Expert Systems With Applications 238 (2024) 122220

Contents lists available at ScienceDirect

Expert Systems With Applications

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

A new framework for deep learning video based Human Action Recognition on the edge

Antonio Carlos Cob-Parro *, Cristina Losada-Gutiérrez, Marta Marrón-Romera, Alfredo Gardel-Vicente, Ignacio Bravo-Muñoz

University of Alcalá, Department of Electronics, Ctra. Madrid-Barcelona, km. 33600, 28805, Alcalá de Henares, Spain

ARTICLE INFO

Keywords:
Machine learning
Embedded systems
Video surveillance
MobileNetV2-SSD
VPU
Edge computing
Deep learning

ABSTRACT

Nowadays, video surveillance systems are commonly found in most public and private spaces. These systems typically consist of a network of cameras that feed into a central node. However, the processing aspect is evolving towards distributed approaches, leveraging edge-computing. These distributed systems are capable of effectively addressing the detection of people or events at each individual node. Most of these systems, rely on the use of deep-learning and segmentation algorithms which enable them to achieve high performance, but usually with a significant computational cost, hindering real-time execution. This paper presents an approach for people detection and action recognition in the wild, optimized for running on the edge, and that is able to work in real-time, in an embedded platform. Human Action Recognition (HAR) is performed by using a Recurrent Neural Network (RNN), specifically a Long Short-Term Memory (LSTM). The input to the LSTM is an ad-hoc, lightweight feature vector obtained from the bounding box of each detected person in the video surveillance image. The resulting system is highly portable and easily scalable, providing a powerful tool for real-world video surveillance applications (in the wild and real-time action recognition). The proposal has been exhaustively evaluated and compared against other state-of-the-art (SOTA) proposals in five datasets, including four widely used (KTH, WEIZMAN, WVU, IXMAX) and a novel one (GBA) recorded in the wild, that includes several people performing different actions simultaneously. The obtained results validate the proposal, since it achieves SOTA accuracy within a much more complicated video surveillance real scenario, and using a lightweight embedded hardware.

1. Introduction

Nowadays, the fast developing of both hardware and software technologies, has caused that sciences such as Artificial Intelligence (AI) have evolved faster. This has allowed the field of computer science to propose interesting solutions in different areas (Pouyanfar et al., 2018; Shinde & Shah, 2018), such as banking management (Aziz & Dowling, 2019; Lee & Shin, 2020), healthcare (Ali et al., 2020; Castiglioni et al., 2021; Esteva et al., 2019; Hinton, 2018; Zhou et al., 2020), or computer vision (Cob-Parro, Losada-Gutiérrez, Marrón-Romera, Gardel-Vicente, & Bravo-Muñoz, 2021; Howard et al., 2017; Wu, Lv, Jiang, & Song, 2020).

Computer vision research has been growing in the last decade (Feichtenhofer, Pinz, & Wildes, 2017). It attempts to provide software solutions through the use of images. In this context, in recent years computer vision researchers have used AI to give solutions to more complex problems such as people detection, where recognition solutions are given for RGB or depth images as shown in the works (Fuentes-Jimenez et al., 2020; Spinello & Arras, 2011) or pattern recognition as explained in the work (Suri, 2000). Another very active area within computer vision is Human Action Recognition (HAR), which focuses on detecting of patterns for people in motion. For this purpose, spatio-temporal analysis techniques are used to identify actions (Kong & Fu, 2022).

HAR has been one of the most studied areas in computer vision in recent years (Pareek & Thakkar, 2021). Taking into account the definition given in Chaquet, Carmona, and Fernández-Caballero (2013), actions are considered to be human activities that can be recognized with less than two seconds of recording, such as jumping, running, sitting, falling, walking, clapping, etc. In the first works related to HAR, RGB video streams were analyzed using different techniques for action classification, being most of these works based on the extraction of ad-hoc feature descriptors (Baptista-Ríos, Martínez-García, Losada-Gutiérrez, & Marrón-Romera, 2016a; Klaser, Marszałek, & Schmid,

* Corresponding author.
E-mail addresses: antonio.cob@edu.uah.es (A.C. Cob-Parro), cristina.losada@uah.es (C. Losada-Gutiérrez), marta.marron@uah.es (M. Marrón-Romera), alfredo.gardel@uah.es (A. Gardel-Vicente), ignacio.bravo@uah.es (I. Bravo-Muñoz).

https://doi.org/10.1016/j.eswa.2023.122220
Received 13 November 2022; Received in revised form 15 October 2023; Accepted 17 October 2023
Available online 24 October 2023
0957-4174/© 2023 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

2008; Oreifej & Liu, 2013; Sulong & Mohammedali, 2014), followed by a classifier. These proposals obtain good results with small datasets, but their performance decreases when the dataset size increases.

With the improvement of technology, the techniques used for classifying actions have gradually migrated towards models based on Deep Learning (DL), which generally extract the features and perform the classification in the network itself. DL based systems are not only used in classification, but also in regression as in Atlam, Torkey, El-Fishawy, and Salem (2021) in which patients with COVID-19 are selected with the best chance of survival and predict the most acute symptoms (characteristics) that affect the probability of survival.

The strength of models built on layered architectures is that they specialize in extracting the most appropriated features to classify the input through training. This training process is usually lengthy and requires extensive and well-organized datasets. Furthermore, despite their excellent results, these systems suffer from the drawback of requiring a high computational cost. Larger network complexity yields superior results but incurs in higher computational overhead.

Within the models that use DL based architectures, there are different types, such as Convolutional Neural Networks (CNNs) (Gupta, Nunavath, & Roy, 2019; Kumaran, Vadivel, & Kumar, 2018; Liu, Campbell, & Guo, 2017; Zhou, Paiement, & Mirmehdi, 2017), primarily used in the field of artificial vision and Recurrent Neural Networks (RNNs) (Selvin, Vinayakumar, Gopalakrishnan, Menon, & Soman, 2017; Wang et al., 2016; Yin, Kann, Yu, & Schütze, 2017) used for the detection and prediction of time series. The latter include not only the classical RNNs, simple neural networks with feedback, but also Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTMs), systems that include the concept of memory.

In this context, in this paper it is proposed a new approach for HAR based on LSTM (Yu, Si, Hu, & Zhang, 2019) and edge computing. To do that, it has been defined a novel light feature vector that is classified using an LSTM, making the model able to accurately classify actions in the wild, with a much lower computational cost than a standard DL model. The proposal includes three different stages. Initially, there is a people detector. It has been chosen a MobileNetV2-SSD (Howard et al., 2017) due to its balance between accuracy and computational cost, however, in this stage, any people detection algorithm capable of extracting Bounding Boxes (BBs) around each identified person in the video frames can be applied. Subsequently, this information is processed to create a ultralight feature vector. These vectors are characterized by requiring a lower computational cost than other alternatives such as optical flow or a position skeletons, for both extraction and inference. These vectors include features related to the spatial position and size of BBs that enhance its capabilities. On the other hand, the proposed architecture for HAR, which includes LSTM, defines the temporal component necessary for HAR.

The depicted architecture is proposed as a modular and scalable system, being able to operate in real-time in embedded systems. Additionally, the system has proven to work well in different environments, since it has been tested with up to five different datasets, obtaining an accuracy in the results comparable to those of the related state-of-the-art (SOTA) works.

Thus, the main contributions of this work are listed below:

1. It is proposed a novel modular architecture for HAR, consisting of a person detector and an action classification. Furthermore, it has been defined an ad-hoc, light feature vector from the people detector output that is then classified using a LSTM based neural network. This allows obtaining low inference times, even in embedded systems, for real-time execution with results comparable to other SOTA proposals.
2. The action detector layers have been carefully defined to reduce the computational cost and optimize the results. In addition, the people detector has been optimized using the OpenVino framework (*OpenVino VPU architectures*, 2023) and integrated with dedicated hardware designed specifically for computer vision such as Visual Processing Units (VPUs) (Intel, 2020). This allows the system to run in an embedded processor such as an UpSquared V2 (UPS2) (Intel, 2022b), which works at 30 Frames Per Second (FPS) and can be powered by portable batteries.
3. Five different datasets have been used to validate the proposal in distinct environments and with diverse people actions. Four of the datasets are publicly available and widely used in HAR works: WVU (Kulathumani, 2011), IXMAS (EPFL, 2006), KTH (KTH, 2004) and WEIZMANN (Gorelick, Blank, Shechtman, Irani, & Basri, 2007). Furthermore, in this work, it is presented a new dataset named GEINTRA Behavior Analysis (GBA) (GEINTRA, 2022a, 2022b) that has been recorded at the Polytechnic School of the University of Alcalá. This dataset is characterized by including sequences in the wild, with real environments and people acting spontaneously. Besides, GBA includes both individual and group actions. This dataset has been made available to the scientific community (GEINTRA, 2022a).
4. The proposal has been evaluated by obtaining metrics for both recognition performance and computational cost on different devices. This evaluation confirms the system's ability to operate in real-time, obtaining results that validate the proposal accuracy.

To summarize, we propose a novel human action detection system optimized to work in embedded systems in the wild. It is a modular system intended to be flexible in software technology allowing the use of new technologies/developments. The proposal first two modules generate an ultralight vector with the most salient features of the BBs. The last one is based on LSTMs that collects the BB feature vectors, and predicts the action performed for each individual. As mentioned above, the system is designed and optimized to work in real-time ($> 30FPS$) and with accuracy levels similar to those in SOTA. Additionally, a particularly focused on actions in the wild, GBA dataset, is also presented and validates the proposal. This allows testing its performance in complex situations where overlaps, brightness and multiple individuals simultaneously carry out different actions.

Regarding the proposal novelty, it is worth to highlight that the main contributions are the original light feature vector described in this work, capable of extracting information about the movement of each detected person in the scene, and the architecture based on LSTMs for HAR, that has been carefully designed and train to detect human actions in different contexts in the wild.

The rest of the paper is organized as follows. Section 2 presents a study of the main previous works related to HAR, as well as different edge computing hardware and software tools used in the video surveillance research area. Then, Section 3 describes the developed algorithms and hardware implementation. After that, Section 4 presents and analyzes the main experimental results performed to validate the proposal. Finally, in Section 5, conclusions and future work are presented.

2. Related works

As mentioned in the introduction, one of the most extensively studied areas in the field of computer vision is HAR. In the beginning, there was a lot of disparity between the concepts of action, activity, behavior, and that is why works like (Ahad, 2011) explains the differences between them based on the time and complexity of the action execution. HAR using RGB images started with the analysis of hand-crafted features (Bregonzio, Gong, & Xiang, 2009; Gorelick et al., 2007; Laptev, Marszalek, Schmid, & Rozenfeld, 2008; Sadanand & Corso, 2012), extracting them through image processing and using machine learning based models to perform classification. For this purpose, systems such as the one proposed by Kong et al. (2019), Tian, Zhou, Zhao, Wei, and Fei (2013), Zeng and Ma (2010) using Histogram of Oriented Gradients (HOG) and filtering with a Support

Vector Machine (SVM) classification model were used, which allowed performing object detection tasks with satisfactory results. However, regarding action recognition, the results were not that remarkable. As technology advanced, innovative solutions were proposed, such as the use of DL (Dargan, Kumar, Ayyagari, & Kumar, 2020; Singh, Ahuja, Kumar, Kumar, & Sachdeva, 2021; Wang et al., 2022) based systems, which extract image features automatically. This type of neural networks are trained with adequate datasets to adjust a set of filters for detecting the desired features (Agatonovic-Kustrin & Beresford, 2000; Han, Kim, Kim, & Youn, 2018).

An example of this type of networks were CNNs, architectures based on convolutional layers that work as filters (adjusted during training) allowing the automatic extraction of features (Alzubaidi et al., 2021; Bhatt et al., 2021). These architectures achieved better results compared to the first mentioned models in which feature extraction was performed manually, especially in detection (Gayathri, Gopi, & Palanisamy, 2021; Hedjazi, Kourbane, & Genc, 2017; Mete & Ensari, 2019). Furthermore, they can also have other applications, such as encoder and decoder for human segmentation in video surveillance systems, as demonstrated in Gruosso, Capece, and Erra (2021).

These CNN-based systems have two main problems regarding HAR (Cho & Yoon, 2018; Ragab, Abdulkadir, & Aziz, 2020). First, most of them require an extremely high computational cost, that is, they need hardware devices such as GPUs (Körez & Barışçı, 2019; László, Szolgay, & Nagy, 2012; Potluri, Fasih, Vutukuru, Al Machot, & Kyamakya, 2011) to be able to operate in real-time. Despite this handicap, there are works that try to solve this situation by using edge technology (Cob-Parro et al., 2021; Rivas-Gomez, Pena, Moloney, Laure, & Markidis, 2018), employing Vision Processing Units (VPU) and specialized frameworks to reduce the computational cost. Secondly, CNNs do not have temporal context, which means that these architectures analyze frame by frame but do not consider the following or previous frames, and as mentioned above, actions have a temporal and spatial component. For adding the temporal component, many works add another dimension to the system input by adding depth such as (Das, Koperski, Bremond, & Francesca, 2017; Xia, Chen, & Aggarwal, 2012; Zhang et al., 2019), or supplementary technologies such as those in Berlin and John (2020), Kumar and John (2016), Tanberk, Kilimci, Tükel, Uysal, and Akyokuş (2020), which used optical flow for HAR from RGB images, since they embed the temporal dependence needed. However, most of these works are still limited by that temporal component.

