, provide better usability and security, reduce delays, and lower the total cost of service. These 5G technologies are expected to not only provide higher throughput and lower latency but also higher connectivity density and mobility range without compromising reliability. However, in order to achieve all these improvements, 5G requires efficient usage of both communications technologies and processing resources. The key for this is to focus on intelligent services that provide very fast responses to the users whilst keeping the network usage as low as possible.

Therefore, to improve the mobile communication technology, 5G enhances its capabilities by three main properties: enhanced Mobile BroadBand (eMBB), massive Machine Type Communication (mMTC), and Ultra-Reliable Low-Latency Communication (URLLC). The main concept of eMBB is to extend frequency resources to a millimeter wave (mmWave) above 6 GHz, while mMTC is defined as the network capacity to simultaneously host millions of devices in an area of 1 km. Lastly, URLLC is the feature that guarantees end-to-end maximum latency of 10 ms. Currently, Cloud Computing is the most used paradigm for network-based processing. The most common architecture to support it is the Centralized Cloud Computing (CCC) (Anitha et al., 2024; Lee et al., 2020). In the CCC architecture, the application server is located

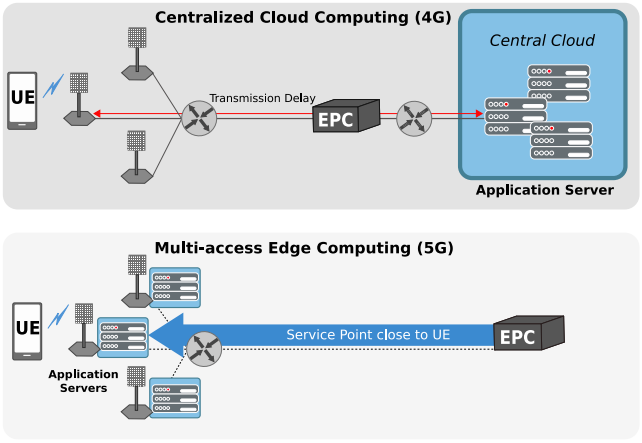


Fig. 1. Centralized Cloud Computing (CCC) and Multi-access Edge Computing (MEC) architectures (Lee et al., 2020).

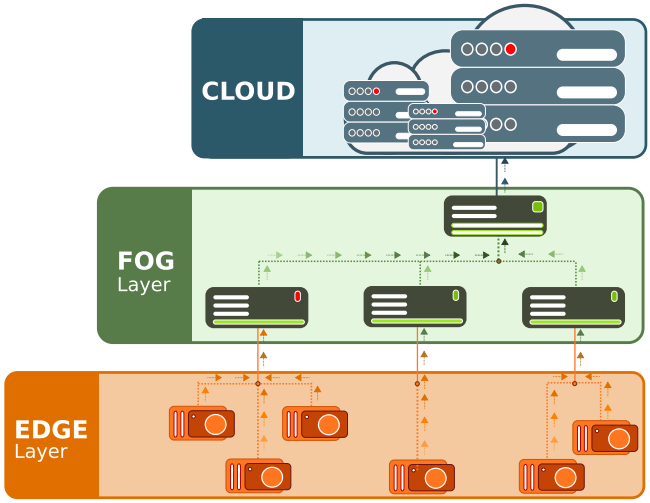


Fig. 2. Network layers.

in the central cloud, causing a delay if the physical distance between the central cloud and the user is high. Thus, the CCC architecture causes the latency to grow large. Moreover, the network is largely used, which may lead to network congestion.

On the other hand, to address the challenges in reducing core network latency, a new network architecture called Multiple access Edge Computing (MEC) has been developed (ETSI Industry Specification Group, 2019). The main functionality of MEC technology is to reduce the physical distance between the user and the application server (Mahbub and Shubair, 2023). For this, the network functions are virtualized in the cloud, in which the application servers are defined, and through the MEC architecture the computing resources which are closest to the user are assigned. MEC architecture allows you to reduce transmission delay by placing the application server and compute resources close to the destination. In short, it allows users to overcome the limitations of mobile terminals and reduce energy consumption while reducing the workload of the cloud. Fig. 1 shows the current 4G architecture (CCC) and the new MEC architecture, as proposed by 5G to achieve URLLC.

3. Methodology

We propose to apply a distributed processing system based on Deep Learning for the detection, classification, and identification of armed persons. This will mitigate the data load that will reach the equipment located at the cloud computing level. As shown in Fig. 2, the processing will be distributed allowing to take advantage of the characteristics of the different devices using the *Fog Computing* approach.

The information will be captured and pre-processed in the Edge layer. Later, the processing will be executed at various levels in the Fog layer. Finally, if it is necessary to identify a person, only the previously processed information will be sent to the Cloud layer. A general scheme can be seen in Fig. 3. The following sections will detail the algorithms implemented in each layer.

3.1. Edge layer

This layer is responsible for capturing the images from the camera sensors. These nodes are equipped with microcontrollers that allow the following tasks to be carried out in consecutive order, as shown in the left part of the diagram of Fig. 4:

- Capture images from camera sensors.
- Apply Background Subtraction (BS). BS is a common and widely used technique for generating a foreground mask (FM). The FM is a binary image containing the pixels belonging to moving objects in the scene. BS calculates the FM performing a subtraction between the current frame and a background model, containing the static part of the scene or, in general, everything that can be considered as background given the characteristics of the observed scene.
- If the changes in the image exceed a certain threshold, the image is sent for processing to the Fog layer, otherwise, it is discarded.
- Finally, the background is updated to adapt to possible changes in the scene.

The pre-processing at the edge layer will only send the relevant information, thus reducing the computational load due to computing unnecessary frames.

3.2. Fog layer

In the Fog layer, the detection of people and weapons is carried out. Besides, in case of a positive detection, the assignment of the weapon to the closest person is carried out, and the face of the assigned person is extracted. Tasks in the Fog layer are performed on two levels, as shown in Fig. 3.

In the first level, the images from the Edge layer are received and the following tasks are carried out:

- Detect people and obtain the region of interest (ROI) that includes each one.
- Detect weapons and obtain the ROI that includes each one. The weights for training the NN have been generated using our own database, composed of 15,600 images.
- Send only the Bounding Box (BB) region of each armed person to the second level of the Fog layer.

In this first level, the information to be processed is reduced with respect to the original, since the image is partitioned to analyze only the parts of interest. To perform an efficient segmentation of the image bodies in devices that do not have a computing capacity as high as those found in the cloud, the use of the YOLOv5 (Jocher, 2021) deep learning neural network is proposed. Pre-trained weights used for people detection are the YOLOv5l6 set.

YOLOv5l6 offers high performance as well as fast computation in high-power GPUs. We propose the application of YOLOv5s because is

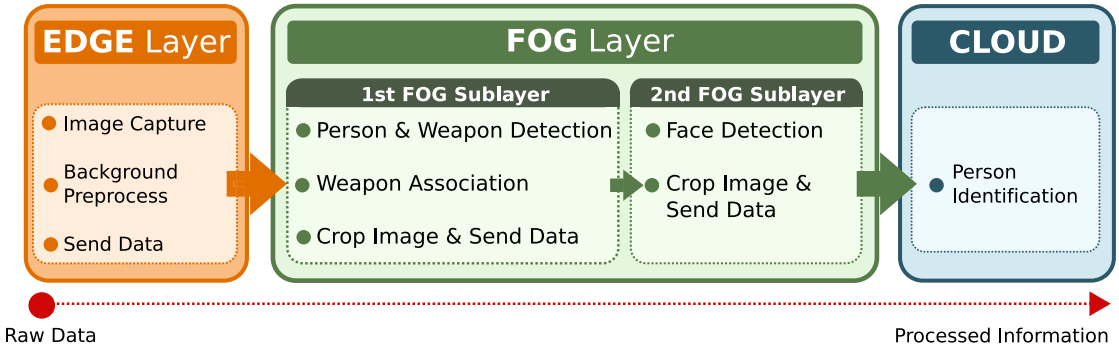


Fig. 3. Block diagram of the processing methodology.

