electronics

MDPI

Article

Optimized Edge-Cloud System for Activity Monitoring Using Knowledge Distillation

Daniel Deniz, Eduardo Ros, Eva M. Ortigosa and Francisco Barranco *

Department of Computer Engineering, Automation and Robotics, Research Centre for Information and Communication Technologies (CITIC-UGR), University of Granada, 18010 Granada, Spain; danideniz@ugr.es (D.D.); eros@ugr.es (E.R.); ortigosa@ugr.es (E.M.O.)
* Correspondence: fbarranco@ugr.es

Abstract: Driven by the increasing care needs of residents in long-term care facilities, Ambient Assisted Living paradigms have become very popular, offering new solutions to alleviate this burden. This work proposes an efficient edge-cloud system for indoor activity monitoring in long-term care institutions. Action recognition from video streams is implemented via Deep Learning networks running at edge nodes. Edge Computing stands out for its power efficiency, reduction in data transmission bandwidth, and inherent protection of residents' sensitive data. To implement Artificial Intelligence models on these resource-limited edge nodes, complex Deep Learning networks are first distilled. Knowledge distillation allows for more accurate and efficient neural networks, boosting recognition performance of the solution by up to 8% without impacting resource usage. Finally, the central server runs a Quality and Resource Management (QRM) tool that monitors hardware qualities and recognition performance. This QRM tool performs runtime resource load balancing among the local processing devices ensuring real-time operation and optimized energy consumption. Also, the QRM module conducts runtime reconfiguration switching the running neural network to optimize the use of resources at the node and to improve the overall recognition, especially for critical situations such as falls. As part of our contributions, we also release the manually curated Indoor Action Dataset.

Keywords: machine learning; internet of things; real-time and embedded systems; distributed systems; healthcare monitoring

Citation: Deniz, D.; Ros, E.; Ortigosa, E.M.; Barranco, F. Optimized Edge-Cloud System for Activity Monitoring Using Knowledge Distillation. *Electronics* **2024**, *13*, 4786. https://doi.org/10.3390/electronics13234786

Academic Editors: Francisco Gómez-Rodríguez, Francisco Luna-Perejón, Lourdes Miró Amarante, Javier Civit Masot and Luis Muñoz

Received: 8 October 2024
Revised: 22 November 2024
Accepted: 27 November 2024
Published: 4 December 2024

1. Introduction

As life expectancy grows, societies are more susceptible to age-related diseases. This situation has led to an increase in care needs for residents in nursing homes. In particular, in European countries such as Spain or Germany, the number of long-term care workers has grown on average by 30% since 2010 [1]. This fact poses a serious social and economic challenge in terms of the resources needed to care for vulnerable older adults.

Multiple works point out that lifestyle monitoring improves well-being and thus, healthy aging for the elderly [2,3]. Moreover, one in every four older adults (older than 65) suffers a fall in a year [4]. Rapid assistance is crucial to avoid aggravating the consequent medical problems such as blood clots or internal damages [5]. Ambient Assisted Living (AAL) systems are ideal for monitoring patient or resident habits and their daily routines to encourage active lifestyles [6]; also, these systems are enabled to provide a rapid response when dangerous situations occur, such as falling or fainting. The assessment of key factors such as the physical and cognitive functions of the elderly may also help adapt the system response to, for example, the level of fall risk [7].

The advances in the Internet of Things (IoT), Information and Communication Technologies (ICT), and System-On-Module (SoM) devices have enabled the development of Cyber–Physical Systems (CPS) which, in turn, facilitate cost-efficient e-Health solutions

for diverse needs [8,9]. The integration of Artificial Intelligence (AI) with CPS has led to Ambient Assisted Living (AAL) solutions that reduce healthcare costs while improving patients' quality of life [10,11]. CPS interconnects distributed elements of the physical world with software components to provide responses to changing environments [12,13]. In our case, these elements are edge processing nodes connected to cameras that are embedded in efficient devices. CPSs provide substantial advantages to AAL systems such as scalability, enabling the automatic deployment of multiple processing nodes in a distributed manner, or adaptation, optimizing the resource usage and delivering highly accurate predictions while providing rapid response to dynamic real-world scenarios [14].

Distributed edge-cloud computing takes advantage of the benefits of local processing by bringing computation closer to the users and sources of data. The edge-cloud paradigm contributes to reducing latency and bandwidth usage via edge processing while enabling increased computational capacity through cloud computing [15]. In general, these edge-cloud approaches use local nodes, which are resource-limited devices that are able to run efficient AI solutions on low-power budgets. At the same time, Edge Computing (EC) inherently ensures privacy by not transmitting any raw sensitive data out of the local network [16]. Nevertheless, the analysis of local non-sensitive information such as accuracy, prediction confidence, or hardware qualities from edge nodes favors global decision-making [17]. Quality and Resource Management (QRM) tools are indeed in charge of these monitoring purposes. QRM tools also perform the analysis of all these parameters for the efficient orchestration of the available computing resources [14,18], optimizing the overall system qualities [11]. This is the approach we follow for our distributed edge-cloud systems, with the exception of the integration of the central server in the local network, so data never leave this network.

Recent advances in AI, and specifically in Deep Learning (DL), have enabled the development of edge-cloud AAL solutions [19,20]. Human Activity Recognition (HAR) is a powerful example that classifies human actions analyzing data from different sources [21,22]: sensor-based HAR uses smart devices such as smart wristbands, inertial sensors, or radio-frequency-based devices especially for real-world healthcare applications [23,24]; video-based HAR uses RGB video data [25]. One of the benefits of video-based methods is that they are less intrusive and easier to adopt, as older people are sometimes reluctant to wear any smart device.

Current DL solutions using video achieve state-of-the-art accuracy for action recognition thanks to 3D convolutional layers that extract spatio-temporal patterns [25]. However, these large models are computationally expensive, with the undesirable effect of increasing inference time, using lots of computational resources, and thus resulting in high power consumption. One of the most popular techniques for facilitating the deployment of accurate and efficient DL models on resource-limited devices for real-time inference is knowledge distillation. Briefly, distillation manages to transfer the knowledge acquired by a complex and accurate DL solution to a more efficient and simpler architecture [26]. The result is a new solution that reduces inference time at the edge node while achieving high accuracy by leveraging the knowledge of the complex model distilled for the simpler architecture. This process can even help reduce the amount of input data streams (e.g., in multimodal or multi-signal solutions) that need to be processed at inference time with minimal loss in accuracy, resulting in a more efficient architecture [24,27]. In this work, we propose the application of knowledge distillation to reduce computational complexity of Deep Learning networks while simultaneously maintaining the recognition performance [28].