To incorporate the temporal component to the neural networks, the use of RNNs (Almiani, AbuGhazleh, Al-Rahayfeh, Atiewi, & Razaque, 2020; Hibat-Allah, Ganahl, Hayward, Melko, & Carrasquilla, 2020; Li, Li, Cook, Zhu, & Gao, 2018) is proposed. Within the RNNs, there are different types of architectures depending on their complexity. Classical RNNs are the simplest, having output feedback in the neurons. Other more recent types of architecture with greater temporal capacity are the Gated Recurrent Unit (GRU) (Ullah et al., 2021) and the LSTMs (Sun, Xu, & Liu, 2021), which add the concept of memory to the RNNs, able to predict time series with limited periods. The most pioneering architectures in the signal analysis are those known as transforms (Mazzia, Angarano, Salvetti, Angelini, & Chiaberge, 2022), which have demonstrated far superior capabilities to GRU and LSTMs but with a higher computational cost.

This type of architectures have the ability to consider previous moments in a series to predict later ones (Raj et al., 2019). It can be used in different fields such as economic series (Naik & Mohan, 2019), where they are used to analyze stock market values or health (Chantamit-o pas & Goyal, 2018) where LSTM are used for stroke prediction. This is precisely the reason for combining these networks with CNNs, aiming to yield a response that encompasses both spatial and temporal responses (Ajao, Bhowmik, & Zargari, 2018; Khaki, Wang, & Archontoulis, 2020; Nasir, Khan, & Varlamis, 2021; Zhou, Li, & Liang, 2020), what allows improving the results against those works than only use CNNs. Transformers have also demonstrated a good performance for several computer vision applications, including action recognition (Liu et al., 2022), but also with a high computational cost, even for those alternatives adapted for mobile devices (Mehta & Rastegari, 2021; Wang, Zhang, Wang, & Yang, 2022).

Besides, these recurrent architectures only exacerbates one of the previously mentioned problems:the high computational cost. By merging CNN and LSTM (Li, Abdel-Aty, & Yuan, 2020; Xia, Huang, & Wang, 2020; Xu, Li, & Deng, 2015), systems require more powerful hardware to achieve the same execution speed. This, together with the silicon crisis (Frieske & Stieler, 2022), makes it necessary to find alternatives and develop architectures to obtain the same results but at a much lower cost.

For this reason, alternatives using CNN and LSTM that require a lower computational cost have been proposed. One approach to accomplishing this cost reduction involves minimizing the complexity of the models' inferences. To this end, one alternative is the use of smaller input datasets. Conventional CNNs used every single pixel of the input image in the process of information extraction, and it is worth mentioning that RGB images contain three times as much information as a gray-scale image, what results in a substantial computational burden during the inference execution. As a consequence, there are solutions that use skeleton models (Berlin & John, 2020; Luvizon, Picard, & Tabia, 2020; Xia et al., 2012; Yan, Xiong, & Lin, 2018; Zhang, Liu, Li, Chen, & Davis, 2017), in which a sequence of vectors representing the joints of an individual are fed into the network to predict the movements, instead of using whole images.

Another alternative is that the CNN itself generates an input vector smaller than the original image (Gu & Sung, 2021; Khan et al., 2021; Sun et al., 2021), that are then passed to the corresponding RNN or LSTM network that analyzes the temporal part of the image. These approaches provide good results, however, they face challenges in distinguishing the actions of individuals when multiple people are present within the image.

The systems based on DL methods usually provide better accuracy that those based on classical techniques, however these DL approaches require much higher computational costs. The hefty computational demands of these methodologies render them unfit for deployment in embedded systems for real-world applications, since they require very powerful hardware for real-time execution. It is, therefore necessary to find a balance between the architecture size and achieved results performance. In this context, the use of low-cost specialized hardware such as VPUs, the optimization of these architectures and the input data preprocessing can improve the computational efficiency to run DL models in real-time on the edge.

Consequently, after the analysis of the SOTA, this paper proposes a system to extract a set of characteristics and translate them into an ultralight vector. This is then processed by an architecture based on LSTMs to obtain the temporal component capable to retrieve the individuals' actions. Thus, a model capable of recognizing human actions in a robust way is proposed, obtaining results that are comparable to other works in the SOTA, but with a lower computational cost what allows its implementation on the edge.

3. Proposed HAR architecture

As explained in the introduction, this paper presents a robust system, based on DL models, to detect human actions. In addition, the proposal has a reduced computational burden, thereby enabling real-time execution in embedded systems while maintaining an accuracy comparable to that of SOTA.

This section explains each component that constitutes the proposal's architectural framework, shown in Fig. 1. As it can be seen, it includes a first module for people detection and tracking. Then, for each detected person, the HAR algorithm is executed, which includes: a preprocessing of the input data, a DL architecture based on LSTMs for extracting features, that are then classified with a Deep Neural Network (DNN).

Below, each of the modules included in Fig. 1 are explained in detail.

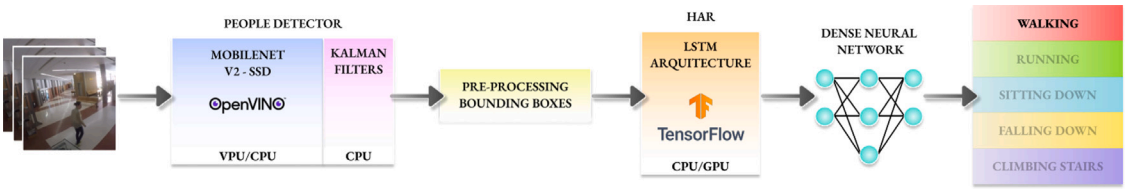

Fig. 1. General architecture of the proposed system for HAR.

3.1. People detection and tracking

As it is shown in Fig. 1, the first module is the people detector, which comprises two different blocks: a CNN, which is a MobileNetV2-SSD, and a bank of Kalman filters.

The reason for using the MobileNetV2-SSD is that it attains a balance between computational cost and accuracy. Different SOTA architectures for people detection have been evaluated and compared, each of them trained using the COCO dataset (Tsung-Yi Lin et al., 2015).

The experiments have been conducted using architectures that are valid and specifically tailored for execution on the VPU provided by the UPS2. A comparative analysis of the different architectures has been conducted, taking into consideration their computational efficiency and cost. It is important to note that the OpenVino framework has been utilized for coding such architectures on the VPU. The official OpenVino documentation (*OpenVino VPU architectures*, 2023) provides a comprehensive list of architectures that have been optimized for execution on a VPU.

To facilitate real-time execution on the embedded platform, there have been prioritized those architectures that demonstrate minimal computational cost while achieving enough accuracy: a mean average precision (mAp) with the COCO dataset above 25%.

Table 1 shows the obtained results for the different architectures analyzed using YOLO (Wang, Bochkovskiy, & Liao, 2022) and MobileNetV2-SSD. There are shown both, mAp and the number of FPS that can be processed within the UPS2.

Moreover, it is worth noting that the designed architecture is based on the optimization of hardware resources to allow the simultaneous analysis of actions by multiple individuals (it has to be able to detect them and to perform the HAR). Hence, it requires the detection process to be lightweight and maintain a stable level of precision when detecting people. Thus, computational cost has been prioritized to ensure that the detector does not become a bottleneck in an embedded system like the UPS2.

Analyzing the values in Table 1, it can be seen that YOLOv2 and YOLOv3 have higher mAp values for COCO, but they do not allow real-time operation ($FPS \geq 30$). Regarding YOLO tiny and MobileNet approaches, YOLOv3 tiny is the one with better mAp, but it cannot run in real-time in the embedded hardware. It can be seen that only YOLOv2 tiny, MobileNetV1-SSD and MobileNetV2-SSD are able to process at least 30 FPS, being the MobileNetV2-SSD the one with higher accuracy.

Thus, as it has been mentioned before, the MobileNetV2-SSD has been chosen due to its balance between accuracy and computational cost, that allows obtaining good results with reduced processing times.

The MobileNetV2-SSD is characterized for using depth-wise convolutional layers (Howard et al., 2017), which are more computationally efficient than standard ones, and for introducing the inverted residual blocks with bottlenecking, that reduce the computational cost.

This architecture has been trained not only with COCO (Lin et al., 2014) (used for comparison), but also with VOC07 (Everingham, Van Gool, Williams, Winn, & Zisserman, 2007) and VOC12 (Everingham, Van Gool, Williams, Winn, & Zisserman, 2012) datasets. It has also been optimized using the OpenVino framework, which allows seeding-up the inference process and fits the architecture to run on edge systems such as VPUs.

Table 1
Comparison of different architectures for object detection running in the VPU, in terms of mAp and computational cost.

Architecture	FPS	mAp with COCO (%)
YOLOv2	5.40	56.43
YOLOv3	2.31	67.72
YOLOv2 tiny	30.80	29.12
YOLOv3 tiny	20.19	39.70
MobileNetV1-SSD	139.14	19.32
MobileNetV2-SSD	142.81	30.75

Table 2
Characteristics of the layers in the SSD module.

Layer	Feature size	Aspect ratio	Number of default boxes
Conv_11	20 × 20	{1,2,1/2}	4
Conv_13	10 × 10	{1,2,1/2}	5
Conv_14	5 × 5	{1,2,1/2}	5
Conv_15	3 × 3	{1,2,1/2}	5
Conv_16	2 × 2	{1,2,1/2}	5
Conv_17	1 × 1	{1,2,1/2}	5

To improve people detection, the MobileNetV2-SSD has been trained with VOC 2007 + VOC 2012 and COCO. The choice of these datasets is driven by the substantial number of person instances available, as demonstrated in the work (Lin et al., 2014), which encompasses nearly one million instances of person detections across various scenarios and actions, providing significant generality to the system. Since these datasets include several classes (not only people), at the output of the DL architecture, all classes are filtered out except the person one. Furthermore, last layers of the architecture, related to the SSD module, have been configured to optimize people detection. Table 2 shows the characteristics of the used SSD modules.

In order to avoid multi-BBs in detection, the system also includes a Non Maximum Suppression (NMS) method. This allows filtering BBs with a low level of intersection over union. The NMS consists of two thresholds, the confidential threshold and the intersection over union one. The first allows discarding those BBs with scores lower than 0.25, whereas the second threshold indicates that the overlap between different BBs must be at least 45% to be accepted by the NMS.

The people detector includes two modules: the MobileNetV2, which extracts the features of the images, and the Single Shot Detection (SSD) (Liu et al., 2016), which manages the classification.

The loss function for the SSD module is comprised of two different loss functions: the confidence loss L_{conf} and the localization loss L_{loc}, as shown in Eq. (1), where N is the number of matched default boxes, and α is a parameter used to balance the L_{conf} and L_{loc}.

$$L(X, C_x, C_y, c, l, g) = \frac{1}{N}(L_{conf}(X, c) + \alpha L_{loc}(C_x, C_y, l, g)) \quad (1)$$

The localization loss function uses a smooth $L1$ loss to quantify the difference between the predicted box (l_i) and the ground-truth one (g_j), where X_{ij}^p represents if a particular default box (i) match with a particular ground truth box (j) of one class p (being 1 if the answer is positive and 0 if there is no match). It is defined for the values of the default box (d), comparing the center point offsets (C_x, C_y) with the default box d, width (W) and height (H) of the BB, as shown in Eq. (2):

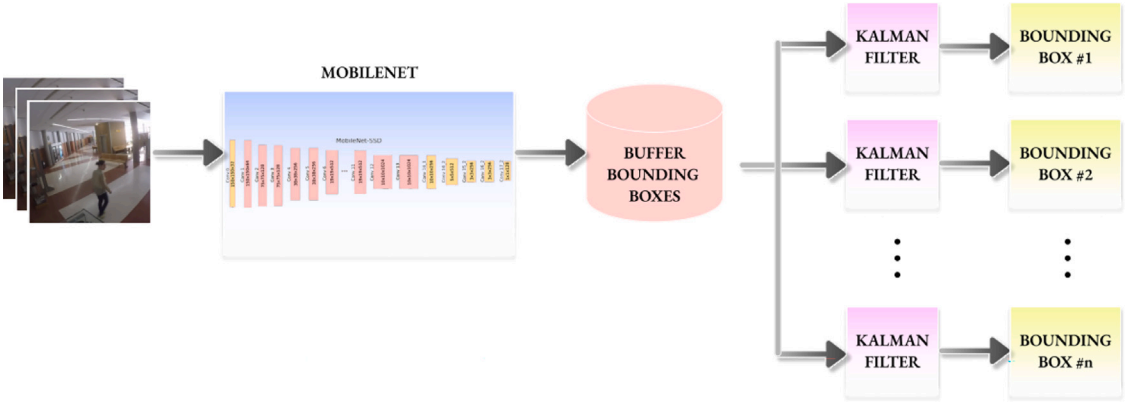

Fig. 2. General block diagram of the people detector module.

$$L_{loc}(X, C_x, C_y, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in C_x, C_y, W, H} X_{ij}^{p} smooth_{L1}\left(l_i^m - \hat{g}_j^m\right) \quad (2)$$

where $\hat{g}_j^{C_x} = \frac{(g_j^{C_x} - d_i^{C_x})}{d_i^W}$, $\hat{g}_j^{C_y} = \frac{(g_j^{C_y} - d_i^{C_y})}{d_i^H}$, $\hat{g}_j^W = log(\frac{g_j^W}{d_i^W})$ and $\hat{g}_j^H = log(\frac{g_j^H}{d_i^H})$.