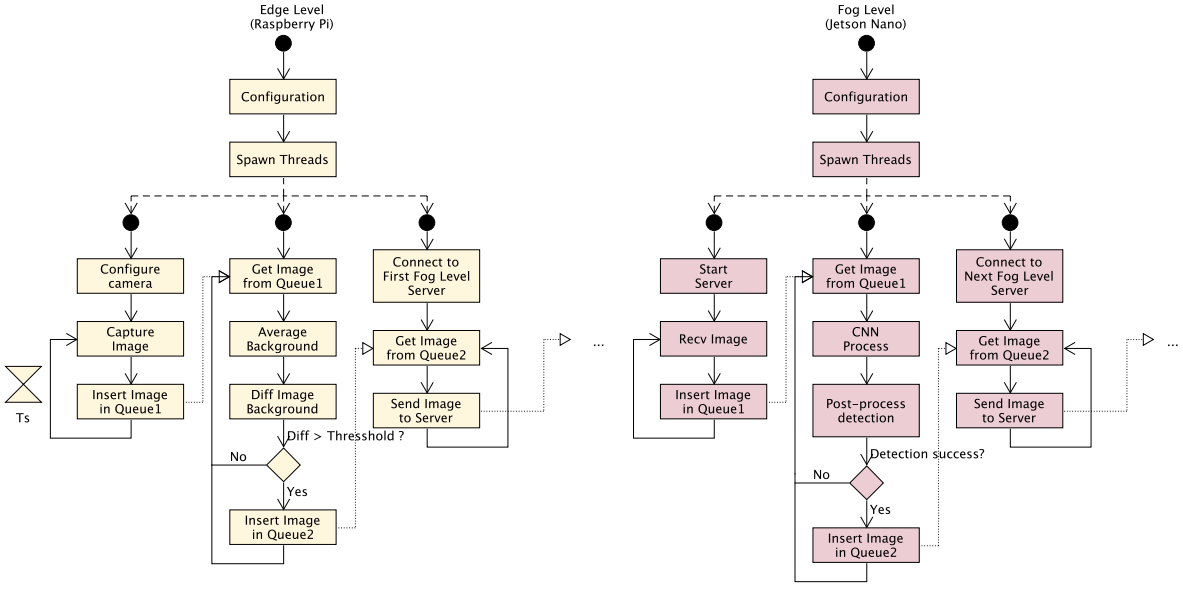


Fig. 4. Execution flow for Edge and Fog nodes.

designed for low-resource devices in the Fog layer, instead YOLOv5l6, or upper, which requires larger memory sizes and thus, it penalizes the overall performance.

In the second level of the Fog layer, the processing nodes are focused on obtaining the face of the subject detected with weapons. For this substage, the weights of the NN have been generated by training the neural network using YOLOv5s6. The tasks that these nodes must perform are:

- Receive the BB with the bodies of the armed people from the first level of the Fog layer.
- Use the neural network to segment the faces of interest.
- Send the face of the detected person to the Cloud layer for identification. In those cases where there is no person with a weapon, no information is sent.

3.3. Cloud layer

This final layer is exclusively responsible for the identification of people. The computation is based on the face-recognition library (Geitgey, 2020) written in Python. In it, the face is matched with a database to identify the person and, thus, to determine if that person is allowed to carry a weapon or not. For instance, a police person is allowed to carry a gun. Therefore, a gun would be detected. Afterward, the person who carries that gun is detected. A suspicious event is sent to the Cloud Layer, in which the person is identified as an authorized person and finally, no alarm would be raised. On the other hand, if the detected person is not identified as an authorized one, the alarm is activated.

3.4. Consumption analysis

For the evaluation of energy consumption, an evaluation model will be built based on measurements of computing time and power consumed by the devices in typical scenarios. The following methodology is applied to both the SmartFog proposal, which uses a distributed pipeline of devices, and to a centralized solution where the entire process runs on a Jetson Nano with 4 GB of RAM, for the purpose of comparison.

We used 5 videos, all of them contains between 1300 and 2000 frames. They represent these scenarios:

1. No people
2. One person without a weapon.
3. Two persons without weapons.
4. One person with a weapon.
5. Two persons with weapons.

We measure the number of processed frames and the time interval that its execution lasts for every stage in the pipeline of the processing:

1. Background analysis.
2. People and weapon detection.
3. Face detection.

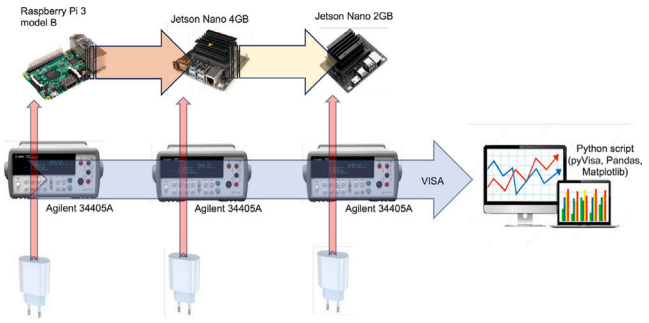


Fig. 5. Diagram of power consumption of the devices.

To measure power consumption, each device's official Power Adaptor is connected to an Agilent 34405A Digital Multimeter to monitor the current flowing through the cable supplying power to each device. The multimeter is connected to a central computer to gather the recorded data. This device has very low input resistance (0.01 Ω at 12 V), so it does not significantly affect the current in the power cable. The multimeter can detect current with a maximum resolution of 0.001 mA at 20 Hz. It also has a USB port for exporting the data using a high-speed connection through the VISA protocol. The VISA specifications, previously maintained by the VXI plug & play Systems Alliance, are now managed by the IVI Foundation. The central computer runs a Python program to receive and save the data from each multimeter. The measurement diagram is shown in Fig. 5.

The data is used to create a graph showing instantaneous power over time. The time axis of the graph is divided into segments based on different scenarios in the scene. The values are averaged using a sliding window to determine the average power for each scenario and device. The energy costs associated with various activities, such as network transmissions, memory accesses for video processing, and similar tasks, are factored into the average power for each scenario.

Using data from processing time and device consumption, we build an analytical model based on frame count to evaluate SmartFog against the alternative centralized solution. Results of the measurements and the final model obtained are described in Section 4.

4. Experimentation

To validate the system, a demonstrator has been deployed with the structure shown in Fig. 2.

- Edge layer: Composed by camera sensors and Raspberry Pi 3 (RPi3) devices.
- Fog layer: Composed by NVIDIA Jetson Nano devices.
- Cloud layer: Composed by an Intel i7–8700 server equipped with a NVIDIA GeForce RTX 2070 GPU.

All the internal hardware, operating system, and software components of both the Edge and Fog layer nodes are depicted in Fig. 6.

The system architecture is described in Fig. 7. Every Fog device of the first level can attend multiple sensor cameras from Edge, and every Fog device of the second level can attend multiple Fog devices from the first level.

This architecture permits low-cost camera devices connected dynamically with GPU resources in the Fog layer. So, the number of GPUs is drastically reduced, favoring the scalability of the network.

The detection and segmentation of people and weapons in the first Fog layer was carried out using Jetson Nano. Timing results were: Preprocessing 0.01 s + Inference 0.43s . To validate the system, different tests have been carried out considering different people with and without weapons in a scene.

4.1. Experimental cases

In order to test the system in real environments, five different schemes have been designed:

4.1.1. Case 1: No person (Background)

Real environments, either indoor or outdoor, affected by changing light conditions without any person are used in this case study. The illumination does not change abruptly, but as in real life, with casted shadows and direct sunlight depending on the scene. The RPi3 edge node is used to detect changes in the background, so, it is expected that no (or little) communication is established with Fog-level devices.

4.1.2. Case 2: One person - no weapons (Body)

Scenes are composed of a single person walking or standing with some illumination variations as in the previous case study. The person on the scene is not carrying or holding any weapon. The case study is expected to make use of the Edge level and the first stage of the Fog level, for person detection. As no weapon is detected no further communication is expected to arise.

4.1.3. Case 3: One person - one weapon (Body + Weapon)

Videos containing a single person standing or walking around and holding a gun are used. These videos have been shot either indoors or outdoors, with non-uniform illumination. The detection of a weapon will provoke a communication transfer for person identification at the Cloud level.

4.1.4. Case 4: Several persons - no weapons (NBodies)

This study case is an evolution of Case 2, with several persons. They are standing or walking around without any weapons. Scenes have been recorded both indoors and outdoors with varying illumination. As there is no weapon, no communication to Cloud is expected to appear.

4.1.5. Case 5: Several persons - several weapons (NBodies + NWeapons)

The final and most demanding test case is when several persons are in the frame holding guns. Some of the persons may be holding several guns at the same time and other persons may be holding none. Moreover, the persons are not holding the weapons all the time during the whole scene length. In this case, the identification process of each person holding a weapon sends messages to the Cloud level, so a higher number of communications will be expected.

4.2. Visual results

First, we will show the effectiveness of the proposal by providing some visual results. These results reveal that the persons and guns are detected in a proper way. In Figs. 8 and 9, two outdoor and indoor images, respectively, from "Case 5" scenario, of the same infrastructure are shown. It is to be remarked that the surveillance cameras are taken from a high angle, and at a large distance from the gunmen (further than 5 m.) Despite that, the guns and the associated persons handling them are correctly detected.

We detect also rifles as it is shown in Figs. 10 and 11, and knives in Figs. 12 and 13. These outdoor images are chosen because Fig. 10 present several armed military units in a real situation, and the picture does not present the arms at the front of the scene. Since the red and orange bounding boxes overlap, the rifle of the red man is assigned to the orange one, these situations are the actual limit and challenge of computer vision systems based on deep learning. Fig. 11 presents real military training, detected correctly. and finally, Figs. 12 and 13 show some terrorists one of them with a knife.

