Enhancing Public Safety Situational Awareness Using Edge Intelligence

Pedro Lira, Stefano Loss, Karine Costa, Daniel Araújo, Aluizio Rocha Neto, Nelio Cacho, Thais Batista, Everton Cavalcante, Frederico Lopes, Eduardo Nogueira
Federal University of Rio Grande do Norte
Natal, Brazil
pedro.varela.117@ufrn.edu.br, momoloss10@gmail.com, {karine.piacentini, daniel, aluizio}@imd.ufrn.br, neliocacho@dimap.ufrn.br, thaisbatista@gmail.com, everton.cavalcante@ufrn.br, {fred, eduardo}@imd.ufrn.br

***Abstract*—Real-time video analytics powered by artificial intelligence (AI) enables public safety agents to effectively perceive and respond to dynamic environments. However, processing large-scale video streams introduces computational and latency challenges. This work presents a framework that combines edge and cloud computing to facilitate efficient AI-based processing of video streams for public safety applications. We evaluated the framework's performance in a face recognition task by comparing edge and cloud processing. Our initial results demonstrate that edge processing achieves lower total latency compared to cloud processing despite higher inference times, primarily due to reduced transmission overhead. The framework also achieves high accuracy in recognition tasks, though with trade-offs in recall.**

Keywords-public safety; situational awareness; edge intelligence; stream analytics

I. INTRODUCTION

Situational awareness (SA) refers to the ability to perceive environmental factors, understand their significance, and anticipate future developments [1]. In contrast to traditional methods relying on radio communication and manual reporting, which present significant delays and inefficiencies, modern technologies like surveillance cameras and body-worn devices can enable real-time data collection and dissemination, thereby improving SA and decision-making [2].

Public safety operations require quick detection, analysis, and response to incidents. To comply with these requirements, real-time processing of video streams can be used to detect threats and support decision-making. While artificial intelligence (AI) techniques can significantly enhance this kind of advanced analysis, the large volumes of data to be handled and the high computational demand of intelligent models typically require cloud resources, which introduce latency due to data transmission and depend on reliable connectivity [3]. An alternative to alleviate these issues is processing AI close to the data source through edge intelligence, i.e., the convergence of AI-based task processing and edge computing. This kind of approach can reduce latency and bandwidth usage while ensuring continuity in low-connectivity environments. Nonetheless, the resource constraints of edge devices and bandwidth costs of cloud transmission demand a careful task distribution strategy.

This paper addresses these issues through SAALSA [4], a framework designed to enable efficient, low-latency video analytics by combining edge and cloud computing. We instantiated SAALSA into a public safety scenario, including the real-time identification of individuals based on face recognition resulting from AI-based processing of video streams. We assessed SAALSA's performance for this task by comparing edge and cloud processing in terms of latency and accuracy. Our preliminary findings have demonstrated the potential of edge intelligence to support critical decision-making in public safety operations.

The remainder of this paper is organized as follows. Section II brings an overview of SAALSA. Section III describes a face recognition use case in public safety that we utilize to demonstrate the framework. Section IV reports a preliminary evaluation of SAALSA's performance in AI-driven face recognition regarding edge and cloud-based processing latency and accuracy. Section V points out final remarks and directions for future work

II. A FRAMEWORK FOR AI-DRIVEN STREAM ANALYTICS IN THE EDGE

SAALSA addresses the fundamental challenge of balancing computational efficiency with latency requirements for AI-based stream analytics [4]. Public safety operations often occur under unstable or absent network conditions, requiring solutions that function independently of the cloud. AI-powered video analytics at the edge addresses this need by directly enabling real-time tasks, such as object detection and face recognition, on local devices. This enables the provision of timely insights that support faster and more accurate decision-making.

The SAALSA's architecture, depicted in Fig. 1, allows distributing processing tasks across three tiers. The *Data Source Tier* captures and collects raw data (e.g., audio and video streams and geolocation data) from various devices. The *Edge Tier* handles initial data processing near data sources, reducing latency and minimizing data transmission overhead to the cloud. The *Cloud Tier* handles computationally intensive tasks and provides centralized coordination and storage. Unlike traditional cloud-based architectures that send raw data to remote servers for processing, SAALSA collects data from several sources,

This work is supported by the SPICI project, funded by the Brazilian Innovation Agency – FINEP (grant 2827/22).