Regarding the confidence loss, it is used when there are multiple classes. Thus, the confidence loss L_{conf} represents the loss of making a class prediction. For every positive match prediction, this loss is penalized according to the confidence score of the corresponding class, whereas for negative match predictions, the loss is penalized according to the confidence score of the class "0" (corresponding to no object is detected). The L_{conf} expression is shown in Eq. (3), where c defines the match confidence

$$L_{conf}(X, c) = -\sum_{i,j=1}^{N} X_{ij}^{p} log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \quad / \quad \hat{c}_i^p = \frac{exp(c_i^p)}{\sum_{\forall p} exp(c_i^p)} \quad (3)$$

In order to enable the system to run robustly in real environments, where multiple individuals are executing different actions simultaneously, a tracking stage is necessary to follow each detected person along a sequence of images. To do that, it is included a set of Kalman filters (Welch, Bishop, et al., 1995) (one for each detected person) similarly to Cob-Parro et al. (2021).

Incorporating the tracking stage also enhances the detection results. Given the nature of the Kalman filter, if the detected person BB disappears (which can occur due to occlusion or a false negative error from the detector) the system is still capable of maintaining a prediction of the BB's position within the scene over several frames, until a new detection is done. This improves the robustness of the people detector against occlusions, as it was demonstrated in Cob-Parro et al. (2021).

Since the number of tracked BB is variable, it is necessary to incorporate an association stage to relate the obtained BBs with the Kalman filters predictions. The association is carried out in a recursive loop, with a Nearest Neighbors (NN) (Konstantinova, Udvarev, & Semerdjiev, 2003) based approach. It worths mentioning that this tracking system further improves the robustness of the proposal against partial or total people occlusions. The general block diagram of the people detector is shown in Fig. 2.

As it has been stated before, due to the modular structure of the proposal, the MobileNetV2-SSD can be easily replaced by other people detection approaches such as classical ones based on HOG descriptors and a SVM classifier (Kong et al., 2019) or a model with higher computational load such as YOLOv7 (Wang, Bochkovskiy, & Liao, 2022), depending on the available hardware resources and the performance requirements. This is because only the information provided by the BB is needed in the following stages of the algorithm. As a result, the model described in this paper can be effortlessly customized to accommodate any other people detector.

3.2. Feature extraction

The next module in the architecture is the feature extraction one, that obtains a light, ad-hoc, feature vector. This module generates a descriptor with 11 components for each detected person, from the BB generated by the people detector.

The vectors obtained from multiple consecutive images within a sequence need to be stacked before introducing them in the action recognition neural network, to insert the temporal component that defines an action. The length of the analyzed sequence must be enough to detect the action, but if it is too long, the computation time increases and it is possible that the sequence includes more than one action, worsening the results. After an experimental analysis, it has been determined that 0.5s is enough to make a decision on the action classification in the video surveillance context. Therefore, the variable L is defined, indicating the number of vectors introduced in the LSTMs architecture for action recognition as shown in Eq. (4).

$$L = FPS/2\ (frames) \quad (4)$$

Fig. 3 shows the different measurements and coordinates that compose the elements of the proposed ultralight vector, where:

- dim_x and dim_y are the image resolution (width and height, respectively).
- U_1 (x-axis) and V_1 (y-axis) correspond to the coordinates of the upper left corner of the BB, and U_2 and V_2 to the lower right corner.
- C_u and C_v are the coordinates of the BB centroid.
- W and H are the width and height of the BB respectively.

It is important to note that, all these variables are in pixels, and they are associated with a specific frame t, thus, all of them are time-dependent except the image resolution dim_x and dim_y.

The elements that comprise the feature vector are described in the following. In order for the LSTM architecture to work optimally, the values are normalized between −1 and 1.

- V_{n_norm} and U_{n_norm}: correspond to the normalized coordinates of both the upper left corner (with $n = 1$) and the lower right corner (with $n = 2$). Eq. (5)shows how the values of U_1, V_1, U_2 and V_2 are normalized.

$$U_{n\_norm}(t) = \frac{2U_n(t) - dim_x}{dim_x} \quad V_{n\_norm}(t) = \frac{2V_n(t) - dim_y}{dim_y}, \quad n = 1, 2. \quad (5)$$

- C_{u_norm}, C_{v_norm}: are the normalized values of the centroids of the BB, obtained as shown in Eq. (6).

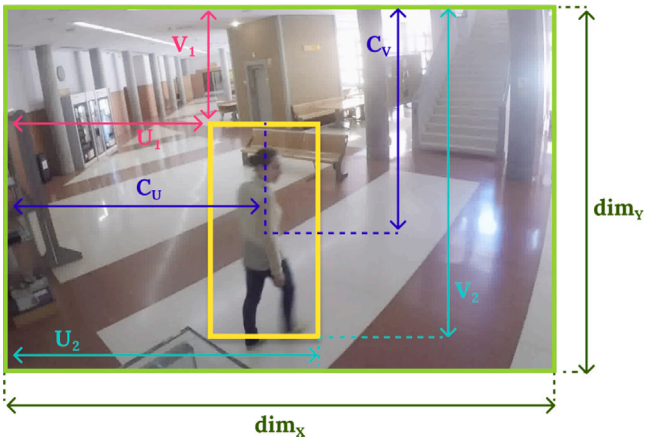

Fig. 3. Elements of the proposed ultralight vector in a sample frame.

Table 3
Recurrent neural networks comparison in terms of performance and computational cost.

Layer	RNNs	GRU	LSTMs
Accuracy	87.32	92.13	99.31
Recall	89.15	93.03	99.09
F1-Score	90.03	91.45	99.06
Time(ms)	0.123	0.241	0.298

$$C_{u\_norm}(t) = \frac{U_2(t) + U_1(t) - dim_x}{dim_x} \quad C_{v\_norm}(t) = \frac{V_2(t) + V_1(t) - dim_y}{dim_y} \tag{6}$$

- H_{norm}, W_{norm}: correspond to the normalized values of the dimensions (height and width respectively) of the BB, as shown in Eq. (7).

$$H_{norm}(t) = \frac{2H(t) - dim_y}{dim_y} \quad W_{norm}(t) = \frac{2W(t) - dim_x}{dim_x} \tag{7}$$

- RhW: defines the relationship between the height ($V_2 - V_1$) and the width ($U_2 - U_1$) of the BB, computed and normalized between −1 and 1 using the expressions in Eq. (8).

$$RhW(t) = \begin{cases} if \quad W(t) < H(t) \quad \rightarrow \quad \frac{W(t)-H(t)}{H(t)} \\ if \quad W(t) > H(t) \quad \rightarrow \quad \frac{H(t)-W(t)}{W(t)} \\ if \quad W(t) = H(t) \quad \rightarrow \quad 0 \end{cases} \tag{8}$$

- A_u, A_v: represent the direction and magnitude of the BB centroid movement between two consecutive frames. Thus, Eq. (9) defines in which direction the BB is moving, and can determine if and how much it moves.

$$A_u(t) = \frac{C_u(t) - C_u(t-1)}{2} \quad A_v(t) = \frac{C_v(t) - C_v(t-1)}{2} \tag{9}$$

After extracting the features, the obtained vectors are concatenated within a temporal window of size L, to add the temporal component to that set of vectors, obtaining a set of L ultralight vectors (see Eq. (10)). So, the concatenated feature vector, with dimensions $11 \times L$, is then fed to the HAR architecture explained in Section 3.3. This vector collects the fluctuations that BBs undergo during L consecutive frames in a sequence. These fluctuations are able to describe the action through the BB position in the image, its change in aspect ratio, and direction and magnitude of this variation. Our hypothesis is that this information about spatio-temporal changes in the BB allows the HAR system to effectively classify different actions.

$$\begin{pmatrix} U_{1\_norm}(t) & U_{1\_norm}(t+1) & \cdots & U_{1\_norm}(t+L-1) \\ V_{1\_norm}(t) & V_{1\_norm}(t+1) & \cdots & V_{1\_norm}(t+L-1) \\ U_{2\_norm}(t) & U_{2\_norm}(t+1) & \cdots & U_{2\_norm}(t+L-1) \\ V_{2\_norm}(t) & V_{2\_norm}(t+1) & \cdots & V_{2\_norm}(t+L-1) \\ C_{u\_norm}(t) & C_{u\_norm}(t+1) & \cdots & C_{u\_norm}(t+L-1) \\ C_{v\_norm}(t) & C_{v\_norm}(t+1) & \cdots & C_{v\_norm}(t+L-1) \\ H_{norm}(t) & H_{norm}(t+1) & \cdots & H_{norm}(t+L-1) \\ W_{norm}(t) & W_{norm}(t+1) & \cdots & W_{norm}(t+L-1) \\ RhW(t) & RhW(t+1) & \cdots & RhW(t+L-1) \\ A_u(t) & A_u(t+1) & \cdots & A_u(t+L-1) \\ A_v(t) & A_v(t+1) & \cdots & A_v(t+L-1) \end{pmatrix} \tag{10}$$

It is noteworthy that matrix in Eq. (10) is computed and processed to obtain the performed action for each detected person, so, as the number of people in the scene is larger, the computational cost increases. However, the proposal has been designed and optimized to take advantage of the parallel processing capacity of the VPU (or GPU if available). Furthermore, the light feature vector (with only $11 \times L$ elements) and the proposed LSTM architecture allows a low computational cost, even when several people are detected in the scene, as it will be shown in Section 4.2.

3.3. Human action recognition

As it has been explained before, the input for HAR is a matrix containing features for L consecutive images. Moreover, it is used an sliding window approach with a shift of one image, providing results (recognized action) for each window.

To perform the recognition task, RNNs have been employed. Within this category of networks, various architectures are available, with the most well-known being RNNs, GRUs and LSTMs. In terms of complexity, RNNs are the simplest, with lower computational costs. Regarding GRUs and LSTMs, although there are several differences between them, the primary distinction lies in the fact that LSTM incorporates three gateways for short-term memory, namely forget, output, and input gates. In contrast, GRU comprises only two gates, namely update and reset gates. Additionally, GRU architectures are less intricate, primarily due to the reduced number of gates compared to LSTM. As a result, LSTM architectures exhibit a more complex and sophisticated design, leading to increased computational costs. Nevertheless, Table 4 3 shows a comparison between the proposed architecture in Fig. 4 and various recurrent networks. The metrics were extracted using the UMN dataset (*Monitoring Human Activity, Robotics and Vision Laboratory, University of Minnesota, Department of Computer Science and Engineering*, 2023), while execution time was measured considering an scenario with only one detected person.

Table 3 demonstrates that the utilization of LSTMs outperforms other recurrent networks in terms of performance. The difference is in execution times: LSTMs take more than twice as long as RNNs. However, it is worth noting that the computational cost for LSTMS remains at a reasonable level real-time processing up to 0.3 ms, obtaining a favorable balance between computational cost and performance metrics.

The proposed architecture for the HAR stage is shown in Fig. 4. As depicted, the model consists of two LSTM layers. The key distinction between them is that the first layer operates unidirectionally, while the second one operates bidirectionally.

The first LSTM layer performs an initial temporal analysis to reorganize the values given by the feature vector. Then, a more complex study is carried out in the next layer with the bidirectional LSTM. After the LSTMs processing, there are four dense layers with a size of 200, 100, 50 and 20 neurons respectively. The chosen activation functions are *tanh*, due to the non-linear behavior of data in the network, and the input normalization.

In addition, a 20% dropout is performed between all the system layers to avoid over-fitting and streamlining the network. A batch normalization layer has also been included immediately after the dropout one. The designed network ends with a *softmax* layer to carry out the actions classification. The global process requires a total number of 3.7 mega FLOPs (floating point operations per second).

The following hyperparameters have been used to train the HAR model: *learning rate* of 1e−4, $\beta_1 = 0.999$, $\beta_2 = 0.9999$, $\epsilon = 1e-6$, a decay of 1e−5. Besides, the number of epochs has been 10,000 and the batch size, 120. These values have been obtained experimentally to increase accuracy while avoiding over-fitting.

Besides, it has been chosen the Adam optimizer, due to two main reasons. First, the system applies to LSTM networks and optimizes its loss function by adjusting the weights that compose it, aiming to minimize this function. On the other hand, the proposal must be deployed in an embedded system, requiring a minimal computational cost. Adam is a first-order gradient stochastic optimization algorithm. This makes Adam the best solution for the framework proposed, as opposed to others that require a higher computational cost or whose loss function optimization is not as accurate.

The input data used for the network training is fundamental for obtaining a good performance in action classification. In this work, initially, the videos were divided into segments, with a sliding window of length L with stride L, and fed into the model during training. This gave poor results comparing with the SOTA when using datasets recorded in the wild. One of the reasons is that some of the segments included more than one action, and a transition between actions instead of only one action during the whole sequence (Fig. 5).

To better test the proposed system, the different sequences in the dataset were conveniently split in order to include only one action for each segment of length L during training.

Fig. 6 shows the process of reorganization and reuse. The frames of a sequence containing an action are extracted from the whole sequence and divided in segments of L frames. In case the extracted sequence length is longer than L frames, some of them can be included in more than one set, as shown in the example of Fig. 6, by dynamically modifying the sliding window stride.