On the other hand, diverse and representative datasets are fundamental for HAR [29] and, generally speaking, for DL solutions that are very data-dependent [30]. In fact, there are several popular action video datasets for enabling the development of DL-based action recognition solutions [29,31,32]. These large-scale datasets represent a wide variety of human actions. However, they suffer from issues that pose a burden to indoor action recognition with, e.g., samples with noisy labels, affected by severe camera motion, or significant motion blur. This is particularly relevant when, as in our case, the actions

to be recognized are a subset of the whole dataset. In order to overcome this problem, we introduce the Indoor Action Dataset that contains daily life activities only in indoor scenarios, with sample recordings that specifically avoid artifacts such as motion blur.

This work proposes an efficient IoT-based edge-cloud CPS for indoor monitoring of a long-term care facility whose core is a set of local nodes distributed in different rooms. These nodes run optimized and distilled Deep Learning (DL) networks—enabling AI on the edge—to recognize actions from video streams. The system overview is depicted in Figure 1. The overall aim of the system is to alleviate the burden of healthcare costs of these institutions (e.g., nursing homes) while improving the quality of life of their residents. Leveraging Edge Intelligence (EI), in the proposed solution, residents in every room in the long-term care institution are kept under observation using smart vision edge nodes. At the same time, orchestration and monitoring are performed by a Quality and Resource Management (QRM) tool that runs on a central server. Firstly, this QRM tool monitors node- and system-level qualities such as energy consumption, temperature, or time performance, and also the overall prediction confidence of the system. By monitoring the action confidence, the QRM tool is able to trigger an alarm to provide rapid assistance as soon as a critical situation is detected. Secondly, the QRM tool orchestrates the optimal use of the limited computational resources distributing computation among the nodes using an Adaptive Load Balancing [33] approach that also must ensure real-time operation while avoiding overloading of particular nodes.

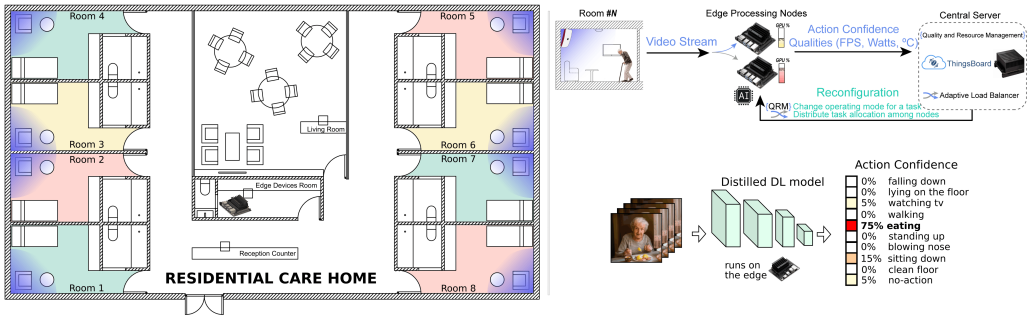

Figure 1. System overview. Cyber–Physical System for efficient indoor activity monitoring. The solution is an edge-cloud platform with: (1) Distributed cameras in the residents' rooms; edge nodes locally run Deep Learning pipelines for action recognition on the videos, via distilled models. Local nodes are low-powered System-On-Module (SoM) devices with integrated GPUs designed for running machine learning solutions. Distillation is used to improve the efficiency of the solutions given the limited computational capacity of the nodes and provides good recognition accuracy. And (2) The central server collects action predictions from the edge nodes and system qualities through the Quality and Resource Management platform ThingsBoard; the central server also triggers reconfiguration commands to balance the load between the processing nodes to ensure real-time operation. In the figure, room background colors indicate the resident's potential risk of falling (green, yellow, and red) according to their pathologies.

Eventually, one of the potential aims of this system is to foster healthy and active lifestyles: the continuous monitoring of indoor actions will be the basis for a future system that reports data to healthcare professionals and provides guidelines for personalized activities.

A key technological challenge is the integration of the aforementioned components into a unique system. However, other tasks such as model architecture selection, model optimization, deployment on the edge for real-time operation, runtime quality, and resource monitoring and management, are also relevant aspects of this work. This integration allows an efficient distributed CPS for action recognition that optimizes the use of limited resources. Bear in mind that model optimization plays a crucial role in the system. Model optimization via knowledge distillation enables the deployment of more accurate and efficient networks at the edge. Moreover, distillation allows us to obtain efficient models to monitor multiple

rooms on a single node while maintaining accuracy comparable to non-distilled resource-intensive solutions.

Briefly, our main contributions are:

- The optimization of Deep Learning models via distillation to achieve efficient and more accurate edge processing nodes while minimizing inference time by transferring knowledge from more complex and even multimodal neural networks.
- The efficient orchestration of the computational resources on the edge addressed by the QRM tool. It distributes processing among the local nodes following a Resource-based adaptive load balancing approach [33,34].
- The Indoor Action Dataset (https://github.com/DaniDeniz/IndoorActionDataset, accessed on 2 October 2024) includes recordings of several common scenes for indoor daily life activities, a valuable resource for others working on action recognition.
- A distributed CPS for efficient action recognition that ensures real-time performance (for up to 250 fps) on the edge and overall resource optimization.

This work is organized as follows: in Section 2, we review existing approaches for action recognition, highlighting the need for optimizing complex architectures for deployment on resource-constrained devices. Then, Section 3 provides a detailed overview of the proposed system's architecture, focusing on edge AI models optimized via knowledge distillation, and the introduction of a Quality and Resource Management (QRM) tool for efficient resource orchestration. Next, in Section 4, we present performance metrics of the neural network models, demonstrating real-time operation and resource efficiency at the edge. Subsequently, Section 5 illustrates the system's practical application, showcasing how resource-efficient orchestration makes the most of limited computational resources. Finally, in the Conclusion, we summarize our contributions and explore future directions for this work.

2. Related Methods

As mentioned before, we focus on video-based solutions, since they provide the best results in terms of accuracy [21,35]. Deep Learning has improved the performance of Human Activity Recognition (HAR) compared to classic solutions based on hand-crafted features [36].

Recent Deep Learning works that apply 3D Convolutional Neural Networks (3D-ConvNets) obtain significant improvements in recognition performance [31] with respect to previous approaches based on neural networks with memory, such as Recurrent Convolutional Networks [37]. In both cases, network models try to capture the temporal evolution of the action to perform classification. In addition, in recent years, some works have introduced Transformer-based action recognition solutions, such as UniFormerV2 [38], ZeroI2V [39], OnmiVec [40], or VideoMAEv2 [41]. However, despite that, these architectures achieve state-of-the-art evaluation performance in different action recognition datasets, their computational complexity is significantly high with up to more than 1 billion parameters [41]. This makes these alternatives very costly to train and fine-tune, given their large number of parameters. On the other hand, another architecture that offers accuracy comparable with current state-of-the-art solutions with less than 25 million parameters is the TwoStream Network [31] (see Figure 2).