Manuscript received May 20, 2025; July 26, 2025; accepted July 25, 2025. Published September 2, 2025.
Issue category: Special Issue on DSD/SEAA 2025 on Works in Progress (WiP) Session, Salerno, Italy, Sept. 2025.
Paper category: Short
DOI: doi.org/10.64552/wipiec.v11i1.88

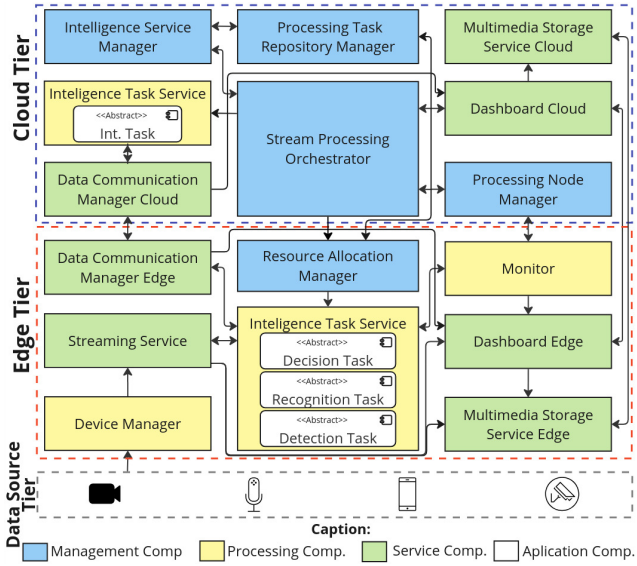

Figure 1. Main components of the SAALSA framework.

processes it at the edge to reduce latency and bandwidth use, and offloads it to the cloud for intensive tasks or long-term storage. Using SAALSA, it is possible to convert real-time data streams into actionable insights in public safety scenarios.

SAALSA comprises processing, management, and service components. Edge nodes handle streaming, detection, and recognition, while the cloud coordinates orchestration and complex analytics. Services include task scheduling, monitoring, multimedia storage, and user interfaces. This modular architecture enables adaptive, distributed video analytics tailored to public safety operations.

Data flows from the *Data Source Tier* to the *Edge Tier*, where the *Device Manager* and *Streaming Service*, built with Kurento,[1] handle media streaming. The *Multimedia Storage Service* records video streams and synchronizes them with the cloud when possible. A shared *Dashboard* provides real-time visualization with geolocation. FogFlow[2] orchestrates tasks between edge and cloud, with the *Resource Allocation Manager* coordinating task distribution between edge and cloud tiers. The *Intelligence Task Service* performs AI-driven task processing by using GStreamer[3] and DeepStream,[4] while Qdrant[5] supports face recognition. Finally, the *Data Communication Manager* handles inter-tier communication and data synchronization.

SAALSA implements dynamic task distribution based on computational requirements and network conditions. Initial processing tasks, such as detection and tracking, occur at the edge to minimize latency. Conversely, intensive tasks, including feature extraction and database queries, can be offloaded to the cloud when network conditions permit and computational demands exceed the edge's capabilities.

III. Face Recognition Use Case in Public Safety

We implemented a face recognition pipeline to demonstrate the feasibility of SAALSA. This use case represents a common public safety requirement where officers need to rapidly identify individuals in the field. Detected faces are first matched locally; if no match is found, embeddings are sent to the cloud. Results are annotated on-screen, enabling efficient, low-latency recognition adapted to hardware constraints.

The pipeline comprises three sequential stages that process video frames for accurate face recognition. Face detection employs the NVIDIA FaceNet model[6] to extract faces within bounding boxes [5]. Next, the FaceNet convolutional neural network [6] performs feature extraction, generating a feature vector (embedding) that maps each face to a compact Euclidean space. Finally, face recognition is achieved by comparing the extracted embedding to those stored in a vector database. Considering the incident response scenario, we adopted high-performance models that strike a balance between robustness and low latency, meeting the resource constraints of edge environments and the variability of public safety conditions.

Given the unpredictable nature of public safety scenarios, we also implemented an accumulation strategy to improve recognition accuracy. The strategy operates by assigning unique identifiers to each detected face, extracting embeddings, and identifying the closest match in the database for each frame. Recognition results are stored for *N* frames, after which the most frequent recognition is selected, and an average distance is computed for the final decision. This strategy prioritizes precision over recall, a crucial feature for public safety applications where false positives can have severe consequences.

IV. Evaluation

Experimental setup. We evaluated SAALSA using two configurations representing typical deployment scenarios. The edge configuration utilized an NVIDIA Jetson Nano[7] 4 GB (ARM Cortex-A57, 128 CUDA cores). The cloud configuration utilized a server equipped with an Intel i5-9300H processor, 64 GB of RAM, and a GTX 1650 GPU.

