
Università Politecnica delle Marche
Corso di Dottorato di Ricerca in Ingegneria dell'Informazione
Curriculum in Ingegneria Informatica, Gestionale e dell'Automazione

---

# Edge AI for human-behavior monitoring: designing lightweight Deep Learning methods on resource-constrained devices

Ph.D. Dissertation of:
**Daniele Berardini**

Advisor:
**Prof. Emanuele Frontoni**

Coadvisor:
**Sara Moccia, PhD**

Curriculum Supervisor:
**Prof. Franco Chiaraluce**

XXXVI edition - new series


Università Politecnica delle Marche
Corso di Dottorato di Ricerca in Ingegneria dell'Informazione
Curriculum in Ingegneria Informatica, Gestionale e dell'Automazione

---

# Edge AI for human-behavior monitoring: designing lightweight Deep Learning methods on resource-constrained devices

Ph.D. Dissertation of:
**Daniele Berardini**

Advisor:
**Prof. Emanuele Frontoni**

Coadvisor:
**Sara Moccia, PhD**

Curriculum Supervisor:
**Prof. Franco Chiaraluce**

XXXVI edition - new series

Università Politecnica delle Marche
Corso di Dottorato di Ricerca in Ingegneria dell'Informazione
Facoltà di Ingegneria
Via Brecce Bianche – 60131 Ancona (AN), Italy

# Abstract

This thesis focuses on the design and development of monitoring systems that use deep learning (DL) methods for real-time analysis of images and videos in indoor environments, following the paradigm of edge artificial intelligence (edge AI). The research explored two main application areas: in the security sector, the focus was on the analysis of RGB data for video surveillance; in the medical sector, the focus was on depth data analysis for monitoring and diagnostic support purposes.

In the context of security, representing the first application scenario of this research, the initial focus was on the design of a multi-camera video surveillance infrastructure. This infrastructure was developed with the objective of efficiently managing data from various sources. Simultaneously, the implementation of DL models capable of effectively detecting objects, while ensuring computational resource efficiency, was pursued. Moving towards the domain of weapon detection, specific analyses were conducted to identify the most suitable low-cost computing device for executing DL-based weapon detectors, comparing the NVIDIA® Jetson Nano with the Google Coral Dev. Tests performed on a specifically acquired dataset (WeaponSenseV0) highlighted that the NVIDIA® Jetson Nano provided better performance, both in terms of efficacy and efficiency. After selecting the NVIDIA® Jetson Nano as the ideal computing device, the research moved towards further developments. The next step was the collection of the WeaponSenseV1 dataset, which paved the way for creating the first edge AI framework to identify, through RGB video recordings, knives and guns held by people. In this DL approach, two cascaded convolutional neural networks (CNNs), optimized for edge devices, were used to address the challenge of recognizing small objects in RGB frames, a theme widely discussed in scientific literature. Although initial results were promising, the framework encountered a significant challenge: a decrease in efficiency in crowded environments. To overcome this obstacle, in the last phase of the research, developed and validated on the WeaponSenseV2 dataset, a super-resolution (SR) branch was integrated into the CNN for weapon detection. This approach was designed to maintain low computational complexity, activating the SR branch only during the training phase and removing it during deployment. The results obtained demonstrated that the new approach not only overcomes the limitations of previous work but also manages to maintain

reduced computational complexity on edge devices, simultaneously improving accuracy in weapon identification.

In the second application scenario, the research focused on developing DL methods for segmenting limbs of preterm infants using depth images acquired in neonatal intensive care units, combining the principles of GreenAI and edge AI. The primary objective was to create a CNN that offered high accuracy, while at the same time being more efficient and deployable in sustainable computational resources. This approach was adopted to reduce energy resource consumption, thus overcoming the limitations posed by state-of-the-art DL models, which tend to operate with huge computational requirements. The adoption of the edge AI paradigm in this area improved the accessibility of artificial-intelligence technologies and enhanced privacy and security by enabling the processing of sensitive data on-site and reducing dependence on external cloud resources. Furthermore, priority was given to increasing the reliability of systems, ensuring their operation even in scenarios with poor or nonexistent connectivity.

# Sommario

Questa tesi si concentra sulla progettazione e lo sviluppo di sistemi di monitoraggio che utilizzano metodi di deep learning (DL) per l'analisi in tempo reale di immagini e video in ambienti indoor, seguendo il paradigma dell'edge artificial intelligence (edge AI). La ricerca ha esplorato due settori applicativi principali: nel settore della sicurezza, l'attenzione si è rivolta all'analisi di dati RGB per la videosorveglianza; nel settore medico, l'analisi si è concentrata su dati di profondità per scopi di monitoraggio e supporto diagnostico.

Nel contesto della sicurezza, che rappresenta il primo scenario applicativo affrontato in questa ricerca, l'attenzione iniziale si è concentrata sul design di un'infrastruttura di videosorveglianza multicamera. Questa infrastruttura è stata sviluppata con l'obiettivo di gestire in modo efficiente i dati provenienti da svariate fonti. Contemporaneamente, si è proceduto all'implementazione di modelli di DL in grado di rilevare oggetti in modo efficace, assicurando al contempo un'efficienza dal punto di vista delle risorse computazionali. Spostandosi verso il dominio del riconoscimento di armi, sono state condotte analisi specifiche per identificare il dispositivo di computazione a basso costo più adatto per il progetto, confrontando l'NVIDIA® Jetson Nano con il Google Coral Dev. I test, eseguiti su un dataset specificatamente acquisito (WeaponSenseV0), hanno evidenziato che l'NVIDIA® Jetson Nano garantiva performance migliori, sia in termini di efficacia che di efficienza. Di conseguenza, tutti gli sviluppi successivi della ricerca sono stati direzionati verso l'utilizzo dell'NVIDIA® Jetson Nano. Dopo aver stabilito il dispositivo di computazione ideale, la ricerca si è spostata verso ulteriori sviluppi. Il passo successivo è stato la raccolta del dataset WeaponSenseV1, che ha aperto la strada alla creazione del primo framework edge AI per identificare, attraverso videoregistrazioni RGB, coltelli e pistole impugnati dalle persone. In questo approccio di DL, due reti neurali convoluzionali (CNN) in cascata, ottimizzate per dispositivi edge, sono state utilizzate per affrontare la sfida del riconoscimento di oggetti di piccole dimensioni nei frame RGB, un tema ampiamente discusso nella letteratura scientifica. Sebbene i risultati iniziali fossero promettenti, il framework ha incontrato una sfida significativa: una riduzione dell'efficienza in ambienti affollati. Per superare questo ostacolo, nell'ultima fase della ricerca, sviluppata e validata sul dataset WeaponSenseV2, è stato integrato un ramo di super-resolution (SR) nella CNN per la detection di armi. Questo approccio è

stato progettato per mantenere bassa la complessità computazionale, attivando il ramo di SR solo durante la fase di addestramento e rimuovendolo in fase di deployment. I risultati ottenuti hanno dimostrato che il nuovo approccio non solo supera le limitazioni dei lavori precedenti, ma riesce anche a mantenere una complessità computazionale ridotta sui dispositivi edge, migliorando contemporaneamente l'accuratezza nell'identificazione di armi.

Nel secondo scenario applicativo, la ricerca si è focalizzata sullo sviluppo di metodi per la segmentazione degli arti dei neonati pretermine tramite immagini di profondità acquisite in terapia intensiva neonatale, coniugando i principi della GreenAI e dell'edge AI. L'obiettivo principale è stato quello di creare una CNN che offrisse un'elevata accuratezza, ma che fosse allo stesso tempo più efficiente e integrabile in risorse computazionali sostenibili. Questo approccio è stato adottato per ridurre il consumo di risorse energetiche, superando così le limitazioni imposte dai modelli di DL allo stato dell'arte, i quali tendono a lavorare con abbondanti risorse computazionali. L'adozione del paradigma dell'edge AI in questo ambito ha migliorato l'accessibilità e la diffusione delle tecnologie di intelligenza artificiale ed ha potenziato la privacy e la sicurezza, permettendo l'elaborazione di dati sensibili in loco e riducendo la dipendenza da risorse cloud esterne. Inoltre, si è data priorità all'incremento dell'affidabilità dei sistemi, garantendone il funzionamento anche in scenari di connettività scarsa o inesistente.

# Acronyms

- Large Language Model (LLM)
- Visual Transformers (ViT)
- Edge Artificial Intelligence (edge AI)
- Internet of Things (IoT)
- deep learning (DL)
- convolutional neural network (CNN)
- Precision Agriculture (PA)
- Unmanned Aerial Vehicle (UAV)
- field of view (FoV)
- super resolution (SR)
- Local Area Network (LAN)
- Internet Protocol (IP)
- Power over Ethernet (PoE)
- Real Time Streaming Protocol (RTSP)
- Single Shot Multibox Detector (SSD)
- Universal Framework Format (*uff*)
- Frames per Second (FPS)
- Video Surveillance System (VSS)
- Closed-Circuit Television (CCTV)
- Single Board Computer (SBC)
- Machine Learning (ML)
- Support Vector Machine (SVM)
- You Only Look Once (YOLO)
- Internet Movie Firearms Database (IMFDB)
- Feature Pyramid Network (FPN)

- Cross Stage Partial (CSP)
- tensor processing unit (TPU)
- application specific integrated circuit (ASIC)
- trillion operations per second (TOPS)
- system-on-chip (SoC)
- 8-bit signed integer (INT8)
- post-training quantization (PTQ)
- half-precision floating-point (FP16)
- single-precision floating-point (FP32)
- TensorRT (TRT)
- Open Neural Network Exchange (*onnx*)
- Average Precision (AP)
- Area Under the Curve (AUC)
- Precision-Recall (PR)
- Precision (Prec)
- Recall (Rec)
- Spatial Pyramid Pooling (SPP)
- Transmission Control Protocol (TPC)
- stochastic gradient descent (SGD)
- Intersection over Union (IoU)
- billion floating point operations (GFLOPs)
- Generative Adversarial Networks (GAN)
- Spatial Pyramid Pooling Fast (SPPF)
- Path Aggregation Network (PANet)
- Enhanced Deep Super Resolution (EDSR)
- high resolution (HR)
- low resolution (LR)

- Hue Saturation Value (HSV)
- neonatal intensive care units (NICUs)
- RGB-depth (RGB-D)
- long short-term memory (LSTM)
- Multiply-Add operations (MACs)
- FLOPs per layer (FPL)
- coefficient of variation (CV)
- transposed convolutions (ConvTranspose)
- Dice similarity coefficient ($DSC$)

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Background and motivation

## 1.1 Dichotomy in digital trends: from complex and generalistic to edge-oriented and specific AI

We live in an ever-changing world, primarily driven by technological progress. In recent years, we have witnessed an increasingly rapid development of new technologies, with artificial intelligence (AI) leading this development.

The main factors contributing over time to this technological explosion are essentially three: the exponential increase in available data, advances in computational power, and the growing interest of the scientific community in the study of increasingly complex and accurate algorithms capable of solving tasks unimaginable until a few years ago [1]. To date, an example of the most important innovations made possible by the combination of these three factors is that of Large Language Models (LLMs), which are particularly advanced AI models that use enormous amounts of text to understand, interpret, translate, and generate natural language in a way that mimics human intelligence. These models are complex not only in their size but also in their ability to learn subtle linguistic and contextual nuances, enabling them to perform a variety of linguistic tasks with precision and naturalness previously inconceivable. However, LLMs present significant drawbacks. Their complexity requires a huge amount of computational power for training and execution, raising issues of environmental sustainability due to the vast energy consumption necessary to power these systems [2]. This aspect highlights the challenge of balancing technological advances with responsibility towards the environment. Furthermore, the high complexity and costs associated with the development and maintenance of these models limit their accessibility and affordability, often making them the prerogative of large organizations or entities with significant financial resources [3]. It is necessary to emphasize that this trend towards a centralization of power and technological control in the field of AI presents significant challenges for society. The risk that control over the development and distribution of AI systems is limited to a narrow group of individuals or entities could lead

to a dominance of their interests and behaviors. This situation could result in stifling innovation and restricting opportunities for a variety of perspectives and competencies. The absence of diversity and representation in decisions concerning AI technology could lead to the development of solutions that do not reflect the needs and perspectives of various groups of people, worsening existing inequalities. It is therefore essential to adopt policies and practices that promote equity, diversity, and inclusion in the field of AI, to ensure that its development and use are beneficial and accessible to all sectors of society.

LLMs clearly exemplify the challenges and limits associated with the use of complex AI models. However, these challenges are not unique to LLMs but also extend to other areas of AI, such as computer vision, which plays a fundamental role in applications ranging from security to medical diagnosis. The field of computer vision, for example, has seen enormous growth in the pre-training of Large Visual Models on vast image datasets. Advanced architectures like Visual Transformers (ViT), Swin Transformers, or SAM models have revolutionized the systems' ability to understand and analyze images, extracting detailed semantic information useful for a wide range of applications. However, computer vision models share the same limitations as LLMs. The need for large volumes of data for training and the demand for high computational power raise similar questions in terms of environmental impact and accessibility. Moreover, managing complex visual systems, due to their high cost and substantial energy consumption, may not be suitable in some contexts, such as in real-time image monitoring and analysis systems.

In the current landscape of constant technological evolution, there is a need to mitigate the aforementioned critical issues arising from the adoption of complex and generalist AI approaches. To address this need, part of recent research is focusing on implementing AI in an edge computing environment. This line of research, known as Edge Artificial Intelligence (edge AI), takes up the main concept of edge computing according to which data processing is performed directly on local devices, at the edge of the network, rather than transmitting large volumes of data to a central server or the cloud for analysis. The goal of edge AI is to make artificial intelligence more accessible, faster, and more energy-efficient. For such purposes, high-precision models requiring enormous computational resources are replaced by lightweight and optimized algorithms, which can be run on hardware with limited computational capabilities, such as smartphones, sensors, and other Internet of Things (IoT) devices. This not only reduces latency – the delay before a data transfer begins following an instruction for its transfer – but also improves privacy and security, as sensitive data can be processed locally without having to send it over a network. Furthermore, edge AI plays a fundamental role in enabling real-time applications and in scenarios where network connection is limited or unreliable. For example, in autonomous

vehicles, edge AI systems can quickly process huge amounts of data from vehicle sensors to make instant driving decisions. Similarly, in sectors such as industrial production or precision agriculture, edge AI allows monitoring and responding to variable conditions in situations where constant cloud communication is not practical or efficient.

The rise of edge AI marks a push towards more sustainable and decentralized solutions in the field of AI, offering a balance between computational power, speed, energy consumption, and privacy. This approach represents a significant paradigm shift, where not only high accuracy is pursued, but also the efficiency and adaptability of AI systems to different needs and usage contexts are valued.

## 1.2 Impact and Challenges of edge AI in Computer Vision

In computer vision, the use of deep learning (DL) has enhanced and directed scientific research and innovations in recent years. The adoption of architectures such as convolutional neural networks (CNNs) has overcome the limitations of traditional methods, thanks to their ability to process and interpret vast amounts of visual data with unprecedented accuracy.

Continuing in the same direction, part of the research continues to focus on the development of increasingly complex and accurate architectures, whose usability depends on the availability of an enormous amount of computational resources. As a result, the use of cloud resources or centralized servers becomes necessary to manage such computational workloads. However, the use of substantial computational resources and a centralized computing model is unsuitable or even impractical in many contexts. In applications that require real-time responses, such as surveillance systems, the latency associated with transferring data to a central server for processing can be a significant obstacle, making it essential to process data directly on the device. Similarly, in remote areas or emergency situations, where Internet connectivity is limited or absent, centralized processing is simply not a viable option. In sectors such as healthcare or public safety, it is crucial to prevent sensitive data from being sent to central servers for processing to protect them from potential breaches. In addition to this, the economic and scalability aspects must be considered: cloud processing requires a significant investment in terms of infrastructure and bandwidth, which may not be sustainable for startups or institutions with limited budgets, with a consequent negative impact on the dissemination and access to technology by everyone. Furthermore, the aspect of environmental sustainability is increasingly relevant, as centralized data centers have a significant environmental impact due to their high energy consumption and CO2

emissions.

The integration of edge AI in the field of Computer Vision is arousing increasing interest in the scientific community due to the numerous advantages it offers over traditional centralized data processing paradigms. This growing interest is reflected in the widespread increase of research works that exploit the combination of AI and edge computing across a broad range of application sectors [4].

As an example, in Precision Agriculture (PA), which aims to enhance crop productivity while reducing costs and environmental impact, the use of Unmanned Aerial Vehicles (UAVs) for image and video acquisition is common [5]. Many of the methods proposed in the literature use highly-demanding DL models that must necessarily be executed in the cloud [6, 7], but these solutions often prove ineffective due to challenges characteristic of PA, such as limited connectivity and bandwidth. Recent PA research therefore relies on edge-based approaches, where the execution of DL algorithms is shifted to edge devices onboard the UAVs [8, 9, 10], so as to overcome the problems related to the lack of connectivity in rural areas and drastically reducing latency times.

The use of edge-oriented methodologies is having a strong impact also in Video Analytics, a subset of Computer Vision specifically tailored to analyze video streams to automatically detect, track and analyze moving objects or behaviors and activities within video footage. The video analysis allows to reveal hidden patterns and connections, thus facilitating well-informed decision making and enabling prediction of future events. The design and application of DL methodologies enabled these systems to greatly outperform human monitoring in terms of accuracy and efficiency. These advantages have fostered the use of intelligent video analysis in a multitude of sectors, from surveillance and security to retail, industry, manufacturing, and healthcare.

The recent transition of video analytics applications to edge AI is driven by the need to tackle both common and application-specific issues that arise from relying on cloud-based data processing.

In the domain of surveillance and security, the most prominent issues relate to privacy and data security. Storing sensitive surveillance footage in the cloud raises significant concerns about unauthorized access and potential misuse of the data. In addition, dependence on stable, high-bandwidth network connections is critical, as any network failures or latency can critically hinder real-time monitoring and responses, which are essential in security operations. In surveillance applications such as intrusion detection or facial recognition, recent approaches utilize edge AI to detect abnormal behaviors, maintain public safety or identify unauthorized accesses, while ensuring efficiency and real-time feedback [11, 12, 13].

The retail sector faces similar challenges, particularly with regard to cus-

tomer privacy. Video analytics in retail often involve customer behavior and demographics, and storing this data in the cloud requires stringent data protection measures. In this sense, the development of edge-oriented methods enabled privacy-preserving customer analytics [14, 15] while also avoiding technical challenges related to the integration of various types of cameras and sensors with cloud platforms [16].

In the context of industry and manufacturing, the main concerns are latency and operational continuity. The slightest delay in processing video data can cause inefficiencies or failures, especially in automated environments. In addition, the high volumes of video data generated in industrial environments can be expensive and require a lot of bandwidth when transmitted and stored in the cloud. Relying solely on cloud services poses risks, in particular in the event of Internet outages, which can halt continuous operational monitoring. In light of these considerations, research has adopted edge solutions to enhance the reliability of systems in intelligent applications for industry, such as detecting manufacturing anomalies or monitoring production processes [17, 18].

Healthcare facilities, on the other hand, are tightly bound by regulatory compliance issues, such as adherence to laws and regulations like HIPAA in the U.S. or GDPR in Europe. Storing patient monitoring videos in the cloud introduces complexities in maintaining compliance. The sensitivity and confidentiality of patient videos also pose significant risks, as any data breach can have disastrous implications for patient privacy. In addition, bandwidth requirements for the transmission of high-quality patient monitoring videos in the cloud can be challenging. These issues are particularly relevant in settings where the healthcare infrastructure may have resource limitations or constraints, as is often the case in countries with publicly funded healthcare systems. In these cases, the use of the edge AI paradigm becomes crucial to contribute to the well-being of society, and research in the field, although lagging behind other application areas, is making progress in this regard [19, 20].

Despite these promising advancements in utilizing edge AI across various domains, significant challenges remain in its integration into Computer Vision, which can sometimes amplify domain-specific issues, potentially hindering progress in certain research areas. A major obstacle is the limited computing power and memory capacity of edge devices, especially when compared to centralized servers. Such limitations can significantly reduce the complexity of DL models that can be implemented on such devices. Moreover, in time-sensitive applications, balancing the need for real-time computations with the limited processing capacity of edge devices becomes critical. In these scenarios, developing DL models with an optimal trade-off between execution speed and accuracy is of crucial importance.

Driven by these considerations, this thesis focuses on the development of

edge-compliant DL methodologies specially designed for monitoring human behavior through video data analysis. It primarily targets two crucial areas where there are still open challenges that can be tackled, achieving significant improvements and pushing forward the state of the art in the integration of edge AI and computer vision: surveillance and security, with the task of weapon detection using surveillance cameras, and healthcare, focusing on pose estimation from depth cameras for monitoring preterm infants.

## 1.3 Aim of the thesis

The ultimate goal of this thesis is to contribute to the current research in the field of the edge AI applied to computer vision, with a specific focus on the design and development of intelligent systems based on DL methods for real-time monitoring of human behavior via analysis of video data acquired in indoor or outdoor environments. The research scope pursued was operationally declined in two different application sectors: the surveillance and security sector, with rgb data analysis for video surveillance purposes, and the healthcare sector, with depth data analysis for monitoring and diagnostic support purposes.

- In the surveillance and security scenario, in collaboration with the Italian company INIM Electronics, a leader in the security systems sector, the task of weapon detection was addressed. Two of the most significant challenges in this scenario are (i) the low accuracy in detection due to the small size of weapons relative to the camera's field of view (FoV) and (ii) the need to perform detection in real-time. The most widespread methods in the literature to mitigate the problem of small-sized weapons involve enlarging the frames to be analyzed through classic interpolation techniques or super resolution (SR), or using complex detection architectures with hundreds of millions of parameters, but both solutions are impractical in an edge context with low computational resources and when real-time feedback is necessary. In light of this, ***the goal is to design surveillance systems which integrate DL methods capable of being executed on edge devices with an optimal speed-accuracy trade-off***.

- In the healthcare scenario, the work focused on the development of efficient methods to segment preterm infants' limbs from depth images, for monitoring and diagnostic support purposes in assessing the quality of the infant's movements. State-of-the-art approaches are based on DL models that require high computational, memory, and energy resources, which limits their applicability only to scenarios with high computational and economic resources. The development of methods following the edge

AI paradigm (i) increases the accessibility and dissemination of such technologies and (ii) ensures greater privacy and security (i.e., sensitive data are processed locally without the need for external cloud resources) and greater reliability (i.e., applications continue to function even with absent or limited Internet connection). In view of this, ***the goal is to develop DL methods that are less onerous in terms of computational, energy and memory resources, thus making them more suitable for use in resource-limited environments***.

## 1.4 Thesis overview

An overview of the thesis structure is proposed hereafter for the sake of readability:

- **Chapter 2**: underlines the video surveillance's pivotal role in modern security systems and presents the crucial need of developing efficient weapon detection algorithms in automatic surveillance systems, for ensuring safety and security. Within chapter subsections, the advantages that the adoption of the edge AI paradigm can bring in this context will be explored, as well as the inherent challenges of the weapon detection task. The limits and open issues in the state of the art will be disclosed and innovative DL methodologies will be proposed to gradually meet the actual needs and tackle the open issues. All the presented analyses and methods have the common objective of exploring, demonstrating and validating the potential and advantages of developing edge-compliant DL methodologies for weapon detection in video surveillance.

- **Chapter 3** highlights the need to implement efficient and sustainable algorithms for preterm infants' limb segmentation. The chapter will present an approach developed considering both environmental and economic aspects. The design of the approach was guided by strategies to minimize the computational resources required for algorithmic computation, contributing to meet the demand for more sustainable and cost-effective solutions.

- **Chapter 4** offers an overview of the conclusions of each work from previous chapters. Then, final considerations and open challenges of healthcare ecosystem are discussed.

## 1.5 Thesis contribution

In surveillance and security, weapon detection on edge is crucial as it enables real-time analysis of surveillance footage, enhancing public safety and responsiveness to eventual threats. During the three years of the PhD, the following publications contributed to expanding the state of the art in the field of edge AI-based surveillance systems for real-time monitoring of human behavior, with focus on the crucial task of weapon detection. The contribution, in journals and conferences, focused on (i) designing multi-camera surveillance systems with the integration of edge AI (ii) exploring and evaluating the performance of edge devices for weapon detection (iii) developing DL methods capable of being executed on edge devices for real-time weapon detection from surveillance videos.

- **Berardini, D.**, Mancini, A., Zingaretti, P., & Moccia, S. (2021, August). Edge artificial intelligence: A multi-camera video surveillance application. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (Vol. 85437, p. V007T07A006). American Society of Mechanical Engineers.

- **Berardini, D.**, Galdelli, A., Mancini, A., & Zingaretti, P. (2022, November). Benchmarking of dual-step neural networks for detection of dangerous weapons on edge devices. In *2022 18th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA)* (pp. 1-6). IEEE.

- **Berardini, D.**, Migliorelli, L., Galdelli, A., Frontoni, E., Mancini, A., & Moccia, S. (2023). A deep-learning framework running on edge devices for handgun and knife detection from indoor video-surveillance cameras. *Multimedia Tools and Applications*, 1-19.