The TwoStream Network comprises two parts: a first 3D-ConvNet stream that processes RGB raw data, and a second 3D-ConvNet stream that learns from pre-computed motion cues (estimated using the Optical Flow TV-L1 method). The Optical Flow estimates describe the 2D motion between consecutive frames in terms of the speed and the direction of the pixels in the scene. In our case, with a fixed camera, motion cues mainly come from the subjects. Later on, the outputs from both streams are combined for the final prediction. Despite being state-of-the-art in terms of accuracy, this model is also a very complex architecture that demands a huge amount of computational resources. Its complexity poses an obstacle to achieving real-time performance for action recognition on local nodes with limited resources.

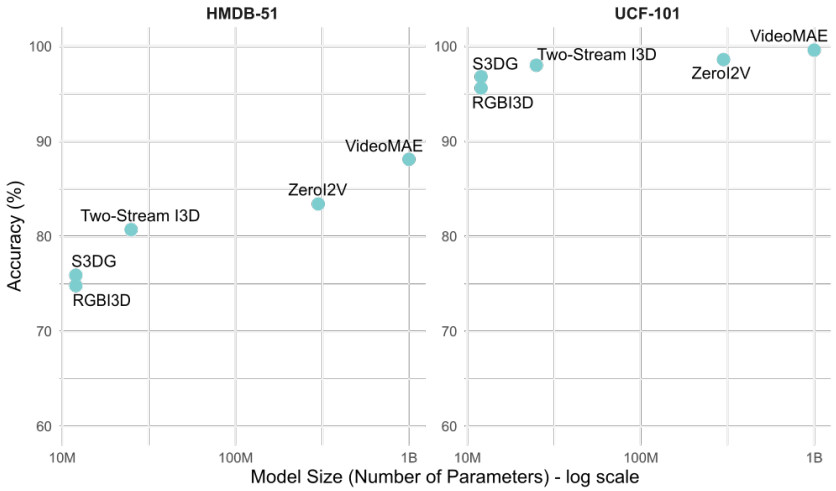

Figure 2. Comparison of accuracy vs. model size in terms of number of parameters of state-of-the-art video action classification solutions on the UCF-101 [32] and HMDB-51 [42] datasets. The best variants (such as VideoMAE, with 1 billion parameters) offer the best evaluation results. However, much lighter alternatives, such as TwoStream I3D, offer comparable results at much lower computational cost, with around only 25 million parameters. Numbers shown in this figure are retrieved from [31,39,41,43].

Simpler 3D-ConvNets architectures, such as RGBI3D [31] or S3DG [43], implement action recognition solutions on the node, providing real-time operation. The RGBI3D [31] is the DL architecture used for processing the RGB information in the TwoStream Network. It is an inflated 3D version of the Inceptionv1 [44] model. This model, comprising stacked Inception blocks, uses convolutions of different sizes to reduce computation and increase the depth and width of the architecture. The S3DG is a model that replicates the RGBI3D architecture, but includes cost-effective designs based on temporally separable convolutions with spatio-temporal feature gating [43]. This results in a model with fewer parameters and computationally more efficient that captures dependencies between feature channels [43]. Figure 3 shows the architecture of the Inception blocks of the RGBI3D and S3DG models, and the procedure for the spatio-temporal feature gating (Gating 3D) operation.

The evolution and advances in the field of DL have led to the development of accurate and unwieldy architectures, generally with millions of parameters used to process the input data doing successive transformations to provide a prediction [44]. However, it poses a challenge to deploy large DL models in devices with limited resources, such as mobile phones or embedded systems, due to their computational complexity and memory requirements [26]. Therefore, model optimization for embedded processing plays a crucial role in enabling the deployment of accurate and computational efficient solutions on the edge. For example, authors in [45] present HAR-SANet, an approach that uses model optimization techniques such as convolution factorization or quantization to reduce the computational complexity and thus, to speed up inference on low-power devices, meeting real-time performance. In short, model optimization strategies are essential to speed up inference of large models on the edge [45,46].

To ease this burden, different model compression and acceleration techniques, such as quantization [47], low-rank factorization [48], or knowledge distillation [28], were introduced. Quantization techniques consist of converting network parameter values from floating-points into integers, reducing the complexity of operations, but also the algorithmic precision. This method reduces memory footprint and inference overhead at the expense of losing some accuracy [49,50]. Low-rank factorization, on the other hand, aims to minimize the number of network parameters through Singular Value Decomposition. As a result, computing is less demanding on the edge. However, it also impacts accuracy negatively [51]. Knowledge distillation is one of the most popular techniques for enabling the deployment

of accurate and efficient DL models with real-time performance on low-power devices. For example, ref. [52] relies on knowledge distillation for producing accurate and lightweight models for visual positioning that run onboard Unmanned Aerial Vehicles with limited resources. Another example is described in [53]: an efficient human pose estimation model that achieves real-time operation on low-power embedded devices. Finally, ref. [54] developed a fast and accurate depth estimation module based on knowledge distillation that runs on mobile phones. One of the benefits of knowledge distillation compared to other alternatives is that it compresses the model without compromising evaluation performance in a non-architecture-dependent way [55]. This means that, for example, an ensemble of large models with multimodal inputs can distill knowledge to a single architecture such as a MobileNet [56]. Finally, regarding action recognition, the S3DG Distilled Network was introduced in [57]. It focuses on improving evaluation performance using limited resources and it is one of the baselines for our evaluation.