Each detected element (person, pistol) is remarked by a colored box with the title of the detected element and the confidence. For instance, in Fig. 8 the gunman is detected as a person with a 0.83 confidence, whilst the pistol is detected with a 0.59 confidence. Moreover, there

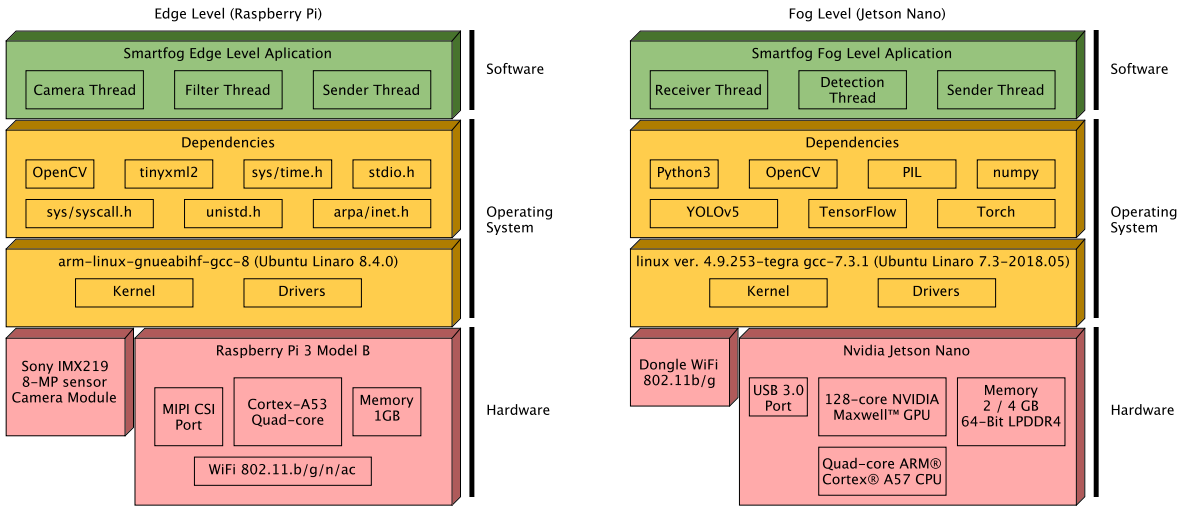


Fig. 6. Components of the Edge and Fog nodes.

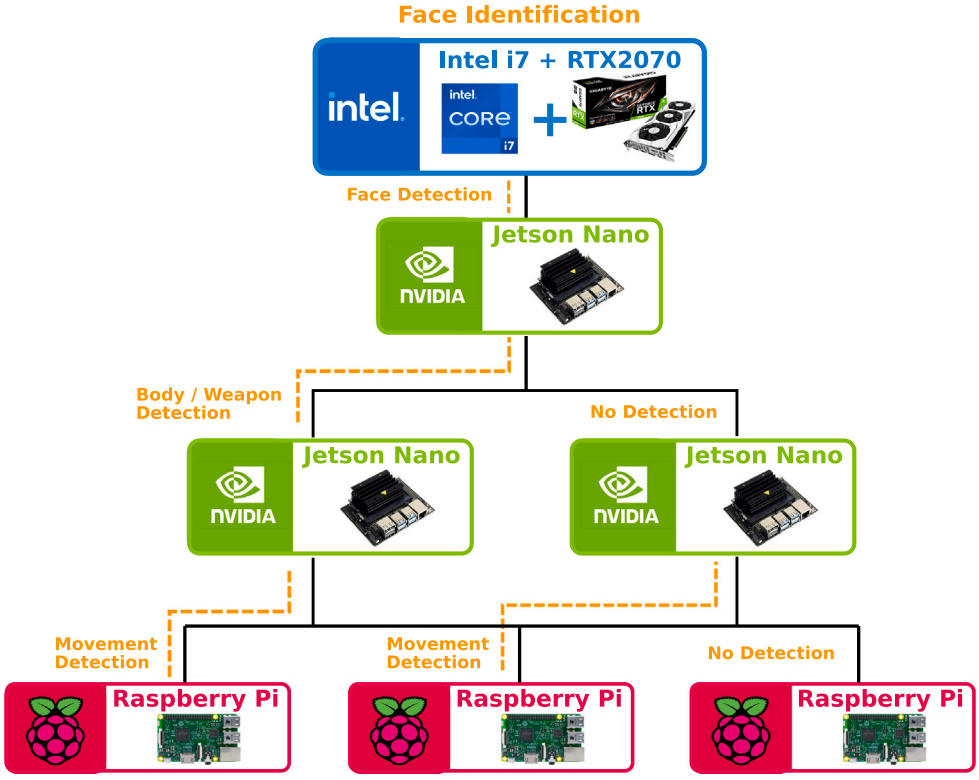


Fig. 7. Deployed system data flow.

is a link between the pistol and the person who handles it. It must be mentioned that the identification of the persons could not be performed because of the use of facemasks that occluded mostly the faces. Therefore, each different person was identified as *personN* for further tracking purposes. Although, this behavior is beyond the scope of this work. Thus, the BBs in orange or pink shown in Figs. 8 and 9 are the result of the distributed processing of:

- (Edge level) detecting a substantial change in a captured frame from the previous one.
- (Fog level, 1st Fog sublayer) detecting a person, producing a BB for each detected person, cropping the image to the BB. In this stage, in parallel with the detection of the person, the detection of weapons is carried out. If a weapon is detected, the BB is sent to the next stage of the Fog level.
- (Fog level, 2nd Fog sublayer), detecting the face of the person within the received BB, cropping the face and the weapon in two different sub-images, linking each weapon BB to its associated person BB.
- (Cloud level) Identification of each detected person, based on the face sub-image sent from the previous level.

4.3. Processing time and network use

As this system is a distributed one, the amount of time taken to process completely a frame involves not only the processing time but

Fig. 8. A frame of an outdoor scene of a gunman aiming at a driver inside a car.

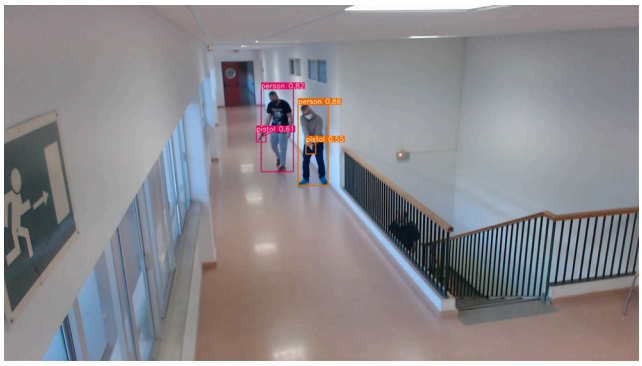


Fig. 9. A frame of an indoor scene of two gunmen in a corridor.

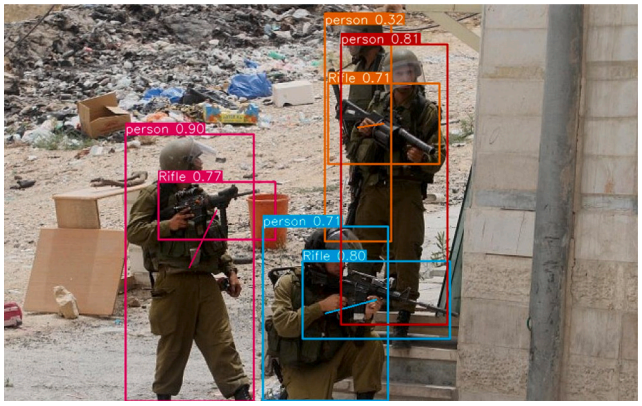


Fig. 10. Outdoor situation involving several military forces with rifles.

Fig. 11. Military assault using a rifle.

Fig. 12. A masked person holding a knife.

also the time expended at the communications between the nodes. In fact, this last type of timing is complex to estimate as it depends on the network status at different moments. If the network is too saturated, there would be a lot of packets trying to access the networking medium so there might be network congestion or some packet collisions. In any case, both problems would make communication slow down significantly, but not in a uniform way.

We have tested the proposed network pipeline as described in Section 3 and collected the amount of data used as inputs at every stage of the distributed system. In order to replicate the same input conditions, the camera at the RPi3 (Edge level node) has been substituted by files that contain each case scenario, and a framebuffer to keep every frame of the video to process it in the same way as if it is coming from the camera.

Table 1 shows the average time spent in each stage of the pipeline for each scenario case. It can be determined that the average time in every stage is similar in each case, or in another case is 0 (as that stage is not activated). Thus, there is a reduction of the overall time in all those cases where the stream can be discarded earlier. Moreover, the 1st and 2nd Fog sublayers, which represent the 2nd and 3rd stages of the pipeline, establish the throughput of the whole pipeline. That is, the computing carried out by the Jetson devices (the ones in the Fog level) limits the speed at which every frame can be processed, with an end-to-end throughput of 3.6 results per second achieved in all case scenarios except in Case 1, where the throughput achieves 23.3 results per second.