Regarding the loss function used for the HAR architecture, first, Eq. (11) shows the *softmax* function, that allows grouping all the predictions in a single vector with values between 0 and 1, being 1 the sum of all elements. In Eq. (11), s is the score, i defines the class, and NC is the number of classes minus one.

$$SF(s)_i = \frac{e^{s_i}}{\sum_j^{NC} e^{s_j}} \tag{11}$$

Then, the HAR architecture uses a loss function of the categorical cross-entropy type (Eq. (12)), being i the class, and g the ground-truth value defined with a one-hot encoding (NumFOCUS, 2022).

$$L_{CrossEntropy} = -\sum_{i=1}^{NC} g_i log(SF(s)_i) \tag{12}$$

4. Experimental results

In this section, the experimental results are presented, analyzing two critical aspects of the system: its performance and the computational cost. For this purpose, five different datasets are used: four of them widely used for HAR, and the other one, a novel dataset recorded and labeled for HAR in the wild. To measure the system's performance, the used metrics are: precision, recall and F1, which evaluate the system's behavior with different datasets and allow comparing the results with other SOTA works. Moreover, power consumption and computational cost are also analyzed to determine if the proposal can run in real-time on the edge.

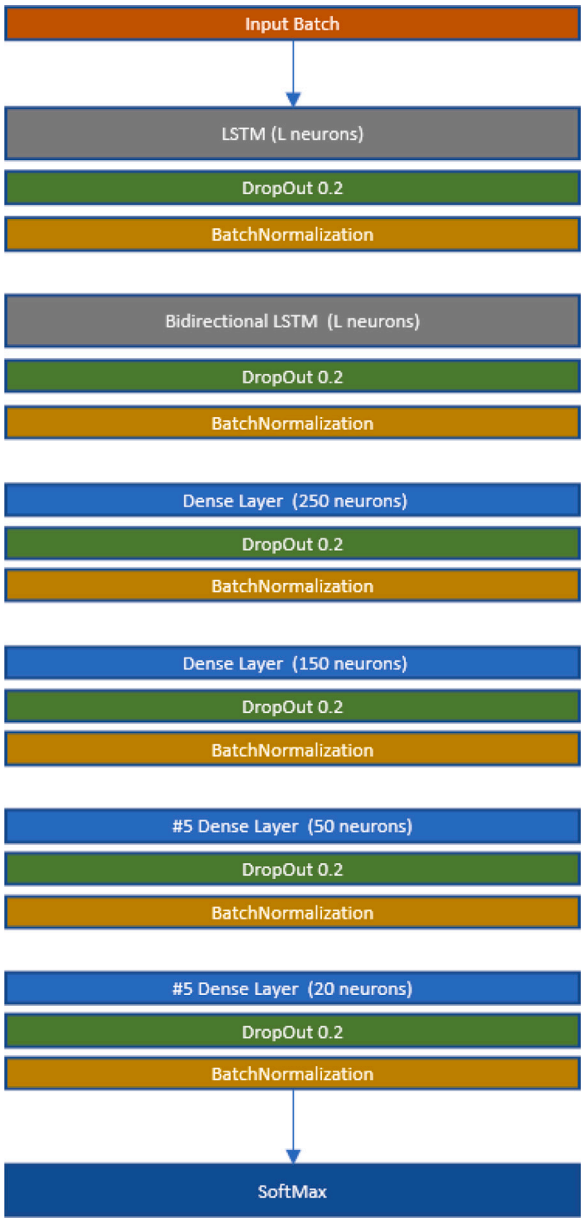

Fig. 4. HAR architecture layers.

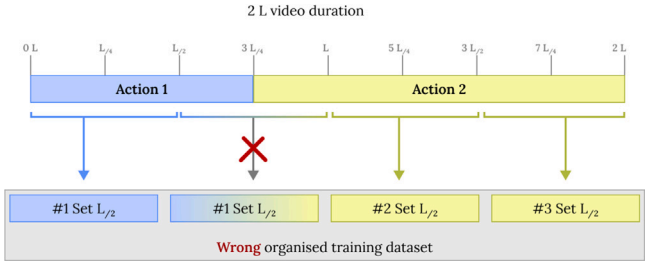

Fig. 5. Example of wrong training set organization.

4.1. Experimental setup

This section presents the setup used for the experimental evaluation of the proposed system. For this purpose, both the hardware and software setup, as well as the datasets used, are explained.

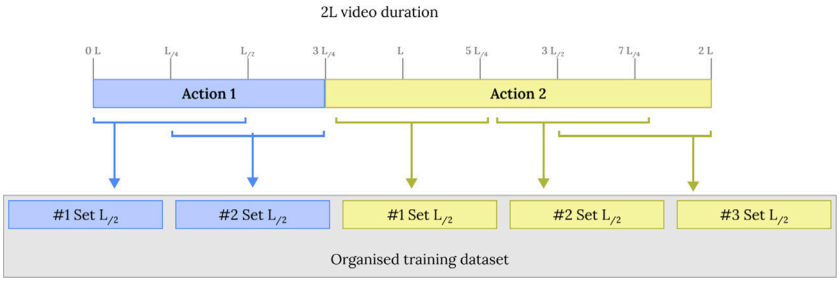

Fig. 6. Training set organization scheme.

4.1.1. Hardware and software setup

The results shown in this section have been obtained using a PC with an Intel© Core™ i7-9700K CPU @ 3.60 GHz × 8 (Intel, 2022a), a RAM memory of 32 GB (Gigabytes), a GPU NVidia GeForce RTX 2080 Ti (NVidia, 2021), and a ROM memory of 500 GB. Moreover, the chosen operating system is Ubuntu 20.04.3 LTS (Canonical, 2022).

Furthermore, the architecture has also been tested on a real embedded system, specifically in the UPS2, because the main objective is to design a model for HAR that is lightweight, robust and with an accuracy similar to the current SOTA for devices with reduced computational power. The hardware features of the UPS2 are the following: an Intel Atom x7-E3950 microprocessor, an 8 GB RAM memory, a 64 GB embedded MultiMediaCard (eMMC) ROM. It is noteworthy that the UPS2 includes an Intel Movidius Myriad-X VPU (Intel, 2020) DL module, that is a System-On-Chip (SoC) which optimizes AI inference models at a low computational cost.

As discussed in Section 3.3, the hyperparameters employed for training the HAR recognition architecture are as follows: *learning rate* of 1e−4, $\beta_1 = 0.999$, $\beta_2 = 0.9999$, $\epsilon = 1e{-}6$, a decay of 1e−5, with a total of 10,000 epochs and a batch size of 120. These values were determined experimentally to optimize accuracy while preventing over-fitting.

Below, in Section 4.1.2, the portion of the dataset used for both the training and testing segments is detailed. Additionally, a k-fold method is employed with the aim of obtaining more generalized results from the system for each dataset.

4.1.2. Datasets

For the study of HAR, there is a wide range of publicly available datasets encompassing numerous classes. Among the most renowned datasets they are UCF101 (Soomro, Zamir, & Shah, 2012) and Kinectics (Kay et al., 2017), which contain hundreds of classes. However, these datasets are not specifically designed for video surveillance systems where the camera remains fixed, and the recording angle is constant. Moreover, in the context of video surveillance, there exists a distinct set of actions that are clearly defined, such as walking, falling, running, jumping, etc. Therefore, it is crucial to focus on developing and evaluating action recognition models tailored explicitly for the unique characteristics and requirements of video surveillance environments. For this reason the experimental evaluation of the proposal has been carried out using five different datasets. Four of them widely used for action recognition: KTH (KTH, 2004), WEIZMANN (Gorelick et al., 2007), WVU (Kulathumani, 2011) and IXMAS (EPFL, 2006), whereas the other one is a novel dataset called GBA, that contains different realistic scenarios in the wild, including several people performing different actions simultaneously. The main characteristic of these datasets are explained below.

- **GBA dataset** (GEINTRA, 2022a): it has been recorded at the Polytechnic School of the University of Alcalá. The main characteristic of this dataset is that it has been recorded and labeled for HAR in the wild, so it includes several people performing different actions in the scene, that can change along a sequence, instead of just a person performing only one action during the whole sequence. A first version of the GBA dataset was presented in the IPIN2016 (Baptista-Ríos, Martínez-García, Losada-Gutiérrez, & Marrón-Romera, 2016b). This first version included 300 sequences with a duration between 15 s and 30 s. They were recorded at 50FPS with a resolution of 1280 × 720 pixels. This first version includes a total of 4 actions (run, walk, fall down, sit down) with an average of 74 sequences per class. The original GBA dataset has been extended adding a new action (stairs) and including new sequences for all the actions. Thus, there have been added 1450 sequences with a duration ranging from 15 s to 1 min and 53 s. In addition, the resolution of new scenes is 1920 × 1080 pixels at 50FPS. Fig. 7 shows examples of the current dataset actions after the update.
  All the sequences in the dataset are labeled including both, the BB for each person and the performed action. These labels are provided in two different formats: plain text and .xml.
  The rest of used datasets have been widely used in the literature for action recognition. All of them include sequences with just one person performing an action. In addition, none of them provide BBs for people location in the video frames.

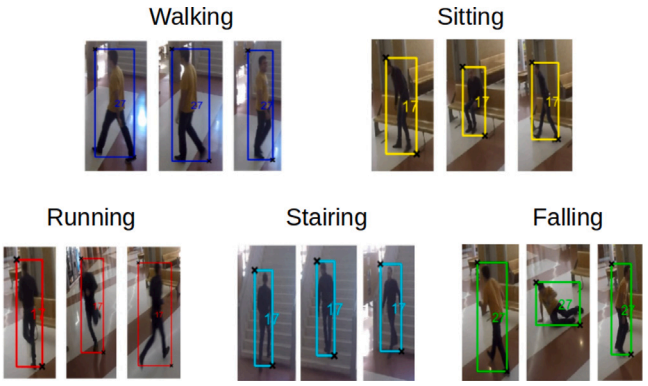

Fig. 7. Example of GBA classes.

- **KTH dataset** (KTH, 2004): consists of a set of videos including 6 classes (boxing, clapping, jogging, handshaking, running, walking), performed by more than 25 subjects in four different scenarios, three of them outdoors. An encoding of 25FPS is used with a total of 600 sequences. The videos have an average duration of 5s, with a total of 100 frames per video. The resolution used for the recordings is 160 × 120 pixels with one color channel (gray-scale). Fig. 8 shows some sample images corresponding to each class in KTH dataset.
- **WEIZMANN dataset** (Gorelick et al., 2007): is a dataset focused in HAR, consisting of a total of 90 videos divided into 10 classes (walking, running, jumping, galloping sideways, bending, one-hand waving, two-hands waving, jumping in place, jumping jack, skipping) that are performed by nine people, as displayed in Fig. 9. The videos have a resolution of 180 × 44 pixels at 25FPS.

Fig. 8. Examples of KTH classes (KTH, 2004).

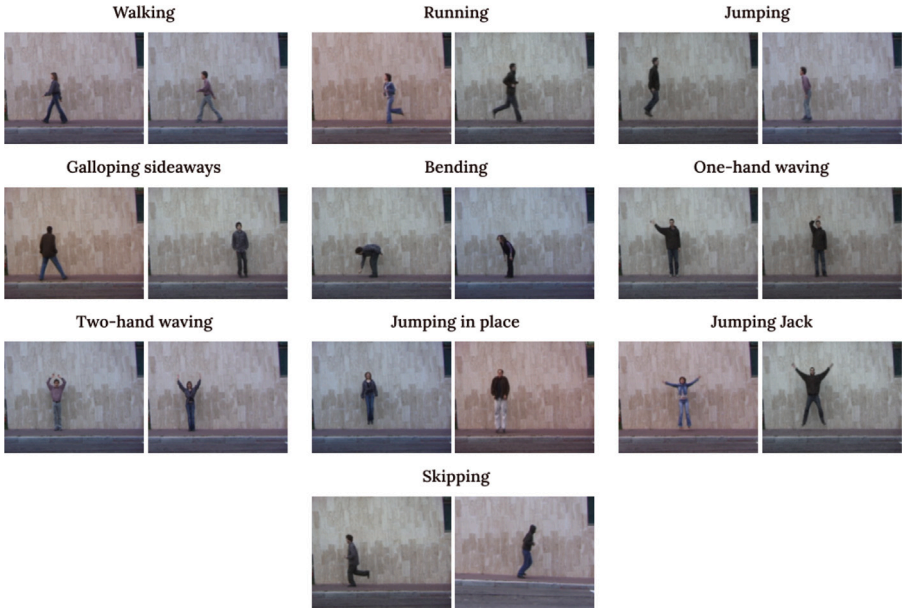

Fig. 9. Examples of WEIZMANN classes (Gorelick et al., 2007).

- **WVU dataset** (Kulathumani, 2011): is focused on action detection. It includes a set of 12 classes (one-hand waving, two-hand waving, bowling, jogging, clapping, picking up, nodding head, kicking, jumping jack, throwing, punching, standing still). This dataset employs different orientations to give diverse perspectives of the actions. A total of 8 cameras and 48 performers were used. 20FPS encoding is used, with a resolution of 640 × 480 pixels in RGB. There are a total of 200 videos. Fig. 10 shows some sample image corresponding to the classes from 2 cameras.
- **IXMAS dataset** (EPFL, 2006): is a dataset oriented to HAR, with a total of 12 classes (checking watch, crossing arms, scratching head, sitting down, getting up, turning around, waving, walking, punching, kicking, picking up and pointing), and interpreted by 11 actors in three different iterations, as shown in Fig. 11. The actors randomly choose their position to give more realism to the dataset. Cameras with different orientations are used to record the sequences. Recording encoding is 23FPS with a resolution of 390 × 291 pixels in gray-scale.

Table 4 includes a summary of the main characteristics for the validation datasets used in this work, and described earlier in this section. This table includes the number of classes, number of actors, number of sequences, image size and FPS.