- **Berardini, D.**, Migliorelli, L., Mancini, A., & Marín-Jiménez, M. J. Edge AI and Super-Resolution for enhanced Weapon Detection in Video Surveillance. *Engineering Applications of Artificial Intelligence* (under review)

In healthcare, monitoring limb movement in preterm infants is crucial to assess the presence of neuro-motor dysfunctions. Although research in this field has developed highly reliable models, computational costs have often been overlooked. These models typically require huge computations which leads to the need for expensive hardware, posing environmental sustainability issues and making their clinical use a privilege, contradicting the goal of creating widely accessible healthcare technologies. With the aim of tackling such issues, the

following contributions deal with (i) the design of edge AI-compliant and cost-efficient methods for clinical applications in preterm infants' pose estimation, (ii) the analysis of the usability of DL methodologies on edge devices, along with improvements in terms of efficiency.

- Cacciatore, A., Migliorelli, L., **Berardini, D.**, Tiribelli, S., Pigliapoco, S., & Moccia, S. (2022, May). Some Ethical Remarks on Deep Learning-Based Movements Monitoring for Preterm Infants: Green AI or Red AI?. In *International Conference on Image Analysis and Processing* (pp. 165-175). Cham: Springer International Publishing.

- Migliorelli, L., Cacciatore, A., Ottaviani, V., **Berardini, D.**, Dellaca', R. L., Frontoni, E., & Moccia, S. (2023). TwinEDA: a sustainable deep-learning approach for limb-position estimation in preterm infants' depth images. *Medical & Biological Engineering & Computing, 61*(2), 387-397.

- **Berardini, D.**, Cacciatore, A., Moccia, S., Mancini, A., & Migliorelli, L. A Methodological Strategy to Develop Sustainable and Cost-Effective Deep Learning Approaches for Green Edge AI. *IEEE Transactions on Sustainable Computing* (under review)

## 1.6 Publications

The following publications, which are only partially related to the topic of my doctorate and will not be discussed in the thesis, are the result of collaborations within the VRAI group and between research groups:

- Migliorelli, L., **Berardini, D.**, Cela, K., Coccia, M., Villani, L., Frontoni, E., & Moccia, S. (2023). A store-and-forward cloud-based telemonitoring system for automatic assessing dysarthria evolution in neurological diseases from video-recording analysis. *Computers in Biology and Medicine, 163*, 107194.

- Gonçalves, C., Lopes, J. M., Moccia, S., **Berardini, D.**, Migliorelli, L., & Santos, C. P. (2023). Deep learning-based approaches for human motion decoding in smart walkers for rehabilitation. *Expert Systems with Applications, 228*, 120288.

- Migliorelli, L., **Berardini, D.**, Rossini, F., Frontoni, E., Carnielli, V., & Moccia, S. (2021, November). Asymmetric Three-dimensional Convolutions For Preterm Infants' Pose Estimation. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)* (pp. 3021-3024). IEEE.

# Chapter 2

# Edge AI in Surveillance Systems for Effective Weapon Detection

## 2.1 Edge AI Preliminaries: a Multi-Camera Video-Surveillance Application

Nowadays, video surveillance plays a crucial role. The increase in the availability of surveillance data, from cameras installed in private places such as homes, offices, educational institutions, and commercial buildings, raises the issue of how to effectively process this data to extract useful information [21]. Monitoring these videos is a time-consuming and tiresome task for humans, particularly when it requires constant supervision. Moreover, the fact that a single high-definition video camera can generate about 10 GB of data per day also points out the challenges associated with data storage.

Over the years, the development of algorithms for automatic processing of surveillance video has become an extremely active field of research trying to overcome these challenges [22]. Such algorithms, in addition to reducing the human workload, enable the storage of only the high-level information derived from the analysis, rather than storing the entire volume of raw video data. The analysis of surveillance videos involves various tasks, such as object detection, action recognition, and classification of those objects or actions as normal or abnormal. Early approaches were based on traditional computer vision techniques [23]; however, in recent years, the potential of artificial intelligence, particularly DL, has overcome the limitations of these traditional methods, achieving much superior results due to its ability to learn from data. In fact, DL algorithms are able to extract information from raw data, such as images and videos, through a training process based on a large volume of annotated data.

Nevertheless, the development of DL algorithms for video processing in surveillance presents several challenges, including the requirement for real-time (or near real-time) processing and the need for cost-effective hardware for their

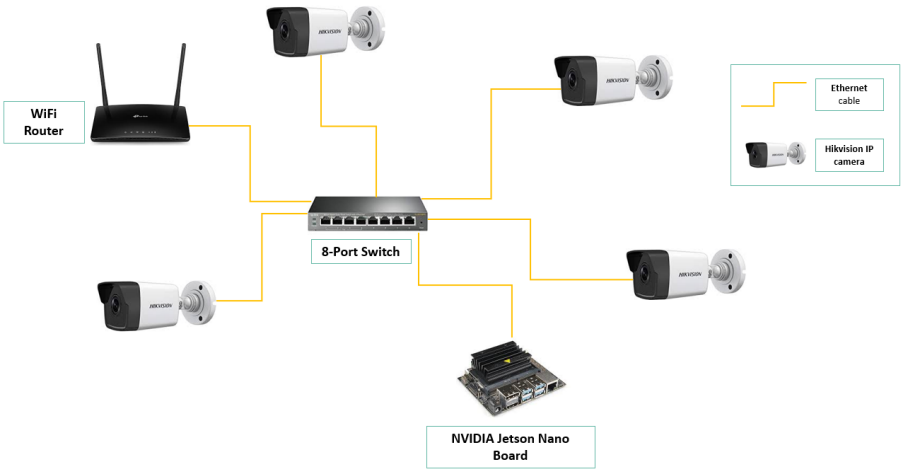



**Figure 2.1** Outline of the proposed infrastructure. The sensing devices and a software module inside the computing device makes up the sub-infrastructure described in Sec. 2.1.1.2. The deep learning based module inside the computing device make up the data processing sub-infrastructure described in Sec. 2.1.1.3.

use. To overcome these issues, edge AI is emerging as a promising solution, combining artificial intelligence, IoT and edge computing [24]. By shifting computing workloads from remote centers, such as cloud servers, to camera devices, it considerably reduces communication overhead and enables accurate real-time analysis. Therefore, edge AI is able to bring significant improvements in the video surveillance domain, enabling faster data processing, reducing latency and providing more efficient data management [25, 26].

Guided by these premises, this preliminary research proposes a multi-camera video surveillance infrastructure integrating the edge AI paradigm. The proposed research aims to provide a low-cost and horizontally scalable solution which makes an efficient use of the state-of-the-art DL techniques for object detection from videos, laying the foundation for using edge-compliant methods via a framework to manage and analyze multiple video streams in real time.

To this end, the work focused on (i) the design of a physical network made up of cameras acting as sensing nodes, which send data through a local area network to a computing device, along with the implementation of an efficient solution to handle multiple-source video data, and (ii) the deployment of an off-the-shelf DL model to perform real-time people and object detection tasks over multiple-source video streams, even in resource-constrained settings.

### 2.1.1 Methods

Figure 2.1 shows an outline of the infrastructure designed for collecting and processing video streams from sensing devices. In this section, first, the sens-

ing devices and computing board employed are introduced, highlighting their main characteristics to motivate their choice (Sec. 2.1.1.1). Next, data acquisition (Sec. 2.1.1.2) and processing (Sect. 2.1.1.3) stages of the proposed infrastructure are presented, focusing on the description of video stream handling for the former and the description of the DL model, as well as its deploy on the edge device, in the latter. For the sake of simplicity, data acquisition and processing are described referring to a single data stream. However, since both acquisition and processing modules are thread-based, in the presence of multiple cameras, a new thread pair (i.e., acquisition thread and processing thread) can be created for each camera from the main process.

#### 2.1.1.1 Acquisition and Computing Devices

To implement the data collection sub-infrastructure, the use of Internet Protocol (IP) cameras as sensing devices was chosen. An IP camera, differing from traditional analog cameras, is capable of sending and receiving data via an IP network. This feature offers several advantages; notably, IP cameras do not require a local recording device but only a Local Area Network (LAN) connection. As a result, they provide higher video quality and resolution compared to traditional cameras. Furthermore, IP cameras can offer Power over Ethernet (PoE) or wireless connections, depending on the requirements, and maintain image clarity over long distances.

As shown in Figure 2.1, the data collection sub-infrastructure in this work consists of four Hikvision® 2MP Fixed Bullet Network Cameras[1] connected to a LAN with PoE.

For the data processing sub-infrastructure, an embedded computing board was chosen to receive and analyze data, so as to implement the edge computing paradigm. These edge computing devices have similar functionalities to a standard computer but are constructed as a single circuit board, which brings computation closer to the data collection sub-infrastructure, offering multiple advantages at a low cost.[27]

Despite the limited computational power of computing boards, adopting the edge computing paradigm presents several advantages over alternatives such as cloud-based solutions. These include faster processing due to reduced latency, enhancing the overall system's responsiveness, and reduced security risks as data may not need to leave the local network. Additionally, privacy is improved, especially when handling video data. The distributed nature of the infrastructure also increases reliability and fault tolerance, while the use of low-cost devices reduces infrastructural costs.

The NVIDIA® Jetson Nano™ Developer Kit[2] was the chosen edge device

[1] https://www.hikvision.com/
[2] https://developer.nvidia.com/embedded/jetson-nano-developer-kit

for the data processing sub-infrastructure. It features a 4 GB RAM, a 4-core ARM A57 CPU, and an on-board GPU with 128 CUDA cores based on Maxwell microarchitecture design, supplemented by a heat sink, enabling the execution of low-power artificial intelligence systems for image classification, detection, and segmentation.

Thus, the Jetson Nano was selected as the core processing module thanks to its computing capabilities, available at a relatively low cost. Its specifications allowed for the development of a lightweight DL algorithm that processes video data from multiple sources simultaneously.

#### 2.1.1.2 Data Acquisition Stage

As detailed in Sec. 2.1.1.1, 4 Hikvision Network cameras were used as sensing nodes in the data acquisition stage. At first, a network switch supporting PoE technology was connected to the LAN router to exploit the PoE capabilities of the IP cameras. The four cameras were then placed in the monitoring area (e.g., a living room) and connected to the network switch via standard CAT6A Ethernet cables, in order to send and receive data over the LAN.

To complete the physical connection between the infrastructure nodes, also the Jetson Nano board was connected to the network switch through an Ethernet cable, thus facilitating its communication with the rest of the infrastructure. It is worth noting that the Jetson Nano, as well as the other devices, was connected to the same switch for practical purposes only. The devices only need to be connected to the same LAN, not necessarily to the same LAN entry point.

Once the physical connections between sensing and computing devices is established, a virtual connection is set up between the cameras and the Jetson Nano, enabling the latter to receive video streams. The Real Time Streaming Protocol (RTSP) is used as the standard protocol for making the streams available over the LAN. RTSP is designed to control streaming media servers within communication systems.

The initial step to create the virtual connection was the definition of a unique RTSP URL for each camera using the Hikvision web service, along with the access credentials for connecting to each device via the URL. Subsequently, a stream handler module was implemented within the Jetson Nano board. This module's purpose is to open each stream towards the cameras, enabling data reception.

A Python script was implemented for video stream handling, which, given the RTSP URL[3] and access credentials, utilizes Python bindings of OpenCV, an open source computer vision library, for interacting with an external program named GStreamer to open the stream and start the data acquisition.

[3] the URL was in the format $rtsp://\langle username\rangle:\langle password\rangle@\langle url\rangle$

GStreamer, a powerful framework for creating streaming media applications, allows to set up a stream handler with desired characteristics (e.g., data type, compression encoding/decoding standards). In detail, the process starts with the creation of the camera handler managing the stream. The main Python process creates a string with appropriate parameters to open a pipeline using GStreamer. Thus, OpenCV interacts with GStreamer and initializes the camera handler. Once the connection between the Jetson Nano and the camera is established, the main Python process generates a thread that begins autonomous data acquisition at a resolution of 640x480 pixel using the previously initialized stream handler.

Thus, at the end of this process, the Jetson Nano begins receiving video data as consecutive frames, which are forwarded to the data processing module for DL analysis.

#### 2.1.1.3 Data Processing Stage

The data processing stage relies on a DL algorithm to analyze the video frames collected in the data acquisition stage. To this end, an off-the-shelf Single Shot Multibox Detector (SSD) MobileNetV2 network [28][29] was employed as lightweight DL algorithm to perform the object and people detection tasks.

The SSD MobileNetV2 network was chosen for its efficient balance of detection speed and accuracy. Unlike two-step architectures like Faster R-CNN [30], SSD localizes objects in one step and employs multi-scale computation for varied object sizes. This approach, along with MobileNetV2 as its feature extraction backbone, allows SSD to provide real-time, accurate detections even in limited-resource environments.

MobileNetV2, a lightweight Convolutional Neural Network, includes a 3x3 convolutional layer and 19 residual bottleneck layers. These layers feature a combination of subsequent 1x1, 3x3, and 1x1 convolutions, with Batch Normalization and ReLU6 activations, using residual connections [31] for efficient feature extraction. SSD extends MobileNetV2 with six convolutional blocks, enabling multi-scale object detection across its fully convolutional structure.

Therefore, to achieve real-time detections over video streams, the Jetson Nano was equipped with an off-the-shelf SSD MobileNetV2 detection model. This model, implemented in TensorFlow[4] — a widely-used Python library for deep learning model training and inference — comes pretrained on the COCO dataset[5]. COCO is a benchmark dataset with over 200k annotated images and 80 object categories, including a wide range of objects, food, vehicles, animals, and people. Given that the work focus was primarily on the developing a

[4]https://www.tensorflow.org/
[5]https://cocodataset.org/

framework to manage and process multiple video streams with edge AI integration, the use of a pretrained model allowed for immediate inference applications without the need for additional training.

In the deployment process of SSD MobileNetV2 into the Jetson Nano, in order to fully exploit the edge device's computing capabilities, NVIDIA's TensorRT — a library for high-performance deep learning inference on NVIDIA® devices valuable for real-time systems and embedded applications — was used. *TensorRT* optimizes models given in Universal Framework Format (*uff*), with ".uff" extension, according to the hardware characteristics of the specific NVIDIA® board. The off-the-shelf TensorFlow model was available as a frozen inference graph in the *protobuf* file format with ".pb" extension. The frozen inference graph, a specific TensorFlow file format, represents the model as a serialized graph with all training weights and can be loaded in a TensorFlow environment for inference. Thus, as a first step, the TensorFlow model was converted into the generic *uff* intermediate format for *TensorRT* compatibility. Then, the *uff* model was converted into a *TensorRT* engine, resulting in a serialized file with model optimizations based on the Jetson Nano hardware. Following these static steps, the final detection model was obtained as a *TensorRT* engine for efficient use during inference on the Jetson Nano board, without the need to rely on TensorFlow or other similar frameworks.

After the adaptation of the off-the-shelf detection model to the edge device characteristics, the data processing pipeline, as mentioned in Sec. 2.1.1.2, was expanded to manage the analysis of the video stream received from the data collection thread. Breaking it down, to carry out the processing stage, the main Python process initially creates a new processing thread, with the initial task of opening the *TensorRT* engine from a file, deserializing it and loading it into the GPU's allocated memory, thereby establishing the necessary context for inference. Subsequently, the processing thread begins to process the frames captured by the acquisition thread from the RTSP stream.

More in details, the processing thread takes the acquired frame, now shared, applies preprocessing to the frame to align with the model's input requirements, namely image resizing at 300x300 pixel, channel order adjustment, and normalization, and inputs it into the SSD MobileNetV2 loaded in the GPU. The model processes the normalized frame, returns the predicted output and the processing thread performs post-processing to generate the final detection coordinates, confidence scores, and predicted label classes, storing them in a shared variable. The variable is accessible to the main Python process, which awaits the results and, upon their receipt, can utilize them for various actuation purposes.

**Table 2.1** Inference speed in Frames Per Second (FPS) comparison on multiple cameras.

| **N. of Cameras** | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| **FPS** | 25.0 [6] | $\sim$ 20.5 | $\sim$ 14.7 | $\sim$ 10.0 |

## 2.1.2 Results


**Figure 2.2** Qualitative results showing the live detection obtained with SSD MobileNetv2 inference on 2 cameras. Image captured under remote working conditions during the COVID-19 pandemic.

Table 2.1 presents the results in terms of Frames per Second (FPS) obtained with up to 4 cameras connected to the LAN, each transmitting over an RTSP stream monitored by the acquisition module.

As detailed in Sec. 2.1.1.2 and in Sec. 2.1.1.3 each acquisition thread on the Jetson Nano managed an RTSP stream, continuously capturing frames at a resolution of 640x480, and the corresponding processing thread then resized each frame to 300x300 and inputted it into the SSD MobileNetV2 network. The network's detection output was subsequently sent back to the main process for visualization or various actuation purposes.

In a single-camera configuration, the system achieved an inference rate of 25.0 FPS. However, with the addition of cameras and parallel data streams, a gradual decline in FPS was observed. With two cameras, the FPS dropped to approximately 20.5. This further decreased to around 14.7 FPS using three cameras, with the Jetson Nano processing data from three different RTSP streams. With four cameras, the inference speed decreased to about 10.0 FPS. It is important to note that the fame rate in the single-camera setup is upper bounded by the Hikvision camera's frame rate, which is 25 FPS.

Figure 2.2 illustrates some qualitative detection results in a living room setting using two cameras. For visualization purposes, a monitor was connected to the edge device. These qualitative results demonstrate the SSD MobileNetV2

[6] 25 FPS is the maximum frame rate of each Hikvision camera.

ability to detect multiple objects of varying sizes while running on an edge device in multi-camera settings. The system effectively identified both a person and a chair from two different perspectives and sizes, as shown in the left and right images of Figure 2.2.

### 2.1.3 Discussion

The designed infrastructure exploited a thread-based approach to handle the video stream coming from each IP camera used for video surveillance. For each camera an acquisition thread was responsible for frame collection and a processing thread analyzed the acquired frames to perform object and people detection, providing results to the main process. The proposed infrastructure achieved promising results in terms of real-time inference even with the 4-camera setting, since an inference speed of 10.0 FPS in the object and people detection tasks guarantees immediate response in case of anomalous event (e.g. detection of specific target objects or people intrusion).

It is worth highlighting that the use of a video surveillance system based on the proposed infrastructure has multiple advantages: (i) the reduction of latency times due to potential feedback delays from external computational resources for data analysis, crucial in time-sensitive applications (ii) the independence from Internet connectivity for the operation of the architecture, avoiding monitoring interruptions in situations of temporary or total lack of external connectivity, fundamental in safety-critical surveillance systems that require continuous monitoring (iii) the extreme horizontal scalability of the system via the addition of new nodes for computation, with all the related benefits [32], (iv) the low cost of the entire infrastructure, enabling a global spread of intelligent surveillance systems, thus democratizing access to the technology and simultaneously increasing the level of overall safety and well-being.

This research, despite employing an off-the-shelf approach for object and people detection without specific task training for the DL model, establishes a foundation for future development. It demonstrates the feasibility of integrating artificial intelligence into video surveillance on edge devices. This is achieved by using a hardware and software infrastructure designed to handle multiple surveillance cameras, employing a thread-based method for real-time video acquisition and analysis through a deep learning model.

Furthermore, the ease of extension and adaptability of the proposed infrastructure also allows its use in a variety of applications that make use of artificial intelligence-based models for video analysis, as the crucial application of weapon detection in video surveillance, explored extensively in this thesis.

## 2.2 A Deep Dive into Weapon Detection in Modern Surveillance: Impact, Challenges and Emerging Trends

Weapon-related crimes, particularly those involving guns and knives, are a significant global issue, accounting for 76% of all homicides [33]. This alarming statistic underscores the critical importance of Video Surveillance Systems (VSSs) in various environments, including airports, hospitals, homes, and offices [34]. VSSs play a pivotal role in enhancing public safety by providing real-time monitoring, which is essential for the early detection of potential threats [34]. The ability of these systems to identify the presence of handguns and knives may enable prompt interventions by security personnel, potentially averting violent incidents and reducing the rate of homicides. Furthermore, installing surveillance cameras serves as a preventive measure, discouraging potential offenders and thereby playing a significant role in reducing crime rates [35].

However, the effectiveness of VSSs is currently limited by the reliance on human operators for round-the-clock monitoring, which is a demanding and costly process [36, 34]. Continuous observation leads to fatigue, reducing the operators' alertness and the overall efficiency of the surveillance [37]. Thus, while VSSs are a crucial tool in addressing weapon-related crimes, its potential is not fully realized due to the limitations of human-based monitoring. This highlights the need for advanced solutions, such as automated detection technologies, to enhance the effectiveness of video surveillance in public safety [38].

To meet these needs, research in the field of video surveillance is largely based on advances in the broader field of generic object detection, adapting automated detection methodologies to the specific context of video surveillance. Indeed, in recent years, the field of generic object detection has been extensively studied and significant progress has been made in the state of the art. With the growing popularity of DL, automatic object detection approaches based on this paradigm have gradually replaced previous approaches based on traditional machine vision techniques (e.g., deformable parts model [39], Selective Search [40]). These DL-based methods have outperformed their predecessors in terms of speed and reliability, establishing themselves as essential tools for augmenting the capabilities of human operators. However, despite research advances in the development of general-purpose object detectors, the nuanced and complex task of efficiently detecting weapons in VSSs remains a significant and unresolved challenge, highlighting an area of ongoing research and development in the field [41, 42].

One of the primary difficulties in detecting weapons, as opposed to generic objects, lies in the small size of weapons compared to the camera's FoV [43]. In Closed-Circuit Televisions (CCTVs), weapons often appear significantly smaller compared to the overall FoV. This is further exacerbated when the weapons are located at a considerable distance from the camera, diminishing their apparent size.

In generic object detection, the problem of identifying small objects has been partially mitigated through the use of complex DL methods, which are paired with higher resolution inputs [44, 45, 46, 47]. These approaches demonstrate improved performance, but they come with significant challenges. Indeed, they necessitate extensive training datasets to achieve optimal effectiveness, and they require considerable computational power for inference processing. As highlighted in [42], this presents practical limitations for their deployment in real-world surveillance settings. Particularly in the domain of video surveillance, the lack of comprehensive, real-world datasets for weapon detection severely hampers the implementation of these complex detection systems [42]. Additionally, the need for powerful computational resources for their deployment can be excessively costly and energy-demanding. This is particularly problematic in settings with limited resources, like smaller public areas or less developed regions, where the implementation of such demanding systems is impractical. Consequently, this poses a major hindrance to the broad adoption of these systems for weapon detection [42]. To improve the detection of small weapons and address the shortage of extensive datasets, solutions leveraging object-segmentation techniques designed to require weaker form of supervision could be employed [48, 49]. However, the inherent complexity of these DL methods inevitably demands improved hardware capabilities. This may pose a barrier to the widespread adoption of such systems.

### 2.2.1 Related Work

Despite the popularity of generic object detection, research efforts in automatic handguns and knives detection from surveillance videos are quite limited, especially in edge computing settings with Single Board Computers (SBCs). Among the seminal works in this field, the authors in [50] proposed an approach for firearms and knives detection from CCTV images based on visual descriptors and Machine Learning (ML). The knife detection algorithm relies on sliding window technique followed by MPEG-7 based feature extraction and Support Vector Machine (SVM) for classification. The firearm detection also includes an image pre-processing step using background subtraction and Canny edge detection algorithms, followed by the sliding window and a classification based on MPEG-7 region shape descriptor. Both detection algorithms were evaluated

**Table 2.2** Summary of the state-of-the-art approaches in weapon detection.

| Work | Method | Dataset | Limitations |
|---|---|---|---|
| Grega et al., 2016 | Sliding Window + MPEG-7 + SVM,<br>handgun and knife classification | custom,<br>CCTV,<br>released w/o box-level annotations | burdensome for edge devices,<br>not real-time |
| Verma et al., 2017 | VGG16 Faster RCNN,<br>handgun detection | IMDB,<br>non-CCTV,<br>public | unrealistic dataset,<br>suffers from small objects,<br>burdensome for edge devices |
| Olmos et al., 2018 | VGG16-based region proposal approach,<br>handgun detection | custom,<br>non-CCTV,<br>released | unrealistic dataset,<br>suffers from small objects,<br>burdensome for edge devices |
| Fernandez-Carrobles et al., 2019 | SqueezeNet Faster RCNN,<br>gun and knife detection | custom/COCO/Olmos et al., 2018<br>non-CCTV,<br>not released | unrealistic dataset,<br>low knife-detection performance |
| Lim et al., 2019 | Multi-Level FPN-based single-stage detector,<br>handgun detection | custom,<br>CCTV,<br>not released | burdensome for edge devices |
| González et al., 2020 | ResNet50 Faster RCNN with FPN,<br>handgun detection | custom,<br>Synthetic/CCTV/non-CCTV,<br>released | burdensome for edge device,<br>low speed for real-time domain |
| Olorunshola et al., 2023 | YOLOv5,<br>person, handgun, rifle and knife detection | custom/Google Open Images,<br>non-CCTV<br>not released | unrealistic dataset,<br>suffers from small objects |

on a custom-built dataset. Despite the valuable contribution, the use of sliding window approach and other time-consuming techniques limit the applicability in real-world scenarios.