This system aims to reduce the communications from Edge (RPi3 nodes with cameras) to Cloud. Table 2 presents the data amount processed by each device. Results from Table 2 show clearly that this hypothesis stands true. For instance, in Case 1, where there are no persons in the scenario, the RPi3 node, set as 1st stage of the networked pipeline of the distributed stream processing engine, sends no data to the following stages. Thus, the network is completely unused. Table 3

Table 1
Average computing time per pipeline node (seconds/frame).

Scenarios	1st stage Raspberry Pi	2nd stage Jetson 4 GB	3rd stage Jetson 2 GB	4th stage Cloud
Case 1: Background	0.0430	0.0000	0.0000	0.0000
Case 2: Body	0.0431	0.2822	0.2415	0.1460
Case 3: Body + Weapon	0.0433	0.2854	0.2379	0.1639
Case 4: NBodies	0.0438	0.2822	0.2390	0.0000
Case 5: NBodies + NWeapons	0.0442	0.2870	0.2437	0.1500

Table 2
Pipeline data processing (MB).

Scenarios	1st stage Raspberry Pi	2nd stage Jetson 4 GB	3rd stage Jetson 2 GB	4th stage Cloud
Case 1: Background	8329.37	0.00	0.00	0.00
Case 2: Body	10 577.83	765.31	0.72	0.02
Case 3: Body + Weapon	10 554.10	1702.66	81.08	2.62
Case 4: NBodies	10 565.96	2568.82	0.65	0.00
Case 5: NBodies + NWeapons	8014.94	1370.43	46.66	1.29

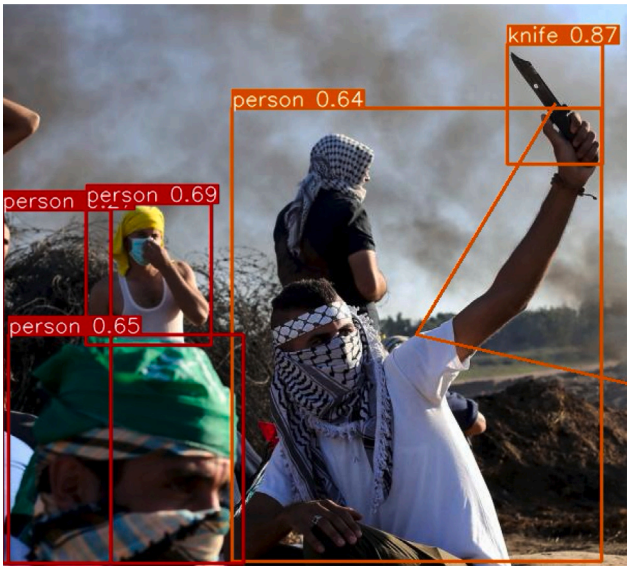


Fig. 13. Some persons, one of which has a knife.

Table 3
Pipeline data transactions (%).

Scenarios	1st → 2nd Stage	2nd → 3rd Stage	3rd → 4th Stage
Case 1: Background	0.00	0.00	0.00
Case 2: Body	7.23	0.01	0.00
Case 3: Body + Weapon	16.13	0.77	0.02
Case 4: NBodies	24.31	0.01	0.00
Case 5: NBodies + NWeapons	17.10	0.58	0.02

shows the percentage of data sent from one stage to the following one in each case.

Results from Table 3, along with those from Table 2, help to understand the reduction of the communications between nodes. Cases 3 and 5 show a decrease in the transactions by means of the proper division of the work. For instance, in Case 3, RPi3 processes 10554.1 MB, but only 16.13% of that data is sent to the 2nd stage of the pipeline (1st Fog sublayer node). Therefore, only 16.13% of the frames show relevant changes to be further processed by the following stages of the pipeline. After that, only 0.77% of the data processed by the Nvidia Jetson 4 GB (which is the 1st Fog sublayer node) set as the 2nd stage of the pipeline is forwarded to the 3rd stage of the pipeline (composed by the 2nd Fog sublayer nodes). This lowering comes from the detection of a body and a weapon, which appear in much fewer frames than the initial stream. Finally, only 0.02% of the frames that contains a body and a weapon are sent to Cloud, for identification. In this final step, the reduction comes from both fewer frames and smaller data (only the face of the person holding the weapon is sent).

4.4. Power consumption

As described in Section 3.4, both the SmartFog distributed system and a centralized solution using a 4 GB RAM Jetson Nano device are analyzed in terms of power consumption.

Using the described logging environment (Fig. 5), several executions of the videos from each Case scenario have been carried out. In the following paragraphs, a specific video from the Case 5 scenario is described in terms of power consumption aligned with the actions of the characters inside the scene.

Fig. 14 shows one frame from a Case 5 scenario video joined to a power consumption plot of each of the involved nodes. The blue line represents the RPi3 node consumption, which is around 3 W on average. It does not change drastically its behavior over time. The second stage node (Jetson Nano 4G) is plotted in green. It can be clearly observed that there are two different states: from the start to around 60 s. and from 60 s. on. The first state is almost stationary around 3.8 W, indicating that there is no processing in the node. This happens during the first 59 s, no changes are in the frame, so the RPi3 node does not send any data to 2nd stage node in the pipeline (the Jetson Nano 4 GB, which is the 1st Fog sublayer node) to detect persons and weapons. After 60 s instant, the power consumption of Jetson Nano 4 GB rises up (ranging from 3.9 W to 9.1 W) with an average of 5.6 W. This behavior occurs because the node looks for persons and weapons.

In the video, the woman takes a gun out in the 90-second moment, which is translated into a peak of the power consumption of the Jetson Nano 2 GB node, plotted in orange. Most of the time, the orange plot is in the range from 0.8 W to 2.5 W (average, 1.2 W), as nothing is computed in that 3rd stage pipeline node (which is the 2nd Fog sublayer). However, at 90 s instant, the power consumption takes a peak of 4.4 W, as the face is searched for in the subframe sent by the 2nd stage node of the pipeline when the weapon is detected.

Moreover, Fig. 15 shows the power consumption analysis for a Case 3 scene, where only one person with a weapon appears on the scene at certain times. On this occasion, the chart has been divided into several intervals to highlight different consumption regimes depending on whether a person and/or weapon is detected. The total consumption signal has been smoothed to facilitate its interpretation. There is an appreciable difference between detection levels: (1) undetected body, *interval a* (~7.5 W); (2) detected unarmed body, *intervals b, d, f, and h* (~8.7 W); and (3) detected armed body, *intervals c, e, and g* (~9.2 W).

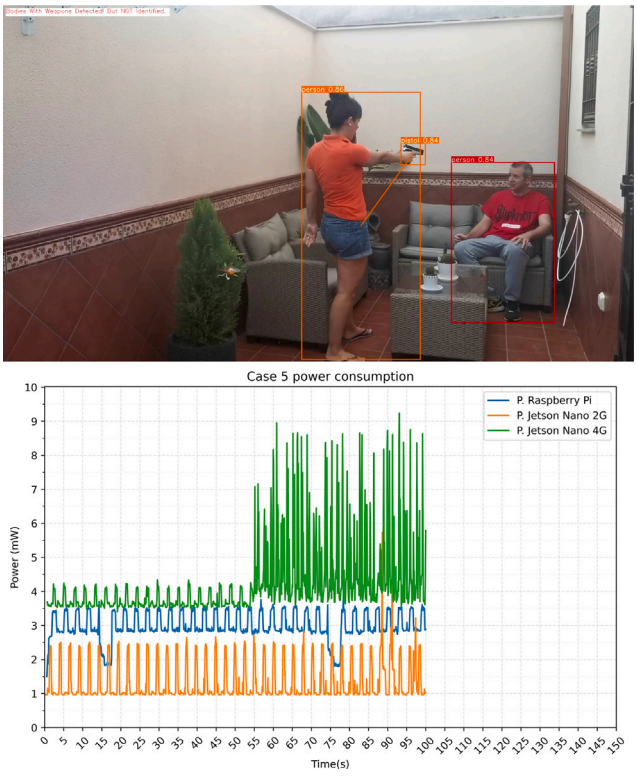


Fig. 14. Power consumption of the networked pipeline nodes for a Case 5 scene.