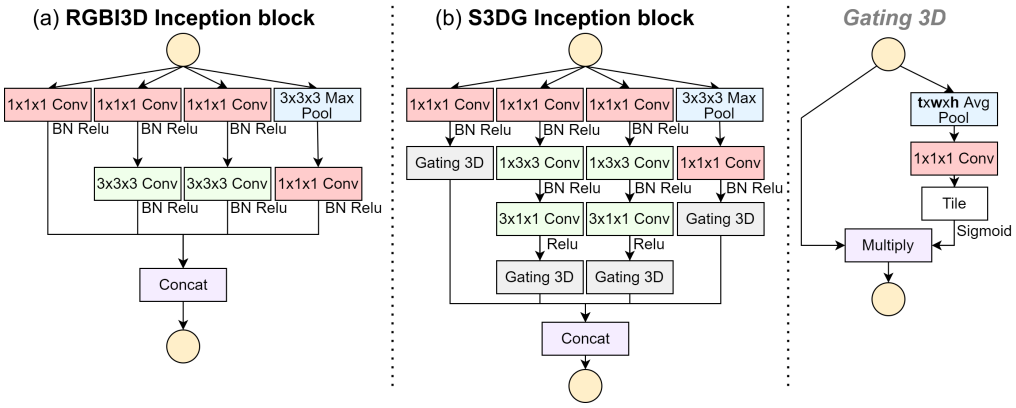

Figure 3. (**a**) Inception block of the RGBI3D [31] architecture, an inflated 3D version of the Inceptionv1 [44]. (**b**) Inception block of S3DG [43] network using temporally separable convolutions and spatio-temporal feature gating (Gating 3D). Note how the spatio-temporal filters with $3 \times 3 \times 3$ kernels from the RGBI3D are split into two convolutions with a spatial ($1 \times 3 \times 3$) and a temporal ($3 \times 1 \times 1$) filter. These filters are computationally more efficient because they compute fewer floating-point operations compared to the spatio-temporal convolution. On the right, the spatio-temporal feature gating operation is shown. This operation is used for capturing dependencies between feature channels.

3. Efficient Distributed CPS for Indoor Action Monitoring

Our system is designed to be deployed in long-term care institutions, such as nursing homes, although it can be easily adapted to hospitals or other residential facilities.

The hardware components of this solution are:

- A set of integrated video processing nodes running on a low budget Jetson Nano System-on-Module (SoM), which is an integrated and miniaturized full system. These are embedded modules equipped with a cost-efficient GPU that is capable of executing machine learning solutions in real time on low-power budgets.
- A central server (Jetson Xavier SoM) that monitors and collects data from the local edge nodes thanks to a QRM tool. The QRM tool orchestrates computing resources using a load balancing tool to efficiently distribute video processing among the edge nodes. Runtime monitoring and reconfiguration also enable the ability to trigger alarms in case of critical situations, such as the fall of a resident.

As shown in Figure 1, the plan is to deploy the system on a residential care home. All local processing nodes and the central server are located inside the facility and data are not to be shared out of the local network due to the sensitive nature of the information. For the final deployment, processing nodes are decoupled from cameras, and a resource management module will ensure the efficient use of resources, processing multiple video feeds at the same edge device. Additionally, note that every room in Figure 1 has a different

background color: green, yellow, or red. Based on an assessment of the cognitive and physical state of the resident conducted by a healthcare professional, this color indicates the resident's fall risk: low-, mid-, and high-level, respectively [7]. The value in the scale determines the likelihood of suffering falls and thus, in our case, determines the analysis to be performed in order to optimize the use of computing resources and ensure the best care of the resident. For example, Alzheimer's patients are known to wander off at night, and thus their risk should be rated as high (red). The module in charge will take this into account and perform the most accurate recognition to robustly detect falls and prevent any further dangerous consequences.

The objective of our efficient distributed CPS can be formulated as the optimization in Equation (1), aiming to finding a system configuration that minimizes the objective,

$$\min \quad \left(\alpha \sum_i \sum_j C_{ij} \cdot x_{ij} + \beta \left(P_{\min} - \frac{\sum_i \sum_j P_{ij} \cdot x_{ij}}{\sum_i \sum_j x_{ij}} \right)^2 \right) \tag{1}$$

$$\text{subject to} \quad \sum_j C_{ij} \cdot x_{ij} \leq R_i \tag{2}$$

where R_i is the total computational capacity of the hardware resource i, and C_{ij} is the computational load of the model j in the hardware resource i. Moreover, $P_{\min}$ is the minimum evaluation performance required for the models in each case (in our case, it is set to 1) and $P_{ij} \in [0, 1]$ is the evaluation performance of the model j in the hardware resource i. Finally, x_{ij} is a binary variable that indicates if the model j is executed in the hardware resource i. We also set a constraint to ensure that the total computational load of all the models running on a particular resource i, do not exceed the computational capacity available on hardware resource i. α and β are the weights that let us modulate the total importance of the computational load and the evaluation performance of the models used for the action monitoring.

In the next subsections, we first explain the different models for indoor action monitoring, the distillation techniques to optimize these computationally intensive DL models, the Quality and Resource Management module that takes care of resource orchestration, and its policy for triggering reconfiguration when the system requires more accurate inference or confirmation of a critical action such as a fall.

3.1. Indoor Action Recognition

Firstly, we designed a TwoStream version of the S3DG network [57], analyzing RGB data and motion cues for indoor action recognition. This alternative achieves high recognition performance, but it is also computationally intensive.

For this reason, we choose efficient DL architectures that achieve reasonable accuracy but with lower computational budget. We also adapted the DL models for processing a reduced temporal and spatial resolution with respect to the original ones; the spatial downsampling of the inputs drastically reduces the resources needed to run an inference at the expense of retraining. Specifically, 3D-ConvNets models based on the RGBI3D [31] and S3DG [57] networks were selected and adapted in this way for the final deployment at the edge.

3.2. Distillation

Through distillation, it is possible to transfer the knowledge extracted by larger models or, as in our case, by an ensemble of individually trained models (using RGB images and motion cues) to a single simpler neural architecture [28]. Distilled models are trained on a dataset, and then a soft target distribution is employed, using as reference the cumbersome model with a high temperature in its softmax. The softmax function transforms the model raw prediction into a probability distribution with values that sum up 1. Equation (3)

shows the softmax function, where z are the raw unnormalized outputs of the model and τ is the temperature.

$$q_i = \frac{exp(z_i/\tau)}{\sum_j exp(z_j/\tau)} \quad (3)$$

The standard softmax function uses $\tau = 1$. For distillation, higher temperature values are set to obtain targets with a softer distribution. Softer targets provide information about the inter-class relationships, which is useful to make the model learn how to generalize [28].

We expect to improve the recognition performance of the DL models using our Indoor Action Dataset through knowledge distillation [28]. We select the most powerful architecture (TwoStream) as the Teacher, and the Students that are efficient DL architectures with sampled inputs.