In recent years, the increasing popularity of DL has prompted the diffusion of new general-purpose architectures for object detection. Among these, some of the most widely used state-of-the-art detectors includes Faster R-CNN, which is based on a two-stage detection process (i.e., a region proposal stage followed by the object localization and classification), and one-stage detectors (i.e., direct object localization and classification) like the You Only Look Once (YOLO) family [51, 52, 53] and SSD [28].

Following this trend, recent works in handguns and knives detection from surveillance videos have focused on exploiting such general-purpose detection architectures.

In [54] a handheld gun detection approach based on DL is proposed. The authors exploited a Faster R-CNN with a VGG16 backbone and compared its performance against several ML methods on the Internet Movie Firearms Database (IMFDB), proving its superiority. Similarly, in [55], a sliding window and a region proposal approach, both based on a VGG16 CNN classifier, were compared. The region proposal approach outperformed the sliding window in terms of speed and detection accuracy on a custom-built dataset.

These works have the merits of having highlighted the validity of DL over standard ML methodologies in video surveillance field. However, the datasets they use to validate the approaches do not fully represent real-world scenarios. Indeed, they are mainly made up of static images not acquired by CCTVs and firearms are often the largest and the only object in the foreground. These datasets oversimplify the detection task, potentially skewing the results. Additionally, the results in [55] on test videos show a high number of false negatives (i.e., missed detections).

While the previous work focused exclusively on the detection of guns, the authors in [56] addressed the detection of both guns and knives. To this end, a Faster R-CNN was trained on a custom-built dataset obtained by collecting data from various sources, including a portion of the dataset in [55] for guns and COCO images for knives. Both GoogleNet and SqueezeNet CNNs were tested as backbones for the Faster R-CNN. While SqueezeNet achieved comparable results to [55] for gun detection, it performed poorly on knife detection. In contrast, the GoogleNet-based architecture showed better performance in knife detection compared to the other methods, even though with relatively low overall detection performances. Similar previous works, this study has limitations in the composition of the dataset, which is not representative of a real-world scenario. In addition, the proposed solution needs two distinct architectures to be effective in the detection of both handguns and knives, limiting

the applicability in resource-constrained environments.

In the last few years, research effort in general-purpose object detection has also focused on the development of DL modules to be added on top of CNN architectures, aiming to improve detection performance. To this end, one of the most popular components is Feature Pyramid Network (FPN) [57], which combines high and low-resolution features from different CNN layers, improving the detection at different scales. Following these improvements over the existing state-of-the-art, more recent works in handguns and knives detection from CCTV have adopted DL architectures with the integration of such components [58, 41], trying to tackle the issues related to small object sizes, discussed in Sec. 2.2. Authors in [58] introduced an approach based on a single-stage object detector integrating a multi-level FPN to enhance localization ability for handgun detection from CCTV. The approach was validated on a custom dataset comprising 5500 images of handguns extracted from CCTV videos. In [41] a Faster R-CNN with a FPN was exploited to perform gun detection on CCTV images. The training was performed on several combinations of non-CCTV data from [55], custom CCTV data and synthetic data. The evaluation on CCTV data highlighted that while the addition of synthetic training data slightly improved the results, the addition of non-CCTV data even decreased detection performances on small objects.

Although both works integrate modules on top of the detection architectures to improve detection performances on the respective CCTV datasets, the results obtained in terms of weapon detection and inference speed do not allow to translate their approaches into the real-world domain. This is clearly expressed by the authors themselves in the conclusions of [41].

With an eye towards focusing on computationally undemanding detection architectures, the authors in [59] present a comparative analysis of the one-stage detectors YOLOv5 and YOLOv7 [60] on a custom dataset of people, handguns, rifles and knives, with images from Google Open Images Dataset, Roboflow Public Dataset and local sources. YOLOv5 outperformed YOLOv7, but the overall results were rather poor, hindering the real-world implementation of the approach. Moreover, as demonstrated by the image samples shown in the paper, the dataset does not mirror a real-world domain. Table 2.2 summarizes the state of the art approaches with their methodological details and limitations.

Despite recent advancements, there is still a lack of approaches that exploit edge AI for weapon detection in surveillance videos. The inherent challenges in developing efficient algorithms for edge devices with limited computational capacity, coupled with the need to achieve good accuracy even when dealing with very small weapons, largely contribute to creating this research gap. An additional major challenge is the difficulty in finding representative surveillance

datasets that accurately reflect various real-world situations, which are essential for training and validating effective detection algorithms. Effectively tackling these challenges would allow to exploit edge AI potential for real-time weapon detection in video surveillance, achieving significant benefits in terms of cost reduction, privacy protection, and latency reduction [61].

With the aim of filling the existing gap and pushing forward the state of the art in weapon detection on edge devices, the proposed research emphasizes addressing the identified challenges and shifting towards approaches centered on the edge AI paradigm. The following sections present:

- **Sec. 2.3**: A detailed description of the fully-CCTV WeaponSense dataset, used to validate the proposed weapon detection approaches. The dataset has three versions, with incremental improvements on the data composition between versions;
- **Sec. 2.4**: A benchmark study performing a comprehensive performance comparison of two low-cost single board computers running a weapon detection algorithm, to determine the edge device with the optimal characteristics;
- **Sec. 2.5**: A novel edge-oriented, DL-based approach for detecting handguns and knives in indoor VSSs, specifically designed to tackle the detection of smaller objects and operate on edge devices with limited computational power;
- **Sec. 2.6**: An innovative approach that integrates deep learning and super-resolution methods, enhancing the detection of small-sized weapons on edge devices without extra computational cost.

(a) (b)

(c) (d)

(e) (f)

**Figure 2.3** Sample of frames extracted from recordings in the WeaponSense dataset are shown to highlight the related challenges (e.g., multiple people, different weapons and non-threatening objects, distance from camera). For visualization purposes only, the handguns and knives have been pointed out in red.

## 2.3 The WeaponSense Dataset

Due to the lack of benchmark datasets and the scarcity of publicly available datasets representative of the real world in the field of weapon detection, part of the research work was spent on the creation of a custom dataset. The WeaponSense dataset was collected and used to validate deep learning methodologies developed for weapon detection on edge devices.

Three versions of the dataset, available upon request, were created: (i) WeaponSenseV0, consisting of 30 video sequences featuring handguns; (ii) WeaponSenseV1, which adds 22 video sequences featuring knives to the previous version; (iii) WeaponSenseV2, which includes 4 additional videos, captured

in a different environment, containing both guns and knives. For WeaponSenseV0 and WeaponSenseV1, acquisition was conducted using a Dahua 4MP Bullet Network Camera, a commercial camera with IP connected to a LAN network with PoE. The choice of the IP camera, suggested by INIM Electronics, an Italian leader in the surveillance systems sector, was driven by the good compromise between cost, performance, and quality in terms of image resolution and compression technology.

The camera was placed in the upper corner of a nearly empty room to capture video sequences in which a variable number of subjects were free to move, with one of them holding a weapon, such as a knife or a gun. The acquisition sessions were carried out using a custom Python script, collecting a total of 52 video sequences of 30 seconds each. The frame rate of the camera was set to 10 FPS with a default resolution of 1280x720 pixels, resulting in 300 frames for each video sequence. In the 52 collected video sequences, 19 different subjects appear holding a gun (30 sequences) or a knife (22 sequences). Furthermore, the same subject appearing in multiple videos wore different clothes. The average number of people per video is 2.88, with a standard deviation of 1.05.

For the creation of WeaponSenseV2, the acquisition of additional video sequences was conducted using a Hikvision 4MP Fixed Turret Network Camera. In this case, the camera was installed on the ceiling of a room set up with shelves containing commercial products, to simulate a commercial activity. The same acquisition protocols described for the first two versions of the WeaponSense dataset were followed. As a result, 4 new sequences were acquired, with 3 different subjects and multiple weapons in each sequence, for a total of 300 frames each. In WeaponSenseV2, which thus comprises 56 videos, the average number of people per video and the standard deviation are 2.87 and 0.99, respectively, remaining almost unchanged from the previous version. The WeaponSense dataset, in all its versions, was gathered with the goal of simulating an indoor real-world application domain, so as to overcome the limitations found in the datasets used in the current state of the art [54, 55]. Some of the challenges of the WeaponSense dataset are shown in Figure 2.3.

The major one is the very small size of the objects to be localized compared to the whole image (i.e., $\sim$0.1% of the image area, computed on the average ground-truth boxes areas), due to the distance of the people from the camera. Another significant challenge is the poor contrast of the objects against the background, along with motion blur in images. Additional challenges include the presence of multiple people who may be holding non-threatening objects (e.g., smartphone as in Fig. 2.3(b)), variations in subjects' poses (e.g., sitting or standing as in Figs. 2.3(a) and 2.3(c)) and orientations (e.g., Fig. 2.3(a) with respect to Fig. 2.3(f)), multiple weapons in the image (as in Fig. 2.3(e)), and variability both in terms of subjects and intra-class objects (i.e., the use

of different objects belonging to the same class).

## 2.4 Benchmark of Cost-Effective Single Board Computers for Weapon Detection

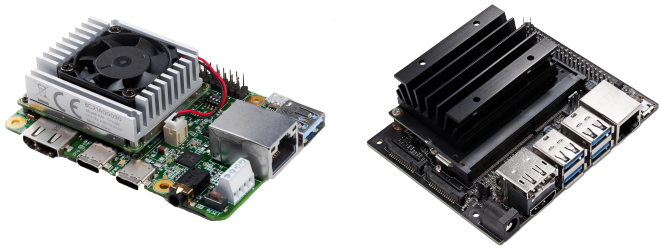
Monitoring threats and preventing criminal activities is an open challenge to safeguard the health of citizens. Modern DL techniques have been shown to outperform traditional techniques in terms of speed and accuracy of results, giving a significant boost in this area by providing real-time information critical for prevention of criminal activities. In this context, edge computing is a powerful paradigm that can be successfully adopted to run artificial intelligence applications while ensuring security, privacy, and flexibility without suffering downtime and latency. The opportunity to leverage the integration of edge computing and artificial intelligence in the domain of video surveillance, as described in Sec. 1.2, has been explored by several researchers in the literature [13]. The authors in [62] presented a video surveillance application leveraging a DL algorithm on edge devices to detect, count and track people. Similarly, in [11], an approach for real-time human detection on resource limited SBCs was proposed. All the contributions emphasized the benefits of implementing artificial intelligence solutions at the edge, instead of relying on cloud processing.

Nevertheless, as far as weapon detection is concerned, the use of SBCs to run DL algorithms is still very limited, with open challenges on how to translate the proposed approaches in a real-world domain [41]. To push forward progress in this context, an essential first step is to make the researchers aware on the limits and capabilities of using SBCs in weapon detection, as done in other domains by existing studies on performance benchmarking of different edge devices [63].

Driven by these considerations, a comprehensive comparison was conducted in this study between two of the most popular SBCs for on-edge analysis: Google Coral Dev board [7] and NVIDIA® Jetson Nano [8]. The benchmark aimed to evaluate the performance in the execution of a lightweight dual-step approach for weapon detection.

More in details, the comparison between SBCs was on a dual step detection architecture for hand-held weapons detection that leverages a prior detection of the people using SSD MobileNetV1 [64], and a subsequent detection of the potential hand-held weapon wielded by each person using YOLOv4-Cross Stage Partial (CSP) [65] network. An extensive performance evaluation comparison - in terms of both inference speed and impact on detection accuracy after quantization - between the two edge SBC devices was conducted using the custom WeaponSenseV0 dataset, introduced in Sec. 2.3, consisting of 1,307

[7] https://coral.ai/products/dev-board
[8] ttps://developer.nvidia.com/embedded/



(a) Google Coral Dev (b) NVIDIA® Jetson Nano

**Figure 2.4** Edge devices selected to benchmark inference performance.

**Table 2.3** Overview of the general features of the two edge SBC devices used in this work.

| Feature | Google Coral Dev | NVIDIA® Jetson Nano |
|---|---|---|
| Memory | 1GB LPDDR4 @1.6GHz | 4GB 64-bt LPDDR4 @1.6GHz |
| Cpu | NXP i.MX 8M SoC Quad-core ARM Cortex-A53, (plus Cortex-M4F) @1.5 GHz | 64-bit Quad-core ARM A57 @1.43GHz |
| OS | Mendel Linux | Jetson4Tegra |
| AI Unit | Google Edge TPU ML accelerator co-processor | 128-core NVIDIA® GPU |
| Optimization Frameworks | TensorFlow Lite PyCoral | TensorRT |

annotated frames.

This work contributes to the future research by providing an insight on the computational limitations to address when developing edge-compliant approaches for weapon detection.

## 2.4.1 Methods

This section gives an overview of the SBCs used in the benchmark analysis and presents the DL model implemented for weapon detection, outlining its deployment on each edge device.

### 2.4.1.1 Target Edge Devices

For the performance benchmark, two popular devices in the edge computing domain were selected: the Google Coral Dev (Fig. 2.4(a)) and the NVIDIA®

Jetson Nano (Fig. 2.4(b)).

In 2020, the Coral Dev board was released by Google for running low power artificial intelligence applications on embedded devices. The Coral provides high performance and high-speed neural network inference thanks to its tensor processing unit (TPU) co-processor, an ad-hoc built-in application specific integrated circuit (ASIC) using the TensorFlow framework. The workflow to create a model for the Edge TPU is based on TensorFlow Lite. The Edge TPU can perform 4 trillion operations (tera-operations) per second (TOPS), using 0.5 watts for each TOPS (2 TOPS per watt). This board offers a fully-integrated system, including NXP's iMX 8M system-on-chip (SoC), eMMC memory, 1GB or 4GB LPDDR4 RAM, Wi-Fi, and Bluetooth. A lightweight open-source operating system derived from Debian Linux, named Mendel, has been developed for the Coral Dev to facilitate the development of fully integrated models.

The second selected SBC, the Jetson Nano, was released by NVIDIA®. This board supports running artificial intelligence applications with low power consumption. The presence of a GPU with 128 CUDA cores based on the Maxwell micro-architecture in the edge device enables high inference performance. It is available in either a 2GB or 4GB RAM version and includes a 4-core ARM® Cortex®-A57 MPCore CPU, achieving a peak performance of 472 GFLOPs for the rapid execution of modern AI algorithms.

To summarize, Tab. 2.3 shows the main features of the two devices utilized for running DL applications on the edge.

#### 2.4.1.2 Deep Learning Approach

Driven by the goal of detecting small dangerous objects held by people, a two-step DL approach was chosen. This dual-step process is necessary to first identify people within the camera's field of view, and then to detect a weapon within the area (bounding box) around the person.

The SSD MobileNetV1 network [64] was utilized for people detection, while YOLOv4-CSP [65] was selected for detecting dangerous weapons held by people. The selection of these two models was based on the hardware constraints of the edge SBCs, as well as on the optimizations (e.g., models' quantization) of the models on these devices. Using a single accurate-but-complex model would not have been compatible with the memory limitations of the SBCs, and many state-of-the-art models are not yet optimized for efficient deployment on edge devices.

The Coral Dev requires Edge TPU-compiled models with full integer quantization to speedup model inference using its Edge TPU co-processor. The people detection model was deployed on the Coral Dev straightforwardly, as Google developers released a ready-to-use, pre-trained and Edge TPU-compiled version with 8-bit signed integer (INT8) quantization. To deploy the weapon detec-

tion model on Coral Dev, it was first converted from Tensorflow to TensorFlow Lite (*tflite*), a framework providing a set of tools that enables on-device DL inference. During this conversion, post-training quantization (PTQ) in INT8 was applied, preparing the model for the edge device. The *tflite* INT8 model was then compiled for Edge TPU to obtain the final optimized version for the execution on Coral Dev.

The Jetson Nano, with its on-board GPU, can run models both with quantization in half-precision floating-point (FP16) format and without quantization in single-precision floating-point (FP32) format, while the INT8 quantization is not supported. To maximize inference speed, FP16 quantization and additional optimizations using *TensorRT* (TRT) were applied to both MobileNetV1 and YOLOv4-CSP. TRT, the framework developed by NVIDIA®, facilitates high-performance DL inference with hardware-specific optimizations. To maximize the throughput for on-device inference, TRT allows to convert a model into a ready-to-run TRT engine in the form of a serialized binary file. The MobileNetV1 model was converted and optimized to TRT with FP16 quantization through the intermediate conversion to *uff*. As regards the YOLOv4-CSP model, the deployment was done via a first conversion of the model from Keras to the Open Neural Network Exchange (*onnx*) format, which is an open format for AI models, and then an optimized FP16 TRT engine was created from the latter format.

After the models deployment on both edge devices, the system was tested in a real-world scenario. During inference, frames were acquired from an IP camera connected to the tested device at a fixed rate of 30 FPS. Following the acquisition and processing setup introduced in 2.1, a separate thread continuously acquired a new frame, putting it in a shared buffer, replacing the previous frame in the buffer. Meanwhile another thread, accessing the shared buffer, asynchronously processed the available frame with the DL models.

## 2.4.2 Experimental protocol

### 2.4.2.1 Data Preparation and Training settings

Starting from the WeaponSenseV0, introduced in Sec. 2.3, each of the 30 video sequences was processed to remove irrelevant frames at the beginning and end of the acquisition, and one frame every four was sampled, so as to increase differences between consecutive frames. Then, each of the resulting 1307 frames, was manually labeled using LabelMe[9], a publicly available annotation tool. Each annotation was performed by drawing a bounding box to tightly enclose the weapon and by assigning it to the class *weapon*.

[9]https://github.com/wkentaro/labelme

An off-the-shelf SSD MobileNetV1 pretrained on the COCO dataset [66] was exploited for inference. The FP16 version of the SSD MobileNetV1 was used for inference with the NVIDIA® Jetson Nano board, while its INT8 version was used for inference with the Google Coral Dev board. To train the YOLOv4-CSP, the dataset WeaponSenseV0 was split in ∼75%, ∼10%, ∼15% for the training, validation and testing set, respectively, with data split at video level. The weapon detector was trained using Tensorflow on a GPU NVIDIA® GeForce RTX™ 3090 with a FP32 format. The training was performed for 80 epochs using Adam optimizer with a batch size of 8 and an initial learning rate of 0.005. The best weights across the epochs were chosen based on the validation loss. After training, PTQ was applied to the weapon detector to obtain both the FP16 Jetson-compliant and the INT8 Coral-compliant models.

#### 2.4.2.2 Performance metrics

To benchmark the edge devices two primary metrics were used: (i) the PASCAL VOC Average Precision (AP) [67] to evaluate the detection performance and (ii) the FPS to evaluate the inference speed. The AP (Eq. 2.1) was computed by taking an approximated Area Under the Curve (AUC) of the Precision-Recall (PR) curve

$$AP_{weapon} = \sum_{k=0}^{n-1} (r_{k+1} - r_k)\rho_{interp}(r_{k+1}) \tag{2.1}$$

where $n$ is the number of recall values in the PR curve and $\rho(\tilde{r})$ is the precision measured at recall $\tilde{r}$.

To further assess the detection performance also Precision (Prec) (Eq. 2.2) and Recall (Rec) (Eq. 2.3) were computed as secondary metrics,

$$Prec = \frac{TP}{TP + FP} \tag{2.2}$$

$$Rec = \frac{TP}{TP + FN} \tag{2.3}$$

where $TP$, $FP$, $FN$ are the correct detections, the wrong detections and the missed detection, respectively, computed at a threshold = 0.4.

The FPS were computed as an exponential moving average (Eq. 2.4)

$$FPS_t = \begin{cases} Y_0, & t = 0. \\ \alpha Y_t + (1 - \alpha) \cdot FPS_{t-1}, & t > 0 \end{cases} \tag{2.4}$$

where $\alpha$ is the coefficient of weighting decrease, set equal to 0.05, $Y_t$ is the instant FPS value at time $t$ and $FPS_t$ is the exponential moving average value at time $t$.

**Table 2.4** Benchmark performance of SSD and YOLO running in sequential mode on the edge devices.

| Device | Accelerator | Datatype | FPS | | | | | AP | Prec | Rec |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | n° people | | | | | | | |
| | | | 0 | 1 | 2 | 3 | 4 | | | |
| Google Coral Dev | TPU | INT8 | **36.5** | 2.9 | 1.5 | 1.1 | 0.9 | 98.8 | 96.8 | 98.9 |
| NVIDIA® Jetson Nano | GPU | FP16 | 23.8 | **4.5** | **2.5** | **1.7** | **1.4** | **99.6** | **100** | **99.6** |

Due to the dependence of the tested approach with respect to the number of people in the camera's FoV, the calculation of FPS was performed taking into account a variable number of people in the camera's field of view. Thus, a different FPS value is computed for each different number of people (i.e., from 0 to 4 people) in the IP camera FoV, to assess in a real-world scenario the impact of increased workload on each SBC. Except for the inference with 0 people, in each inference run there is exactly a weapon held by a person.

### 2.4.3 Results

Table 2.4 shows the results obtained in terms of AP, Prec, Rec and FPS on each edge device. Jetson Nano achieved the highest results in terms of primary PASCAL VOC and secondary metrics (AP = 99.6, Prec = 100.0 and Rec = 99.6), while the INT8-quantized framework running on Coral achieved slightly worse results (AP = 98.8, Prec = 96.8 and Rec = 98.9). Comparing the two edge devices in terms of FPS, it turns out that the Coral outperformed the Jetson Nano when doing inference with 0 people in the camera FoV (FPS = 35.6 for the Coral and FPS = 23.8 for the Jetson Nano). On the other hand, increasing the number of people in the camera FoV led to better results for the Jetson Nano with respect to the Coral, going from FPS = 4.5 (on the Jetson Nano) against FPS = 2.9 (on the Coral) for 1 person to FPS = 1.4 (on the Jetson Nano) against FPS = 0.9 (on the Coral) for 4 people. Fig. 2.5 shows the FPS trend of both the edge devices when varying the number of people in the camera FoV.

### 2.4.4 Discussion

Although the introduction of DL enhanced existing approaches for the early detection of criminal activities, there remains potential for further improvements, especially in terms of edge computing perspectives.

The work compared two SBC running a two-step approach for identifying dangerous weapons in the camera FoV. The results obtained, detailed in Table 2.4, shows the strengths and weaknesses of each edge device in terms of AP,

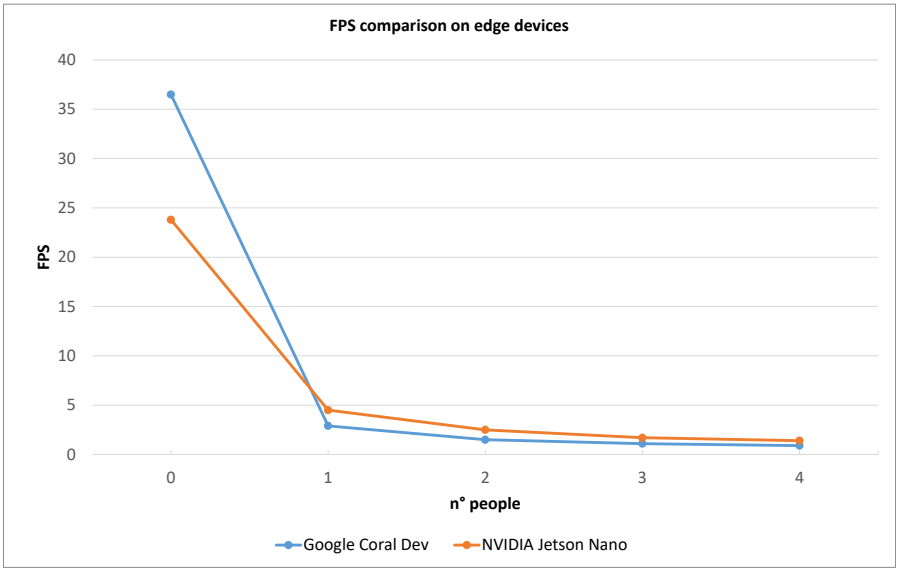



**Figure 2.5** FPS comparison on edge devices.

Prec, Rec and FPS.

The FP16-quantized framework running on the Jetson Nano achieved better results with respect to the INT8-quantized framework for both the PASCAL VOC AP and secondary metrics (i.e., Prec and Rec). The higher performance of the former could be attributed to its FP16 weights' representation, which is closer to the original FP32 weights' representation used in the training phase compared to the INT8 weights' one. Indeed, although the post-training quantization of the FP32 framework to INT8 representation allowed to obtain a lighter-memory model with respect to the FP16 post-training quantization (i.e., 8 bits per weight instead of 32), the overall expressivity of the network turned out to be lower, resulting in slightly worse detection performances.

As regards benchmark on the inference speed of the edge devices, the Coral Dev obtained impressive results when running inference with 0 people in the camera FoV, while from 1 to 4 people the Jetson Nano outperformed the Coral Dev in terms of FPS, showing a smoother FPS degradation (Fig. 2.5). The reason for this trend may be attributed to the memory management of the edge devices. The Coral Dev, beyond the shared 1GB RAM, has an additional 8MB SRAM (See Tab. 2.3) that can cache model parameters for the fastest possible data transfer towards the TPU, while the Jetson Nano solely relies on a shared 4GB RAM. Thus, since the no-people inference use only the first-step SSD MobileNetV1, the Coral can cache the whole model's parameters (i.e., model size of 7MB) in the 8MB SRAM once and do high-speed continuous inference. When dealing with one or more people, the second-step YOLOv4-CSP

model comes into play and the Coral Dev faces an additional time overhead for rewriting SRAM to switch among the two models. Furthermore, the YOLOv4-CSP's parameters (i.e., model size of 56MB) do not fit entirely in the SRAM, requiring also external memory readings (i.e., shared RAM) to do inference. On the other side, the Jetson Nano directly loads both models in its shared 4GB RAM, resulting in a slower FPS for no-people inference but maintaining higher FPS values with respect to the Coral Dev when both models are used.