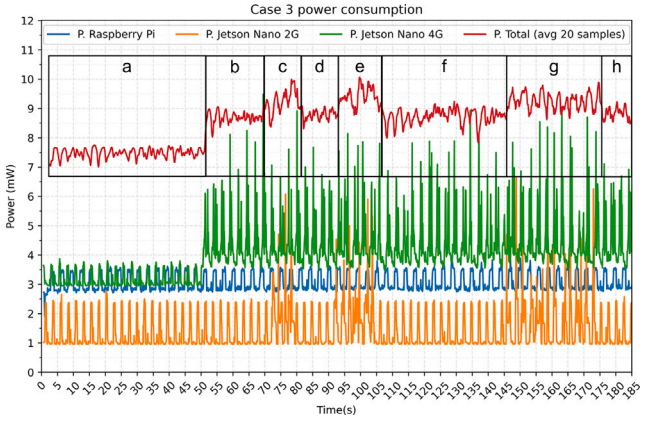


Fig. 15. Power consumption of the networked pipeline nodes for a Case 3.

This experiment shows the power consumption behavior of the nodes of the infrastructure of the distributed stream processing networked pipeline. It demonstrates that the high-performance computing (HPC) required for the most demanding tasks (body, weapon, and face detection) is fired only in specific instants, allowing for continuous tracking but keeping the power consumption of the system in a controlled state.

In order to build the analytical consumption model, the power consumption of each deployment is shown in Table 4. In that, P stands for Power Consumption (in Watts, W), epf is energy per frame (in Joules, J), and tpf is time per frame (in seconds, s).

With the empirical results provided in Table 4, we deduced a power consumption model for each deployment.

The SmartFog deployment is composed of three devices: Edge (e), Fog Level 1 ($f1$), and Fog Level 2 ($f2$). Edge level device was a Raspberry Pi 3, Fog Level 1 device was a Jetson Nano 4 GB, and Fog Level 2 was a Jetson Nano 2 GB. On the other hand, Edge Computing deployment consisted in only one device (E) capable of carrying out all the procedures. In this case, a Jetson Nano 4 GB was used as the Edge Computing device.

Three scenarios have been considered within any scene: the part of the scene in which the camera only captures "background without people" (noted as b), the part of the scene in which there are "people without weapons" (indicated as p), and finally, the section of the scene in which "people with weapons" are present (noted as w). According to this description, we could determine the following values:

- n_b: Number of frames of b ("background without people").
- n_p: Number of frames of p ("people without weapons")
- n_w: Number of frames of w ("people with weapons")

Combining all the above, we can provide values for each part in a compact way. For instance, $epf_{e,p}$ corresponds to energy per frame (epf) in a Raspberry Pi 3 device in Edge level (e) in a "people without weapons" scenario (p). The experiments showed that energy consumption to be 0.389 J.

4.4.1. Energy model for the SmartFog deployment

The SmartFog deployment power consumption model considering the number of cameras (C) is described in Eq. (1).

$$\begin{aligned} Energy_{SmartFog} = \\ & n_b \cdot \left(C \cdot epf_{e,b} + tpf_{e,b} \cdot P_{f1,b} + tpf_{e,b} \cdot P_{f2,b}\right) + \\ & n_p \cdot \left(C \cdot \left(epf_{e,p} + \cdot epf_{f1,p}\right) + tpf_{e,p} \cdot P_{f2,p}\right) + \\ & n_w \cdot C \cdot \left(epf_{e,w} + epf_{f2,w} + epf_{f2,w}\right) \end{aligned} \tag{1}$$

4.5. Energy model for the Edge Computing deployment

The mathematical model is much simpler than the SmartFog energy model, as there is only one device per camera. The Edge Computing device (E) energy consumption model is described in Eq. (2).

$$Energy_E = C \cdot \left(n_b \cdot epf_b + n_p \cdot epf_p + n_w \cdot epf_w\right) \tag{2}$$

4.5.1. Comparative results

We selected several scenarios for comparing the results from the energy models of the SmartFog and Edge Computing deployments. In the following, we present each scenario:

1. Scenario 1: Scene in a low-traffic place.
   (a) Frames without people: 900.
   (b) Frames with unarmed people: 100.
   (c) Frames with armed people: 0.
   (d) Number of cameras: 1, 4.
2. Scenario 2: Scene in a busy and safe place.
   (a) Frames without people: 400.
   (b) Frames with unarmed people: 600.
   (c) Frames with armed people: 0.
   (d) Number of cameras: 1, 4.
3. Scenario 3: Scene in a busy place with an armed person incident.
   (a) Frames without people: 200.
   (b) Frames with unarmed people: 700.
   (c) Frames with armed people: 100.
   (d) Number of cameras: 1, 4.

After the execution of this scenarios, the energy consumption of the model are stated in Table 5. The Edge Computing version has better energy consumption behavior in scenarios with low–traffic (very

Table 4
SmartFog vs Edge Computing power consumption.

SmartFog deployed system									
	Background			Unarmed people			Armed people		
	P (W)	tpf (s)	epf (J)	P (W)	tpf (s)	epf (J)	P (W)	tpf (s)	epf (J)
Raspberry Pi (Edge level)	3	0.096	0.29	3	0.13	0.389	3	0.131	0.393
Jetson 4 GB (Fog level 1)	3.7	–	–	4.75	0.28	1.349	4.75	0.286	1.359
Jetson 2 GB (Fog level 2)	1.4	–	–	1.4	–	–	1.6	0.228	0.365
Edge Computing deployed system									
	P (W)	tpf (s)	epf (J)	P (W)	tpf (s)	epf (J)	P (W)	tpf (s)	epf (J)
Jetson 4 GB (Edge Computing)	4.9	0.043	0.21	7.65	0.29	2.234	7.8	0.482	3.76

Table 5
Comparison of the global energy consumption of the Edge Computing version vs the SmartFog version.

Scenario	Cameras	Edge	SmartFog
1	1	0.41 kJ	0.89 kJ
	4	1.66 kJ	2.19 kJ
2	1	1.43 kJ	1.46 kJ
	4	5.70 kJ	4.94 kJ
3	1	1.98 kJ	1.71 kJ
	4	7.93 kJ	6.17 kJ

few persons). Besides, in scenarios with multiples cameras, SmartFog outperforms the Edge Computing version.

We decided to detect where the time was spent in each case. Therefore, we plotted chord diagrams, connecting the time spent by each stage in each level (Edge/Fog Level 2/Fog Level 2) with each scenario. In the Edge Computing case, as there is only one device, the chord diagrams represent the percentage of time spent in each stage.

Fig. 16 shows the comparison between the Edge Computing and SmartFog in all the scenarios with 4 cameras. The Edge Computing behaves better than SmartFog only in the Scenario 1. Although, 90% of the frames of the scene are background without any person, and only 10% of the frames are with unarmed people, Edge Computing almost take 50% of the processing time in the detection of the unarmed people process. However, due to the efficiency of the Jetson Nano 4 GB device, the power consumption is reduced. On the other hand, as the Raspberry Pi 3 is not an efficient device, the Edge level (the camera acquisition, the background subtraction, and change detection threshold) at the SmartFog model consumes greater than the Edge Computing version. Besides, although there are no armed people frames in the scenario, there is still a power consumption at the Fog Level 2 (3rd stage). We call that basal power consumption.

Scenario 2, in which there are many more frames with persons, the second stage gets much more processing time. In this case, SmarFog outperforms the Edge Computing. In this case, the Jetson Nano 4 GB device in the Edge Computing is very powerful but it is not as power consumption efficient as the SmartFog distribution model, as only 1 Jetson Nano 4 GB of the 2nd stage of the Fog Level 1 device of the SmartFog consumes is required for several cameras. Therefore, in this case, the distributed structure of SmartFog makes the best of it.

The most exigent scenario, Scenario 3, provides best results for the SmartFog model. The distributed infrastructure of SmartFog along with its intelligent decision-making intrinsic behavior is able to reduce the power consumption of further stages when they are not needed. This enhances the efficiency of the proposal in terms of power consumption of the SmartFog.

5. Discussion

Regarding the Visual Results, the system correctly detects people. The test was conducted manually, checking whether the detection was accurate. In some frames, the person goes undetected due to occlusions or unusual body positions that cause detection errors. However, as soon as the person moves, they are detected again.

People holding guns are well detected in Scenario Cases 3 and 5. Again, when appears occlusions or other factors, weapons may also go undetected punctually. However, the weapons are detected again once the occlusion clears. There are some frames where a false weapon is detected. In Cases 2 and 4, shown in Table 3, the false positives are only 0.01% of the total data sent from the first stage of the pipeline. These misdetections occur due to the similarity of objects to guns, rifles, or knives. Nonetheless, these errors last only a few frames. Finally, in the identification phase at the Cloud level, a small part of the image containing the face of the person holding the weapon is sent. The detected faces have been correctly identified in all cases.

Regarding network computation, this distributed scheme combines the best aspects of both CCC and MEC architectures, minimizing their negative effects. The processing is efficiently distributed among the network nodes. The Edge node discards frames without activity. The first Fog level node detects people, whether armed or not. Consequently, the second Fog level node receives only the bounding boxes of armed individuals. Finally, the ROI of this face is sent to the Cloud server for identification. This approach significantly reduces the amount of exchanged data. For example, in Case 3, Table 2 shows a reduction from 10554.10 MB (approximately 10.31 GB) to just 2.62 MB. This represents a reduction of 99.99975% in the data sent to the Server.