For distillation, the next steps were followed: (1) Input data are augmented with operations such as random rotation, resizing, or cropping; (2) the Teacher and Student networks are fed with the same input data, (although the Student input is downsampled); (3) the distillation loss ($\mathcal{L}_{KL}$) is computed using Kullback–Leibler divergence [58] to evaluate the difference between the probability distribution of predictions between the Teacher and the Student; (4) the student cross-entropy loss ($\mathcal{L}_{CE}$) with respect to the sample label is also computed; (5) the final loss (see Equation (4)) is obtained combining the losses computed in steps 3 and 4 setting $\lambda = 0.1$ to weigh the contribution of $\mathcal{L}_{KL}$ (90%) and $\mathcal{L}_{CE}$ (10%) to the global loss. In Equation (4), τ refers to the temperature, ψ to the softmax function, Z_s and Z_t are respectively the student and teacher predictions, and y stands for the true label of the sample. Hence, the Student is optimized primarily to match the predictions of the Teacher; and (6) back-propagation is applied on the Student to minimize $\mathcal{L}_{global}$.

$$\mathcal{L}_{global} = (1-\lambda)\mathcal{L}_{KL}(\psi(Z_s/\tau), \psi(Z_t/\tau)) + \lambda\mathcal{L}_{CE}(\psi(Z_s), y) \quad (4)$$

Note also that every model is distilled using the temperature values $\tau \in [1, 9]$. High temperature values help to extract information about the inter-class relationships. This is useful for transferring how the Teacher tends to generalize.

Figure 4 shows an overview of the approach followed to distill the knowledge from the multimodal TwoStream architecture that uses RGB and motion cues, to more efficient DL models that use lower-resolution RGB data. Sampling the input size negatively impacts the baseline accuracy but also reduces the amount of floating-point operations (FLOPS) required to perform an inference. This is fundamental for embedding more efficient DL models that achieve lower inference times while maintaining recognition performance.

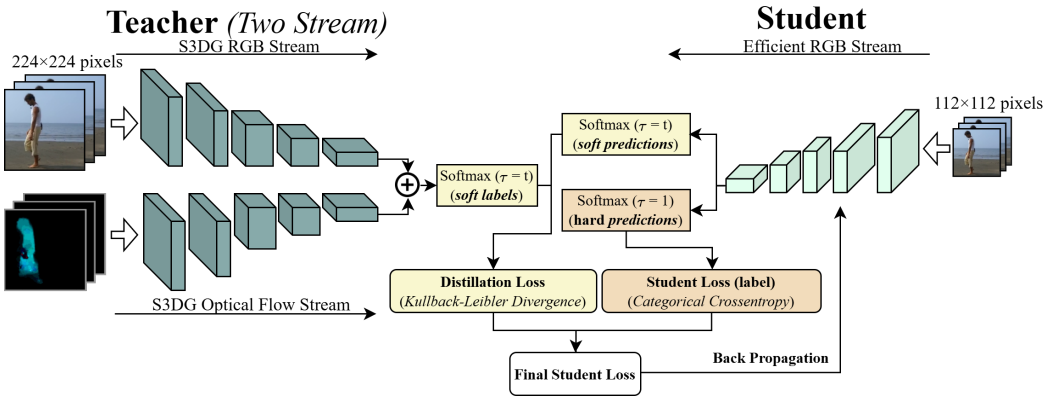

Figure 4. Distillation architecture example. We use the most complex and accurate alternative (TwoStream network) as a Teacher. Then, we select a Student, which is more efficient and receives a similar input with a lower temporal and spatial resolution. We train the Student, taking into account how different the distribution of its prediction is compared to the Teacher. Note that motion cues are only required in training and distillation phases to feed the Teacher. Student distilled models are fed only with RGB videos. Higher temperature (τ) leads to softer distribution between classes.

3.3. Quality and Resource Management (QRM)

In a resource-constrained environment, it is crucial to optimize the use of computational resources for running tasks, ensuring real-time operation [59]. Moreover, an adaptive distributed edge processing approach helps to manage the efficient use of local limited computational resources for monitoring residents' daily activities. The decision-making on how to allocate local devices to the computation to ensure real-time processing is carried out by the QRM on the central server. The QRM module monitors the action confidence and node hardware qualities, such as time performance or energy consumption.

Note that, in our case, the DL models are GPU intensive, and their performance is highly dependent on the computational capacity of the GPU. For this reason, the average GPU usage, therefore, serves as an indicator of the computational load of the local nodes, as done in previous works [60,61]. Particularly, the estimated computational load for real-time execution for each model is pre-calculated offline before deploying the system to avoid the monitoring overhead. This implies that every machine learning architecture is executed beforehand at the edge, and the average GPU utilization is monitored for a fixed time (30 s) while processing a video feed and stored to be checked when the system is running. The mean GPU usage reflects the estimated computational load for analyzing a particular room depending on the resident's risk. From now on, we will refer to this quality simply as the computational load.

3.3.1. Resource Orchestration

The QRM tool follows a Resource-based Adaptive Load Balancing approach [33]. The load balancing algorithm distributes video processing favoring the edge node with the lowest computational load to prevent overloading while guaranteeing real-time operation. The load balancing method collects computational load information from the local nodes in a Load Information Table. The computation distribution considers the number of rooms and the risk level of the residents analyzed by each processing node. A Resource-based orchestration is followed, and the processing is distributed depending on the resources available on the local nodes and the computational load of the tasks (see Figure 5). This dynamic load balancing method monitors the task allocation periodically to equally distribute processing across the nodes [34]. The process for dynamic load balancing starts with our algorithm retrieving a list of the node workloads from the Load Information Table. This information includes the rooms that each node is processing and the computational load of each task. Afterwards, the dynamic load balancing monitor evaluates the consequences of reallocating one of the tasks from the node with the highest workload into the node with the lowest workload. Job reallocation is done if it minimizes the difference between the computational load from both nodes.

3.3.2. Runtime Reconfiguration Policy

The model to perform activity analysis depends on each resident's fall risk. Resource-intensive but more accurate alternatives are assigned to residents with higher risk. Moreover, when the system detects that no significant action takes place or that a room is empty, action recognition is turned off, and a simple Motion Detection method is run. The motion detection component is based on thresholding, and simply computes the absolute difference of pixel values between two non-consecutive frames. Analyzed frames are separated by a one-second time interval since the elderly tend to move very slowly, so differences between the adjacent frames might be subtle. If the absolute value of the sum of differences is significant, the module considers that there is motion in the scene. According to our experiments, setting this threshold between 10 and 15% to rate pixel-level differences yields the best results. This module aims at reducing computation in a simpler way, without adding significant processing overhead. This simpler estimation greatly reduces resource usage while still allowing to monitor the room, and running action recognition when meaningful activity is actually detected.