The evaluation considered streaming a recorded video from a simulated device to the edge and the cloud (see Fig. 2). The stream, sent via GStreamer using the RTSP protocol, was processed by a DeepStream-based face recognition pipeline (Section III), which used Qdrant for vector-based identity retrieval. Both setups used identical versions of DeepStream (6.0) and Qdrant (1.12.1), with consistent configurations and quantization levels: `INT8` for detection and `FP32` for embedding. We implemented tracking using NVIDIA's discriminative correlation filter. While we are aware that the hardware used in our evaluation does not currently represent the state-of-the-art of

[1] https://kurento.openvidu.io/
[2] https://fogflow.readthedocs.io/
[3] https://gstreamer.freedesktop.org/
[4] https://developer.nvidia.com/deepstream-sdk
[5] https://qdrant.tech/
[6] https://catalog.ngc.nvidia.com/orgs/nvidia/teams/tao/models/facenet
[7] https://developer.nvidia.com/embedded/jetson-nano

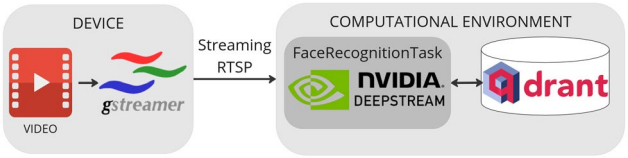

Figure 2. Face recognition setup used in the evaluation.

specialized AI hardware, we aimed to provide valuable insights into trade-offs between edge and cloud processing.

In addition to edge-only and cloud-only experiments, we evaluated a hybrid configuration where detection is performed at the edge, and only cropped faces are sent to the cloud for recognition. We measured transmission times for both full frames (1920 × 1080, RGB) and cropped faces (160 × 160), encoded in `base64`. We also established a secure Tailscale[8] network to simulate realistic cloud latency.

It is worth mentioning that our study focused on the overall pipeline processing rather than the latency due to transmitting data, as network conditions can introduce variability that must be accounted for in real-world deployments. Future work can explore controlled environments with dedicated network configurations to provide a more precise evaluation of transmission delays in edge and cloud scenarios.

Data sets. We constructed the dataset used in the evaluation from two publicly available face recognition datasets: CelebA [7] and Labeled Faces in the Wild (LFW) [8]. CelebA comprises 10,177 individuals, while LFW includes 5,749 individuals. For this study, we randomly selected 5,000 individuals from CelebA and 800 individuals from LFW, ensuring each individual had one to four images. This selection resulted in a total of 5,800 individuals and 9,199 images. Additionally, four volunteers from our research group contributed three images each, captured from different face angles (frontal and both sides profiles), adding 12 more images to the experimental dataset. We used two videos: Video V1 (duration 1'45") features a single individual per frame for accurate timing analysis, and Video V2 (duration 3'10") features the four volunteers in dynamic outdoor settings to assess the model's robustness under real-world conditions.

A. *Processing Time Assessment*

The first experiment evaluated the computational cost of each stage in the face recognition pipeline, aiming to identify performance bottlenecks. This experiment utilized Video V1, which includes one person per frame, enabling consistent measurement across all frames. We timed four tasks: (i) face detection, (ii) face tracking, (iii) embedding extraction, and (iv) database query, on both edge and cloud environments. As shown in Table I, all stages were faster in the cloud. Embedding extraction was the most expensive task, especially on the edge (288.37 ms per frame), due to a non-optimized ONNX model[9] that did not fully exploit DeepStream's acceleration. In contrast, face detection used a native DeepStream model, enabling much faster inference. Database queries also exhibited higher latency on the edge, primarily due to slower communication between

TABLE I
AVERAGE PROCESSING TIME FOR EACH PIPELINE OPERATION

Operation	Cloud	Edge
Face detection	4.73 ± 1.02 ms	47.27 ± 1.96 ms
Face tracking	2.47 ± 2.31 ms	10.48 ± 11.07 ms
Embedding extraction	8.85 ± 4.14 ms	288.37 ± 10.58 ms
Database query	7.05 ± 1.26 ms	29.87 ± 3.20 ms

TABLE II
TOTAL PROCESSING TIME PER FRAME

Metrics	Edge	Hybrid	Cloud
Average frame processing time	375.99 ms	65.74 ms	23.20 ms
Average frame transmission time	108.85 ms*	606.75 ms	3,019.42 ms
Total processing time per frame	484.84 ms	672.49 ms	3,042.62 ms

*Time to transmit a Full HD frame via Gigabit Ethernet

DeepStream and Qdrant, limited memory bandwidth, and lower processing capacity.

Table II summarizes the total frame processing time. Edge processing had lower overall latency than cloud, despite slower inference. The hybrid approach, which balanced computation and communication, was slightly slower than the edge-only approach. These results reflect the trade-offs between computation and transmission, as well as the benefits of minimizing unnecessary data transfer.