The results in Tables 2 and 3 show that network congestion, a major limitation of the CCC architecture, is effectively overcome.

Tests in Section 4.4 demonstrate that the power consumption of the proposed Fog Computing approach is much lower than that of a typical Edge-level (MEC) server, which usually consumes tens of watts per hour. As shown in Fig. 14, the peak power consumption of each node adds up to 3.5 mW + 5.5 mW + 9.2 mW = 18 mW. These values represent peak consumption, meaning real consumption is lower. Fig. 15 confirms this pattern: frames with a person holding a weapon show higher power consumption, but overall, it ranges between 7 and 10 mW.

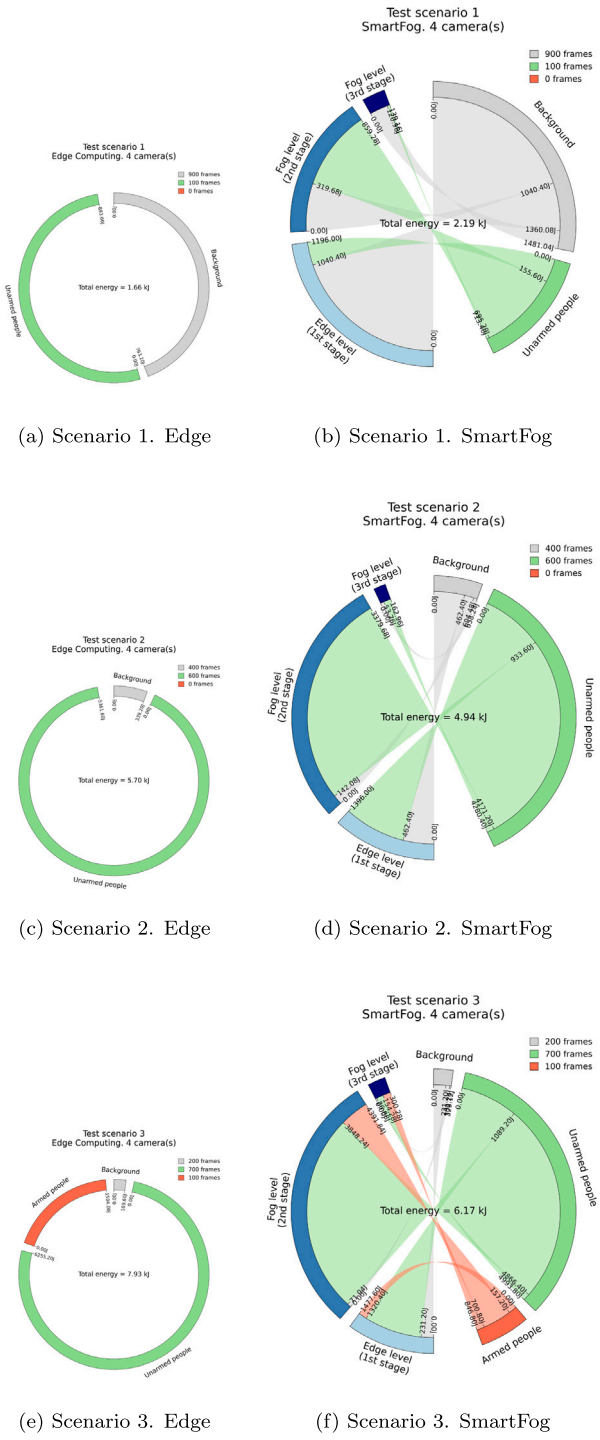
In conclusion, this proposal outperforms the MEC architecture regarding power consumption and reduces the costs of deploying numerous servers.

6. Conclusions

We have presented a new surveillance system for detecting people and weapons, optimized for Fog Computing. This approach is suitable for inclusion in 5G deployments. It efficiently utilizes the Edge and Fog layers, with several Deep Learning algorithms implemented in embedded GPUs in the Fog layer. The system is trained to detect guns, rifles, and knives.

As a result, a large amount of data and communications are saved—only about 2% of the original information with weapons in the scene is sent to the Cloud layer, and no data is sent if no weapons are detected.

The system is useful not only with cameras in which the weapons and people stay still in a first plane but also in typical security cameras in which people are far and down the cameras, as is shown in the scenes presented in this article.



(a) Scenario 1. Edge (b) Scenario 1. SmartFog

(c) Scenario 2. Edge (d) Scenario 2. SmartFog

(e) Scenario 3. Edge (f) Scenario 3. SmartFog

Fig. 16. Scenarios with 4 cameras.

Most scientific works are dedicated only to gun detection, however, we include rifles, guns, and knives. Usually, without including face detection and recognition after the weapon detection, as we do.

Weapon detection is usually carried out in Cloud/Server, and sometimes in Edge, without taking advantage of Fog possibilities. Centralized systems require complete data transmission, suffering scalability problems. Edge systems require embedded expensive GPUs.

The SmartFog distributed and intelligent network processing infrastructure reduces the power consumption of the overall system, maintaining large accuracy, concerning the Edge Computing version. It

Fig. 17. Live demonstration in FEINDEF.

is much more flexible and adaptable to increase the amount of cameras. The actual bottleneck is the edge level camera device selected in this proposal, the Raspberry Pi 3, which has a considerable energy consumption in basal mode. However, despite that, the SmartFog proposal shows lower overall power consumption in highly demanding scenarios with persons and weapons.

The main benefit of our proposal is the data reduction and the flexible and dynamic computation network infrastructure. So, Edge devices include simple and cheap processors just to decide if a change in the scene succeeds. Fog-embedded GPUs can attend several Edge cameras, reducing the system cost, the data transmission, and favoring the scalability of the system. Computer vision (CV) and artificial intelligence (AI) have significantly augmented the impact of Edge and Fog computing by enabling intelligent processing of visual data closer to the source (Tuli et al., 2023). It is not necessary to transmit large amounts of raw data to centralized servers. By combining CV and AI with Edge and Fog computing it is possible to achieve faster response times, reduce bandwidth usage, enhance privacy and security, and unlock new applications and services.

Finally, we presented a demonstrator of our system working for three consecutive days at FEINDEF 2023, the “International Defense Fair of Spain” (Feindef, 2023). Fig. 17 shows how it operates in a real scenario with many people. Note that the person on the left is the only one with a gun, thus, this person is detected as armed, and his BB is orange, whilst the rest of the BBs people are red.

7. Future works

This paper shows the possibilities of implementing Fog computing in low-resource camera sensor deployments. In fact, we are planning to substitute the Edge level device, the Raspberry Pi 3, with a much lower power consumption device such as ESP32–CAM, with only 0.5 W. In the future, we aim to evaluate a scenario with hundreds of camera sensors and a wide level of fog nodes. Also, we pretend to introduce security mechanisms for low-resource devices, in particular, we will integrate the mechanisms described in Alcaraz Velasco et al. (2021, 2024).

CRediT authorship contribution statement

Héctor Martinez: Software, Resources, Methodology. **Francisco J. Rodriguez-Lozano:** Software, Resources, Methodology. **Fernando León-García:** Visualization, Resources, Methodology, Investigation. **Jose M. Palomares:** Writing – original draft, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Formal analysis. **Joaquín Olivares:** Writing – original draft, Validation, Supervision, Project administration, Methodology, Investigation, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work has been partly supported by the Spanish Ministry of Science, Innovation, and Universities grant *Intelligent distributed processing architectures in Fog level for the IoT paradigm (Smart-Fog)* RTI2018-098371-B-I00, and, the Advanced Informatics Research Group – GIIA (TIC-252). H. Martínez is a postdoctoral fellow supported by the *Consejería de Transformación Económica, Industria, Conocimiento y Universidades de la Junta de Andalucía*.

References

Alcaraz Velasco, F., Palomares, J.M., Olivares, J., 2021. Lightweight method of shuffling overlapped data-blocks for data integrity and security in WSNs. Comput. Netw. 199, 108470. http://dx.doi.org/10.1016/j.comnet.2021.108470.

Alcaraz Velasco, F., Palomares, J.M., Olivares, J., 2024. GS3: A lightweight method of generating data blocks with shuffling, scrambling, and substituting data for constrained IoT devices. IEEE Internet Things J. 1–20. http://dx.doi.org/10.1109/JIOT.2024.3395543.

Anitha, P., Vimala, H., Shreyas, J., 2024. Comprehensive review on congestion detection, alleviation, and control for IoT networks. J. Netw. Comput. Appl. 221, 103749. http://dx.doi.org/10.1016/j.jnca.2023.103749.