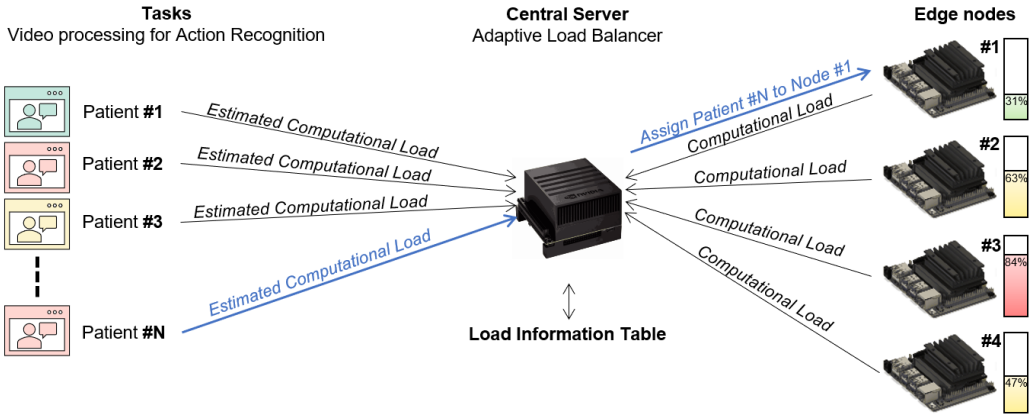

Figure 5. Resource-based adaptive load balancing diagram example. The central server gathers the computational load from the processing nodes, filling in the load information table (center). Tasks are distributed between edge nodes following a Resource-based approach. This means that video processing is assigned to the nodes with the lowest computational load to equally distribute computation among the edge nodes. In the figure, the example shows a new resident room to be monitored (blue lines, resident #N) and how processing is deployed to edge_node_#1 which, at the moment, is the node with the lowest workload.

The runtime reconfiguration is carried out to confirm whether a resident might have suffered a potentially harmful situation. This is addressed by re-analyzing the video using the most accurate DL alternative to reduce false alarms and improve the recognition of dangerous situations. This analysis is offloaded to the central server using the most accurate alternative when the DL model on the node predicts a fall with medium-low confidence.

The QRM follows the policy shown in Figure 6. When a critical situation is not identified with high confidence on the edge, the video is offloaded to the central server and re-analyzed with the most accurate model (in this case, Distilled S3DG_64_196) to confirm whether to trigger an alarm. This runtime reconfiguration contributes to optimizing the use of resources at the edge while reducing false alarms and improving the recognition of critical situations, reaching a sensitivity of up to 93.75% and precision above 99% for fall identification. In addition, the re-analysis of the action looking for confirmation takes less than 2 s, giving enough time to the caregivers for immediate assistance.

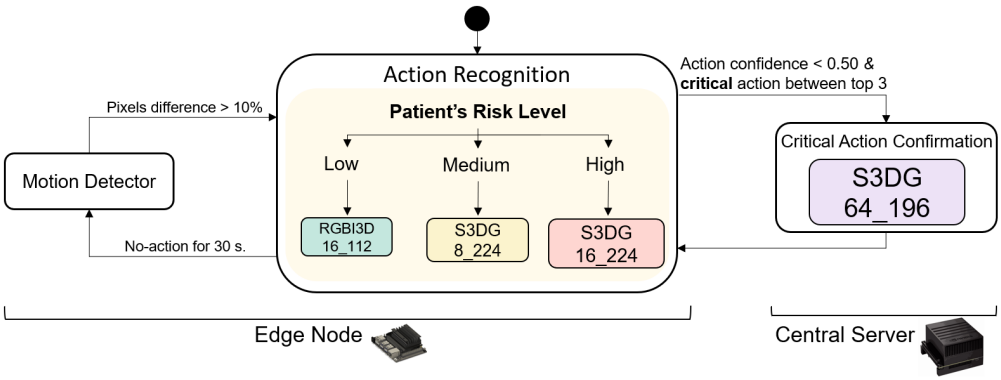

Figure 6. Runtime reconfiguration policy. Action recognition is addressed on the node using the fittest operating mode to the resident's risk level. If no activity is detected for more than 30 s, only the Motion Detector is performed. Furthermore, when the action recognition module on the edge cannot distinguish properly whether a critical situation occurred, videos are re-analyzed with the most accurate model, offloading the computation to a more powerful embedded device (Jetson Xavier, central server). The offloaded task takes less than 2 s, enabling rapid assistance from caregivers.

4. Discussion and Results

In this section, we first describe the nature of our Indoor Action Dataset, which is a collection of common indoor daily life activities. Next, we analyze the performance of several DL architectures for action recognition and compare them with their distilled alternatives. By distilling the knowledge from complex to simpler and less resource-intensive architectures, we aim to design a computational- and energy-efficient solution.

Regarding local edges, our NVIDIA Jetson SoM devices run the action recognition solutions. These power-efficient embedded devices have four ARM microprocessors and a 4GB GPU with 128 CUDA cores. These are miniaturized full systems that are very limited in terms of computational capabilities compared to high-end GPUs. However, these SoMs are also optimized for efficient execution as stand-alone platforms in real scenarios.

Models running on local edges are optimized through TensorRT to enhance their throughput. In particular, their memory footprint and power budgets are reduced by limiting bitwidths for floating-point operations, fusing kernels and layers, or selecting the best hyperparameters for convolutional layers. Finally, the Triton Inference Server is integrated for low-latency deployment of multiple DL alternatives on the edge.

4.1. Indoor Action Dataset

To overcome the lack of quality and representative video data for indoor action recognition we designed a new dataset, namely the Indoor Action Dataset. This dataset was built recording samples of an average of 10 min videos with people carrying out indoor actions. Afterwards, the videos were manually curated, assigned to training, validation or testing split, and sampled into 3 to 5 s labeled clips. In total, five different subjects collaborated in the recordings, in a variety of indoor scenarios such as bedrooms, kitchens, or living rooms.

The dataset includes samples of ten different classes of activities such as cleaning, eating, sitting down, standing up, blowing the nose, walking, watching tv, as well as classes that represent potentially risky situations such as falling down or lying on the floor. Additionally, we included a no-action class with indoor spaces where no activity is carried out or no person is on the scene. This class is useful to reduce DL models bias from the scene background. This representative Indoor Action Dataset is limited in terms of the number of total samples (around 1300) compared to large-scale datasets, but it serves the purpose of our work. Data are split into training (55%), validation (15%), and testing (30%). Bear in mind that every clip extracted from each video is assigned to the same split. The average number of samples per class is 122 ($\pm 63\sigma$), although it is imbalanced. We select a larger proportion of elements for testing than for validation because we aim to retain a sufficiently large test set to accurately assess the model's generalization capabilities across a variety of indoor actions and scenarios, while ensuring adequate representation of classes for obtaining reliable performance metrics.