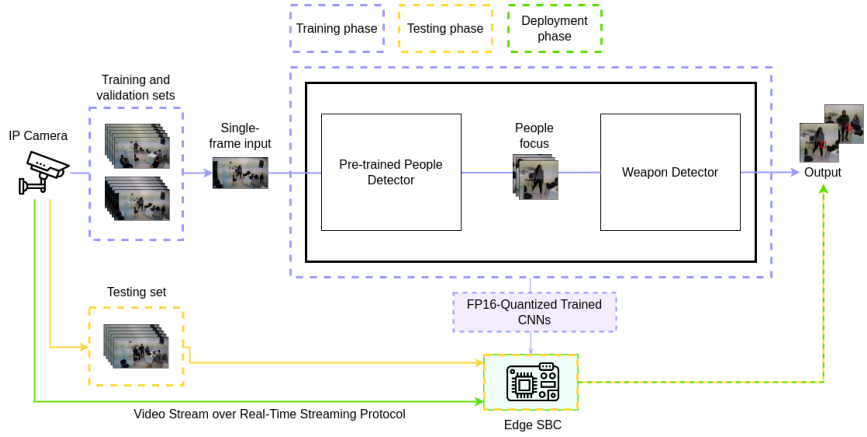
To conclude, the research delineated herein did not concentrate on examining the optimal DL architecture for undertaking the task of interest, namely, the automated detection of weapons from surveillance videos. The exploration of CNN architectures is reserved for subsequent sections of this thesis. The findings from this investigation were instrumental in guiding the selection of the NVIDIA® Jetson Nano as the most appropriate computational device. This decision was based on the congruence of the device's capabilities with the specific requirements of the experimental framework, as evidenced by the aforementioned research outcomes.

## 2.5 Edge-Driven Deep Learning Framework for Handgun and Knife Detection in Indoor Video Surveillance

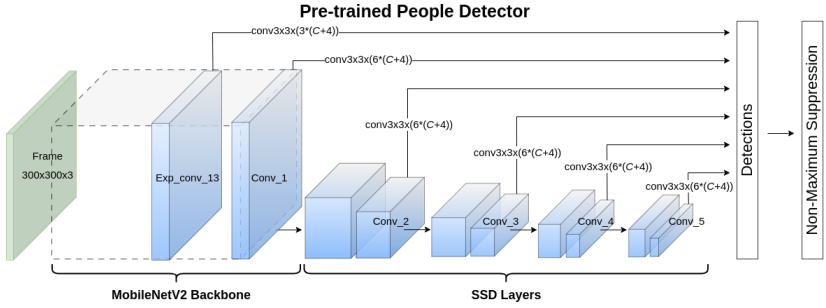
As extensively discussed in Sec. 2.2, the primary challenge in integrating edge computing and artificial intelligence lies in developing methods that are both accurate and light enough to be executed on edge devices. The specific challenges of weapon detection in video surveillance, which include the need to accurately recognize small-sized weapons in non-static images (i.e., extracted from videos), further hinder the development of edge-oriented methodologies. Here, it is crucial to find an optimal balance between the computational complexity of the model, the execution speed, and the accuracy. Much research in the literature focused on maximizing the accuracy of weapon detection methods, neglecting aspects of execution speed and computational compatibility with edge devices. Conversely, the rare approaches that focused on developing edge-oriented methods, trying to balance all the necessary components, show gaps in terms of applicability to real domains. In fact, these approaches were validated on datasets that do not adequately reflect the real context of video surveillance (e.g., they use static images with foreground or non-handled weapons), resulting in a drastic reduction in accuracy when applied in real scenarios.

Summing up the challenges and requirements in this still relatively unexplored research field, there is the need to find an effective yet efficient approach for small handheld weapons detection in CCTV under resource-constrained settings. Motivated by the exploratory benchmark on the edge devices presented in Sec. 2.4, and with the aim of taking a step towards the resolution of the still-open challenges, this work presents a DL-based approach oriented to the edge computing paradigm for handgun and knife detection from indoor surveillance videos. The innovative contribution of the work is the proposal of an approach robust to the small-object size yet deployable on edge devices with limited computing capacity – and consequently costs. To this end, it leverages a first CNN to obtain a prior detection of the people in the frame, and a second CNN to perform a subsequent detection of the potential handgun or knife within each person's bounding box. The approach was validated on the fully-CCTV WeaponSenseV1 dataset, introduced in 2.3.

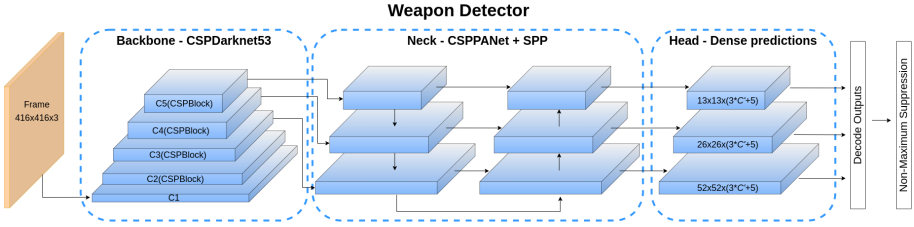
The following sections (i) describe the implemented approach and the deployment on the edge (Sec. 2.5.1); (ii) delineate the ablation studies, including the training settings and the performance metrics used to validate the approaches (Sec. 2.5.2); (iii) show the experiments carried out and the results obtained (Sec. 2.5.3); and (iv) present the discussion of results (Sec. 2.5.4).




(a)




(b)




(c)

**Figure 2.6** (a) Workflow of the proposed approach for indoor handgun and knife detection. After proper dataset preparation (described in Sec. 2.5.2.1) the weapon detector was trained using the output of the people detector (as detailed in Sec. 2.5.1.1) and the mean average precision performance was computed on the test set. Both convolutional neural networks were quantized in half-precision (i.e., FP16 quantization) and deployed in the NVIDIA® Jetson Nano (as in Sec. 2.5.1.2)) for real-time processing of the IP camera video stream. The details on the convolutional structure of (b) the people detector and (c) the weapon detector are shown, too.

To enable fair comparisons, the codes are available on GitHub[10].

## 2.5.1 Methods

### 2.5.1.1 Dual-Stage Deep Learning Approach

The proposed approach, whose workflow is shown in Fig. 2.6(a), is based on the observation that a weapon becomes dangerous only when carried by a human. Thus, a two-step detection process is proposed, involving a prior detection of the people within each frame, followed by the detection of the potential handgun or knife within each person's bounding box. Each step relies on a specific DL architecture, aiming at maximizing the speed-accuracy trade-off. The architectural choices also take into account the SBC hardware constraints in terms of memory footprint, as this is a primary concern when dealing with edge devices.

To carry out the prior people's detection, the SSD MobileNetV2 network [28, 29] was used (Fig. 2.6(b)), since it represents a good compromise between computational speed and people detection accuracy. Although its detection performance on the COCO dataset is lower than other architectures [29], it achieves nearly-optimal results when the performance evaluation is restricted on the *person* category, as shown in [68]. The SSD meta-architecture was chosen since it performs object localization by adopting a single-stage approach as opposed to other two-stage architectures which enhances accuracy to the detriment of speed (e.g., [30]). This allowed to reduce inference time.

MobileNetV2 was adopted as backbone for features extraction to further increase the speed of the SSD. MobileNetV2 is a lightweight CNN with a 3×3 convolutional layer followed by 19 inverted residual blocks [29], made up of three 1×1, 3×3, 1×1 convolutions interleaved with batch normalization and ReLU6 activation function, with a residual connection [31] between the 1×1 layers. The peculiar blocks' structure reduces the number of network parameters, thus increasing inference speed. The SSD meta-architecture stacks on top of the MobileNetV2 six output convolutional blocks, obtaining six different scales of detection for each input image.

An intermediate processing on each person's bounding box within the frame was performed before the weapon detection step. In particular, with the aim of preserving the objects' aspect ratio, a square crop from the original image was computed, according to both the center and the maximum side (between width and height) of each person's bounding box. Each crop was then fed to the subsequent step, resizing it according to the needs.

Once the prior information on each person's location within the frame was obtained, the subsequent step performed the detection of potential weapons

[10]https://github.com/daniebera/on-the-edge-weapon-detection

carried by a subject. The YOLOv4-CSP network [65] was implemented for handgun and knife detection (Fig. 2.6(c)) due to its ability in detecting objects with respect to other state-of-the-art detectors, while attaining a good inference speed in resource-constrained hardware. Such results are highlighted by the comparison in both [69] and [29] on the COCO dataset. The YOLOv4-CSP was designed starting from the YOLOv4 network, originally introduced in [69]. As regards the backbone, the YOLOv4-CSP exploits the existing CSPDarknet53 (i.e., a Darknet53 with CSP stages each made up of 1,2,8,8,4 residual layer, respectively) and converts only the first CSP stage into an original Darknet residual layer for efficiency purposes. Instead, as regards the neck, it introduces CSP connections in the Path Aggregation Network architecture of the YOLOv4 by transforming the original reversed darknet layers of YOLOv4 in reversed CSP darknet layers, maintaining the Spatial Pyramid Pooling (SPP) module. As output layers, in its original configuration YOLOv4-CSP has three $1 \times 1$ convolutions with 255 filters each, so as to obtain detection at three different scales. As a result, the YOLOv4-CSP obtained a substantial gain in terms of trade-off between speed and accuracy, making it suitable for challenging detection tasks in resource-constrained settings.

To accomplish the detection on two classes (i.e., *knife*, *gun*), each of the three original output layer of the YOLOv4-CSP was replaced with a $1 \times 1$ convolution having $3 \times (2+5) = 21$ filters. In this way, each output layer provides a $n \times n \times 21$ map in which each of the $n \times n$ spatial locations encodes the information (i.e., coordinates, class scores and probability of containing an object or *objectness* score) of 3 candidate bounding boxes, so that 3 candidate bounding boxes $\times$ (2 class scores + 4 bounding box coordinates + 1 *objectness* score) = 21. The final selection of the most promising bounding boxes among candidates was performed according to the non-maximum suppression algorithm.

#### 2.5.1.2 Deployment on Edge Devices

Guided by the findings in Sec. 2.4, the SBC to deploy the DL framework was the NVIDIA® Jetson Nano Developer Kit, due to its capability to run low-power artificial intelligence applications effectively on its onboard GPU, making it suitable for the proposed research. As in Sec. 2.4.1.2, TRT framework was used to optimize DL models and convert each model in a serialized and FP16-quantized engine to enable higher performance inference on the Jetson Nano. The MobileNetV2 model was converted in a straightforward way from Tensorflow to the *uff* format for TRT-compatibility and from *uff* to an optimized TRT engine. The YOLOv4-CSP model was first converted from Keras to *onnx* format, then, a TRT engine was created from the *onnx*-like model. Since an internal default parameter of a Keras layer (i.e., upsampling) caused incompatibility issues for a fully optimized inference with TRT, the

**Table 2.5** Number of annotated frames — prior to online data-augmentation application — and number of video sequences related to each class of interest.

| | **n° frames** | **n° videos** |
|---|---|---|
| *knife* | 1118 | 22 |
| *gun* | 1307 | 30 |
| total | 2425 | 52 |

**Table 2.6** Number of video sequences related to train, validation and test datasets for each class. In round brackets is given the number of total frames in each set for each class, obtained by summing the number of labeled frames of each video belonging to the set considered.

| | **Train** | **Validation** | **Test** |
|---|---|---|---|
| *knife* videos (frames) | 17 (870) | 2 (98) | 3 (150) |
| *gun* videos (frames) | 24 (1030) | 2 (103) | 4 (174) |
| total videos (frames) | 41 (1900) | 4 (201) | 7 (324) |

Keras model structure was re-implemented with a custom upsampling layer. Moreover, two custom plugins in TRT were used to allow post processing of the model predictions and to apply Non-maximum Suppression algorithm, otherwise not supported in TRT engine.

Once the TRT engines were obtained, following the work in Sec. 2.1 and in Sec. 2.4, a thread-based pipeline was exploited to acquire frames from the IP camera and to process them first with the SSD Mobilenetv2 and then (potentially) with the YOLOv4-CSP. After the opening of a video stream via RTSP between the IP camera and the Jetson Nano, a thread was responsible for handling video data as consecutive frames and forwarding them to the DL algorithms for processing. The people detector and the weapon detector were implemented as distinct processes that communicate via the Transmission Control Protocol (TCP), enabling them, in principle, to be physically decoupled (i.e., in an edge computing architecture with multiple nodes each process can communicate with the others independently from its physical location). The entire pipeline was designed to be suitable even in multi-camera settings via the opening of multiple camera-to-device RTSP video streams.

## 2.5.2 Experimental Protocol

### 2.5.2.1 Dataset Preparation

The WeaponSenseV1 was used, and further processed to create the final dataset for validating the DL algorithms. Following the same protocol described in Sec.2.4.2.1, for each new video sequence integrated to the previous version of the WeaponSense, one frame in every 4 was sampled. The resulting 1118 new frames, carefully labeled with *knife* annotations, were added to the 1307 frames with *gun* annotations.

Table 2.5 summarizes main information on the dataset, including number of frames and videos for each class (i.e., *knife* and *gun* classes).

During the inference, in the second step of the proposed approach, the YOLOv4-CSP for handguns and knives detection takes as input a square crop centered on each person's location instead of the original frame. Thus, the dataset was further processed to enable a faster training of the algorithm. A new dataset was constructed by square cropping on the original frame (having a resolution of 1280×720 pixels) the detected person holding the weapon. Each crop was then resized to 416×416 pixels (i.e., the network's input size) and the ground-truth bounding box coordinates were adjusted accordingly, to obtain the dataset used in training phase.

Table 2.6 summarizes the splitting strategy. The split was explicitly performed at video level (i.e., without mixing frames extracted from the same video across train, validation and test set) to attenuate possible bias. As a result of this strategy, the 78.3%, 8.3% and 13.4% of the available frames were used for training, validation and testing, respectively.

To improve DL algorithms' generalization capabilities, online data augmentation strategies were implemented on the training dataset. The applied data-augmentation transformations were: the change in brightness level, so as to simulate a scenario where artificial and natural lighting might change throughout the day, and the horizontal flipping to switch the weapon grip.

#### 2.5.2.2 Training Settings

The SSD MobileNetV2 was implemented using Tensorflow. The available weights obtained with the pre-training of the model on the COCO dataset were exploited for inference. The YOLOv4-CSP network was implemented and trained in Keras, a Python library running on top of TensorFlow. To train the YOLOv4-CSP the fine tuning methodology was applied. Starting from the pre-trained weights on COCO dataset, the model was trained with stochastic gradient descent (SGD) for 300 epochs using an initial learning rate of 0.001 and a batch size of 16. The learning rate reduction on plateau policy was applied with a reduction factor of 0.5 after 10 epochs with no improvements on the validation loss. Early stopping was also applied, with training termination after 75 epochs with no improvements on the validation loss. The optimal combination of batch size, optimizer and initial learning rate was found after the tuning of each hyper-parameter through manual search. The best weights configuration among epochs was retrieved according to the lowest loss value achieved on the validation set. The training was performed on a GPU NVIDIA® GeForce RTX™ 3090.

**Table 2.7** Proposed ablation study.

| Name of the architecture | First-step | Second-step |
|---|---|---|
| SSD-MobileNetV2$^2$ | SSD-MobileNetV2 | SSD-MobileNetV2 |
| YOLOv4-CSP$^2$ | YOLOv4-CSP | YOLOv4-CSP |
| **Proposed** | **SSD-MobileNetV2** | **YOLOv4-CSP** |

#### 2.5.2.3 Ablation Study and State of the Art Comparison

Table 2.7 outlines the ablation studies conducted, including the DL models used in each step and the name of each approach evaluated. As a first ablation study, the use of SSD Mobilenetv2 in both steps of the proposed approach was investigated (i.e., SSD-MobileNetV2$^2$), to evaluate the impact on the detection performances. Since the work aims at developing an approach to maximize the speed-accuracy trade-off, also the use of YOLOv4-CSP in both steps was investigated (i.e., YOLOv4-CSP$^2$), mainly to evaluate its influence on the inference speed.

The proposed dual-step approach for handguns and knives detection was compared also with the state-of-the-art methods in [55, 41, 59] developed for weapons detection task, as well as with other popular object detectors.

Moreover, to point out the impact of using different image input sizes on detection performances, further comparison with state-of-the-art detectors with varying input sizes was performed. The rationale for such comparison lies on the fact that in general-purpose object detection the size of the input image can affect the performance. As a matter of fact, bigger input sizes often leads to more accurate but slower detection while smaller input sizes leads to faster but less accurate detection [70].

For a fair comparison, all the approaches were investigated using the same data split and were trained on the same computational hardware.

#### 2.5.2.4 Performance Metrics

To validate the proposed approach and compare it against the other state-of-the-art approaches, embracing the main literature in the field [69, 28], the detection performance was assessed using the standard COCO detection metrics as follows:

- AP as primary metric computed as the mean AP over the *knife* and *gun* classes and over 10 Intersection over Union (IoU) thresholds from 0.50 to 0.95 with a step size of 0.05 (0.50:0.95);
- AP50 as the AP computed at IoU 0.50, corresponding to the primary PASCAL VOC[11] metric, detailed in Sec. 2.4.2.2;

[11] http://host.robots.ox.ac.uk/pascal/VOC/

**Table 2.8** COCO standard evaluation metric and inference speed comparisons for the ablation study.

| Ablations | AP | APm | APs | AP50 | AP75 | $FPS_{nano}$ |
|---|---|---|---|---|---|---|
| SSD-MobileNetV2[2] | 21.20 | 26.50 | 19.40 | 56.80 | 8.90 | **13.60** |
| YOLOv4-CSP[2] | 79.30 | 50.10 | 49.30 | 99.60 | 93.90 | 2.80 |
| **Proposed** | **79.30** | **50.10** | **49.30** | **99.60** | **93.90** | 5.10 |

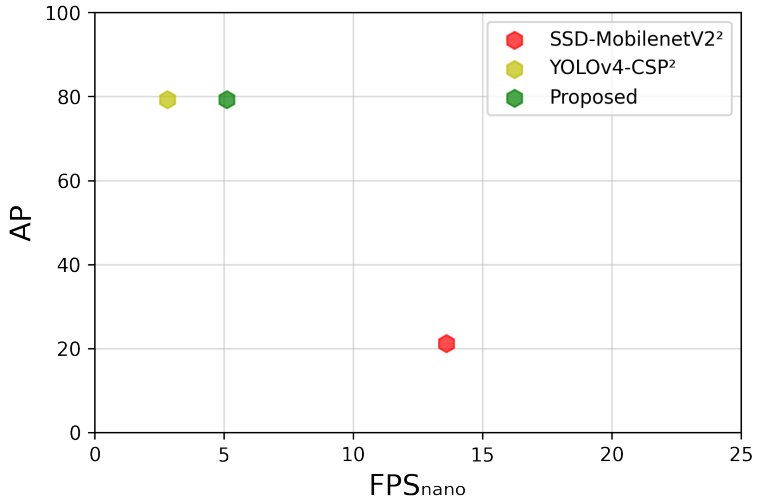



**Figure 2.7** Comparison of the speed-accuracy trade-off in terms of frame per second on the Jetson Nano ($FPS_{nano}$) and Average Precision (AP) for the ablation study.

- AP75 as a strict metric computed as the AP at IoU 0.75;
- APs and APm as the AP at IoU 0.50:0.95 for small (where object area $< 32^2$ pixels) and medium (where $32^2 <$ object area $< 96^2$ pixels) objects, respectively. The APl for large objects was not included since the weapons' related bounding-box area is always smaller than $96^2$ pixels in the WeaponSenseV1 dataset.

To further evaluate the presented approaches, efficiency metrics were also computed. Specifically, (i) billion floating point operations (GFLOPs) were computed when comparing the proposed approach with the others in the state of the art and (ii) inference speed on the Jetson Nano board ($FPS_{nano}$) in terms of FPS was computed for the ablation studies. Following the literature in closer fields [69, 71, 72, 73], both GFLOPs and FPS were plotted against AP. Additionally, to assess if significant differences exist among the approaches in the ablation studies, the one-way ANOVA (significance level = 0.05) with post hoc test was conducted. The considered population for each approach was the set of APs computed individually for each video in the test set.

**Table 2.9** COCO standard evaluation metric for comparisons between the proposed approach and other state-of-the-art architectures.

| Compared Approaches | AP | APm | APs | AP50 | AP75 | GFLOPs |
|---|---|---|---|---|---|---|
| Faster-RCNN-VGG16$_{640x640}$ [55] | 10.40 | 14.50 | 6.70 | 28.20 | 3.10 | 138.12 |
| Faster-RCNN-ResNet50-FPN$_{1280x720}$ [41] | 30.50 | 34.70 | 27.50 | 67.60 | 20.80 | 223.68 |
| Faster-RCNN-ResNet50-FPN$_{640x640}$ | 15.80 | 21.20 | 10.20 | 39.50 | 8.80 | 99.64 |
| Faster-RCNN-ResNet50-FPN$_{416x416}$ | 7.00 | 9.60 | 4.50 | 19.50 | 1.40 | 44.56 |
| SSD-MobileNetV2$_{416x416}$ | 9.80 | 17.50 | 2.90 | 26.40 | 5.10 | **1.18** |
| YOLOv4-CSP$_{416x416}$ | 9.00 | 11.50 | 1.10 | 32.00 | 1.90 | 25.17 |
| YOLOv5$_{416x416}$ [59] | 10.10 | 17.20 | 3.60 | 23.30 | 7.80 | 47.90 |
| **Proposed** | **79.30** | **50.10** | **49.30** | **99.60** | **93.90** | 26.35 |

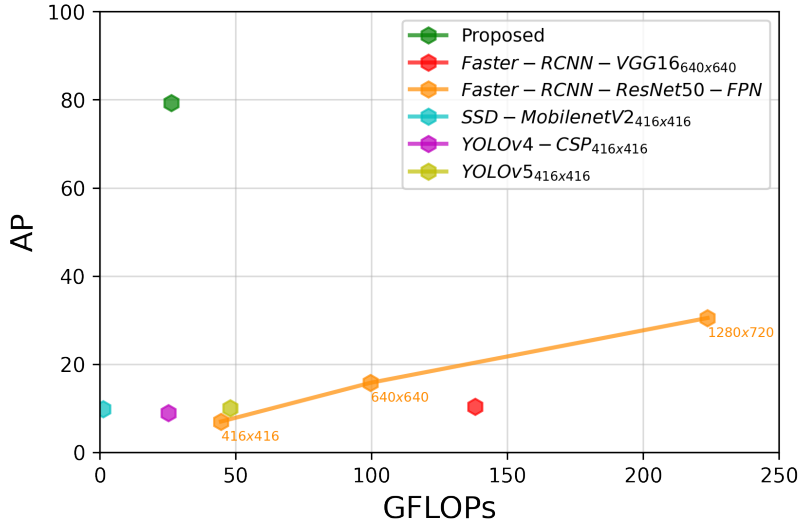



**Figure 2.8** Comparison of the complexity-accuracy trade-off in terms of billions floating point operations (GFLOPs) and Average Precision (AP) for the comparison against the state-of-the art approaches. The yellow values in the chart indicate the image input sizes for the Faster-RCNN-ResNet50-FPN architecture. The proposed approach outperforms the state-of-the-art weapon detectors while having fewer GFLOPs.

### 2.5.3 Results

Table 2.8 summarizes the performance comparison in terms of AP, APm, APs, AP50, AP75 and $FPS_{nano}$ of the approaches in the ablation study.

The proposed approach achieved the highest results in all the COCO metrics, with an AP = 79.30 averaged over all classes, as well as an AP50 = 99.60, which represents the PASCAL VOC traditional metric computed at a single IoU of 0.50. The YOLOv4-CSP[2] approach obtained the same results of the proposed one for all the COCO metrics, while it achieved the worst results in terms of inference speed ($FPS_{nano}$ = 2.80) on the Jetson Nano board. In contrast, with the use of the SSD-MobileNetV2[2] approach the inference speed reached the highest value ($FPS_{nano}$ = 13.60), but the AP dropped significantly (AP = 21.20 with 58.10 points drops), along with all the other COCO metrics. In particular, the SSD-MobileNetV2[2] approach resulted in very low performance when computed at IoU of 0.75 (AP75 = 8.90) with a drop of 85.00 points with respect to the proposed approach. Significant differences were found (p-value $< 0.05$) between the approaches in the ablation studies. The speed-accuracy trade-off of the proposed approach with respect to the ablations is shown in Fig. 2.7.

When compared with the other state-of-the-art single-step approaches, the proposed one obtained by far the best performances for all the COCO metrics (shown in Tab. 2.9). Moreover, the proposed approach required GFLOPs = 26.35, achieving the best results in terms of trade-off between complexity and detection performance (as pointed out in Fig. 2.8).