B. *Recognition Assessment*

The second experiment evaluated the face recognition performance by using Video V2. To analyze the accuracy of the system, we employed precision and recall metrics, considering two parameters: the number of accumulated frames used to confirm an identity and a Euclidean distance threshold T that defines a valid match. Recognition was considered correct (true positive, TP) when the average distance between the detected face and its closest match in the database was below T, and the predicted identity matched the ground truth. Conversely, a false positive (FP) occurred when the distance was acceptable, but the identity was incorrect, while a false negative (FN) indicated a correct identity with a distance above the threshold. Precision is computed using Equation 1, while recall is computed using Equation 2:

$$Precision = \frac{TP}{TP + FP} \quad (1) \qquad Recall = \frac{TP}{TP+FN} \quad (2)$$

According to the results shown in Fig. 3, precision reached 100% upon accumulating $N = 110$ frames and $T = 0.8$, indicating no false positives. However, recall remained at 0.39 due to a high false negative rate. This configuration prioritizes accuracy over coverage and is suitable for public safety applications where minimizing false alarms is crucial. We observed that recall improved significantly when we relaxed the threshold; however, this came at the cost of reduced precision, with an increase in false positives. This trade-off illustrates the balance between identifying as many individuals as possible and maintaining a high level of confidence in each recognition. These findings

[8] https://tailscale.com/

[9] https://onnx.ai/

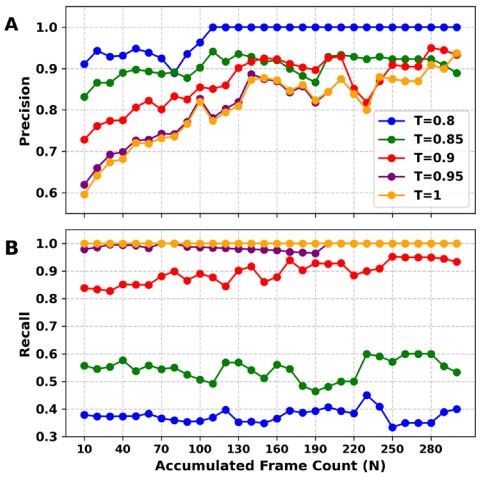

Figure 3. Precision and recall results for face recognition for different accumulation frame counts (*N*) and Euclidean distance thresholds (*T*).

suggest that in public safety scenarios, where reliability is paramount, using stricter parameters can help ensure that only highly confident identifications are acted upon, even if fewer matches are detected overall.

V. Conclusion

This work presented preliminary results for an edge-cloud video analytics framework tailored to public safety applications. Our initial experiments demonstrated the feasibility of edge processing for real-time video analytics, with edge deployment achieving the lowest total latency despite computational constraints. The face recognition use case illustrated the framework's potential while highlighting key trade-offs between accuracy and performance. The accumulation strategy successfully eliminates false positives, which is crucial for public safety applications, though at the cost of reduced recall.

The preliminary evaluation revealed several key insights into the performance of edge-cloud video analytics. Despite computational constraints, edge processing achieves lower total latency due to reduced transmission overhead, challenging the assumption that cloud processing is always superior for AI-driven tasks. The accumulation strategy effectively eliminates false positives but at the cost of increased false negatives, representing a critical trade-off in public safety applications. Additionally, selective offloading of computationally intensive tasks shows promise but requires careful consideration of network conditions and latency requirements.

Our future work will address current limitations through comprehensive evaluation and optimization, considering modern hardware, such as NVIDIA Jetson Orin and Raspberry Pi 5, in the evaluation. Future evaluation will include large-scale testing with realistic public safety scenarios and dynamic task distribution algorithms for optimal edge-cloud task allocation. An energy efficiency analysis will also examine the implications for power consumption and battery life.

References

[1] M. R. Endsley, "Toward a theory of situation awareness in dynamic systems," *Human Factors*, vol. 37, no. 1, pp. 32–64, Mar. 1995.

[2] D. Minoli, A. Koltun, and B. Occhiogrosso, Situational awareness for law enforcement and public safety agencies operating in smart cities – Part 2: Platforms, in S. Rani, V. Sai, and R. Maheswar, Eds. *IoT and WSN based smart cities: A machine learning perspective*. Switzerland: Springer International Publishing, 2022, pp. 139–162.

[3] L. U. Khan, I. Yaqoob, N. H. Tran, S. M. A. Kazmi, T. N. Dang, and C. S. Hong, "Edge-computing-enabled smart cities: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 200–10 232, Oct. 2020.

[4] S. Loss et al., "A framework for live situational awareness in stream-based 5G applications," in *1st IEEE Latin American Conference on Internet of Things*. USA: IEEE, 2025, to appear.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition*. USA: IEEE, 2016, pp. 779–788.

[6] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition*. USA: IEEE, 2015, pp. 815–823.

[7] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep Learning Face Attributes in the Wild," in *2015 IEEE International Conference on Computer Vision*. USA: IEEE, 2015, pp. 3730–3738.

[8] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, *Labeled Faces in the Wild: A database for studying face recognition in unconstrained environments*. University of Massachusetts, Amherst, USA, Tech. Rep. 07-49, October 2007.