We also use a large-scale dataset selecting examples for indoor activities only, from publicly available datasets: Kinetics [31], Charades [29], STAIR [62], Moments in Time [63], or UCF-101 [32]. This Heterogeneous Activity Dataset, with more than 14,000 videos, with an average duration of 10 s, is a refined version of the Combined Indoor Action Video (CIAV) dataset [11]. Around 75% of clips are assigned to the training split, 10% to the validation split, and we reserve 15% for testing purposes. The CIAV dataset is manually curated to avoid feeding the network with erroneous labels and to homogenize action labels, especially considering that our application focuses only on the recognition of indoor activities, a small subset of the whole dataset. However, the quality of the data representing indoor actions is still poor (e.g., some samples suffer from sudden and fast camera motion or severe motion blurring). Nevertheless, the use of a larger dataset enables generalization, preventing overfitting.

4.2. Indoor Action Recognition Evaluation

Several alternatives based on 3D convolutions and Recurrent Convolutional networks for action recognition are compared in this section. Concretely, we selected the RGBI3D [31] and S3DG [57] 3D-ConvNets architectures as the baselines for our models. For the sake of completeness, we have also designed a Recurrent 2D DL model using a 2D convolutional backbone (MobileNetV2 [56]) trained jointly with a Long Short-Term Memory (LSTM) layer [64].

4.2.1. Training Procedure

Different configurations for the network architectures (RGBI3D, S3DG, and Recurrent2D) were considered with variations in spatial resolution and the temporal dimension. Specifically, we sampled equally distanced frames, selecting $[8, 16, 32, 64]$ equidistant frames out of short sequences of 64 frames; also, inputs were sampled at different spatial resolutions $[112, 140, 168, 196, 224]$.

For training, first, every alternative was pre-trained on the large-scale Heterogeneous Activity Dataset. Next, the resulting model was fine-tuned with the Indoor Action Dataset. Following this approach helps to take advantage of the knowledge provided by the large-scale dataset improving the model generalization and thus, it fosters specialization for actions of interest in our Indoor Action Dataset.

Subsequently, the model alternatives were distilled using a TwoStream network as a Teacher to improve their recognition performance while maintaining time performance or resource usage. Note that the TwoStream S3DG network selecting 64 frames with 224×224 resolution is used as the Teacher. The TwoStream S3DG is the most complex model (250 GFlops) and it reaches 89.47 F1-Score, running at less than 7 fps at our edge nodes, far away from real-time performance (at least 25 fps).

4.2.2. Analysis

We have compared the different models in terms of time performance and accuracy, using the test set from the Indoor Action Dataset. On the one hand, our accuracy metric is the macro F1-Score (simply referred as F1-Score from now on). The F1-Score is the harmonic mean between the precision and the recall per class and, also, it equally weighs the contribution of each class. On the other hand, the time performance is measured in frames per second (fps) running the inference on our edge nodes. The computational requirements of the models in terms of resource usage determine the inference time. More specifically, lower resource usage also means faster inference (more fps) and lower energy budgets compared to more computationally complex alternatives that take more time to perform inference and require higher power consumption.

In particular, Table 1 shows a selection of the distilled architectures and highlights the percentage improvement in the F1-Score after distillation. Architectures are labeled in the way 'model_tmpSampling_spResolution': where 'model' is the model backbone architecture, 'tmpSampling' is the temporal resolution, and 'spResolution' the spatial resolution. For example, RGBI3D_16_112 is a model based on the RGBI3D model which uses 16 frames (out of 64) of 112×112 spatial resolution.

Notably, before distillation, the mean F1-Score was 80.62 ± 6.82, and after distillation, the average F1-Score improvement reached 3.2%. Although it may seem a marginal improvement, it significantly varies among the proposed models, reaching up to 8.3% improvement for RGBI3D_16_112 and 7.1% for S3DG_64_140. More importantly, the inference time for processing the video feed of 2560 ms (64 frames at 25 fps) does not increase despite this accuracy improvement.

Table 1. Distillation improvement on DL architectures (the percentage before ↑ indicates the improvement with respect to the F1-Base).

Architecture	Time (ms)	F1-Base	F1-Distilled
RGBI3D_8_224	401	80.85	84.88 (5.0% ↑)
RGBI3D_16_112	239	76.34	82.65 (8.3% ↑)
RGBI3D_16_224	751	86.16	86.52 (0.4% ↑)
RGBI3D_64_140	1275	85.23	88.06 (3.3% ↑)
RGBI3D_64_168	1804	84.65	87.56 (3.4% ↑)
RGBI3D_64_196	2408	87.67	90.55 (3.3% ↑)
S3DG_8_224	416	84.41	85.63 (1.4% ↑)
S3DG_16_112	256	78.12	79.90 (2.3% ↑)
S3DG_16_224	807	86.55	88.13 (1.8% ↑)
S3DG_64_140	1306	78.56	84.12 (7.1% ↑)
S3DG_64_168	1842	88.18	89.13 (1.1% ↑)
S3DG_64_196	2553	87.04	91.74 (5.4% ↑)
Recurrent2D_8_224	133	75.19	76.64 (1.9% ↑)
Recurrent2D_16_112	96	64.32	66.78 (3.8% ↑)
Recurrent2D_16_224	303	74.98	76.91 (2.6% ↑)
Recurrent2D_64_168	633	71.80	73.16 (2.1% ↑)

For visualization purposes, Figure 7 only shows the most representative models, considering those with good accuracy vs. time performance trade-offs. The left axis shows the F1-Score achieved by the DL models (F1-Score before distillation in orange). The red line denotes the inference time in fps reached by the optimized DL architectures running on the edge devices (see right axis). Models are shown in the horizontal axis, using stacked columns that show the F1-Score in orange and the improvement reached after distillation in blue. Note the efficiency vs. accuracy trade-off for the different models: on the right part of the figure, models reach the highest accuracy at the expense of higher inference times; the left part shows models with time performance above 250 fps, but achieving around 10 less points of F1-Score. In summary, training very accurate alternatives with the maximum F1-Score has a huge impact in terms of resource usage and time performance. Also bear in mind that the input size of the DL models (temporal and spatial resolution) directly impacts accuracy and time performance: lower resolution leads to more efficient (but less accurate) alternatives.