The approach in [55] (i.e., Faster-RCNN-$VGG16_{640\times640}$) achieved low values for all the metrics, and particularly for AP, APs and AP75, with 10.40, 6.70 and 3.10, respectively. The same holds for the approach in [59], with AP = 10.10, AP50 = 23.30 and AP75 = 7.80. The approach in [41] (i.e., Faster-RCNN-ResNet50-$FPN_{1280\times720}$) required the highest GFLOPs (i.e., 223.68), while obtained the nearest performance to the proposed approach with AP = 30.50, AP50 = 67.60 and AP75 = 20.80, yet showing consistent degradation in performance when the IoU threshold increases from 0.50 to 0.75. Moreover, decreasing the input size on the same architecture led to a significant reduction of all the metrics (AP = 15.80 and AP = 7.00 for Faster-RCNN-ResNet50-$FPN_{640\times640}$ and Faster-RCNN-ResNet50-$FPN_{416\times416}$, respectively).

In particular, when evaluating the approach in [41] using the same input size as the proposed approach (i.e., 416 × 416 pixels), the worst results were obtained in terms of AP, APm, AP50 and AP75 with values 7.00, 9.60, 19.50 and 1.40, respectively.

Both the architectures SSD-MobileNetV2 and YOLOv4-CSP in single-step settings (i.e., trained to directly detect the weapons from the original frames) obtained very low performance, with the worst value on small objects achieved

by YOLOv4-CSP (APs = 1.10). On the other hand, the SSD-MobileNetV2 required the smallest amount of GFLOPs (i.e., 1.18), pointing out the lightweight design of the model.

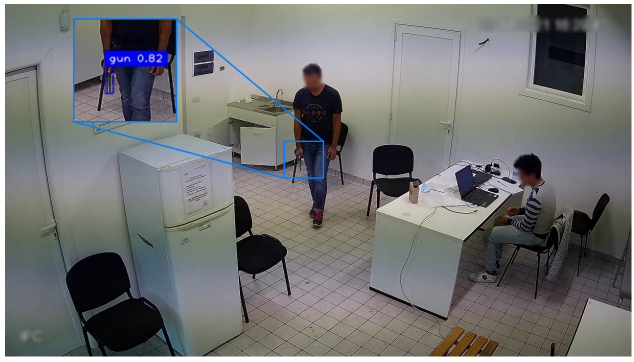
Qualitative results of the proposed approach are shown in Fig. 2.9. The samples include weapons from both classes (i.e., *knife* in Fig. 2.9(a) and *gun* in Figs. 2.9(b) and 2.9(c)).

### 2.5.4 Discussion

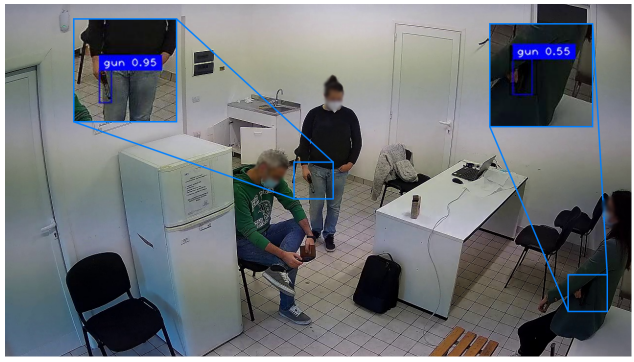
Automatic weapon detection from CCTV plays a crucial role in preventing crimes and enabling a prompt response by law enforcement agencies. Despite its relevance, the survey of the literature highlighted the lack of effective yet efficient approaches in coping the open challenges in the field, such as handling small-object sizes and achieving real-time responses [41] especially in on-the-edge settings. As a first step to solve such issues, the presented work addressed the challenging task of the on-the-edge indoor detection of handguns and knives while keeping near real-time performance.

The proposed double-step approach achieved satisfactory detection results with AP and AP50 equal to 79.30 and 99.60, respectively. The choice of YOLOv4-CSP as weapons detector in the second step allowed to obtain accurate detection with good localization capability, with marginal differences in AP for small and medium-sized objects. The impact of the YOLOv4-CSP as second-step detector is visible from the comparison with the SSD-MobileNetV2[2] approach. In the latter, the SSD-MobileNetV2 detector used in the second step was unable to achieve good localization at higher IoU and also suffered on small weapons detection (APs = 19.40), meaning that the feature extracted from the person's crop were not strong enough to localize challenging objects (e.g., very thin objects, objects with low background contrast). On the other hand, the SSD-MobileNetV2[2] approach achieved the best inference speed thanks to the higher lightness of the SSD-MobileNetV2 with respect to YOLOv4-CSP. Neverthless, the proposed approach still achieved the best speed/accuracy trade-off among the approaches in the ablation study. Its accuracy is also evidenced by the qualitative results, with high confidence in localizing and predicting each correct weapon class. Also, in extremely challenging scenarios (i.e., in Fig. 2.9(c), the vaguely visible *gun* on the right side) the proposed approach localized the weapon, even if with lower confidence compared to other detections. The low confidence in such situations could be attributed to the detection hardness resulting from the low weapon/background contrast. In comparison with YOLOv4-CSP[2], while there is no difference in AP due to the simplicity of the people detection task for both YOLOv4-CSP and SSD-MobileNetV2 models (i.e., in the first step all the people were correctly


(a)




(b)




(c)

**Figure 2.9** Samples of qualitative results. For the sake of clarity, each object detected has been zoomed in to point out both the predicted bounding box and the related classification score. Predicted *gun* and *knife* bounding boxes are highlighted in blue and red, respectively.

identified in the frames), the speedup in the proposed approach is given by the use of the lighter model in the first step. When compared with the state-of-the-art approaches, the proposed one achieved the highest performance. The low performances of [55] could be related to the hardness in the localization of handheld weapons whose size is very small compared to the frame size. In support of such a consideration, the worst metrics of [55] were the APs and the AP75. As regards [41], despite the addition of the FPN module slightly increased the detection ability on the small objects, the low performance paired with the high GFLOPs does not allow the use of the approach in the actual on-the-edge practice. Furthermore, reducing the input size makes the achieved result even worse. The state-of-the-art detectors (i.e., SSD-MobileNetV2 and YOLOv4-CSP) were evaluated at the same input size of the proposed approach and despite the small GFLOPs values highlight the small complexity of the approaches, they obtained very low performance. The poor results may be attributed to the small-sized weapons in the images, which almost disappear when the original frame size (i.e., 1280×720) is resized to match the detectors' input size (i.e., 416×416). In the proposed approach, thanks to the prior focus on the people, the size of the weapon with respect to the camera FoV does not affect the detection performances so heavily.

A limitation of the proposed approach lies in the dependence of its speed on the number of people in the FoV at the same time (i.e., the second step of the approach process an image for each detected person). However, it still ensures near real-time performance in non-crowded environments (e.g., home surveillance systems).

This work, by proposing and validating an approach both effective and efficiently executable on edge devices, represents a step in the incremental progress towards addressing some of the crucial challenges in weapon detection. Although there is considerable room for improvement, the developed methodology demonstrated the feasibility of effective weapon detection in resource-limited settings, enabling to leverage the potential of edge computing with artificial intelligence.

## 2.6 Edge AI and Super-Resolution for Enhanced Weapon Detection in Video Surveillance

The need to develop edge-oriented methods in weapon detection has been partially addressed by the research presented in Sec. 2.5, proposing a deep learning approach for weapon recognition that is effective and at the same time executable in near real time on low-power edge devices. Despite its improvements to the state of the art in weapon detection, the speed of the proposed approach being dependent on the number of people present in the camera FoV limits its applicability to non-crowded contexts. In this sense, there is still much work to be done to effectively solve the most relevant problems in detecting weapons: (i) the low detection accuracy due to the small size of the weapons with respect to the camera FoV, and (ii) the need to perform real-time detection. As already discussed in Sec. 2.2, to solve the small-object challenge, a popular method in many fields is the use of SR. Naive approaches, as in [74], apply SR to enlarge images or image regions, then sequentially apply detection methods. In more recent approaches, like [75] and [76], the potential of Generative Adversarial Networks (GAN) for SR is exploited to jointly super-resolve and detect objects. Despite the improvement in detecting small-sized objects, the major limitation of these approaches when edge-oriented solutions are necessary is the significant complexity introduced. Indeed, beside using an additional model, having enlarged the image, the number of GFLOPs significantly increases for the same model used, as shown in Tab. 2.9. Due to these limitations, the solutions are impractical in an edge context and when real-time feedback is needed. Nevertheless, the basic idea of integrating SR techniques brings significant advantages for the problem of small object detection. Therefore, the operational goal of this research is to leverage these advantages while minimizing the drawbacks, to overcome the limitations of the previous approach, discussed in Sec. 2.5.4. This would guarantee the applicability, in any video surveillance context, of models on edge devices in real time without dependence on the number of people present in the camera FoV, allowing for further improvement of the state of the art, approaching the research goal of this thesis.

To this end, inspired by [77], this work proposes YOLOSR, an architecture which exploits a SR branch trained in conjunction with a lightweight weapon detector, to enhance the detection performance especially on small-sized objects. The SR branch is then discarded in the inference phase, so as to improve weapon detection accuracy without impact on the computational complexity of the detection architecture. The approach was validated on the WeaponSenseV2 dataset, described in Sec. 2.3.

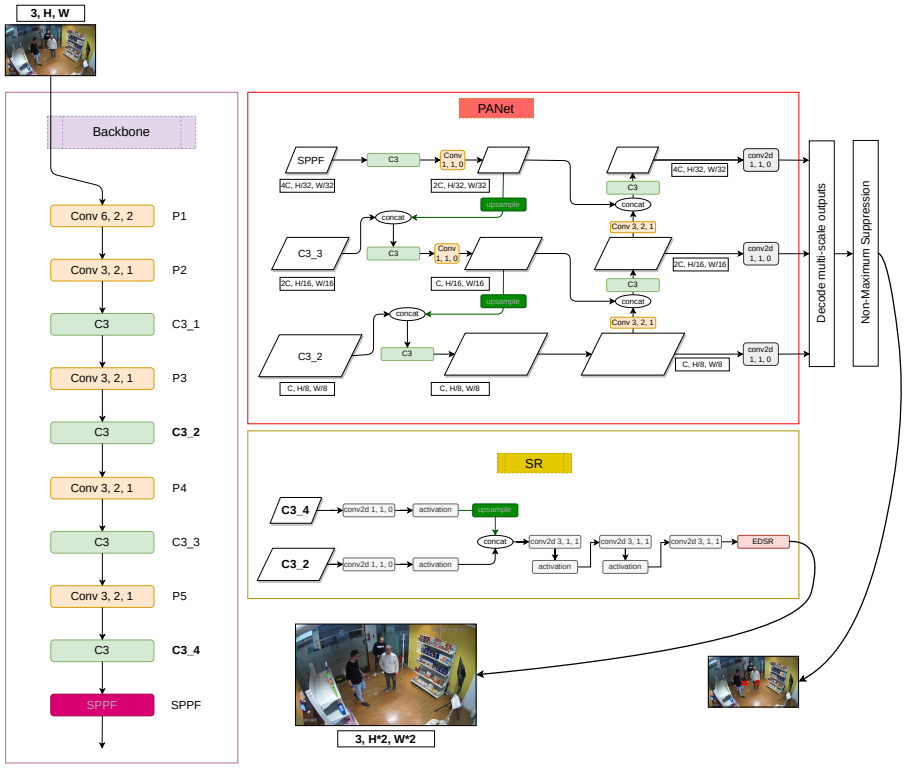



**Figure 2.10** Architectural view of the proposed YOLOSR, comprising the baseline detector YOLOv5-small (Backbone + PANet) and the SR branch (SR).

## 2.6.1 YOLOSR Architecture

The architecture of the YOLOSR is detailed in Fig. 2.10. The two main components of the architecture are the baseline YOLOv5-small (YOLOv5s) detector and the SR branch. This section first reviews the most popular YOLOv5s for weapon detection, then presents the proposed SR branch in detail.

### Baseline YOLOv5s

The YOLOv5 object detection model comprises several variants (i.e., nano, small, medium, large, extra large), each designed to balance differently the trade-off between speed, size, and accuracy. The YOLOv5s is the small variant of the YOLOv5, which enhance speed and efficiency over accuracy. Despite being designed to be suitable for real-time applications on resource-constrained devices, it still maintains a reasonable level of accuracy. The YOLOv5s consists of three main components: the backbone, the neck and the head. The backbone, which processes the input image extracting low- and high-level semantic features, is based on the CSPDarknet53, presented in Sec. 2.5.1.1. The

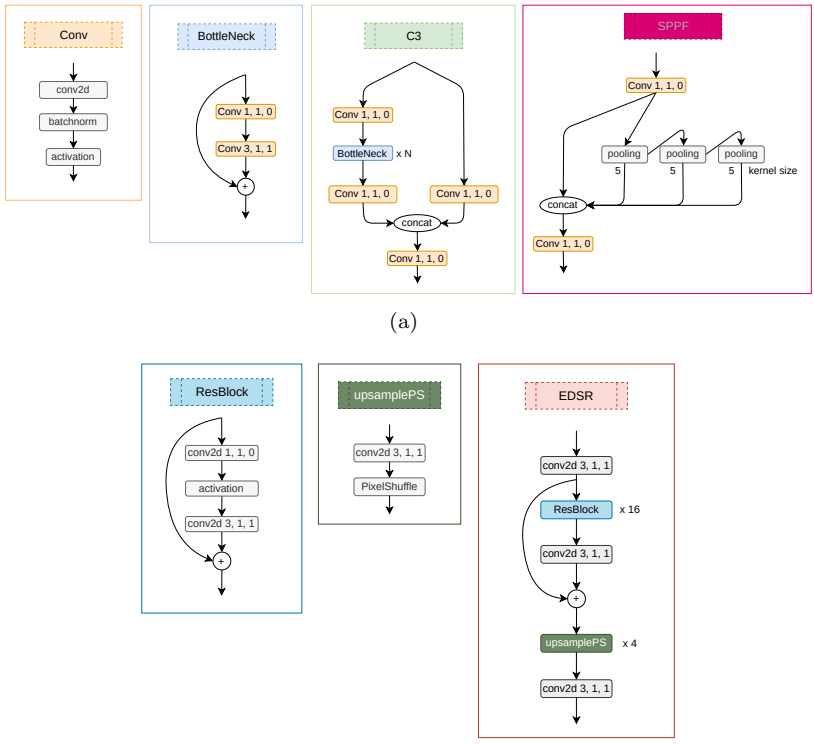



**Figure 2.11** (a) Submodules of the the baseline YOLOv5-small and (b) submodules of the SR branch are shown.

CSP design applies to each stage of the Darknet53 and splits the feature map into two parts, processes them separately and then merges them again (i.e., C3 stages shown in Fig. 2.11(a)), enhancing learning efficiency and reducing computational cost. For the sake of efficiency, in the YOLOv5s the structure of the Darknet53 is reduced in depth with respect to the other YOLOv5 versions, with 1,2,3,1 bottleneck blocks in each CSP stage, respectively. An initial 6×6 convolution with padding and stride 2 increases the channel depth while reducing spatial dimensions, enabling the network to capture finer details from the beginning. Then, after each CSP stage the backbone halves the spatial dimensions, resulting in a total network stride of 32. At the end of the backbone YOLOv5s uses SPP Fast (SPPF) an improved version of the SPP module, which pools the feature map at different scales and concatenates them to maintain spatial information. This design makes the model robust to variations in object size and shape.

The neck of YOLOv5s, which maintains the CSP-sized version of C3 stages introduced in [65], aggregates features from three different scales of the backbone, using Path Aggregation Network (PANet) to enhance feature-level communication between different scales. PANet connects the top-down and bottom-up pathways in a bidirectional manner, which improves the propagation of low-level features and helps the model in detecting smaller objects.

The head consists of three detection layers, each responsible for detecting objects at a different scale (large, medium, small). For each scale, a final 1×1 convolutional layer predicts, for each portion of the feature map, three bounding boxes in terms of: (i) bounding box coordinates (x, y, width, height), (ii) an objectness score indicating the likelihood of an object's presence, and (iii) class scores for each class the model is trained on. Thus, for weapon detection on the WeaponSenseV2, each detection layer was implemented having 21 filters so as to predict, for three bounding boxes: four bounding box coordinates, an objectness score and two classes (i.e., *Knife*, *Handgun*). The selection of the most accurate bounding boxes among the candidates generated by the YOLOv5s model was refined using the non-maximum suppression algorithm.

#### Super Resolution Branch

The design of the SR branch was inspired by [77], and can be conceptualized as an encoder-decoder architecture. From this perspective, the first part of the encoder is shared with the detection architecture, and is responsible for extracting features relevant to both the SR and weapon detection tasks. The second part of the encoder, related only to the SR branch, uses the low- and high-level features coming from the second and the fourth C3 stages of the shared backbone, respectively. First, after applying of a 1×1 convolution to refine both low- and high-level features, the encoder upsamples the high-level

**Table 2.10** WeaponSenseV2 composition, pointing out (i) the number of frames and number of video sequences containing the *Handgun* class, the *Knife*, or both (i.e., mixed); (ii) the total number of labeled instances for each class.

| | **n° frames** | **n° videos** | **n° instances** |
|---|---|---|---|
| *Handgun* | 1421 | 30 | 1798 |
| *Knife* | 1266 | 23 | 1541 |
| mixed (*Handgun* & *Knife*) | 275 | 3 | - |
| total | 2952 | 56 | 3339 |

feature to match the dimension of the low-level feature and concatenates them. Then, three consecutive 3×3 convolutions process the features, merging spatial and semantic information. The structure of the decoder, shown in Fig. 2.11(b) is based on the Enhanced Deep Super Resolution (EDSR) network [78]. To reconstruct the high resolution (HR) image, the decoder starts with a head consisting of a 3×3 convolution. Following this, the body of the decoder, made up of 16 residual blocks and another 3×3 convolution, processed the features. The output from the body is then added to the head's output, via residual connection. Then, the tail of the decoder employs the PixelShuffle [79] operation, which implements efficient sub-pixel convolutions, applied four times to upsample the features. This process achieves a total upsampling of x16 on the decoder's input features, thus reconstructing a x2 HR image from the low-resolution (LR) original image given as YOLOSR architecture's input. The process concludes with a final 3×3 convolution with three filters to produce the RGB HR output image.

## 2.6.2 Experimental Protocol

### 2.6.2.1 Dataset Preparation

The WeaponSenseV2, described in 2.3, was used for validating the YOLOSR. This dataset builds upon the 2425 frames from WeaponSenseV1, augmented with an additional 527 frames, possibly containing multiple labeled instances. These new frames resulted from the annotation of the *Handgun* and *Knife* classes and the sampling one frame in every two from the new videos, following 2.4.2.1. All the frames have a size of 1280×720 pixels.

Table 2.10 summarizes the dataset-related details, including the number of frames and videos containing the *Handgun* class, the *Knife*, or both, as well as the number of labeled instances for each class.

### 2.6.2.2 Training Settings and Performance Metrics

The YOLOSR was implemented in PyTorch and the experiments in this study run on a NVIDIA® RTX™ A6000 GPU. Table 2.11 presents the data splitting

**Table 2.11** Number of videos in the train, validation, and test splits for each class, including the mixed videos, containing both classes' instances. The total number of frames in each set is given in round brackets, computed by summing the labeled frames from all videos in the respective set

| | **Train** | **Validation** | **Test** |
|---|---|---|---|
| *Handgun* videos (frames) | 24 (1030) | 2 (103) | 4 (174) |
| *Knife* videos (frames) | 17 (870) | 3 (221) | 3 (150) |
| mixed (*Handgun* & *Knife*) videos (frames) | 2 (264) | - | 1 (140) |
| total videos (frames) | 43 (2164) | 5 (324) | 8 (464) |

methodology used for the WeaponSenseV2. To reduce potential bias, the split was conducted at the video level, ensuring that frames from a single video were not distributed across the training, validation, and test sets. Following this approach, 73.3%, 11.0%, and 15.7% of the total frames were allocated for training, validation, and testing purposes, respectively. To accomplish the joint training of the weapon detector and the SR branch, the 1280×720 frames were used as HR ground truth for the SR branch, and were downsampled to 640×360 during the training and used as input for the YOLOSR. Consistently, the validation and testing of the YOLOSR, discarding the SR branch, was always performed on the downsampled 640×360 frames.

The overall loss of the YOLOSR was a weighted sum of the detection loss and the SR reconstruction loss, defined in Eq. 2.5.

$$Loss = \alpha L_w + \beta L_{sr} \tag{2.5}$$

The detection loss $L_w$ was the standard YOLO loss [51], defined in Eq. 2.6, where $\lambda_{box}$, $\lambda_{obj}$ and $\lambda_{cls}$ are the weights for the box, objectness, and classification loss, respectively.

$$L_w = \lambda_{box} \cdot L_{box} + \lambda_{obj} \cdot L_{box} + \lambda_{cls} \cdot L_{cls} \tag{2.6}$$

The SR reconstruction loss was the L1 loss, computed between the SR branch prediction $\hat{Y}$, and the ground truth HR image $Y$, as in Eq. 2.7.

$$L_{sr} = \left\| \hat{Y} - Y \right\|_1 \tag{2.7}$$

To train the YOLOSR, fine-tuning strategy was applied. The weapon detector was initialized with YOLOv5s pre-trained weights from the COCO dataset, while for the SR branch the Kaiming initialization was used. The model was trained using SGD optimizer with momentum of 0.9 and weight decay of 0.0005 for 300 epochs using an initial learning rate of 0.05 and a batch size of 32. The learning rate linear-reduction policy was applied with a final learning rate of 0.0005. Early stopping was also applied, with training termination after

100 epochs with no improvements on the validation loss. The optimal combination of batch size, optimizer and initial learning rate was found after the tuning of each hyper-parameter through manual search. The best weights configuration among epochs was retrieved according to the lowest detection loss value achieved on the validation set. To increase data variability, minimizing overfitting risks and improving model's generalization ability, online data augmentation strategies were implemented on the training dataset. The randomly applied data-augmentation transformations were: Hue Saturation Value (HSV) adjustments, image translation and scaling, left-right flipping, and mosaic augmentation.

Following [77], the performance of the YOLOSR was evaluated using the AP50, as the AP defined in Eq. 2.1, computed using an IoU of 0.5. This metric was chosen as the primary measure for assessing and comparing the detection accuracy against the other methods. The global AP was calculated as the mean of the AP values for both the *Handgun* and *Knife* classes. To evaluate the YOLOSR in terms of efficiency against the other approaches, inference speed in FPS on the Jetson Nano board was computed, so as to validate the method on edge devices. In addition, to measure and compare the computational complexity of each method, the GFLOPs were computed.

#### 2.6.2.3 Ablation Studies

With the aim of evaluating the effectiveness of the proposed method in increasing the weapon detection accuracy without adding complexity, a series of comparisons with other architectures and ablation experiments were conducted.

As a first ablation study, several baseline architectures having different complexities were compared on the weapon detection task. The baseline YOLOv5s, used in YOLOSR, was compared against the YOLOv5-medium (YOLOv5m), YOLOv5-large (YOLOv5l), YOLOv4-CSP and YOLOv3.

The second ablation study then evaluated the integration of the SR branch into the most promising baseline architectures identified in the first ablation, selected according to their effectiveness in balancing complexity and detection performance.

In the final ablation study, the impact of the SR branch design on the weapon detection performance was investigated. To this end, a different version of the SR branch, named SR-early, was designed and integrated into the most promising baseline architectures of the first ablation. In this SR version, the shared backbone between the weapon detector and the SR branch was shallower. Specifically, the SR-early branch utilized the low- and high-level features coming from the first and the third C3 stages of the backbone (i.e., C3_1 and C3_2 in Fig. 2.10). In accordance, within the SR-early branch was modified the upscaling process, reducing the number of PixelShuffle operations from four to

three, so as to generate the HR image of the correct size (i.e., doubled with respect to the LR input image).

### 2.6.3 Results

**Table 2.12** Comparisons of the baseline architectures in terms of FPS on Jetson Nano and GFLOPs, and AP50 assessed for handgun, knife and as a mean between the two classes (All).

| Model | AP50 | | | GFLOPs | FPS |
|---|---|---|---|---|---|
| | Handgun | Knife | All | | |
| YOLOv5s | 55.90 | 23.50 | 39.70 | **15.90** | **15.4** |
| YOLOv5m | 59.50 | 25.40 | 42.40 | 48.00 | 5.9 |
| YOLOv5l | 67.80 | 41.20 | **54.50** | 107.90 | 3.2 |
| YOLOv4-CSP | 50.10 | 18.60 | 34.40 | 53.10 | 4.7 |
| YOLOv3 | 66.70 | 38.60 | 52.70 | 154.90 | 2.4 |

The comparative results for different baseline architectures in terms of complexity, efficiency (measured in GFLOPs and inference speed in FPS on Jetson Nano, respectively) and predictive accuracy (indicated by AP50) computed for the handgun, knife and as a mean of the two classes, are presented in Table 2.12. The evaluated architectures include YOLOv5s, YOLOv5m, YOLOv5l, YOLOv4CSP, and YOLOv3. A specific focus on the GFLOPs and FPS of each architecture reveals the following order of increasing complexity and decreasing efficiency: YOLOv5s with 15.90 GFLOPs running at 15.4 FPS, YOLOv5m with 48.00 GFLOPs running at 5.9 FPS, YOLOv4-CSP with 53.10 GFLOPs running at 4.7 FPS, YOLOv5l with 107.90 GFLOPs running at 3.2 FPS, and YOLOv3 with 154.90 GFLOPs running at 2.4 FPS.