Ashraf, A.H., Imran, M., Qahtani, A.M., Alsufyani, A., Almutiry, O., Mahmood, A., Attique, M., Habib, M., 2022. Weapons detection for security and video surveillance using CNN and YOLO-v5s. Comput. Mater. Continua 70 (2), 2761–2775. http://dx.doi.org/10.32604/cmc.2022.018785.

Asrith, M.J.S.K., Reddy, K.P., Sujihelen, 2018. Face recognition and weapon detection from very low resolution image. In: 2018 International Conference on Emerging Trends and Innovations in Engineering and Technological Research. ICETIETR, pp. 1–5. http://dx.doi.org/10.1109/ICETIETR.2018.8529108.

Azizi, S., Shojafar, M., Abawajy, J., Buyya, R., 2022. Deadline-aware and energy-efficient IoT task scheduling in fog computing systems: A semi-greedy approach. J. Netw. Comput. Appl. 201, 103333. http://dx.doi.org/10.1016/j.jnca.2022.103333.

Bhatti, M.T., Khan, M.G., Aslam, M., Fiaz, M.J., 2021. Weapon detection in real-time CCTV videos using deep learning. IEEE Access 9, 34366–34382. http://dx.doi.org/10.1109/ACCESS.2021.3059170.

Bhowmik, S., 2017. Cloud Computing. Cambridge University Press, http://dx.doi.org/10.1017/9781316941386.

Buyya, R., Srirama, S.N., 2019. Fog and Edge Computing: Principles and Paradigms. John Wiley & Sons, Ltd.

Castillo, A., Tabik, S., Pérez, F., Olmos, R., Herrera, F., 2019. Brightness guided preprocessing for automatic cold steel weapon detection in surveillance videos with deep learning. Neurocomputing 330, 151–161. http://dx.doi.org/10.1016/j.neucom.2018.10.076.

Castillo-Secilla, J.M., Aranda, P.C., et al., 2010. Experimental procedure for the characterization and optimization of the power consumption and reliability in ZigBee mesh networks. In: 2010 Third International Conference on Advances in Mesh Networks. pp. 13–16. http://dx.doi.org/10.1109/MESH.2010.16.

Cob-Parro, A.C., Losada-Gutiérrez, C., Marrón-Romera, M., Gardel-Vicente, A., Bravo-Muñoz, I., 2021. Smart video surveillance system based on edge computing. Sensors 21 (9), 2958. http://dx.doi.org/10.3390/s21092958.

Cootes, T., Edwards, G., Taylor, C., 2001. Active appearance models. IEEE Trans. Pattern Anal. Mach. Intell. 23 (6), 681–685. http://dx.doi.org/10.1109/34.927467.

Dao, T., Khalil, K., Roy-Chowdhury, A.K., Krishnamurthy, S.V., Kaplan, L., 2017. Energy efficient object detection in camera sensor networks. In: 2017 IEEE 37th International Conference on Distributed Computing Systems. ICDCS, pp. 1208–1218. http://dx.doi.org/10.1109/ICDCS.2017.152.

Dao, N.-N., Tran, Q.D., Dinh, N.-T., Cho, S., Braun, T., 2020. Edge computing architectures. In: Edge Computing: Models, Technologies and Applications. Institution of Engineering and Technology, pp. 27–44. http://dx.doi.org/10.1049/pbpc033e_ch2.

Debnath, R., Bhowmik, M.K., 2021. A comprehensive survey on computer vision based concepts, methodologies, analysis and applications for automatic gun/knife detection. J. Vis. Commun. Image Represent. 78, 103165. http://dx.doi.org/10.1016/j.jvcir.2021.103165, URL: https://www.sciencedirect.com/science/article/pii/S104732032100105X.

Deloitte, 2021. The Open Future of Radio Access Networks. Deloitte.

Dhirani, L.L., Newe, T., Lewis, E., Nizamani, S., 2017. Cloud computing and internet of things fusion: Cost issues. In: 2017 Eleventh International Conference on Sensing Technology. ICST, IEEE, pp. 1–6. http://dx.doi.org/10.1109/icsenst.2017.8304426.

ETSI Industry Specification Group, 2019. Multi-access edge computing (mec); framework and reference architecture, etsi gs mec 003 v2.1.1. Technical Report, ETSI.

Fei-Fei, L., Fergus, R., Perona, P., 2004. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. In: 2004 Conference on Computer Vision and Pattern Recognition Workshop. p. 178. http://dx.doi.org/10.1109/CVPR.2004.383.

Feindef, 2023. 2023 Feria internacional de defensa y seguridad de España. https://www.feindef.com/.

Geitgey, A., 2020. face-recognition 1.3.0: Recognize faces from Python or from the command line. pypi.org, 20-Feb-2020. [Online]. Available: https://pypi.org/project/face-recognition/.

Gelana, F., Yadav, A., 2019. Firearm detection from surveillance cameras using image processing and machine learning techniques. In: Smart Innovations in Communication and Computational Sciences. Advances in Intelligent Systems and Computing, Vol. 851. Springer, pp. 25–34. http://dx.doi.org/10.1007/978-981-13-2414-7_3.

Girshick, R., 2015. Fast R-CNN. In: 2015 IEEE International Conference on Computer Vision. ICCV, pp. 1440–1448. http://dx.doi.org/10.1109/ICCV.2015.169.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 580–587. http://dx.doi.org/10.1109/CVPR.2014.81.

Harris, C., Stephens, M., 1988. A combined corner and edge detector. In: Alvey Vision Conference. pp. 147–152. http://dx.doi.org/10.5244/C.2.23.

iGillott, 2020. Open RAN Integration: Run With It. White Paper. Technical Report, iGillott Research Inc.

ITU-T, 2018. Transport Network Support of IMT2020/5G. White Paper. Technical Report, International Telecomunication Union.

Jocher, G., 2021. YOLOv5. GitHub https://github.com/ultralytics/yolov5.

Khalid, O., Khan, I.A., Rais, R.N.B., Malik, A.W., 2020. An insight into 5G networks with fog computing. In: Fog Computing. John Wiley & Sons, Ltd, pp. 505–527. http://dx.doi.org/10.1002/9781119551713.ch20.

Lee, W., Suh, E.S., Kwak, W.Y., Han, H., 2020. Comparative analysis of 5G mobile communication network architectures. Appl. Sci. 10 (7), http://dx.doi.org/10.3390/app10072478.

León-García, F., Palomares, J.M., Olivares, J., 2018. D2R-TED: Data—Domain reduction model for threshold-based event detection in sensor networks. Sensors 18 (11), 3806. http://dx.doi.org/10.3390/s18113806.

León-García, F., Rodríguez-Lozano, F.J., Olivares, J., Palomares, J.M., 2019. Data communication optimization for the evaluation of multivariate conditions in distributed scenarios. IEEE Access 7, 123473–123489. http://dx.doi.org/10.1109/ACCESS.2019.2936918.

Liyanage, M., Braeken, A., Shahabuddin, S., Ranaweera, P., 2023. Open RAN security: Challenges and opportunities. J. Netw. Comput. Appl. 214, 103621. http://dx.doi.org/10.1016/j.jnca.2023.103621.

Mahbub, M., Shubair, R.M., 2023. Contemporary advances in multi-access edge computing: A survey of fundamentals, architecture, technologies, deployment cases, security, challenges, and directions. J. Netw. Comput. Appl. 219, 103726. http://dx.doi.org/10.1016/j.jnca.2023.103726.

Meng, Y., Naeem, M.A., Almagrabi, A.O., Ali, R., Kim, H.S., 2020. Advancing the state of the fog computing to enable 5G network technologies. Sensors 20 (6), 1754. http://dx.doi.org/10.3390/s20061754.

Mohanapriya, D., Mahesh, K., 2020. An efficient framework for object tracking in video surveillance. In: The Cognitive Approach in Cloud Computing and Internet of Things Technologies for Surveillance Tracking Systems. Elsevier, pp. 65–74. http://dx.doi.org/10.1016/b978-0-12-816385-6.00005-2.

Nakib, M., Khan, R.T., Hasan, M.S., Uddin, J., 2018. Crime scene prediction by detecting threatening objects using convolutional neural network. In: 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering. IC4ME2, pp. 1–4. http://dx.doi.org/10.1109/IC4ME2.2018.8465583.

Narejo, S., Pandey, B., vargas, D.E., Rodriguez, C., Anjum, M.R., 2021. Weapon detection using YOLO V3 for smart surveillance system. In: Ali, Z.A. (Ed.), Math. Probl. Eng. 2021, 1–9. http://dx.doi.org/10.1155/2021/9975700.

Olivares, J., Palomares, J., Soto, J., Gámez, J., 2010. License plate detection based on genetic neural networks, morphology, and active contours. In: Trends in Applied Intelligent Systems. IEA/AIE 2010. Lecture Notes in Computer Science, Vol. 6098. pp. 301–310. http://dx.doi.org/10.1007/978-3-642-13033-5_31.