4.2. Performance for action recognition

In this section, the behavior of the proposal is analyzed quantitatively. For this purpose, there have been extracted the accuracy, precision, recall and F1 metrics, as well as the confusion matrices for the different classes of the five analyzed datasets. Moreover, Section 4.4 presents the comparison of the obtained results with related SOTA works.

Table 4
Summary of the main characteristics of the used datasets.

Dataset	# classes	# actors	# seqs.	Size (pixels)	FPS
KTH	6	20	600	160 × 120	25
WEIZMANN	10	9	90	180 × 144	25
WVU	12	48	200	640 × 480	20
IXMAS	12	10	1148	390 × 291	23
GBA	5	17	1450	1920 × 1080	50

4.2.1. Results for KTH dataset

Fig. 12 shows the confusion matrix for the KTH dataset. To obtain these results, the available data have been divided, using the 80% of sequences for training and the other 20% for testing. Furthermore, to ensure the system generality, a k-fold of 5 has been applied. The results presented in the confusion matrix show that the accuracy is above 98% for all actions of the dataset. The system shows a higher confusion between running and jogging actions, as they are similar activities; even so, the error does not exceed 2%. The rest of actions have an accuracy higher than 98%. It is worth noting that the average values are above 99% in the performance metrics used.

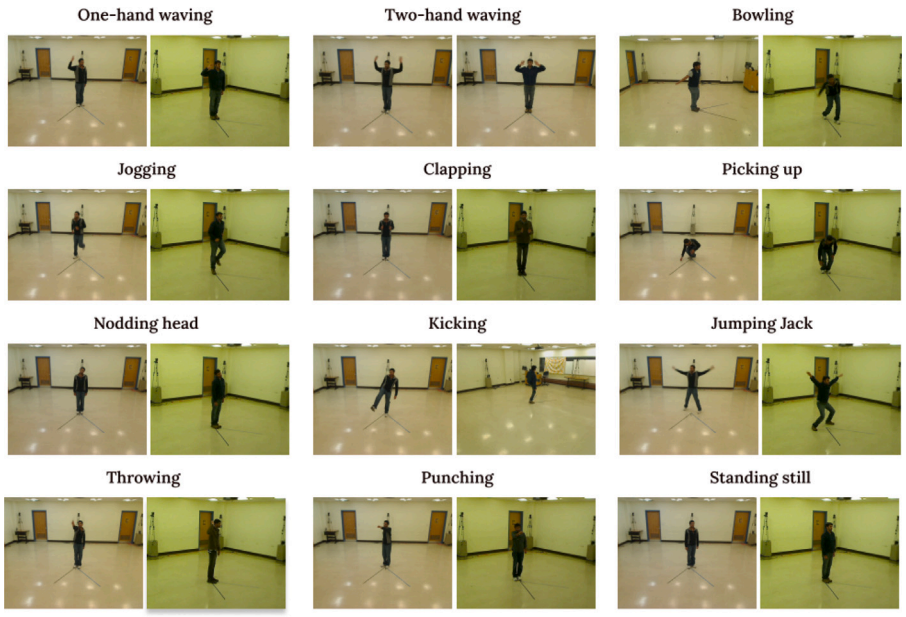

Fig. 10. Examples of WVU classes (Kulathumani, 2011).

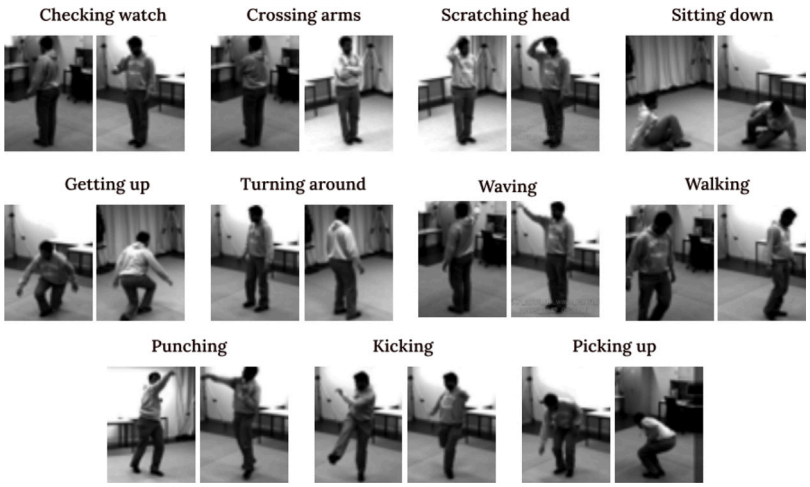

Fig. 11. Example of IXMAS classes (EPFL, 2006).

Table 5
Metrics values obtained for KTH dataset.

Action	Precision	Recall	F1
Boxing	99.84	100.00	99.61
Clapping	99.56	99.37	98.92
Handshaking	99.66	98.83	99.17
Jogging	98.54	99.04	98.93
Running	98.78	98.66	98.78
Walking	99.32	98.80	99.01
AVERAGE	99.28	99.11	99.06

Table 5 shows the values for each metric (Precision, Recall and F1-score) obtained for each class in the KTH dataset. The last row shows the average of all classes for each metric, being in all cases above 99%.

4.2.2. Results for WEIZMANN dataset

The next dataset analyzed is WEIZMANN. In this case, it has been used a 80% training and a 20% for testing, with a k-fold of 5. In Fig. 13, it is shown the confusion matrix corresponding to the results obtained using the WEIZMANN dataset. Similar to KTH, the most significant difficulty is found in the running and walking actions, where the highest error is 1.5%. The rest of classes are above 98%. It can be seen that the results are similar to those obtained for KTH, even with 5 more classes, which suggests that the system is scalable, thus able to work robustly regardless the number of classes.

Table 6 shows metrics values for each action included in WEIZMANN dataset. Additionally, in the table's last row, an average of the three metrics is added, being above 99% in any case.

4.2.3. Results for IXMAS dataset

At it has been explained, IXMAS has a total of 12 classes. This dataset is characterized by low resolution and gray-scale images, but even so, the detection network is able to obtain the BB correctly.

The network has been trained with a 70% of the data and tested with the other 30%. This is because there were more samples than in the previous cases for training. The dataset has several cameras recording the scenes, and all of them have been used for both the training and testing processes. In addition, a k-fold of 5 has been performed to obtain an average of training and test.

The obtained results are similar to the two previous datasets, with a precision over a 98% in all classes, as it is shown in Fig. 14. It

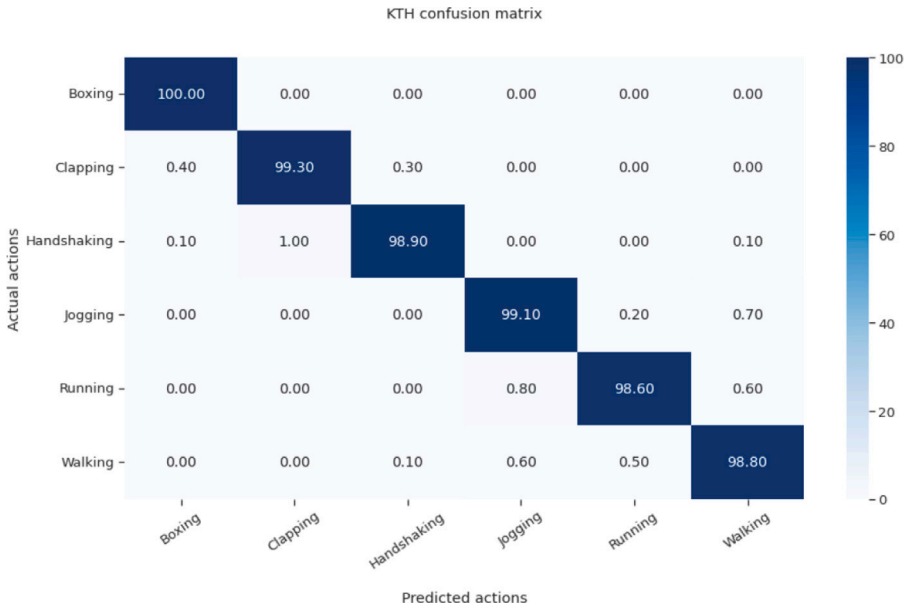

Fig. 12. Confusion matrix obtained for KTH dataset.

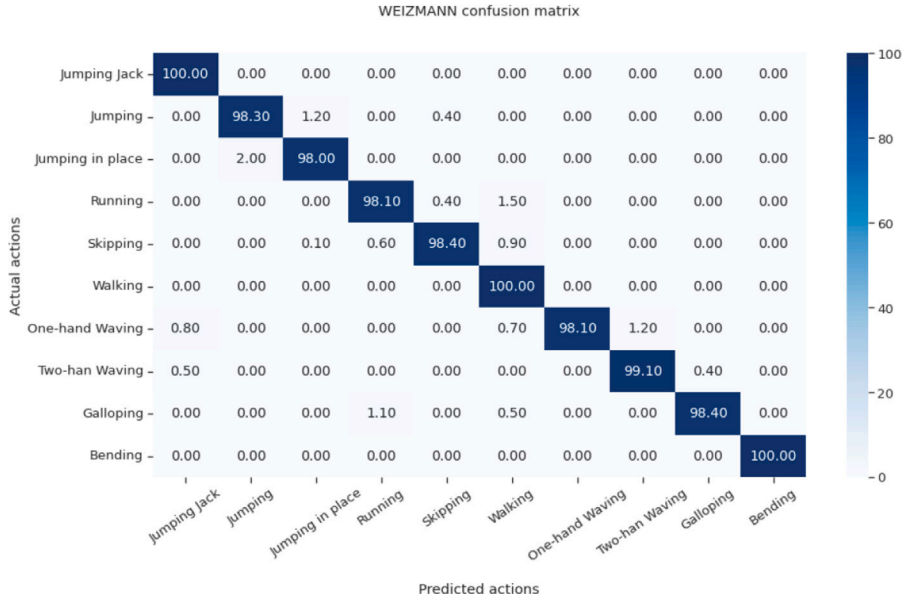

Fig. 13. Confusion matrix obtained for WEIZMANN dataset.

Table 6
Metrics values obtained for WEIZMANN dataset.

Actions	Precision	Recall	F1
Jumping Jack	99.10	100.00	99.54
Jumping	98.18	98.31	98.71
Jumping Place	99.56	98.07	99.44
Running	99.08	98.10	99.03
Skipping	98.23	98.43	98.86
Walking	99.57	100.0	99.06
One-hand Waving	99.86	98.10	98.13
Two-hand Waving	99.67	99.19	99.23
Galloping	99.05	98.38	99.84
Bending	99.15	100.00	99.31
AVERAGE	99.14	98.91	99.11

can be observed that this dataset comprises more classes compared to previous ones, however, the results remain stable. According to the confusion matrix, it can be seen that the highest confusion is produced between the pick-up and point classes, unlike the other datasets where the confusion was between walking and running classes.

Table 7 shows the values obtained for the different analyzed metrics in each action. It can be seen that for all classes, the results are above 98%. The table's last row shows the average of all actions being above 98% in the three metrics.

4.2.4. Results for WVU dataset

There has also been analyzed the WVU dataset, with 12 actions. As it has been explained before, this dataset is recorded with 8 cameras from different positions. That is why this dataset is the most challenging one, being the 80% of available sequences used for training and the remaining 20% for testing. In addition, a k-fold of 5 has been performed to obtain an average of training and test. However, despite the difficulty of the dataset, the system is able to correctly relate the movements generated by BBs to the performed actions. Confusion matrix, in Fig. 15,

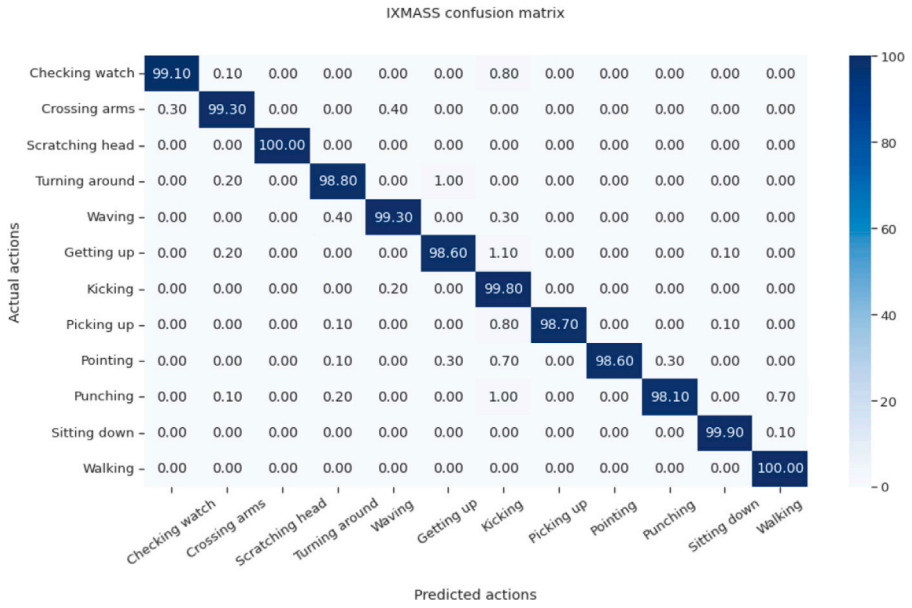

Fig. 14. Confusion matrix obtained for IXMAS dataset.

Table 7
Metric values obtained for IXMAS dataset.