When it comes to predictive accuracy, YOLOv5l leads with an AP50 computed as a mean of the two classes of 54.50, followed by YOLOv3 (52.7), YOLOv5m (42.40), YOLOv5s (39.70), and YOLOv4CSP (34.40). Notably, for each of these networks, the AP50 values were higher for handguns compared to knives. This trend suggests a more effective detection performance for handguns across all evaluated models.

**Table 2.13** Models' comparisons with SR branch in terms of AP50 assessed for handgun, knife and as a mean between the two classes (All).

| Model | AP50 | | |
|---|---|---|---|
| | Handgun | Knife | All |
| YOLOSR (proposed) | 68.80 | 34.20 | **51.50** |
| YOLOv5mSR | 65.90 | 32.90 | 49.40 |
| YOLOv5lSR | 64.30 | 35.00 | 49.70 |

The search for an architecture that balances both effectiveness and efficiency led to the exclusion of YOLOv3 and YOLOv4-CSP for further tests on the

**Table 2.14** Models' comparisons with SR-early branch which uses the low and high-level features from the first and the third C3 stages of the backbone. Performance are assessed in terms of AP50 for handgun, knife and as a mean between the two classes (All).

| Model | AP50 | | |
|---|---|---|---|
| | **Handgun** | **Knife** | **All** |
| YOLOSR-early | 60.40 | 32.50 | 46.50 |
| YOLOv5mSR-early | 62.60 | 35.80 | **49.20** |
| YOLOv5lSR-early | 56.80 | 35.00 | 45.90 |

SR branch. Indeed, YOLOv3, besides being the most complex and inefficient with 154.90 GFLOPs and 2.4 FPS, also fell short in terms of effectiveness, not achieving the highest AP50 score. On the other hand, YOLOv4-CSP was the least effective, with an AP50 of 34.40, and ranked third in complexity and efficiency. The focus thus shifted to the remaining architectures – YOLOv5s, YOLOv5m, and YOLOv5l among which YOLOv5l stood out for its highest predictive accuracy (AP50=54.50) and YOLOv5s for its low complexity (15.90 GFLOPs) and superior efficiency, running at 15.4 FPS on the Jetson Nano.

In Table 2.13, the focus shifts to the three main baselines of interest namely YOLOv5s, YOLOv5m, and YOLOv5l, each integrated with the SR branch. The results demonstrate that the proposed architecture, YOLOv5s with the SR branch (referred to as YOLOSR), achieved the highest performance, recording an AP50 of 51.50 across all classes. The integration of the SR branch into the YOLOv5m baseline, resulting in YOLOv5mSR, led to an AP50 of 49.50 for both classes. Meanwhile, the YOLOv5lSR, which is the YOLOv5l baseline with the SR branch, showed slightly better results with an AP50 of 49.70 for both classes. Interestingly, the addition of the SR branch led to a decrease in AP50 for the less computationally efficient YOLOv5l baseline. However, for the other two baselines, YOLOv5s and YOLOv5m, there was a noticeable increase in performance. In particular, the proposed architecture experienced the most remarkable enhancement, with an increase of 12.8 percentage points in handgun detection and 10.7 percentage points in knife detection.

The final result, presented in Table 2.14, involves an ablation study to determine the most effective placement of the SR branch within the architecture. The results from this ablation study indicated a decline in performance for all baselines when the SR branch's position was altered as described in Sec.2.6.2.3. The most significant negative impact was observed in the YOLOv5l baseline with the early integration of the SR branch (referred to as YOLOv5lSR-early), which recorded an average AP50 of 45.90 across the two classes. This was followed by both YOLOv5s baseline (YOLOSR-early) and YOLOv5m baseline (YOLOv5mSR-early), registering an average AP50 across the two classes of 46.50 and 49.20, respectively. It is worth noting that the integration of both

the SR and SR-early branches only affects accuracy, while it does not influence the speed nor the complexity during inference.

### 2.6.4 Discussion

In the context of video surveillance, real-time recognition of weapons on low-cost devices is an open problem. If effectively solved, it would ensure the global spread of such technology, leading to a significant increase in public safety in many contexts. To this end, in this study was presented the YOLOSR, an approach that leverages the integration of SR technologies, enhancing performance in weapon recognition without the drawback of increased computational complexity. The proposed architecture was built on a baseline weapon detector and a SR branch used only during training. Among the various baselines validated, the YOLOv5s used stood out for its low complexity, making it perfect for edge computing contexts. Although, by design, this baseline prioritizes speed over accuracy, the integration of the SR branch into the YOLOv5s helped in enhancing the resolution of the feature maps within the model's shared backbone. Thus, the structures of the objects become clearer and more defined, which turned out to be extremely beneficial for detecting smaller objects, otherwise difficult to identify at lower resolutions. Since the SR branch was discarded during inference, the YOLOSR maintained same GFLOPs of the baseline YOLOv5s but with a substantial increment of 11.80 points in AP across the two classes. The benefits brought by the integration of the SR branch make YOLOSR the fastest architecture with the best results in AP among the presented ones, thus showing the best balance between speed and accuracy. On the contrary, an interesting finding regards the YOLOv5l baseline, in which the shared backbone becomes too deep for the SR task, leading to poor results. This, in a multi-task context, negatively impacts the detection performance. Diving deeper into the details, to comprehend why the SR-early branch led to worse results compared to SR branch, a premise needs to be made. Generally, due to the tiny size of small objects, details related to these objects are gradually lost in the high-level feature maps (i.e., feature maps processed by the deeper layers of the network). In fact, the most relevant information about small objects is extracted from the shallow layers, which is why architectures like YOLO use a multi-scale approach, also combining information from low-level feature maps to achieve detection on smaller-sized objects. In light of these premises, although the negative influence of the SR task still led to worse results than the YOLOv5l baseline, an hypothesis on why the SR branch performed better than the SR-early branch can be made. Since the negative impact of the SR-early branch affected the shared backbone's earlier layers more, while the SR branch had a greater impact on the deeper

layers, YOLOv5lSR was able to achieve better detection performance because its earlier layers were able to extract more relevant information compared to the earlier layers of YOLOv5lSR-early. This hypothesis is also supported by the fact that, moving from YOLOv5lSR-early to YOLOv5lSR, the highest increase in AP was related to the *Knife* class (i.e., +7.8 points), whose objects are even smaller than those in the *Handgun* class. Nevertheless, additional investigation on this aspects is needed. Although the solution proposed in this study makes it suitable for edge deployment and scenarios requiring real-time feedback, its accuracy performance, compared with the study presented in Sec. 2.5, underscores the importance of continued research in this domain.

## 2.7 Conclusion and Future Perspective

The increasing prevalence of surveillance recordings in a variety of both public and private settings, such as homes, offices, and educational facilities, underscores the challenge of handling large data volumes. Moreover, the demanding task of human supervision, particularly in the context of round-the-clock surveillance, together with the complexities of data storage, emphasizes the urgency for designing solutions in this domain.

Answering to these challenges, the development of automated algorithms for surveillance video analysis has gained momentum, transitioning from traditional computer vision techniques to more advanced artificial intelligence methods, particularly DL. These DL algorithms have innovated the field by significantly reducing the human burden and enabling the storage of processed, high-level information instead of vast volumes of raw footage. However, implementing these algorithms in video surveillance is not without its challenges, notably the requirement for real-time processing, and the need for cost-effective and efficient hardware are still topical concerns.

To address these issues edge AI, which integrates artificial intelligence with the IoT and edge computing, emerges as a promising approach. This integration enables the shifting of processing tasks from remote servers to local devices, significantly cutting down communication expenses, enabling real-time data analysis, and providing enhanced security in data handling compared to cloud-based systems. Particularly in video surveillance, edge AI is aimed at accelerating data processing speeds, minimizing latency, and boosting overall data management efficiency. The research detailed in this chapter is aligned with these objectives, delving into the application of edge AI in video surveillance systems to effectively tackle the aforementioned challenges.

Specifically, a first research effort in this Chapter dealt with a multi-camera video surveillance infrastructure designed to be cost-effective and scalable which adopted cutting-edge DL techniques for object detection over multiple video

streams. The work focused on establishing a network of camera sensors, efficiently managing data from multiple sources and applying DL models capable of real-time, resource-efficient detection tasks. This method represented a considerable improvement in the development of surveillance systems, aiming to reduce resource intensity and enhance their capability to manage the complexities inherent in simultaneous security loads. Furthermore, it laid the foundation for ongoing advancements, primarily in the proposed DL methodologies which aligned with the edge AI paradigm.

Following this research, a thorough investigation was undertaken to determine the most optimal SBC system for computation. This exploration was performed using the first version of the custom-built WeaponSense dataset (i.e., WeaponSenseV0) and involved a comparative analysis between the Google Coral Dev and the NVIDIA® Jetson Nano. The comparison is based on the deployment of a CNN aimed at detecting weapons yielded by a subject in an indoor scenarios. The outcome of this comparison revealed that the NVIDIA® Jetson Nano board outperformed its counterpart, leading to its consistent use in subsequent studies for the development of algorithms that ensure both efficiency and effectiveness.

Following this outcome, subsequent research focused on the development of a system for identifying handgun and knives from the WeaponSenseV1. This focus addresses a significant challenge that remains a topic of keen interest in the literature: the detection of small objects. To tackle the challenge, this research deployed two consecutive CNNs on the NVIDIA® Jetson Nano: the first is tasked with identifying the person, while the second CNN is dedicated to recognizing the handheld weapon. This dual-network strategy represents a targeted and methodical approach to weapon detection in surveillance scenarios however, a major limitation such as poor efficiency in densely populated settings stimulated the latest proposed work. Thus, last efforts focused on the use of a single detector, with a specific emphasis on improving its efficacy during training via a SR module. This module was designed to be detachable from the architecture being evaluated, ensuring that it did not introduce additional computational costs when the CNN is deployed on the NVIDIA® Jetson Nano. This enhancement, tested on the WeaponSenseV2, while significantly optimizing the detector's performance without adding processing demands, underscored the vast potential for further innovation, especially from a methodological standpoint. With such a view, future research directions will include the improvement – both in terms of expansion and diversification – of the WeaponSense dataset, with an emphasis on collecting and annotating data from outdoor scenarios. Another area of interest will be the implementation of tracking modules, as in [80], which could provide insights into an individual's intentions by analyzing their movements once a weapon is detected. This would

add a predictive element to the system. Furthermore, integrating video-based data with information from different sensing devices, such as passive infrared [81] will be useful to enhance the overall reliability of the system.

# Chapter 3

# Advancing Preterm Infants' Movement Monitoring with Edge AI: Bridging the Technological Gap

## 3.1 Monitoring Preterm Infants through Sustainable Vision Systems: Challenge and Perspectives

The World Health Organization estimates that each year, over one in ten infants is born prematurely (before the 37th week of pregnancy)[1]. Although there have been significant advancements in survival rates, premature birth still has a deep impact on the neurodevelopment of infants. Common and enduring effects of preterm birth include delayed language development, cognitive deficits, and behavioral and motor disorders [82].

The timely identification of signs of atypical neuronal development is crucial for enabling clinicians to intervene. This early intervention is vital for enhancing infants' brain plasticity and facilitating damage compensation and recovery procedures [83]. With such a view, monitoring limbs' movement of preterm infants in neonatal intensive care units (NICUs) may provide valuable insights on their neuromotor development [84]. Nevertheless, despite its clinical significance, the movements' assessment often remains qualitative and reliant on visual observation by trained clinicians in NICUs [85].

Close, quantitative and non-intrusive monitoring is essential to examine infants' spontaneous motility and responses to various stimuli and interventions [86]. In this context, clinical decision support systems based on vision sensors have emerged as a promising tool for non-invasive and quantitative assessment of preterm infants [87]. These systems, as exemplified by studies such as [88, 89, 90], integrate artificial intelligence – and specifically DL – algorithms

[1] https://www.who.int/news-room/fact-sheets/detail/preterm-birth

for data analysis, and have the potential to provide meaningful insights into infants' movements, gestures, and postures. Nevertheless, notwithstanding their undeniable clinical relevance, it is crucial to recognize that the implementation of these tools needs the employment of either centralized or cloud-based server infrastructure [87, 91]. This architectural choice introduces multifarious challenges. In the case of cloud-based systems, these challenges primarily revolve around issues related to privacy, security, and potential reliance on a consistent internet connection [92]. Conversely, centralized infrastructures, lacking in scalability, pose different issues, notably in terms of computational costs, which, while advantageous for data processing, may also give rise to significant environmental concerns due to increased energy consumption [93]. This can consequently affect affordability through both higher operational expenses and potential sustainability concerns [94].

In the field of artificial intelligence, high operational costs have long posed a significant barrier to innovation and practical deployment, as highlighted in [95]. This challenge is particularly pronounced in the healthcare sector, where the development of costly systems raises substantial concerns related to the ethical principle of distributive justice in technology, as discussed in [96]. However, a transformative change is forthcoming with the advent of edge AI. This computational paradigm empowers real-time data processing and decision-making, thereby alleviating the strain on centralized cloud resources and effectively overcoming the economic barriers that have long been associated with artificial-intelligence implementation [97]. Moreover, the adoption of edge AI may also align with the emerging concept of GreenAI [98]. By significantly reducing energy consumption and the need for extensive cloud server infrastructures, edge-AI-based systems may contribute to a more sustainable and environmentally friendly approach to AI deployment, especially in healthcare settings [99].

Following these considerations, the proposed contribution presents a DL-based approach tailored to the edge computing paradigm, with a specific focus on segmenting preterm infants' limbs from depth images collected in the NICU of the *G. Salesi* Hospital in Ancona, Italy. This research addresses the critical need for efficient limb segmentation algorithms as a prior for infants' movement monitoring, while emphasizing sustainability in accordance with contemporary GreenAI principles [98]. Drawing inspiration from the eco-conscious guidelines outlined by Schwartz et al. [98], the work develops a sustainability-oriented methodology, encompassing both environmental and economic considerations. The approach explores strategies to minimize the computational resources required for algorithmic computation, contributing to a more environmentally friendly and cost-effective solution. As a concrete realization of this strategy, the selected edge device for deploying the proposed algorithms is the NVIDIA®

Jetson Nano [100]. This transition towards greener and more efficient AI implementations holds the potential to reduce environmental impacts while advancing crucial applications such as the quantitative assessment of preterm infants' movements in NICUs.

## 3.2 Related Work

In recent years, the field of neonatal care has witnessed a significant surge in the utilization of vision-based methods for monitoring preterm infants' movement. These methods, predominantly employing RGB and RGB-depth (RGB-D) cameras, have undergone a remarkable evolution from implementing traditional computer vision algorithms to embracing advanced DL techniques.

Initially, these systems relied on classical computer vision approaches, such as thresholding and morphological operations. However, the landscape shifted markedly with the advent of DL, which offered enhanced capabilities in analyzing complex motion patterns. Pioneering this transition, McCay et al. [101] employed OpenPose, a CNN initially devised for adult pose estimation, to assess infant's movements. They further refined this approach by developing a secondary CNN model to evaluate the quality of general movements in preterm infants, extracting insights from joint orientation and displacement histograms derived from RGB video data. In Moro et al., [102], developed a system to analyze and categorize the motion patterns of infants from 2D video recordings, focusing on identifying unusual movements. Their approach combines computer vision and machine learning techniques. The process involves three main steps: first, identifying key body points on infants using a deep learning-based detector; second, deriving quantitative measures from the movement trajectories of these points; and third, applying machine-learning-based classifiers like SVM, to distinguish between normal and abnormal motion patterns. Similarly, the works of Reich et al. [103], Sakkos et al. [104], and Schmidt et al. [105], Moccia et al. [88] also leveraged the infant's joints detection as a prior for limbs' movement monitoring. Their methodologies varied, ranging from employing shallow artificial networks to integrating long short-term memory (LSTM) networks, all aimed at improving movement's assessment. While these surveyed approaches have achieved good results, they fall short in aligning with the principles of energy efficiency and economic sustainability. Indeed, these systems, especially those reliant on advanced DL methods, require significant computational resources and energy to run. This not only heightens the economic impact but also challenges their integration in environments with limited computational resources, such as remote or under-resourced medical facilities. Schwartz et al., [105, 98], categorize this approach as "Red AI" characterized by striving for enhanced accuracy while overlooking its economic and environmental impacts.

In contrast, "Green AI" supports DL models that meet or surpass existing performance standards while also minimizing energy use, thus advocating for models that are both efficient and sustainable.

Following the Green AI principles, the work in [89], inspired by [88], implemented an approach completely based on DL for preterm infants' pose estimation from depth video recordings acquired in NICUs. This was based on two subsequent CNNs. The first CNN performed the preliminary identification of limbs positions, while the second serves to refine the results obtained from the initial CNN. In the work, there is a preliminary hint of the necessity to enhance the sustainability of the algorithms, which is exemplified through an ablation analysis of the proposed architectural framework. The latter, specifically, was optimised from a computational point of view through the implementation of asymmetric convolutions that had the property of reducing the number of trainable parameters of the neural network while keeping almost unchanged the results [106].

As the field advances, it is crucial to find a balance between technical advancement and the development of economically and energetically sustainable solutions as to ensure the equitable distribution of the benefits yielded by these decision-support systems [86].

With this vision, the research in [94], presents an all GreenAI paradigm-oriented study: proposing the TwinEDA architecture designed for limb segmentation in preterm infants using depth images. TwinEDA's design integrates elements from two distinct CNN models: it incorporates the efficient lightweight asymmetric and dilated convolutions from [107], as well as the more complex bi-branch Unet structure from [88]. This architecture aims to match the performance of the Red-AI oriented network proposed in [88], but with enhanced efficiency akin to the model in [107].

Pursuing this research, this Chapter focuses on addressing two principal issues identified in the TwinEDA study [94], which underscore the need for additional investigation in specific domains:

- The original design of TwinEDA was devoted to minimizing computational demands. However, the assessment of computation costs was limited to just memory requirements and the number of trainable parameters. This approach was found to be suboptimal, as evidenced by the relevant literature [98]. **In response to this issue, a more efficient variant, TwinEDA Light, has been introduced. Its primary goal is to enhance the computational efficiency of TwinEDA. The effectiveness of this improvement is assessed through an in-depth analysis of FLOPs, providing a more thorough evaluation of computational performance.**

- The transition towards edge computing is identified as a pivotal development, enabling the application of algorithms on widely accessible computing devices. This would adhere to the principles of distributive justice and fairness in artificial intelligence, thereby expanding the global accessibility of high-quality neonatal care [96]. **As a result, the proposed architecture, along with other comparative DL approaches, has been deployed on the NVIDIA Jetson Nano device. This implementation allows for an empirical assessment of their performance, considering both predictive accuracy and inference speed, in edge computing contexts.**

In the research presented in [94], where the undersigned contributed as a coauthor, a specific domain of investigation (i.e., the development of an architecture along the lines dictated by GreenAI) was established. The work provides a crucial basis for the analyses that will follow in this chapter, which represent the further development of the vision system for automatic preterm infants' limbs segmentation from depth images (Figure 3.1). Indeed, the goal of this research is to (i) refine the DL approach proposed in [94] through an in-depth analysis of FLOPs so that it will be even more efficient, (ii) fully orient the research following the edge computing paradigm, in this regard all the outcomes presented will be the result of the deployment on the NVIDIA® computing device. The impact of the research is twofold, while on the one hand an operational methodology for optimizing a network will be proposed so that the deployment in current clinical practice results as energy efficient as possible, on the other hand it will testify how SBC-type computing devices ensure the possibility of deploying advanced monitoring systems without cost barriers. This latter integration, and the subsequent discussion of results, introduces an innovative approach in computer vision, particularly in its application to clinical decision-making tools. This effort seeks to establish a new benchmark in literature, illustrating how advanced decision support systems can be effectively implemented in environments with limited resources and to the best of the author's knowledge this is among the first work in literature in the field of monitoring by vision systems.

### 3.2.1 From TwinEDA to TwinEDA Light

TwinEDA was conceived to put together the best features from two baseline architectures: EDANet [107] and a CNN we previously designed [89] for preterm infants' limbs segmentation. Since EDANet was intended for real-time image semantic segmentation, it was designed to be compute-efficient. For this reason, its main computational blocks rely on lightweight surrogates of convolutions, such as dilated convolutions [108] and asymmetric convolutions. When com-

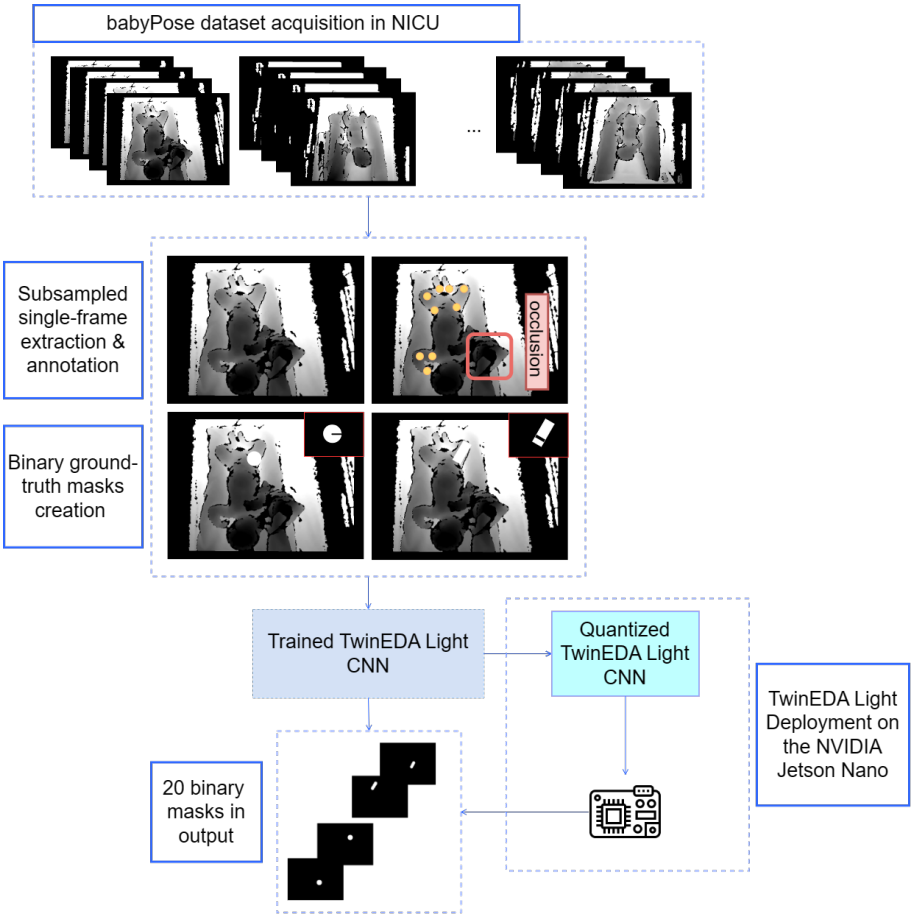



**Figure 3.1** Workflow of the proposed approach to monitor preterm infants' limb-movement.

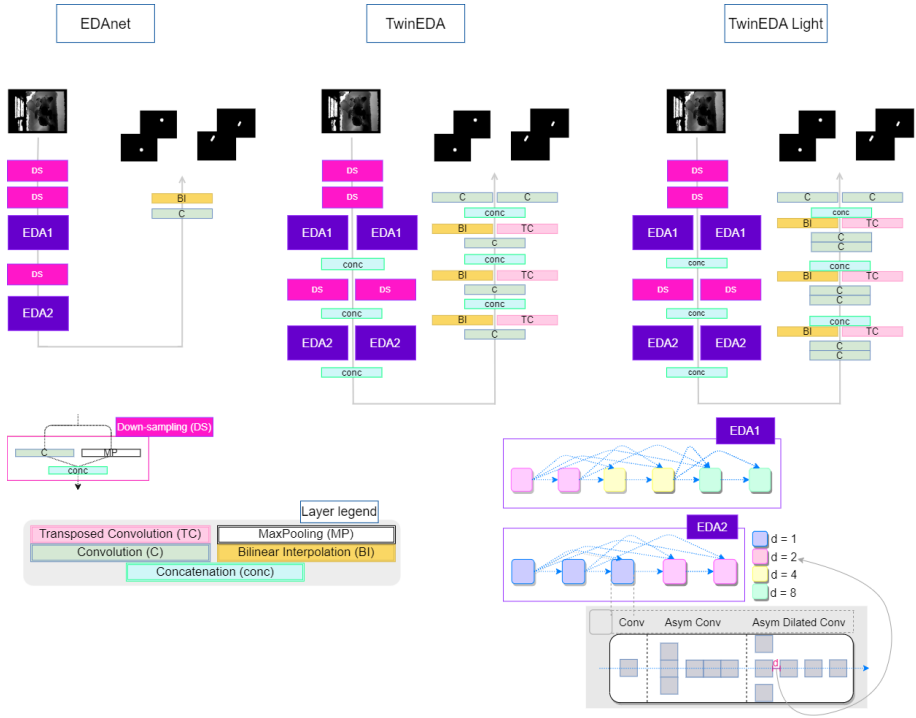



**Figure 3.2** Architecture of EDANet, TwinEDA, and TwinEDA Light. Every block or layer is explained in the bottom part of the image. *Conv* stands for convolution and *Asym* for Asymmetric. EDA1 and EDA2 are the two processing units of EDANet, that we maintain in both TwinEDA and TwinEDA Light. EDA1 (EDA2) consists of six (five) densely-connected consecutive convolutional blocks, each of which processes the data via a normal convolution, an asymmetric convolution, and an asymmetric and dilated convolution, with increasing dilation factor (d) througout EDA1 and EDA2. The values for d are powers of 2 and are color-coded in the image.

pared to the network in [88], EDANet is a much higher throughput but weaker performance. Of course, this is explained by the two different approaches used when the two CNNs were designed (high speed for EDANet, strong performance for the CNN proposed in [88]). TwinEDA puts together the elements that make EDANet fast and efficient (namely, lightweight operations like dilated and asymmetric convolutions), and the CNN in [88] highly-performing on the task (namely, the bi-branch structure of each stage inside UNet [109]).