Olmos, R., Tabik, S., Herrera, F., 2018. Automatic handgun detection alarm in videos using deep learning. Neurocomputing 275, 66–72. http://dx.doi.org/10.1016/j.neucom.2017.05.012.

O'Shea, K., Nash, R., 2015. An introduction to convolutional neural networks. CoRR arXiv:1511.08458, URL: http://dblp.uni-trier.de/db/journals/corr/corr1511.html#OSheaN15.

Ostrowski, K., Małecki, K., Dziurzański, P., Singh, A.K., 2023. Mobility-aware fog computing in dynamic networks with mobile nodes: A survey. J. Netw. Comput. Appl. 219, 103724. http://dx.doi.org/10.1016/j.jnca.2023.103724.

Palomar, R., Palomares, J., Castillo, J., Olivares, J., Gómez-Luna, J., 2010. Parallelizing and optimizing LIP-Canny using NVIDIA CUDA. In: Trends in Applied Intelligent Systems. IEA/AIE 2010. Lecture Notes in Computer Science, Vol. 6098. Springer, pp. 389–398. http://dx.doi.org/10.1007/978-3-642-13033-5_40.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. arXiv:1506.02640.

Ren, S., He, K., Girshick, R., Sun, J., 2017. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. 39 (6), 1137–1149. http://dx.doi.org/10.1109/TPAMI.2016.2577031.

Rodriguez-Lozano, F.J., León-García, F., Ruiz de Adana, M., Palomares, J.M., Olivares, J., 2019. Non-invasive forehead segmentation in thermographic imaging. Sensors 19 (19), 4096. http://dx.doi.org/10.3390/s19194096.

Sánchez, J., Monzón, N., Salgado, A., 2018. An analysis and implementation of the Harris corner detector. Image Process. On Line 8, 305–328. http://dx.doi.org/10.5201/ipol.2018.229.

SanMiguel, J.C., Cavallaro, A., 2017. Energy consumption models for smart camera networks. IEEE Trans. Circuits Syst. Video Technol. 27 (12), 2661–2674. http://dx.doi.org/10.1109/TCSVT.2016.2593598.

Santos, T., Oliveira, H., Cunha, A., 2024. Systematic review on weapon detection in surveillance footage through deep learning. Comp. Sci. Rev. 51, 100612. http://dx.doi.org/10.1016/j.cosrev.2023.100612, URL: https://www.sciencedirect.com/science/article/pii/S1574013723000795.

Sasikaladevi, V., Mangai, V., 2018. Colour based image segmentation using hybrid kmeans with watershed segmentation. Int. J. Mech. Eng. Technol. 9, 1367–1377, URL: https://ssrn.com/abstract=3474793.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y., 2014. OverFeat: Integrated recognition, localization and detection using convolutional networks. arXiv:1312.6229.

Songhorabadi, M., Rahimi, M., MoghadamFarid, A., Haghi Kashani, M., 2023. Fog computing approaches in IoT-enabled smart cities. J. Netw. Comput. Appl. 211, 103557. http://dx.doi.org/10.1016/j.jnca.2022.103557.

Taha, B., Shoufan, A., 2019. Machine learning-based drone detection and classification: State-of-the-art in research. IEEE Access 7, 138669–138682. http://dx.doi.org/10.1109/ACCESS.2019.2942944.

Tiwari, R.K., Verma, G.K., 2015. A computer vision based framework for visual gun detection using Harris interest point detector. Procedia Comput. Sci. 54, 703–712. http://dx.doi.org/10.1016/j.procs.2015.06.083.

Tuli, S., Mirhakimi, F., Pallewatta, S., Zawad, S., Casale, G., Javadi, B., Yan, F., Buyya, R., Jennings, N.R., 2023. AI augmented Edge and Fog computing: Trends and challenges. J. Netw. Comput. Appl. 216, 103648. http://dx.doi.org/10.1016/j.jnca.2023.103648.

Warsi, A., Abdullah, M., Husen, M.N., Yahya, M., 2020. Automatic handgun and knife detection algorithms: A review. In: 2020 14th International Conference on Ubiquitous Information Management and Communication. IMCOM, pp. 1–9. http://dx.doi.org/10.1109/IMCOM48794.2020.9001725.

Xiong, B., Liu, Y., Liu, R., Zhao, J., He, S., Zhao, B., Yang, K., Li, K., 2024. ActiveGuardian: An accurate and efficient algorithm for identifying active elephant flows in network traffic. J. Netw. Comput. Appl. 224, 103853. http://dx.doi.org/10.1016/j.jnca.2024.103853.

Yadav, P., Gupta, N., Sharma, P.K., 2023. A comprehensive study towards high-level approaches for weapon detection using classical machine learning and deep learning methods. Expert Syst. Appl. 212, 118698. http://dx.doi.org/10.1016/j.eswa.2022.118698, URL: https://www.sciencedirect.com/science/article/pii/S0957417422017286.

Zhang, J., Xing, W., Xing, M., Sun, G., 2018. Terahertz image detection with the improved faster region-based convolutional neural network. Sensors 18 (7), 2327. http://dx.doi.org/10.3390/s18072327.

Zhao, Z.Q., Zheng, P., Xu, S.T., Wu, X., 2019. Object detection with deep learning: A review. IEEE Trans. Neural Netw. Learn. Syst. 30 (11), 3212–3232. http://dx.doi.org/10.1109/TNNLS.2018.2876865.

Zhou, M., Elmore, J.A., Samiappan, S., Evans, K.O., Pfeiffer, M.B., Blackwell, B.F., Iglay, R.B., 2021. Improving animal monitoring using Small Unmanned Aircraft Systems (sUAS) and deep learning networks. Sensors 21 (17), 5697. http://dx.doi.org/10.3390/s21175697.

Zolghadri, M., Asghari, P., Dashti, S.E., Hedayati, A., 2024. Resource allocation in Fog–Cloud Environments: State of the art. J. Netw. Comput. Appl. 227, 103891.

Héctor Martínez is a Computer Science Ph.D. member at GIIA research group lead by Joaquin Olivares Bueno at University of Cordoba (Spain). His predoctoral research was done at the HPC&A group led by Enrique S. Quintana-Orti at Jaume I University (Spain).

His thesis focused on developing a DNA/RNA Aligner and optimizing different bioinformatic applications with HPC techniques (MPI, OpenMP, Cuda, ...). Currently, he has more than 12 JCR journal papers and more than 12 International conferences. The last few years, his work has focused on IoT (either, Fog and Edge Computing), Neuronal Networks, and Image Processing. His interests cover HPC programming models like OpenMP, Cuda, SIMD, and MPI, High-performance networks, and low-power processors.

Francisco J. Rodriguez Lozano received a B.Sc. in Computer Science, M.Sc. in Distributed Renewable Energies, at Ph.D. in Computer Engineering at the University of Córdoba, Spain, in 2016, 2017, and 2020 respectively.

He is a member of the Department of Electronics and Computer Engineering at the same university, where he is a full-time Assistant Professor since 2018. He has been the main author in several papers in high-rank international journals and has participated in different national and international conferences, as well as in national and international research projects. His research interests are in the field of computer vision, machine learning and embedded systems.

Fernando León García holds a Ph.D. in Advanced Computing, a master's degree in Distributed Renewable Energies, and a bachelor's degree in Computer Engineering. He conducted his doctoral research at the Advanced Computing Research Group (GIIA) at the University of Córdoba, Spain, where he has previously served as a temporary substitute Lecturer and currently holds the position of Assistant Professor.

Dr. León-García's research interests encompass a wide array of fields, including Internet of Things, Distributed Architectures, Embedded Systems, Fog/Edge Computing, and Communication Compression Techniques.

Jose M. Palomares was born in Motril, Spain, in 1975. He received his Ph.D., M.Sc., and B.Sc. degrees in Computer Engineering from the Universidad de Granada, Granada, Spain in 2011, 1998, and 1996, respectively. Since 2000, he has been working as Lecturer, Assistant, Associate Professor, and, currently, as a Professor at the Universidad de Córdoba, Córdoba, Spain. He is co-founder of the Advanced Informatics Research Group (GIIA), Universidad de Córdoba, Córdoba, Spain. He has research interests in image and video processing, real-time systems, wireless sensor networks, IoT, and computer architecture.

Joaquín Olivares received the B.S. and M.S. degree in Computer Sciences specialized in artificial intelligence in 1997, and 1999, respectively, and the M.S. degree in Electronics Engineering in 2003, all from the Universidad de Granada, Spain. His Ph.D. degree was a patented chip for motion estimation in video coding designed in FPGA in 2008 at the Universidad de Córdoba, Spain. Between 2000 and 2001 he worked as a software architect with the Orangee Group in Rome, Italy. He is a Full Professor in the Electronic and Computer Engineering Department at the Universidad de Cordoba, where he was hired in 2001. He is the founder and head of the Advanced Informatics Research Group. His research interests are the Internet of Things, Embedded Systems, Computer Vision, and High-Performance Computing.