Actions	Precision	Recall	F1
Checking watch	98.54	99.14	98.55
Crossing arms	99.26	99.21	99.01
Scratching head	99.01	100.0	99.50
Turning around	98.39	98.76	98.69
Waving	98.67	99.39	98.61
Getting up	98.23	98.73	98.95
Kicking	98.91	99.67	98.68
Picking up	99.15	98.70	99.22
Pointing	99.09	98.60	98.43
Punching	98.75	98.18	97.91
Sitting down	99.03	99.82	98.90
Walking	98.87	100.0	99.27
AVERAGE	98.85	99.18	98.80

Table 8
Metrics values obtained for WVU dataset.

Actions	Precision	Recall	F1
Standing still	99.81	99.10	99.42
Picking up	97.36	99.12	98.25
Nodding head	100.0	99.83	99.96
Clapping	99.64	99.11	99.52
One-hand waving	99.48	99.28	99.79
Two-hand waving	99.73	100.00	99.81
Jumping jack	99.30	100.00	99.16
Kicking	99.03	99.82	99.44
Throwing	98.98	98.29	98.13
Bowling	99.12	98.61	99.20
Jogging	99.29	99.84	99.55
Punching	99.55	99.85	99.70
AVERAGE	99.27	99.40	99.32

is similar to the other analyzed datasets, with more difficulties in the actions corresponding to jogging and walking. This is due to the similarity of BB movements for the two activities. Even so, it can be observed that the results are above 98% for all classes.

Table 8 shows each metric extracted for each class in the dataset. Again, last row shows all classes average for each metric, being in all cases above 99%.

Table 9
Metrics values for GBA dataset.

Actions	Precision	Recall	F1
Walking	97.68	97.02	97.35
Running	85.63	91.06	88.26
Sitting down	99.29	96.99	98.12
Falling down	99.93	98.96	99.44
Stairing	100.00	98.98	99.49
AVERAGE	96.34	96.56	96.44

4.2.5. Results for GBA dataset

Finally, the GBA dataset, developed by the University of Alcalá, has been used. Unlike the other datasets, this one shows more realistic situations where several people move through a real scenario. In this dataset, there are five different actions: walking, running, falling and climbing stairs. It is noteworthy that the dataset is unbalanced concerning the number of labels, including much more sequences for walking than for other actions, such as falling or climbing stairs. Therefore, to avoid over-fitting with the "walking" class, the number of walking sequences in training has been limited to the same number as the label with the least available samples. This way, the training is more realistic. The network has been trained with 80% of the data and tested with the remaining 20%. In addition, a k-fold of 5 has been performed to obtain an average of training and test.

The obtained results are shown in Fig. 16 and Table 9. It can be observed that the greatest confusion is between running and walking classes, where movements are very similar, while taking into account the added complexity of GBA videos. Even though it is worth mentioning that at least 97% of precision is reached in the remaining of the classes.

4.3. Computational cost

This section evaluates the computational cost of the proposed system for HAR in different hardware platforms. As it has been stated in the introduction, one of the objectives of this work is to design a system capable of running in real-time on embedded systems.

The computational cost for the two main stages of the algorithm has been obtained independently: people detection and tracking module,

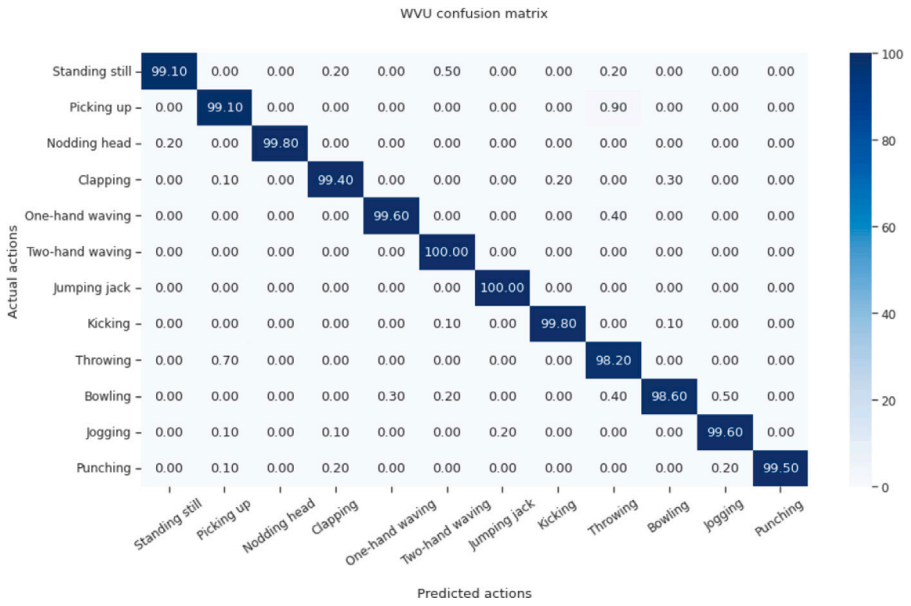

Fig. 15. Confusion matrix obtained for WVU dataset.

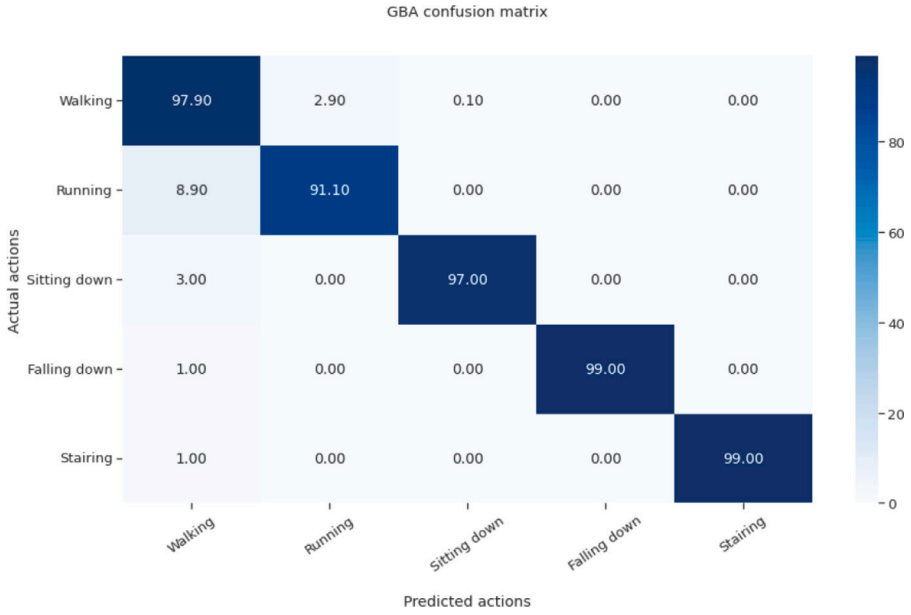

Fig. 16. Confusion matrix obtained for GBA dataset.

and action recognition. This computation has been obtained for two different hardware platforms which were previously described in Section 4.1: a high performance PC (HPC) with a GPU and an embedded platform UPS2 with a VPU.

In the UPS2, the people detection system based on a MobileNetV2-SSD has been optimized with the OpenVino framework for being executed in the available VPU (Myriad-X). In regards to the HAR module, since it is not based on CNN, the VPU could not be used to process the architecture based on LSTMs. As a consequence, the integrated CPU (Intel Atom x7-E3950) has been used for HAR.

Regarding the PC, the people detection module has been run in the CPU, and the inference for the HAR proposal has been executed on the GPU (NVidia GeForce 2080 GTX Ti NVidia, 2021) for improving the parallelization, reducing the computation time.

To evaluate the computational cost corresponding to the people detector, 500 frames of each dataset have been used, obtaining the average time. In the case of HAR stage, since the feature vector extraction and classification is carried out for each detected person, it has been computed the processing time for different number of people from one to ten, in order to determine if there is a relationship between the number of people and the time. Since available datasets for HAR described previously are characterized by only one person performing the action, computational cost study for HAR depending on the number of people in the scene has been carried out for GBA. It has been chosen because in this dataset there are scenes with more than one person performing different actions. Thus, for this analysis, there have been used 5000 frames from GBA dataset including from 1 to 10 people.

Table 10 shows the obtained values for the average and standard deviation of the processing time of one frame (in ms) for each of the algorithm stages: People detection and tracking (PD&T) and HAR. As it has been explained before, the time for HAR is shown for different numbers of people in the scene, between one and ten. In this table, it can be seen that the highest time consumption is associated to the

Table 10
PC and UPS2 computational cost in ms/frame.

	# people	Time (ms)			RAM usage
		PD&T	HAR	Total	(%)
UPS2	1	6.90 ± 0.09	0.300 ± 0.008	7.200	68.5
	2		0.312 ± 0.010	7.212	68.4
	3		0.321 ± 0.011	7.221	69.1
	4		0.338 ± 0.009	7.248	68.9
	5		0.350 ± 0.012	7.250	69.3
	10		0.410 ± 0.014	7.310	70.2
HPC	1	2.24 ± 0.06	0.097 ± 0.002	2.337	32.3
	2		0.098 ± 0.001	2.338	31.8
	3		0.097 ± 0.003	2.337	33.0
	4		0.099 ± 0.002	2.339	33.4
	5		0.100 ± 0.004	2.340	33.7
	10		0.108 ± 0.003	2.348	34.2

detection and tracking stages, with an average value of 6.90 ms/frame in the UPS2, and 2.24 ms/frame in the PC, representing over the 95% of the total time consumption.

Regarding the computational cost for the HAR stage, for the UPS2, it can be seen that the processing time increases slightly as the number of people is larger, but the required time is under 0.4 ms/frame for all analyzed cases. Furthermore, it has not been observed an overload on the microprocessor. On the other hand, results for the PC are very similar, independently of the number of people.

Thereby, it can be seen that although the computational cost depends on the number of people, the increase in time as the number of people grows is not very significant with respect to the total processing time. This can be explained due to the parallelization capability of the used devices, and the low computational cost of the HAR module as light feature vectors are used as input data.

In Table 10, the standard deviation values are also included, which demonstrates that the obtained values are close to the average and indicates consistent results

The fifth column in Table 10 includes the total average time, adding times for each stage. The obtained values are lower than 7.3 ms/frame for the UPS2 and under 2.4 ms/frame for the PC. It allows us to assert that the system is capable of running the proposed solution in real-time (≤ 30 ms/frame) in an embedded system.

Finally, Table 10 also shows the percentage of RAM usage for both devices: the UPS2 (16 GB) and the HPC (32 GB). These values increase slightly as the number of people is larger. Considering up to ten people, the UPS2 is always under 71% of RAM usage, whereas the PC is under 35%.

4.4. Comparison with the SOTA

In this section, there is presented a comparison of the results obtained for the proposal against other SOTA approaches that provide results for the same datasets. This comparison is carried out both, in terms of performance and computational cost.

Table 11 shows the comparison of performance for each dataset. There are shown the metrics provided in each work again those obtained with the proposal in the same dataset.

It is observed that the results obtained in the classification for KTH dataset are superior in the precision (99.28%) and F1 (99.06%) metrics compared to other works, being only outperformed in recall by Afza et al. (2021), that uses a feature vector obtained from the image optical flow. In addition, the proposal outperforms the rest of works in WEIZMANN dataset, with an accuracy of 99.14%, a recall of 98.91% and an F1 of 99.11%. This can be because, in this dataset, the actors exaggerated performed actions.

Regarding the WVU and IXMAS datasets, the main characteristic with respect to KTH, GBA or WEIZMANN is the increase in the number of classes up to 12. In IXMAS datasets, the proposed system outperforms

Table 11
Performance comparison of SOTA versus current work.

Datasets	Works	Precision	Recall	F1
KTH	Our method	**99.28**	99.11	**99.06**
	Afza et al. (2021)	–	**100.00**	99.00
	Nasaoui, Bellamine, and Silkan (2022)	95.45	95.29	–
	Vishwakarma (2020)	–	–	96.66
	Dash, Mishra, Srujan Raju, and Narasimha Prasad (2021)	–	–	90.00
WVU	Our method	99.27	99.40	99.32
	Sharif, Khan, Zahid, Shah, and Akram (2020)	99.28	**99.79**	99.785
	Khan et al. (2021)	**99.78**	99.00	99.38
	Ullah et al. (2021)	99.81	99.75	**99.99**
IXMAS	Our method	98.85	**99.18**	98.88
	Khan et al. (2021)	97.25	97.18	97.21
	Nida et al. (2022)	–	–	78.24
	Ullah et al. (2021)	**99.84**	96.61	**99.63**
WEIZMANN	Our method	**99.14**	**98.91**	**99.11**
	Afza et al. (2021)	–	97.93	97.38
	Sharif et al. (2020)	97.77	97.80	98.11
	Abdelbaky and Aly (2021)	–	–	90.00
	Vishwakarma (2020)	–	–	96.00

Table 12
Computational cost comparison with SOTA versus the proposal in ms.

Dataset	Our method	(Khan et al., 2021)
KTH	0.315	7.15
WVU		8.60
IXMAS		2.41

other works in RECALL. Furthermore, both in WVU and IXMAS, the obtained precision and F1-score are very close to those obtained by other works. In WVU precision and F1-score are over 99%, whereas in IXMAS (the most complex of the analyzed dataset), the precision and F1-score are over 98%, outperforming results of proposals in Khan et al. (2021), Nida, Yousaf, Irtaza, and Velastin (2022). It is noteworthy that, the systems included in Table 11 for the IXMAS dataset are based on CNNs requiring a high computational burden, which are not appropriate for running on edge devices.