Fig. 3.2 shows the architecture of EDANet, TwinEDA, and newly proposed TwinEDA Light. TwinEDA expands EDANet's architecture both in the encoding path, by parallelizing its blocks in a bi-branch structure, and in the decoding path, by using more up-sampling stages and including trainable transposed convolutions in each of them. TwinEDA has 3.73 million trainable parameters, more than five times as many as EDANet (around 0.69 million).

The downsampling path uses two bi-branch blocks (inspired by [88]) that employ 3×3 strided convolutions with a stride of 2 and maxpooling layers to reduce the image/feature maps size. It includes two EDA modules (from [107]), EDA1 and EDA2, comprising six and five densely connected sub-blocks, respectively. Each sub-block features a 1×1 convolution to decrease the number of feature maps, an asymmetric 3×3 convolution, and a dilated 3×3 asymmetric convolution. The dilation factor in these convolutions, inspired by EDANet, increases progressively, with EDA1 using dilation factors of 2, 4, and 8, and EDA2 using factors of 1 and 2. This dense data flow approach in each EDA module allows for efficient processing of features. The outputs of the EDA1 modules are concatenated and then passed through additional down-sampling blocks and the EDA2 modules.

The upsampling path in TwinEDA mirrors the bi-branch structure of the downsampling path. It processes the data initially through a single 1×1 convolutional layer, followed by two parallel layers. One layer implements a 3×3 transposed convolution (as in [88]), while the other performs a bilinear interpolation operation (as in [107]), which is a demand-driven process.

As reported in the literature [98], in order to assess the computational requirements and efficiency of a CNN, only relying on the number of trainable parameters can be misleading and does not provide a clear picture. Indeed, different architectures can use the same amount of parameters for different operations (e.g., the same set of parameters $[a, b]$ can be used to perform $a + b$ or $a^b$, the latter of which clearly requires more computation), or in a different layers (e.g., CNNs typically have a better inductive bias than Fully-Connected networks, which implies that the latter need more parameters and more samples to generalize on unseen data [110]). Therefore, to obtain a further reduction in inference time and compute, this research is focused more on FLOPs

or Multiply-Add operations (MACs)[2], rather than on model parameters as in [94]. Although the number of parameters and the number of FLOPs are highly correlated (displaying a correlation of 0.772 in CNNs and 0.994 for transformers [111]), the number of FLOPs is the exact measure of the amount of compute that a program requires, in terms of single summations and multiplications.

The current research landscape reveals a notable gap in studies evaluating the efficiency of CNNs based on the number of FLOPs. While Tang et al. [112] proposed a method involving a loss function that incorporates FLOPs minimization, leading to model pruning for reduced computation, the approach presented here, particularly in the context of TwinEDA Light, differs significantly. Instead of pruning, this method focuses on optimizing an existing architecture (TwinEDA) by rearranging computation-intensive operations to address computational inefficiencies.

## 3.3 Methods

### 3.3.1 TwinEDA Light

This research investigates the hypothesis that the distribution of FLOPs within a CNN is crucial for identifying potential "compute bottlenecks" and, consequently, increased latency. It is hypothesized that a CNN exhibiting an uneven FLOPs distribution across its layers is likely to be more efficiently optimized by focusing on the layers with the highest FLOPs demands. An uneven FLOPs distribution is tentatively defined as a scenario where a single layer accounts for more than 5-10% of the total computation. The ptflop library [113] was used to estimate the FLOPs for the entire network and each layer individually. The FLOPs per layer (FPL) percentage is calculated by dividing the FLOPs in a layer by the total network FLOPs, and standard deviation (std) is used as a measure of dispersion, indicating the unevenness of the FLOPs distribution. An even FPL distribution would theoretically have a std of 0, meaning that each layer requires the same amount of compute. In order to compare the std values between relatively similar architectures, the coefficient of variation (CV) is additionally reported for the FPL distribution in the network, defined as the ratio between mean and std, so as to normalize the degree of deviation with respect to the mean network size.

The FPL distribution for TwinEDA (shown in Fig. 3.3) has std=3.14 and CV=6.44. The bar plot clearly shows a huge discrepancy between the majority

[2]Nowadays, most architectures rely on Fuse Multiply-Add operations (MACs), that perform a sum and a multiplication ($x*a+b$) in just one FLOP, so one FLOP includes two MACs. Although the two numbers are linearly dependent and express the same quantity, we will refer to FLOPs in this paper, in order to give a more realistic description of the compute inside CNNs.

of layers (many of which are not shown in the plot, as their FPL is below 3%) and three of the final layers, each of which takes up for 27% of total FLOPs. A deeper analysis, revealed that these three layers are the only three transposed convolutions in TwinEDA (the pink layers in Fig. 3.2, also marked with TC). Their disproportionate consumption of computational resources, compared to other layers, indicates inefficiencies in their design, potentially leading to higher computational costs in the network. Thus, it is hypothesized that a faster and less compute-demanding CNN can be achieved by reducing the std for this distribution. As per EDANet (Fig. 3.4), the FPL distribution also shows that one layer absorbs around 27% of the total FLOPs in the network (in particular).

Of course, reducing the std by completely changing the architecture would not allow for a fair comparison between the two CNNs, nor for a fair evaluation of the method. Therefore, the strategy involves a reduction in TwinEDA's hyperparameters (height, width, and depth of the feature maps) in the region of the three transposed convolutions (ConvTranspose). In particular, in order to keep the two CNNs as similar as possible, the spatial size (width and height) was preserved and choose to operate on depth, i.e., reducing the number of feature maps that the ConvTranspose layers process. This is achieved by introducing a squeeze with a 1x1 convolutions before each of these ConvTranspose, whose aim is to halve the number of feature maps that these layers have to process. In particular, the three ConvTranspose layers now respectively receive 67%, the 66%, and the 50% of the number of input feature maps as before.

As can be seen from Fig. 3.5, these small changes lead to a decrease in std in the FPL distribution from 3.14 (TwinEDA, Fig. 3.3) to 2.68, which is very close to the value for EDANet (2.60, Fig. 3.4). Similarly, CV decreases from TwinEDA's (6.44) to TwinEDA Light's (5.01), which is, however, much bigger than EDANet's (3.01).

### 3.3.2 Preterm Infants' Kinematic Model and Ground Truth Preparation

As in the work [88, 94, 89], 12 binary masks for the infants' limb-joint and 8 for the connections between the joints were prepared as ground-truth. The 12 joints considered by the kinematic model were: wrists, elbows, shoulders, ankles, knees, and hips. The 8 joint-connections were: forearms, arms, legs, and thighs. A joint-mask consists of all the pixels inside a radius $r$ centered on the annotation site. A connection-mask is a rectangular region of thickness $r$ lying above the straight line connecting two consecutive joints. Individual masks were built for each joint and joint-connection to handle possible self- or external- occlusions (Fig. 3.1, central part).

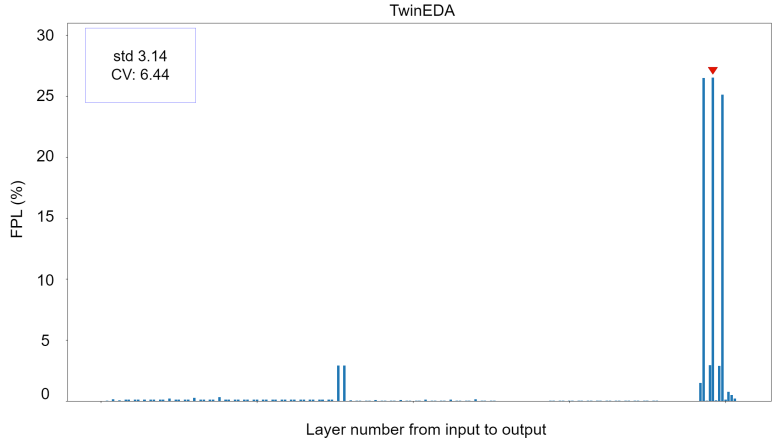



**Figure 3.3** The percentage of FLOPs per layer (FPL) for TwinEDA. The maximum FPL in the network is highlighted by a red triangle (27%). The coefficient of variation (CV), defined as the ratio between mean and (standard deviation) std, is also reported (along wtih sd) for the FPL distribution in the network.

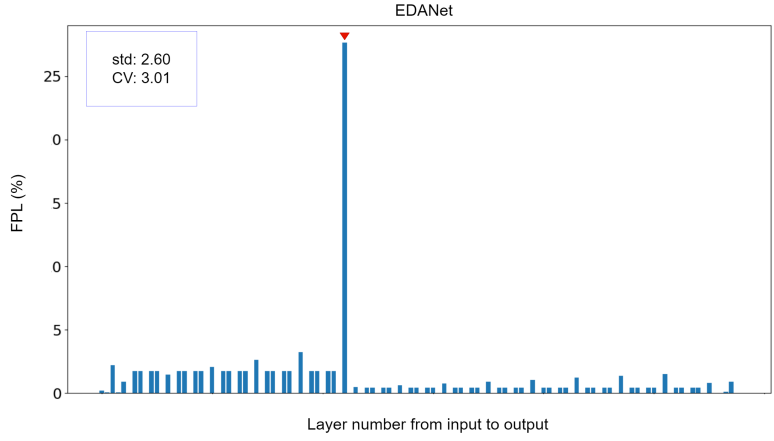



**Figure 3.4** The percentage of FPL for EDANet. The maximum FPL in the network is highlighted by a red triangle (27%). The coefficient of variation (CV), defined as the ratio between mean and (standard deviation) std, is also reported (along wtih sd) for the FPL distribution in the network.

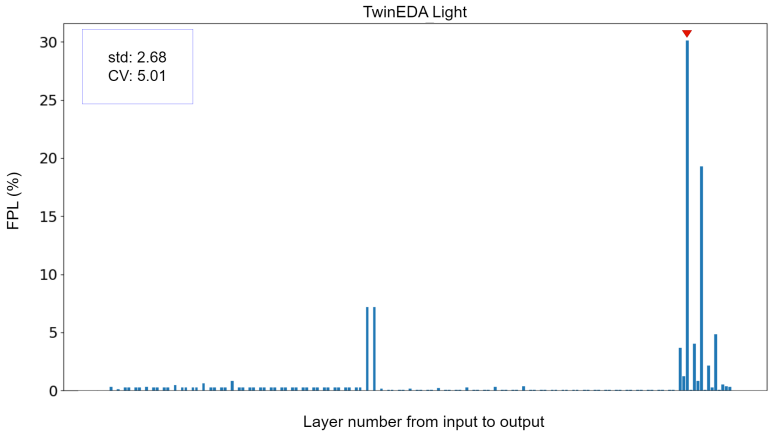



**Figure 3.5** The percentage of FPL for TwinEDA Light. The maximum FPL in the network is highlighted by a red triangle (30%). The coefficient of variation (CV), defined as the ratio between mean and (standard deviation) std, is also reported (along wtih sd) for the FPL distribution in the network.

### 3.3.3 Deployment on Edge Device

As stated in Chapter 2, to fully leverage the computing capabilities of the Jetson Nano and improve the TwinEDA light inference speed, the *TensorRT* framework was used. Developed by NVIDIA®, *TensorRT* optimizes a DL model for specific hardware, converting it into a serialized engine for high-performance inference on GPUs.

Initially, the model was transformed from Pytorch to *onnx*, an open format that facilitates interoperability among Artificial Intelligence frameworks. Following this, the final *TensorRT* engine was crafted from the *onnx*-compatible model, incorporating GPU-specific enhancements such as layer fusions and precision calibration.

This process yielded an optimized TwinEDA light engine, which was then employed to assess performance on the Jetson Nano.

## 3.4 Experimental Protocol

### 3.4.1 Dataset

For the experiments, the babyPose dataset [114] was expanded by 11 depth videos. This expansion resulted in a dataset comprising 27 depth videos from 27 spontaneously breathing preterm infants. These videos were acquired in the NICU of G. Salesi Hospital in Ancona, Italy, following approval from the Ethics Committee of the "Ospedali Riuniti di Ancona" (ID: Prot. 2019-399) and upon

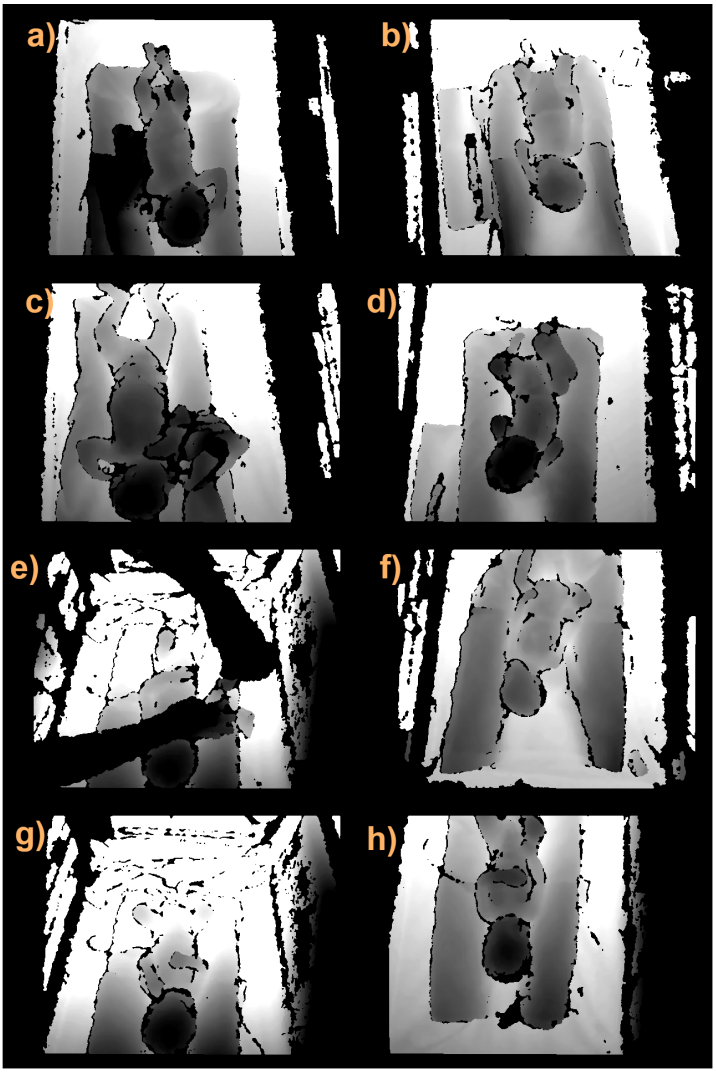



**Figure 3.6** Samples of depth frames from the babyPose dataset.

receiving written consent from the infants' legal guardians. The videos, each 5 minutes in length, were acquired using the Astra Mini S - Orbbec® at a frame rate of 30 FPS and an image size of 640x480 pixels. To address privacy concerns, only depth videos were acquired, and all networks were trained and tested exclusively on depth frames. Samples of depth frames are shown in Figure 3.6.

Consistent with the average movement frequency of preterm infants [115], one frame was extracted from each video every 5 seconds. Out of these extracted frames, 1000 frames per infant were annotated with the assistance of clinical partners using a publicly available, custom-built annotation tool[3]. A random selection of 700 frames per infant was used to train the proposed architectures, while the remaining 300 frames were reserved for testing.

### 3.4.2 Training Settings

To manage the training time and memory requirements, the resolution of all depth frames was reduced to 128x96 pixels. In preprocessing these frames, the mean intensity was removed as described in [88, 94]. A radius of 4 pixels was selected for constructing the ground-truth masks.

A batch size of 256 was set, along with an initial learning rate of 0.05. The cosine annealing scheduler with five restarts policy was applied to the learning rate. The best combination of loss function, learning rate schedule, and optimizer was determined through a grid-search analysis. Following this analysis, the networks were trained for 200 epochs using Adam as the optimizer and per-pixel binary cross entropy as the loss function. The optimal configuration of weights across epochs was determined based on the lowest loss value obtained on the validation set. The training of these architectures was conducted using Pytorch 1.12.0 on a GPU NVIDIA® RTX 3090 with 24 GB of RAM.

### 3.4.3 Comparison with Other Architectures

The proposed architecture (i.e., TwinEDA Light) was compared with the original TwinEDA and EDANet, all deployed on the NVIDIA Jetson Nano computing device. The comparison with the first architecture, i.e., the TwinEDA, serves to prove the research hypothesis that the TwinEDA Light is an efficient version of the TwinEDA built on the basis of an in-depth FLOPs analysis on the individual blocks peculiar to the network. On the other hand, the comparison with EDANet serves to demonstrate how, in accordance with the Green AI paradigm, it is essential to design sustainable architectures without significant performance degradation particularly in the medical field. It should be empha-

[3]https://github.com/roccopietrini/pyPointAnnotator

| Network | GFLOPs | N° params (M) | mean *DSC* | FPS (16bit) | FPS (32bit) |
|---|---|---|---|---|---|
| TwinEDA | 7.18 | 3.73 | 81.8 | 22.7 | 19.9 |
| TwinEDA Light | 2.86 | 2.8 | 82.3 | 41.9 | 30.5 |
| EDANet | 0.42 | 0.69 | 77.3 | 82.3 | 57.7 |

**Table 3.1** The table shows, for each architecture, the number of Giga FLOPs (GFLOPs), the number of trainable parameters the average *DSC* values, and the inference speed in FPS. FPS were assessed on NVIDIA Jetson Nano in two distinct formats: FP16 (or half-precision floating-point) and FP32 (single-precision floating-point). To distinguish the post-quantization and the not-quantized architectures' throughput, the nomenclatures FPS (16bit) and FPS (32bit) were used.

sised that all architectures were retrained on the same split dataset described in the previous section.

### 3.4.4 Performance Metrics

The performance in terms of efficacy was measured via the Dice similarity coefficient ($DSC$) computed between the ground-truth binary masks and the predicted ones.

$$DSC = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{3.1}$$

where $TP$ and $FP$ are the true joint (or joint-connection) and background pixels classified as joints, respectively. $FN$ are the pixels belonging to a joint (or joint-connection) wrongly ascribed as background.

Following Sec. 2.4.1.2, the efficiency of each model was evaluated by both reporting the number of FLOPs for each CNN but also by measuring the inference speed in FPS. This assessment was carried out for models in two distinct formats: those utilizing FP16 (half-precision floating-point) post-training quantization and those operating in FP32 (single-precision floating-point) format.

## 3.5 Results

Table 3.1 presents the outcomes of the conducted experiments. The performance of each CNN is reported in terms of Giga FLOPs (GFLOPs), number of parameters, mean $DSC$, and FPS on NVIDIA Jetson Nano. FPS assessments were conducted following two different weights' precisions: FP16 (after a quantization process), and FP32. The $DSC$ values are not significantly influenced by the type of quantization, leading to the decision to report only those obtained using FP32 weights.

A notable trend is the decrease in the number of parameters: TwinEDA Light, despite its increased depth due to the application of 1x1 convolutions for feature map channel reduction, has fewer parameters (2.8 million) compared

to TwinEDA (3.73 million). Simultaneously, TwinEDA Light demonstrates a slight yet significant improvement in performance, achieving a *DSC* of 82.3, compared to TwinEDA's 81.8 and EDANet's 77.3.

Examining the FPS for both post-training quantization methodologies EDANet remains the fastest network, with 82.3 FPS in 16-bit and 57.7 in 32-bit formats, however, TwinEDA Light significantly enhances performance over TwinEDA, elevating the computation speed from near real-time to true real-time. Specifically, TwinEDA Light achieves FPS of 41.9 in 16-bit format and 30.5 in 32-bit format, compared to TwinEDA's 22.7 (16-bit) and 19.9 (32-bit), respectively.

## 3.6 Discussion

This chapter delves into recent advancements in DL, specifically within the healthcare sector, emphasizing the need to integrate edge AI computing to strengthen and operationalize the GreenAI principles, in order to overcome barriers such as high costs and environmental concerns. The research hinges on the development of a DL-based approach, particularly tailored for edge computing, and focuses on segmenting preterm infants' limbs from depth images acquired in NICUs.

The discussion is further extended by addressing two primary issues identified in the TwinEDA study [94]: the optimization of computational efficiency and the transition towards edge computing for broader accessibility of neonatal care. The research introduces TwinEDA Light, a variant designed to enhance computational efficiency assessed via FLOPs analysis.

When evaluating quantitative performance in terms of *DSC*, TwinEDA Light is the one that achieved better results with an increase of +0.5 *DSC* from TwinEDA to its Light version and +5 *DSC* from EDANet to TwinEDA Light. This follows the precepts of GreenAI, according to which an eco-aware network is designed to achieve adequate performance with low computational costs. In fact, TwinEDA Light has a lower number of parameters (- 0.93 M) than TwinEDA and a significant reduction in GFLOPs (-4.32 G). This was made possible by implementing a strategy for evaluating the GFLOPs absorbed by each layer of the TwinEDA, trying to reduce as much as possible any absorption peaks also clearly visible in the Figure 3.3. Indeed, TwinEDA Light was developed with the objective of minimizing the std of the FPL distribution. This goal is achieved; however, it is important to note that in Fig. 3.5, the most computationally intensive layer in TwinEDA Light accounts for 30% of the total FLOPs in the network. This is the same ConvTranspose layer that absorbs 27% of the total FLOPs in TwinEDA (Fig. 3.3). Having a stronger outlier in the distribution seems contradictory to the aim of reducing the std of the distribution. Nonetheless, this approach results in a reduction of FPL for

the other two ConvTranspose layers, which consume 19% and 4% of the total FLOPs in TwinEDA Light, compared to 26% and 15% in TwinEDA.

It is worth noting that this study does not aim to conclusively prove the hypothesis that a negative correlation exists between the standard deviation in FPL distribution and the model efficiency and throughput. Rather, the design of TwinEDA Light serves as a proof of concept for the method, with further research planned in this area. This work encourages that other researchers also consider this metric in the design and optimization of CNN-based models.

## 3.7 Conclusion and Future Perspective

This research marks a significant shift in the development of DL models, with a strong emphasis on sustainability and adherence to the principles of distributive justice and fairness in artificial intelligence. Central to this study are two objectives: first, refining the DL approach as detailed in [94] through a comprehensive analysis of FLOPs to boost efficiency; second, embracing the edge computing paradigm, demonstrated by the deployment of TwinEDA Light network on the NVIDIA® computing device. This approach is not just about achieving energy efficiency in clinical practice, but is also a incentive to making advanced monitoring systems globally accessible and free from cost barriers, thus expanding the reach of high-quality neonatal care.

The study's forward-looking plans include extending the research beyond prototype testing in NICUs. The focus is on further optimizing convolutional neural networks for edge computing devices, which involves developing innovative methods to minimize the energy consumption of AI models throughout their lifecycle, from training to deployment. This aspect is particularly crucial in healthcare, where the need for accessible, equitable, and cost-effective technology solutions is paramount. In order to minimize costs, some well-studied techniques could be used, like Knowledge Distillation [116], to boost the performance of an inexpensive model thanks to the supervision of a pre-trained teacher model. Moreover, as mentioned in Sec. 3.6, further work will be dedicated to investigate the hypothesis underlying the proposed method, *i.e.*, the existence of a negative correlation between the standard deviation in FPL distribution and model efficiency.

By integrating these methodologies, the research not only sets a new trend in computer vision, especially in the realm of clinical decision-making tools, but also strives to establish a new benchmark in the literature.

# Chapter 4

# Conclusive remarks

## 4.1 Conclusion

Driven by the extreme importance of fostering technological progress oriented toward the development of approaches that are efficient, sustainable, and affordable for all, the journey of this thesis began with the ultimate scope of contributing significantly to research in the field of edge AI applied to computer vision, with the aim of designing and developing intelligent systems for real-time monitoring of human behavior. In pursuit of this scope, as outlined in the objective of this thesis, the three-year research mainly targeted two application domains: surveillance and security, focusing on weapon detection from surveillance cameras, and healthcare, focusing on limb segmentation from depth cameras for preterm infants monitoring. A deep dive into the considered domains enabled the identification of limitations still existing in state-of-the-art approaches, as well as the gaps that need to be addressed to contribute to technological progress in the development of edge-compliant methodologies.