Since most of the analyzed proposals do not include data about the computational cost, the comparison has been carried out against the work (Khan et al., 2021). In the related work, authors use 10000 frames per class in each dataset to extract the time metrics, so for a fair comparison, in this paper, the experiment has been replicated, computing the average processing time for the HAR stage for 10000 frames. It is worth noting that the experiments in Khan et al. (2021) have been performed with a PC that has a Core i7 CPU with 16 GB of RAM and 8 GB NVidia GPU (not being specified the model), whereas the times computed for our proposal are obtained in the embedded platform UPS2 with much lower computational resources.

Even so, in Table 12 it can be observed that, in the three datasets, the processing time obtained are more than seven times lower that those provided in Khan et al. (2021). Moreover, since the proposal does not process the image but a feature vector that is obtained from the detected BB, the processing time does not depend on the dataset or its input images size. Besides, although it depends on the number of detected people, the increasing in time as the number of people is larger, is not particularly significant.

5. Conclusions and future works

5.1. Conclusions

This paper describes a proposal for real-time people detection and activity recognition in the wild, based on edge computing. The proposed system is built on two modules: the first one detects and tracks

people by using a CNN (MobileNetV2-SSD) and a set of Kalman filters, whereas the second one recognizes the actions that is performing the tracked person at each video frame through an LSTM neural network, whose input is a lightweight feature vector extracted from its corresponding BB.

The proposed system has been evaluated in four datasets that are widely used in the related literature (KTH, IXMAS, WVU, WEIZMANN), computing metrics such as precision, recall and F1-score. Moreover, in the paper it is also presented a novel dataset for HAR in the wild named GBA that includes several people performing different actions simultaneously in realistic and not segmented scenes.

The obtained metric results are comparable to the SOTA ones, and outperform some other works on WEIZMANN dataset. More specifically, in the first four datasets, they have been obtained values in the three metrics above 98%. Within KTH, precision of 99.28% and F1 of 99.06% are remarkable. It has also to be highlighted the recall of 99.18% obtained within IXMAS. Finally when tested the proposal in WEIZMANN dataset, a precision of 99.14%; recall of 98.00% and F1 of 99.11% overcome comparable works in the SOTA. Regarding results within GBA, precision, recall and F1 are above 96.00%, what allows validating the proposal also in an in the wild environment.

On the other hand, since one of the objectives was running the proposal on real-time on edge devices, the computational cost of the video surveillance processing has also been thoroughly evaluated and compared against the one presented in SOTA works. Computational cost has been obtained both, for a high performance PC and for an UPS2 embedded platform, and the obtained results show that the proposal is run in real-time in both platforms, with a computation time seven times lower than the one of SOTA.

Since the proposal for HAR is designed to run in real-time even in embedded hardware platforms, the proposed feature vector only include characteristics related to the BB of each detected person. Due to that, the kind of actions that can be detected is limited to those that produce a change in the BB parameters. Thus, although the proposal obtains good results for the actions usually found in a video surveillance context, it may not be able to classify actions that involve small movements such as making up or teeth brushing.

Thus, it can be concluded that the proposed system is self-contained and able to be executed in platforms with reduced hardware capabilities. In addition, the proposed system performance has been demonstrated to achieve similar detection metrics as other SOTA works. As an added contribution in the paper, a new dataset that emulates a realistic video surveillance scene has been used and prepared for the scientific community, in order to verify the system reliability to work in real environments.

5.2. Future works

In this paper, we propose a system for human action detection in video surveillance systems. When designing the architecture, we encountered a significant limiting factor: computational load. The system needed to run on an embedded processor with limited hardware resources. Therefore, the system had to be as lightweight as possible. To address this, a method was devised to process not the entire image but rather the BB. This approach initially reduces the computational cost of the system. However, it imposes a limitation on the number of actions that can be accurately classified. This is because the system analyzes features related to the fluctuations in BBs across different frames to classify the action. With a larger number of actions, these fluctuations may not provide enough information for accurate classification. To overcome this issue, introducing additional features or making a more complex architecture would be necessary. However, these solutions would contradict the earlier objective of a low computational cost system.

In future research, we aim to apply this BB analysis approach in different environments, such as the healthcare field, to monitor and supervise the actions of patients or elderly individuals. Additionally, with the improvement in the technology, the MobileNetV2-SSD can be replace by other people detector that can also be executed in real-time in an embedded system. Furthermore, it can also be explored the use of other DL approaches such as those based on Transformers, that are providing good results in different computer vision applications. Finally, the ultimate step would involve deploying this software architecture in production environments using MLOps technology, where inference would be performed on specialized hardware systems rather than on the embedded system's VPU.

CRediT authorship contribution statement

Antonio Carlos Cob-Parro: Methodology, Software, Validation, Writing – original draft. **Cristina Losada-Gutiérrez:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing. **Marta Marrón-Romera:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing. **Alfredo Gardel-Vicente:** Conceptualization, Project administration, Funding acquisition. **Ignacio Bravo-Muñoz:** Conceptualization, Project administration, Funding acquisition.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgments

This research has received funding from the European Union Horizon 2020 Research and Innovation Programme under PALAEMON project (Grant Agreement n. 814962), by the Spanish Ministry of Economy and Competitiveness under project EYEFUL-UAH (PID2020-113118RB-C31), by the University of Alcalá under project ARGOS+, Spain (PIUAH21/IA-016) and METIS, Spain (PIUAH22/IA-037) and by the Community of Madrid, Spain under project CONCORDIA (CM/JIN/2021-015).

References

Abdelbaky, A., & Aly, S. (2021). Human action recognition using three orthogonal planes with unsupervised deep convolutional neural network. *Multimedia Tools and Applications*, *80*(13), 20019–20043.

Afza, F., Khan, M. A., Sharif, M., Kadry, S., Manogaran, G., Saba, T., et al. (2021). A framework of human action recognition using length control features fusion and weighted entropy-variances based feature selection. *Image and Vision Computing*, *106*, Article 104090.

Agatonovic-Kustrin, S., & Beresford, R. (2000). Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of Pharmaceutical and Biomedical Analysis*, *22*(5), 717–727.

Ahad, M. A. R. (2011). *Computer vision and action recognition: A guide for image processing and computer vision community for action understanding, vol. 5*. Springer Science & Business Media.

Ajao, O., Bhowmik, D., & Zargari, S. (2018). Fake news identification on twitter with hybrid cnn and rnn models. In *Proceedings of the 9th international conference on social media and society* (pp. 226–230).

Ali, F., El-Sappagh, S., Islam, S. R., Kwak, D., Ali, A., Imran, M., et al. (2020). A smart healthcare monitoring system for heart disease prediction based on ensemble deep learning and feature fusion. *Information Fusion*, *63*, 208–222.

Almiani, M., AbuGhazleh, A., Al-Rahayfeh, A., Atiewi, S., & Razaque, A. (2020). Deep recurrent neural network for IoT intrusion detection system. *Simulation Modelling Practice and Theory*, *101*, Article 102031.

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., et al. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, *8*(1), 1–74.

Atlam, M., Torkey, H., El-Fishawy, N., & Salem, H. (2021). Coronavirus disease 2019 (COVID-19): survival analysis using deep learning and cox regression model. *Pattern Analysis and Applications*, *24*(3), 993–1005.

Aziz, S., & Dowling, M. (2019). Machine learning and AI for risk management. In *Disrupting finance* (pp. 33–50). Palgrave Pivot, Cham.

Baptista-Ríos, M., Martínez-García, C., Losada-Gutiérrez, C., & Marrón-Romera, M. (2016a). Human activity monitoring for falling detection. a realistic framework. In *2016 International conference on indoor positioning and indoor navigation* (pp. 1–7). IEEE.

Baptista-Ríos, M., Martínez-García, C., Losada-Gutiérrez, C., & Marrón-Romera, M. (2016b). Human activity monitoring for falling detection. a realistic framework. In *2016 International conference on indoor positioning and indoor navigation* (pp. 1–7). IEEE.

Berlin, S. J., & John, M. (2020). Spiking neural network based on joint entropy of optical flow features for human action recognition. *The Visual Computer*, 1–15.

Bhatt, D., Patel, C., Talsania, H., Patel, J., Vaghela, R., Pandya, S., et al. (2021). CNN variants for computer vision: History, architecture, application, challenges and future scope. *Electronics*, *10*(20), 2470.

Bregonzio, M., Gong, S., & Xiang, T. (2009). Recognising action as clouds of space-time interest points. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 1948–1955). IEEE.

Canonical (2022). Ubuntu documentation. https://ubuntu.com/download/desktop (Last Accessed 26 July 2022).

Castiglioni, I., Rundo, L., Codari, M., Di Leo, G., Salvatore, C., Interlenghi, M., et al. (2021). AI applications to medical images: From machine learning to deep learning. *Physica Medica*, *83*, 9–24.

Chaquet, J. M., Carmona, E. J., & Fernández-Caballero, A. (2013). A survey of video datasets for human action and activity recognition. *Computer Vision and Image Understanding*, *117*(6), 633–659.

Cho, H., & Yoon, S. M. (2018). Divide and conquer-based 1D CNN human activity recognition using test data sharpening. *Sensors*, *18*(4), 1055.

Cob-Parro, A. C., Losada-Gutiérrez, C., Marrón-Romera, M., Gardel-Vicente, A., & Bravo-Muñoz, I. (2021). Smart video surveillance system based on edge computing. *Sensors*, *21*(9), 2958.

Dargan, S., Kumar, M., Ayyagari, M. R., & Kumar, G. (2020). A survey of deep learning and its applications: a new paradigm to machine learning. *Archives of Computational Methods in Engineering*, *27*(4), 1071–1092.

Das, S., Koperski, M., Bremond, F., & Francesca, G. (2017). Action recognition based on a mixture of RGB and depth based skeleton. In *2017 14th IEEE international conference on advanced video and signal based surveillance* (pp. 1–6). IEEE.

Dash, S. C. B., Mishra, S. R., Srujan Raju, K., & Narasimha Prasad, L. (2021). Human action recognition using a hybrid deep learning heuristic. *Soft Computing*, *25*(20), 13079–13092.

EPFL (2006). IXMAS actions – new views and occlusions. https://www.epfl.ch/labs/cvlab/data/data-ixmas10/. (Last Accessed 06 July 2022).

Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., et al. (2019). A guide to deep learning in healthcare. *Nature Medicine*, *25*(1), 24–29.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2007). The PASCAL visual object classes challenge 2007 (VOC2007) results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2012). The PASCAL visual object classes challenge 2012 (VOC2012) results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Feichtenhofer, C., Pinz, A., & Wildes, R. P. (2017). Spatiotemporal multiplier networks for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4768–4777).

Frieske, B., & Stieler, S. (2022). The semiconductor crisis as a result of the Covid-19 pandemic and impacts on the automotive industry and its supply chains. In *35th International electric vehicle symposium and exhibition*. URL https://elib.dlr.de/187015/.

Fuentes-Jimenez, D., Martin-Lopez, R., Losada-Gutierrez, C., Casillas-Perez, D., Macias-Guarasa, J., Luna, C. A., et al. (2020). DPDnet: A robust people detector using deep learning with an overhead depth camera. *Expert Systems with Applications*, *146*, Article 113168.

Gayathri, S., Gopi, V. P., & Palanisamy, P. (2021). Diabetic retinopathy classification based on multipath CNN and machine learning classifiers. *Physical and Engineering Sciences in Medicine*, *44*(3), 639–653.

GEINTRA (2022a). GEINTRA behaviour analysis dataset. https://www.geintra-uah.org/datasets/gotpd1. (Last Accessed 19 July 2022).

GEINTRA (2022b). GEINTRA-web. http://www.geintra-uah.org/en. (Last Accessed 28 July 2022).

Gorelick, L., Blank, M., Shechtman, E., Irani, M., & Basri, R. (2007). Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence*, *29*(12), 2247–2253.

Gruosso, M., Capece, N., & Erra, U. (2021). Human segmentation in surveillance video with deep learning. *Multimedia Tools and Applications*, *80*, 1175–1199.

Gu, B., & Sung, Y. (2021). Enhanced reinforcement learning method combining one-hot encoding-based vectors for CNN-based alternative high-level decisions. *Applied Sciences*, *11*(3), 1291.

Gupta, T., Nunavath, V., & Roy, S. (2019). Crowdvas-net: A deep-CNN based framework to detect abnormal crowd-motion behavior in videos for predicting crowd disaster. In *2019 IEEE international conference on systems, man and cybernetics* (pp. 2877–2882). IEEE.

Han, S.-H., Kim, K. W., Kim, S., & Youn, Y. C. (2018). Artificial neural network: understanding the basic concepts without mathematics. *Dementia and Neurocognitive Disorders*, *17*(3), 83–89.

Hedjazi, M. A., Kourbane, I., & Genc, Y. (2017). On identifying leaves: A comparison of CNN with classical ML methods. In *2017 25th signal processing and communications applications conference* (pp. 1–4). IEEE.

Hibat-Allah, M., Ganahl, M., Hayward, L. E., Melko, R. G., & Carrasquilla, J. (2020). Recurrent neural network wave functions. *Physical Review Research*, *2*(2), Article 023358.

Hinton, G. (2018). Deep learning—a technology with the potential to transform health care. *Jama*, *320*(11), 1101–1102.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., et al. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

Intel (2020). Intel® movidius™ myriad™ X VPU. https://www.movidius.com/myriadx. (Last Accessed 16 marzo 2020).

Intel (2022a). Processor intel® core™ i7-9700K. https://www.intel.es/content/www/es/es/products/sku/186604/intel-core-i79700k-processor-12m-cache-up-to-4-90-ghz/specifications.html. (Last Accessed 16 July 2022).

Intel (2022b). UP squared AI vision development kit. https://www.intel.com/content/www/us/en/developer/topic-technology/edge-5g/hardware/up-squared-ai-vision-dev-kit.html. (Last Accessed 16 July 2022).

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., et al. (2017). The kinetics human action video dataset. arXiv preprint arXiv:1705.06950.

Khaki, S., Wang, L., & Archontoulis, S. V. (2020). A cnn-rnn framework for crop yield prediction. *Frontiers in Plant Science*, *10*, 1750.

Khan, S., Khan, M. A., Alhaisoni, M., Tariq, U., Yong, H.-S., Armghan, A., et al. (2021). Human action recognition: a paradigm of best deep learning features selection and serial based extended fusion. *Sensors*, *21*(23), 7941.

Klaser, A., Marszałek, M., & Schmid, C. (2008). A spatio-temporal descriptor based on 3d-gradients. In *Procedings of the british machine vision conference 2008*.

Kong, Y., & Fu, Y. (2022). Human action recognition and prediction: A survey. *International Journal of Computer Vision*, *130*(5), 1366–1401.

Kong, X., Meng, Z., Nojiri, N., Iwahori, Y., Meng, L., & Tomiyama, H. (2019). A HOG-SVM based fall detection iot system for elderly persons using deep sensor. *Procedia Computer Science*, *147*, 276–282.

Konstantinova, P. D., Udvarev, A., & Semerdjiev, T. (2003). A study of a target tracking algorithm using global nearest neighbor approach.. *3*, In *Compsystech* (pp. 290–295).

Körez, A., & Barışçı, N. (2019). Object detection with low capacity GPU systems using improved faster R-CNN. *Applied Sciences*, *10*(1), 83.

KTH (2004). KTH - recognition of human actions dataset. https://www.csc.kth.se/cvap/actions/. (Last Accessed 28 July 2022).

Kulathumani, V. (2011). WVU - multi-view action recognition dataset. https://community.wvu.edu/~vkkulathumani/wvu-action.html. (Last Accessed 6 July 2022).

Kumar, S. S., & John, M. (2016). Human activity recognition using optical flow based feature set. In *2016 IEEE international carnahan conference on security technology* (pp. 1–5). IEEE.

Kumaran, N., Vadivel, A., & Kumar, S. S. (2018). Recognition of human actions using CNN-GWO: a novel modeling of CNN for enhancement of classification performance. *Multimedia Tools and Applications*, *77*(18), 23115–23147.

Laptev, I., Marszalek, M., Schmid, C., & Rozenfeld, B. (2008). Learning realistic human actions from movies. In *2008 IEEE conference on computer vision and pattern recognition* (pp. 1–8). IEEE.

László, E., Szolgay, P., & Nagy, Z. (2012). Analysis of a gpu based cnn implementation. In *2012 13th international workshop on cellular nanoscale networks and their applications* (pp. 1–5). IEEE.

Lee, I., & Shin, Y. J. (2020). Machine learning for enterprises: Applications, algorithm selection, and challenges. *Business Horizons*, *63*(2), 157–170.

Li, P., Abdel-Aty, M., & Yuan, J. (2020). Real-time crash risk prediction on arterials based on LSTM-CNN. *Accident Analysis and Prevention*, *135*, Article 105371.

Li, S., Li, W., Cook, C., Zhu, C., & Gao, Y. (2018). Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5457–5466).

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., et al. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740–755). Springer.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., et al. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21–37). Springer.

Liu, X., Campbell, D., & Guo, Z. (2017). Single image density map estimation based on multi-column CNN and boosting. In *2017 IEEE global conference on signal and information processing* (pp. 1393–1396). IEEE.

Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., et al. (2022). Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 3202–3211).

Luvizon, D. C., Picard, D., & Tabia, H. (2020). Multi-task deep learning for real-time 3D human pose estimation and action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(8), 2752–2764.

Mazzia, V., Angarano, S., Salvetti, F., Angelini, F., & Chiaberge, M. (2022). Action transformer: A self-attention model for short-time pose-based human action recognition. *Pattern Recognition*, *124*, Article 108487.

Mehta, S., & Rastegari, M. (2021). Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. arXiv preprint arXiv:2110.02178.

Mete, B. R., & Ensari, T. (2019). Flower classification with deep cnn and machine learning algorithms. In *2019 3rd International symposium on multidisciplinary studies and innovative technologies* (pp. 1–5). IEEE.

Monitoring human activity, robotics and vision laboratory, university of minnesota, department of computer science and engineering. (2023). http://mha.cs.umn.edu/index.shtml. (Last Accessed 1 June 2023).

Naik, N., & Mohan, B. R. (2019). Study of stock return predictions using recurrent neural networks with LSTM. In *International conference on engineering applications of neural networks* (pp. 453–459). Springer.

Nasaoui, H., Bellamine, I., & Silkan, H. (2022). Human action recognition using squeezed convolutional neural network. In *2022 11th International symposium on signal, image, video and communications* (pp. 1–5). IEEE.

Nasir, J. A., Khan, O. S., & Varlamis, I. (2021). Fake news detection: A hybrid CNN-RNN based deep learning approach. *International Journal of Information Management Data Insights*, *1*(1), Article 100007.

Nida, N., Yousaf, M. H., Irtaza, A., & Velastin, S. A. (2022). Video augmentation technique for human action recognition using genetic algorithm. *ETRI Journal*, *44*(2), 327–338.

NumFOCUS (2022). Pandas library. https://pandas.pydata.org/. (Last Accessed 16 July 2022).

NVidia (2021). Geforce 2080 rtx Ti. https://www.nvidia.com/es-es/geforce/20-series/. (Last Accessed 16 July 2022).

OpenVino VPU architectures. (2023). https://docs.openvino.ai/2022.3/openvino_docs_OV_UG_supported_plugins_VPU.html. (Last accessed 08 July 2023).

Oreifej, O., & Liu, Z. (2013). Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 716–723).

Pareek, P., & Thakkar, A. (2021). A survey on video-based human action recognition: recent updates, datasets, challenges, and applications. *Artificial Intelligence Review*, *54*(3), 2259–2322.

Chantamit-o pas, P., & Goyal, M. (2018). Long short-term memory recurrent neural network for stroke prediction. In *International conference on machine learning and data mining in pattern recognition* (pp. 312–323). Springer.

Potluri, S., Fasih, A., Vutukuru, L. K., Al Machot, F., & Kyamakya, K. (2011). CNN based high performance computing for real time image processing on GPU. In *Proceedings of the joint INDS'11 & ISTET'11* (pp. 1–7). IEEE.

Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M. P., et al. (2018). A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys*, *51*(5), 1–36.

Ragab, M. G., Abdulkadir, S. J., & Aziz, N. (2020). Random search one dimensional CNN for human activity recognition. In *2020 International conference on computational intelligence* (pp. 86–91). IEEE.

Raj, J. S., Ananthi, J. V., et al. (2019). Recurrent neural networks and nonlinear prediction in support vector machines. *Journal of Soft Computing Paradigm (JSCP)*, *1*(01), 33–40.

Rivas-Gomez, S., Pena, A. J., Moloney, D., Laure, E., & Markidis, S. (2018). Exploring the vision processing unit as co-processor for inference. In *2018 IEEE international parallel and distributed processing symposium workshops* (pp. 589–598). IEEE.

Sadanand, S., & Corso, J. J. (2012). Action bank: A high-level representation of activity in video. In *2012 IEEE Conference on computer vision and pattern recognition* (pp. 1234–1241). IEEE.

Selvin, S., Vinayakumar, R., Gopalakrishnan, E., Menon, V. K., & Soman, K. (2017). Stock price prediction using LSTM, RNN and CNN-sliding window model. In *2017 International conference on advances in computing, communications and informatics* (pp. 1643–1647). IEEE.

Sharif, M., Khan, M. A., Zahid, F., Shah, J. H., & Akram, T. (2020). Human action recognition: a framework of statistical weighted segmentation and rank correlation-based selection. *Pattern Analysis and Applications*, *23*(1), 281–294.

Shinde, P. P., & Shah, S. (2018). A review of machine learning and deep learning applications. In *2018 Fourth international conference on computing communication control and automation* (pp. 1–6). IEEE.

Singh, S., Ahuja, U., Kumar, M., Kumar, K., & Sachdeva, M. (2021). Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment. *Multimedia Tools and Applications*, *80*(13), 19753–19768.

Soomro, K., Zamir, A. R., & Shah, M. (2012). UCF101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402.

Spinello, L., & Arras, K. O. (2011). People detection in RGB-D data. In *2011 IEEE/RSJ international conference on intelligent robots and systems* (pp. 3838–3843). IEEE.

Sulong, G., & Mohammedali, A. (2014). Human activities recognition via features extraction from skeleton. *Journal of Theoretical & Applied Information Technology*, *68*(3).

Sun, L., Xu, W., & Liu, J. (2021). Two-channel attention mechanism fusion model of stock price prediction based on CNN-LSTM. *Transactions on Asian and Low-Resource Language Information Processing*, *20*(5), 1–12.

Suri, J. S. (2000). Computer vision, pattern recognition and image processing in left ventricle segmentation: The last 50 years. *Pattern Analysis & Applications*, *3*(3), 209–242.

Tanberk, S., Kilimci, Z. H., Tükel, D. B., Uysal, M., & Akyokuş, S. (2020). A hybrid deep model using deep learning and dense optical flow approaches for human activity recognition. *IEEE Access*, *8*, 19799–19809.

Tian, Q., Zhou, B., Zhao, W.-h., Wei, Y., & Fei, W.-w. (2013). Human detection using HOG features of head and shoulder based on depth map. *JSW*, *8*(9), 2223–2230.

Tsung-Yi Lin, M. M., et al. (2015). Microsoft COCO: Common objects in context. arXiv:1405.0312.

Ullah, A., Muhammad, K., Ding, W., Palade, V., Haq, I. U., & Baik, S. W. (2021). Efficient activity recognition using lightweight CNN and DS-GRU network for surveillance applications. *Applied Soft Computing*, *103*, Article 107102.

Vishwakarma, D. K. (2020). A two-fold transformation model for human action recognition using decisive pose. *Cognitive Systems Research*, *61*, 1–13.

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. arXiv preprint arXiv:2207.02696.

Wang, C., Wang, X., Zhang, J., Zhang, L., Bai, X., Ning, X., et al. (2022). Uncertainty estimation for stereo matching based on evidential deep learning. *Pattern Recognition*, *124*, Article 108498.

Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W. (2016). Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2285–2294).

Wang, X., Zhang, L. L., Wang, Y., & Yang, M. (2022). Towards efficient vision transformer inference: A first study of transformers on mobile devices. In *Proceedings of the 23rd annual international workshop on mobile computing systems and applications* (pp. 1–7).

Welch, G., Bishop, G., et al. (1995). *An introduction to the Kalman filter*. NC, USA: Chapel Hill.

Wu, D., Lv, S., Jiang, M., & Song, H. (2020). Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. *Computers and Electronics in Agriculture*, *178*, Article 105742.

Xia, L., Chen, C.-C., & Aggarwal, J. K. (2012). View invariant human action recognition using histograms of 3d joints. In *2012 IEEE computer society conference on computer vision and pattern recognition workshops* (pp. 20–27). IEEE.

Xia, K., Huang, J., & Wang, H. (2020). LSTM-CNN architecture for human activity recognition. *IEEE Access*, *8*, 56855–56866.

Xu, Z., Li, S., & Deng, W. (2015). Learning temporal features using LSTM-CNN architecture for face anti-spoofing. In *2015 3rd IAPR Asian conference on pattern recognition* (pp. 141–145). IEEE.

Yan, S., Xiong, Y., & Lin, D. (2018). Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence, vol. 32, no. 1*.

Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). Comparative study of cnn and rnn for natural language processing. arXiv preprint arXiv:1702.01923.

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural Computation*, *31*(7), 1235–1270.

Zeng, C., & Ma, H. (2010). Robust head-shoulder detection by pca-based multilevel hog-lbp detector for people counting. In *2010 20th International conference on pattern recognition* (pp. 2069–2072). IEEE.

Zhang, G., Liu, J., Li, H., Chen, Y. Q., & Davis, L. S. (2017). Joint human detection and head pose estimation via multistream networks for RGB-D videos. *IEEE Signal Processing Letters*, *24*(11), 1666–1670.

Zhang, H.-B., Zhang, Y.-X., Zhong, B., Lei, Q., Yang, L., Du, J.-X., et al. (2019). A comprehensive survey of vision-based human action recognition methods. *Sensors*, *19*(5), 1005.

Zhou, X., Li, Y., & Liang, W. (2020). CNN-RNN based intelligent recommendation for online medical pre-diagnosis support. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *18*(3), 912–921.

Zhou, X., Liang, W., Kevin, I., Wang, K., Wang, H., Yang, L. T., et al. (2020). Deep-learning-enhanced human activity recognition for internet of healthcare things. *IEEE Internet of Things Journal*, *7*(7), 6429–6438.

Zhou, K., Paiement, A., & Mirmehdi, M. (2017). Detecting humans in RGB-D data with CNNs. In *2017 Fifteenth IAPR international conference on machine vision applications* (pp. 306–309).