The two key challenges to be faced in weapon detection from surveillance video are the small size of the weapons to be detected and the need to perform real-time detections. In the edge context, these challenges are amplified by the low computational capacity of edge devices, which limit the complexity of usable models, raising the need to develop models with the best possible balance of complexity, speed, and accuracy. To progressively tackle such challenges, the research effort into the domain of weapon detection in surveillance scenarios led to multiple outcomes, each building upon the findings of the previous ones. These outcomes represent incremental steps toward achieving the primary aim of developing surveillance systems based on DL methods with an optimal speed-accuracy balance on low-power edge devices.

A preliminary study explored and underscored the feasibility of integrating the edge AI paradigm within the context of video surveillance in multi-camera settings, also demonstrating the ability to achieve real-time responsiveness in these environments. Motivated by these preliminaries, the following research addressed the specific task of weapon detection. The study analyzed and quan-

titatively evaluated the performance of edge devices in this context when running DL models, demonstrating their limitations and computational capabilities, and enabling the selection of the NVIDIA Jetson Nano as the most suitable SBC. Based on these findings, the research outlined in this thesis moved forward with the development of a deep learning method for weapon detection that could optimally balance three key components: speed, accuracy, and complexity, when performed on edge devices. The work compared the proposed method with the state of the art, emphasizing both its validity and the lack in the current state of the art of methodologies balanced in all the components (i.e., speed, accuracy, complexity). Nevertheless, the still-existing limitations in the proposed approach encouraged further research in this direction. Thus, the last research work in this thesis proposed an approach for weapon detection without any limitation on applicability, which at the same time significantly enhances the accuracy without increasing complexity, further advancing the state of the art in the adoption of edge AI paradigm in weapon detection.

Moving into the healthcare domain for preterm infants' monitoring, the literature review pointed out that the current research does not align with the principles of efficiency and economic sustainability. The approaches in literature require significant computational resources, impacting the economic feasibility and hindering their integration in settings with limited resources. The research in this thesis addressed such needs by proposing an enhancement over the existing approaches, with the development of a more efficient method for preterm infants' limb segmentation. The validation of the approach via both the analysis on the computational complexity and the execution on edge devices emphasized its applicability in real contexts, as well as its superiority over the other state-of-the-art approaches.

The findings in both the surveillance and healthcare domains not only demonstrate the robustness and applicability of the proposed methods, but also lay the foundation for future explorations in similar areas. Indeed, the research pursued in this thesis showed the potential and benefits that the integration of edge AI and computer vision can bring, while also proving its feasibility in challenging contexts.

## 4.2 Impact

The research conducted in this thesis discloses significant practical implications in both domains examined. In the domain of surveillance and security, the main goal pursued through the development of new methodologies for weapon detection is the increase in the effectiveness of security measures. In the healthcare sector, efforts to improve automatic systems for monitoring preterm infants are aimed at better treatments and diagnoses. These goals, however, are not

an end in themselves. Enhancing the effectiveness of security systems leads to crime prevention and enables prompt intervention in dangerous situations, while better treatments and diagnoses lead to faster patient recovery and early discovery of potential diseases. These implications, as a result, improve people's well-being, which is the ultimate and real research goal. For both weapon detection and preterm infants monitoring, the design and development of advanced methodologies that, besides being effective, are also efficient and sustainable, aim to achieve the same ultimate goal. The benefits of using technologies based on the edge AI paradigm in weapon detection are significant in ensuring privacy and data security, and in increasing the responsiveness of automated surveillance systems. In monitoring preterm infants, the adoption of edge AI paradigm brings benefits on patient privacy and on ensuring operational continuity of systems, as well as real-time feedback. All of these benefits, explored in this thesis and pursued during the three years of research, paired with the real possibility of making these technologies affordable to everyone, converge in the same direction by providing the greatest drive towards the ultimate goal of the research, the well-being of society.

# Bibliography


[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[2] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in nlp," *arXiv preprint arXiv:1906.02243*, 2019.

[3] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the dangers of stochastic parrots: Can language models be too big?" in *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, 2021, pp. 610–623.

[4] J. K. P. Seng, K. L.-m. Ang, E. Peter, and A. Mmonyi, "Artificial intelligence (ai) and machine learning for multimedia and edge information processing," *Electronics*, vol. 11, no. 14, p. 2239, 2022.

[5] P. Radoglou-Grammatikis, P. Sarigiannidis, T. Lagkas, and I. Moscholios, "A compilation of uav applications for precision agriculture," *Computer Networks*, vol. 172, p. 107148, 2020.

[6] Y. Ampatzidis, V. Partel, and L. Costa, "Agroview: Cloud-based application to process, analyze and visualize uav-collected data for precision agriculture applications utilizing artificial intelligence," *Computers and Electronics in Agriculture*, vol. 174, p. 105457, 2020.

[7] O. E. Apolo-Apolo, M. Pérez-Ruiz, J. Martinez-Guanter, and J. Valente, "A cloud-based environment for generating yield estimation maps from apple orchards using uav imagery and a deep learning technique," *Frontiers in plant science*, vol. 11, p. 1086, 2020.

[8] I. Gallo, A. U. Rehman, R. H. Dehkordi, N. Landro, R. La Grassa, and M. Boschetti, "Deep object detection of crop weeds: Performance of yolov7 on a real case dataset from uav images," *Remote Sensing*, vol. 15, no. 2, p. 539, 2023.

[9] T. de Camargo, M. Schirrmann, N. Landwehr, K.-H. Dammer, and M. Pflanz, "Optimized deep learning model as a basis for fast uav mapping of weed species in winter wheat crops," *Remote Sensing*, vol. 13, no. 9, p. 1704, 2021.

[10] J. Su, X. Zhu, S. Li, and W.-H. Chen, "Ai meets uavs: A survey on ai empowered uav perception systems for precision agriculture," *Neurocomputing*, vol. 518, pp. 242–270, 2023.

[11] H. H. Nguyen, T. N. Ta, N. C. Nguyen, H. M. Pham, D. M. Nguyen *et al.*, "Yolo based real-time human detection for smart video surveillance at the edge," in *2020 IEEE Eighth International Conference on Communications and Electronics (ICCE)*. IEEE, 2021, pp. 439–444.

[12] T. Qian, F. Zhang, and S. U. Khan, "Facial expression recognition based on edge computing," in *2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*. IEEE, 2019, pp. 410–415.

[13] Q. Zhang, H. Sun, X. Wu, and H. Zhong, "Edge video analytics for public safety: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1675–1696, 2019.

[14] A. Çalışkan, V. Özdemir, E. Baytörk, O. M. Öztörk, O. D. Kefeli, and A. Üzengi, "Real time retail analytics with computer vision," in *2022 Innovations in Intelligent Systems and Applications Conference (ASYU)*. IEEE, 2022, pp. 1–4.

[15] K. R. Kanjula, V. V. Reddy, J. S. Abraham *et al.*, "People counting system for retail analytics using edge ai," *arXiv preprint arXiv:2205.13020*, 2022.

[16] M. Paolanti, R. Pietrini, A. Mancini, E. Frontoni, and P. Zingaretti, "Deep understanding of shopper behaviours and interactions using rgb-d vision," *Machine Vision and Applications*, vol. 31, pp. 1–21, 2020.

[17] C.-F. Lai, W.-C. Chien, L. T. Yang, and W. Qiang, "Lstm and edge computing for big data feature recognition of industrial electrical equipment," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2469–2477, 2019.

[18] D. Hästbacka, J. Halme, L. Barna, H. Hoikka, H. Pettinen, M. Larrañaga, M. Björkbom, H. Mesiä, A. Jaatinen, and M. Elo, "Dynamic edge and cloud service integration for industrial iot and production monitoring applications of industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 1, pp. 498–508, 2021.

[19] S. Vimal, Y. H. Robinson, S. Kadry, H. V. Long, and Y. Nam, "Iot based smart health monitoring with cnn using edge computing," *Journal of Internet Technology*, vol. 22, no. 1, pp. 173–185, 2021.

[20] V. Hayyolalam, M. Aloqaily, Ö. Özkasap, and M. Guizani, "Edge intelligence for empowering iot-based healthcare systems," *IEEE Wireless Communications*, vol. 28, no. 3, pp. 6–14, 2021.

[21] D. Liciotti, M. Paolanti, E. Frontoni, A. Mancini, and P. Zingaretti, "Person re-identification dataset with rgb-d camera in a top-view configuration," in *Video Analytics. Face and Facial Expression Recognition and Audience Measurement*. Springer, 2016, pp. 1–11.

[22] G. Sreenu and M. S. Durai, "Intelligent video surveillance: a review through deep learning techniques for crowd analysis," *Journal of Big Data*, vol. 6, no. 1, pp. 1–27, 2019.

[23] A. Mancini, E. Frontoni, P. Zingaretti, and V. Placidi, "Smart vision system for shelf analysis in intelligent retail environments," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 55911. American Society of Mechanical Engineers, 2013, p. V004T08A045.

[24] F. Jauro, H. Chiroma, A. Y. Gital, M. Almutairi, M. A. Shafi'i, and J. H. Abawajy, "Deep learning architectures in emerging cloud computing architectures: Recent development, challenges and next research trend," *Applied Soft Computing*, vol. 96, p. 106582, 2020.

[25] J. Chen, K. Li, Q. Deng, K. Li, and S. Y. Philip, "Distributed deep learning model for intelligent video surveillance systems with edge computing," *IEEE Transactions on Industrial Informatics*, 2019.

[26] X. Xu, Q. Wu, L. Qi, W. Dou, S.-B. Tsai, and M. Z. A. Bhuiyan, "Trust-aware service offloading for video surveillance in edge computing enabled internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[27] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the internet of things," *IEEE access*, vol. 6, pp. 6900–6919, 2017.

[28] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *European Conference on Computer Vision*. Springer, 2016, pp. 21–37.

[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv preprint arXiv:1506.01497*, 2015.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[32] R. Mayer and H.-A. Jacobsen, "Scalable deep learning on distributed infrastructures: Challenges, techniques, and tools," *ACM Computing Surveys (CSUR)*, vol. 53, no. 1, pp. 1–37, 2020.

[33] United Nations Office on Drugs and Crime - UNODC, "Global study on homicide 2019," 2019, Vienna, https://www.unodc.org/unodc/en/data-and-analysis/global-study-on-homicide.html (Accessed on 29/01/2022).

[34] O. Elharrouss, N. Almaadeed, and S. Al-Maadeed, "A review of video surveillance systems," *Journal of Visual Communication and Image Representation*, vol. 77, p. 103116, 2021.

[35] A. L. Thomas, E. L. Piza, B. C. Welsh, and D. P. Farrington, "The internationalisation of cctv surveillance: Effects on crime and implications for emerging technologies," *International Journal of Comparative and Applied Criminal Justice*, vol. 46, no. 1, pp. 81–102, 2022.

[36] F. Porikli, F. Bremond, S. L. Dockstader, J. Ferryman, A. Hoogs, B. C. Lovell, S. Pankanti, B. Rinner, P. Tu, and P. L. Venetianer, "Video surveillance: past, present, and now the future [dsp forum]," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 190–198, 2013.

[37] N. Cohen, J. Gattuso, and K. MacLennan-Brown, *CCTV operational requirements manual 2009*. United Kingdom: Home Office Scientific Development Branch St. Albans, 2009.

[38] C. Fontes, E. Hohma, C. C. Corrigan, and C. Lütge, "Ai-powered public surveillance systems: why we (might) need them and how we want them," *Technology in Society*, vol. 71, p. 102137, 2022.

[39] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[40] J. R. R. Uijlings, K. E. A. Van De Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[41] J. L. S. González, C. Zaccaro, J. A. Álvarez-García, L. M. S. Morillo, and F. S. Caparrini, "Real-time gun detection in cctv: An open problem," *Neural networks*, vol. 132, pp. 297–308, 2020.

[42] P. Yadav, N. Gupta, and P. K. Sharma, "A comprehensive study towards high-level approaches for weapon detection using classical machine learning and deep learning methods," *Expert Systems with Applications*, p. 118698, 2022.

[43] C. Deng, M. Wang, L. Liu, Y. Liu, and Y. Jiang, "Extended feature pyramid network for small object detection," *IEEE Transactions on Multimedia*, vol. 24, pp. 1968–1979, 2021.

[44] W. Wang, J. Dai, Z. Chen, Z. Huang, Z. Li, X. Zhu, X. Hu, T. Lu, L. Lu, H. Li *et al.*, "Internimage: Exploring large-scale vision foundation models with deformable convolutions," *arXiv preprint arXiv:2211.05778*, 2022.

[45] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum, "Dino: Detr with improved denoising anchor boxes for end-to-end object detection," *arXiv preprint arXiv:2203.03605*, 2022.

[46] S. Qiao, L.-C. Chen, and A. Yuille, "Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 213–10 224.

[47] K. Tong and Y. Wu, "Deep learning-based detection from the perspective of small or tiny objects: A survey," *Image and Vision Computing*, p. 104471, 2022.

[48] D. Zhang, J. Han, L. Yang, and D. Xu, "Spftn: A joint learning framework for localizing and segmenting objects in weakly labeled videos," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 2, pp. 475–489, 2018.

[49] P. Huang, J. Han, N. Liu, J. Ren, and D. Zhang, "Scribble-supervised video object segmentation," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 2, pp. 339–353, 2021.

[50] M. Grega, A. Matiolański, P. Guzik, and M. Leszczuk, "Automated detection of firearms and knives in a cctv image," *Sensors*, vol. 16, no. 1, p. 47, 2016.

[51] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[52] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[53] ——, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[54] G. K. Verma and A. Dhillon, "A handheld gun detection using faster r-cnn deep learning," in *Proceedings of the 7th international conference on computer and communication technology*, 2017, pp. 84–88.

[55] R. Olmos, S. Tabik, and F. Herrera, "Automatic handgun detection alarm in videos using deep learning," *Neurocomputing*, vol. 275, pp. 66–72, 2018.

[56] M. M. Fernandez-Carrobles, O. Deniz, and F. Maroto, "Gun and knife detection based on faster r-cnn for video surveillance," in *Iberian conference on pattern recognition and image analysis*. Springer, 2019, pp. 441–452.

[57] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[58] J. Lim, M. I. Al Jobayer, V. M. Baskaran, J. M. Lim, K. Wong, and J. See, "Gun detection in surveillance videos using deep neural networks," in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2019, pp. 1998–2002.

[59] O. E. Olorunshola, M. E. Irhebhude, and A. E. Evwiekpaefe, "A comparative study of yolov5 and yolov7 object detection algorithms," *Journal of Computing and Social Informatics*, vol. 2, no. 1, pp. 1–12, 2023.

[60] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint arXiv:2207.02696*, 2022.

[61] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: A survey," *Future Generation Computer Systems*, vol. 97, pp. 219–235, 2019.

[62] A. C. Cob-Parro, C. Losada-Gutiérrez, M. Marrón-Romera, A. Gardel-Vicente, and I. Bravo-Muñoz, "Smart video surveillance system based on edge computing," *Sensors*, vol. 21, no. 9, p. 2958, 2021.

[63] A. Baobaid, M. Meribout, V. K. Tiwari, and J. P. Pena, "Hardware accelerators for real-time face recognition: A survey," *IEEE Access*, 2022.

[64] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[65] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-yolov4: Scaling cross stage partial network," in *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, 2021, pp. 13 029–13 038.

[66] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.

[67] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, pp. 303–338, 2010.

[68] W. Rahmaniar and A. Hernawan, "Real-time human detection using deep learning on embedded platforms: A review," *Journal of Robotics and Control (JRC)*, vol. 2, no. 6, pp. 462–468, 2021.

[69] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[70] A.-A. Tulbure, A.-A. Tulbure, and E.-H. Dulf, "A review on modern defect detection models using dcnns–deep convolutional neural networks," *Journal of Advanced Research*, vol. 35, pp. 33–48, 2022.

[71] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," *Advances in neural information processing systems*, vol. 32, 2019.

[72] Y. Lee, J. Kim, J. Willette, and S. J. Hwang, "Mpvit: Multi-path vision transformer for dense prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7287–7296.

[73] Y. Li, M. Shao, B. Fan, and W. Zhang, "Multi-scale global context feature pyramid network for object detector," *Signal, Image and Video Processing*, pp. 1–9, 2022.

[74] B. Na and G. C. Fox, "Object detection by a super-resolution method and a convolutional neural networks," in *2018 IEEE international conference on big data (Big data)*. IEEE, 2018, pp. 2263–2269.

[75] Y. Bai, Y. Zhang, M. Ding, and B. Ghanem, "Sod-mtgan: Small object detection via multi-task generative adversarial network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 206–221.

[76] S. M. A. Bashir and Y. Wang, "Small object detection in remote sensing images with residual feature aggregation-based super-resolution and object detector network," *Remote Sensing*, vol. 13, no. 9, p. 1854, 2021.

[77] J. Zhang, J. Lei, W. Xie, Z. Fang, Y. Li, and Q. Du, "Superyolo: Super resolution assisted object detection in multimodal remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–15, 2023.

[78] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 136–144.

[79] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1874–1883.

[80] J. Zhao, K. Dai, P. Zhang, D. Wang, and H. Lu, "Robust online tracking with meta-updater," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[81] Z. Gu, "Home smart motion system assisted by multi-sensor," *Microprocessors and microsystems*, vol. 80, p. 103591, 2021.

[82] H. Turpin, S. Urben, F. Ansermet, A. Borghini, M. M. Murray, and C. Müller-Nix, "The interplay between prematurity, maternal stress and children's intelligence quotient at age 11: a longitudinal study," *Scientific Reports*, vol. 9, no. 1, pp. 1–9, 2019.

[83] G. S. Mallmann, A. L. N. França, P. R. Almeida, L. S. Oliveira, L. S. F. Merey, and D. A. Soares-Marangoni, "Association between the general movement optimality score and clinical features in newborns during hospitalization: A cross-sectional study," *Early Human Development*, vol. 177, p. 105720, 2023.

[84] T. Zhao, T. Griffith, Y. Zhang, H. Li, N. Hussain, B. Lester, and X. Cong, "Early-life factors associated with neurobehavioral outcomes in preterm infants during nicu hospitalization," *Pediatric Research*, vol. 92, no. 6, pp. 1695–1704, 2022.

[85] C. Yildirim, A. Asalioğlu, Y. Coşkun, G. Acar, and İ. Akman, "General movements assessment and alberta infant motor scale in neurodevelopmental outcome of preterm infants," *Pediatrics & Neonatology*, vol. 63, no. 5, pp. 535–541, 2022.

[86] L. Migliorelli, S. Tiribelli, A. Cacciatore, B. Giovanola, E. Frontoni, and S. Moccia, "Accountable deep-learning-based vision systems for preterm infant monitoring," *Computer*, vol. 56, no. 5, pp. 84–93, 2023.

[87] K. Raghuram, S. Orlandi, P. Church, M. Luther, A. Kiss, and V. Shah, "Automated movement analysis to predict cerebral palsy in very preterm infants: An ambispective cohort study," *Children*, vol. 9, no. 6, p. 843, 2022.

[88] S. Moccia, L. Migliorelli, R. Pietrini, and E. Frontoni, "Preterm infants' limb-pose estimation from depth images using convolutional neural networks," in *2019 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. IEEE, 2019, pp. 1–7.

[89] L. Migliorelli, E. Frontoni, and S. Moccia, "An accurate estimation of preterm infants' limb pose from depth images using deep neural networks with densely connected atrous spatial convolutions," *Expert Systems with Applications*, vol. 204, p. 117458, 2022.

[90] A. Ruiz-Zafra, D. Precioso, B. Salvador, S. P. Lubián-López, J. Jiménez, I. Benavente-Fernández, J. Pigueiras, D. Gómez-Ullate, and L. C. Gontard, "Neocam: An edge-cloud platform for non-invasive real-time monitoring in neonatal intensive care units," *IEEE Journal of Biomedical and Health Informatics*, 2023.

[91] C. O. Alenoghena, A. J. Onumanyi, H. O. Ohize, A. O. Adejo, M. Oligbi, S. I. Ali, and S. A. Okoh, "ehealth: A survey of architectures, developments in mhealth, security concerns and solutions," *International Journal*

*of Environmental Research and Public Health*, vol. 19, no. 20, p. 13071, 2022.

[92] Y. Alghofaili, A. Albattah, N. Alrajeh, M. A. Rassam, and B. A. S. Al-Rimy, "Secure cloud infrastructure: A survey on issues, current solutions, and open challenges," *Applied Sciences*, vol. 11, no. 19, p. 9005, 2021.

[93] M. Laroui, B. Nour, H. Moungla, M. A. Cherif, H. Afifi, and M. Guizani, "Edge and fog computing for iot: A survey on current research activities & future directions," *Computer Communications*, vol. 180, pp. 210–231, 2021.

[94] L. Migliorelli, A. Cacciatore, V. Ottaviani, D. Berardini, R. L. Dellaca', E. Frontoni, and S. Moccia, "Twineda: a sustainable deep-learning approach for limb-position estimation in preterm infants' depth images," *Medical & Biological Engineering & Computing*, vol. 61, no. 2, pp. 387–397, 2023.

[95] Y. K. Dwivedi, L. Hughes, E. Ismagilova, G. Aarts, C. Coombs, T. Crick, Y. Duan, R. Dwivedi, J. Edwards, A. Eirug *et al.*, "Artificial intelligence (ai): Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy," *International Journal of Information Management*, vol. 57, p. 101994, 2021.

[96] B. Giovanola and S. Tiribelli, "Weapons of moral construction? on the value of fairness in algorithmic decision-making," *Ethics and Information Technology*, vol. 24, no. 1, p. 3, 2022.

[97] R. Singh and S. S. Gill, "Edge ai: a survey," *Internet of Things and Cyber-Physical Systems*, 2023.

[98] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green ai," *Communications of the ACM*, vol. 63, no. 12, pp. 54–63, 2020.

[99] R. Verdecchia, J. Sallou, and L. Cruz, "A systematic review of green ai," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, p. e1507, 2023.

[100] S. Cass, "Nvidia makes it easy to embed ai: The jetson nano packs a lot of machine-learning power into diy projects-[hands on]," *IEEE Spectrum*, vol. 57, no. 7, pp. 14–16, 2020.

[101] K. D. McCay, E. S. Ho, H. P. Shum, G. Fehringer, C. Marcroft, and N. D. Embleton, "Abnormal infant movements classification with deep learning on pose-based features," *IEEE Access*, vol. 8, pp. 51 582–51 592, 2020.

[102] M. Moro, V. P. Pastore, C. Tacchino, P. Durand, I. Blanchi, P. Moretti, F. Odone, and M. Casadio, "A markerless pipeline to analyze spontaneous movements of preterm infants," *Computer Methods and Programs in Biomedicine*, vol. 226, p. 107119, 2022.

[103] S. Reich, D. Zhang, T. Kulvicius, S. Bölte, K. Nielsen-Saines, F. B. Pokorny, R. Peharz, L. Poustka, F. Wörgötter, C. Einspieler *et al.*, "Novel ai driven approach to classify infant motor functions," *Scientific Reports*, vol. 11, no. 1, p. 9888, 2021.

[104] D. Sakkos, K. D. Mccay, C. Marcroft, N. D. Embleton, S. Chattopadhyay, and E. S. Ho, "Identification of abnormal movements in infants: A deep neural network for body part-based prediction of cerebral palsy," *IEEE Access*, vol. 9, pp. 94 281–94 292, 2021.

[105] W. T. Schmidt, M. Regan, M. C. Fahey, and A. Paplinski, "General movement assessment by machine learning: why is it so difficult?" *Journal of Medical Artificial Intelligence*, vol. 2, no. July, p. 15, 2019.

[106] H. Yang, C. Yuan, B. Li, Y. Du, J. Xing, W. Hu, and S. J. Maybank, "Asymmetric 3d convolutional neural networks for action recognition," *Pattern recognition*, vol. 85, pp. 1–12, 2019.

[107] S.-Y. Lo, H.-M. Hang, S.-W. Chan, and J.-J. Lin, "Efficient dense modules of asymmetric convolution for real-time semantic segmentation," in *Proceedings of the ACM Multimedia Asia*, 2019, pp. 1–6.

[108] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[109] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.

[110] Z. Li, Y. Zhang, and S. Arora, "Why are convolutional nets more sample-efficient than fully-connected nets?" *arXiv preprint arXiv:2010.08515*, 2020.

[111] R. Desislavov, F. Martínez-Plumed, and J. Hernández-Orallo, "Trends in ai inference energy consumption: Beyond the performance-vs-parameter laws of deep learning," *Sustainable Computing: Informatics and Systems*, vol. 38, p. 100857, 2023.

[112] R. Tang, A. Adhikari, and J. Lin, "Flops as a direct optimization objective for learning sparse neural networks," *arXiv preprint arXiv:1811.03060*, 2018.

[113] V. Sovrasov. (2023) ptflops: a flops counting tool for neural networks in pytorch framework. [Online]. Available: https://github.com/sovrasov/flops-counter.pytorch

[114] L. Migliorelli, S. Moccia, R. Pietrini, V. P. Carnielli, and E. Frontoni, "The babypose dataset," *Data in Brief*, vol. 33, p. 106329, 2020.

[115] B. Fallang, O. D. Saugstad, J. Grøgaard, and M. Hadders-Algra, "Kinematic quality of reaching movements in preterm infants," *Pediatric Research*, vol. 53, no. 5, p. 836, 2003.

[116] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.