As for the improvement via distillation, the blue color in columns (also annotated), shows how much the F1-Score of the DL models is improved through distillation by acquiring knowledge from the TwoStream Teacher network. For example, the F1-Score of the most efficient network shown is enhanced up to 82.65, an 8% accuracy increase maintaining inference time. This means that the accuracy gap between the most efficient distilled alternative Distilled RGBI3D_16_112 and the most accurate architecture S3DG_64_196 (before distillation) is reduced from 12 to only 4 F1-Score points. Moreover, the distilled alternative uses 87% less computational resources (14 vs. 102 GFlops), meaning a 10× reduction in inference time. In other words, before distillation and comparing these two models, the RGBI3D_16_112 offer 5.45 points of F1-Score per GFlop. After distillation, this efficient model offers 8% more points of F1-Score per GFlop, which is significantly higher compared to the S3DG_64_196 that offer 0.85 F1-Score points per GFlop.

Figure 8 shows the F1-Score vs. inference time on the edge trade-off from the distilled models. This figure aims to clarify the selection of the fittest models to be deployed on the edge. One should choose the architectures with higher throughput (lower inference times) and higher accuracy values. In the figure, these models are highlighted at the top left (connected by a yellow line): Distilled RGBI3D_16_112, Distilled S3DG_8_224, and Distilled S3DG_16_224. More accurate models do not allow us to run inference for more than one resident in real-time in the same edge node and thus, they are not good candidates for deployment. Running less resource-intensive models enables simultaneously processing

more than one video stream in the same edge node, optimizing the overall computing resources of the distributed system. Finally, and despite of the last sentence, we do select the Distilled S3DG_64_196 model for confirming potentially risky situations, such as falls. In this case, we require a very accurate (although costly) model to reduce false positives that will be executed on the central server.

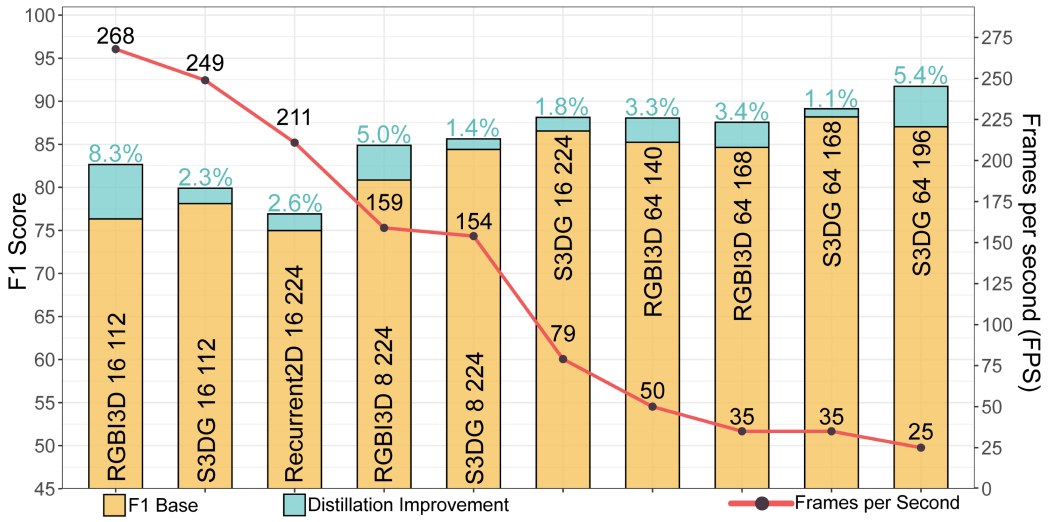

Figure 7. Evaluation performance of the selected alternatives for action recognition. The F1-Score value is given in the left vertical axis and each stacked column is also annotated with the F1-Score improvement achieved through distillation (blue). The right vertical axis shows the time performance on edge devices (fps); models (columns) are ordered from higher to lower time performance from left to right (red line). For instance, distilled alternatives located in the middle of the horizontal axis offer great accuracy vs. time performance trade-off. Specifically, models such as Distilled S3DG_16_224 or Distilled RGBI3D_64_140 achieve higher accuracy compared to the most resource-intensive non-distilled architecture (S3DG_64_196—25 fps), with an F1-Score above 88% and meeting real-time performance running at 79 and 50 fps respectively.

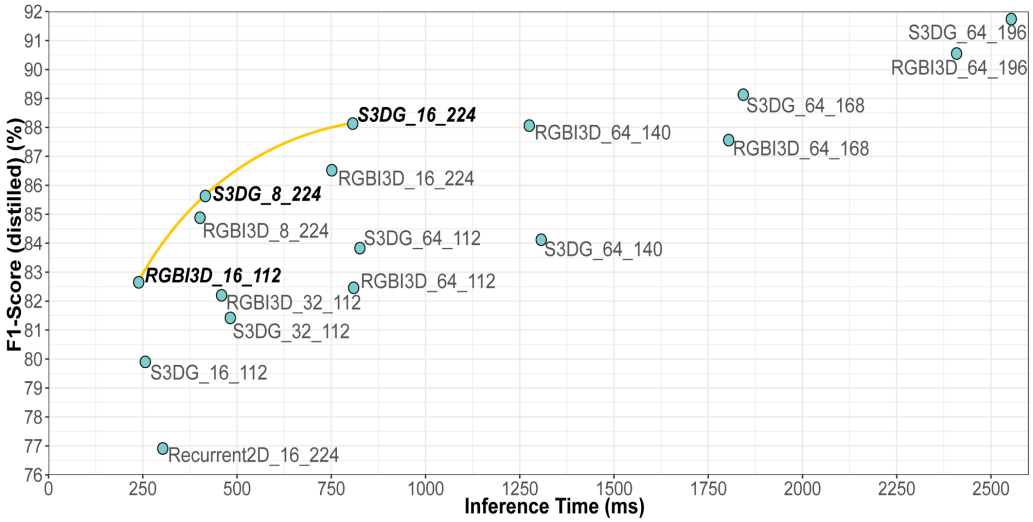

Figure 8. F1-Score vs. inference time trade-off from distilled student models. The inference time in milliseconds indicates the time required to perform an inference for a sample clip of 2560 ms (64 frames at 25 fps) on the edge node. In the chart, one should look for models with high throughput (lower inference times) and high F1-Score values. Therefore, for the final system deployment we select the alternatives located on the top left side of the figure (the front connected by a yellow line), because they provide a better F1-Score vs. resource usage trade-off.

4.3. Distributed Edge Processing

Briefly, analyzing the residents' habits through distributed local processing offers several advantages such as efficient processing and real-time execution, privacy protection, bandwidth usage, and cost reduction compared to using high-end expensive cluster platforms.

As mentioned before, the proposed system uses the selected distilled models in Table 2, where the models for low-, mid-, and high-level fall risk residents are described. The characterization includes the average F1-Score, the computational load, and the F1-Score for falling down, a critical action for our system. Note how the model assigned to residents with higher risk offers better F1-Score value for the class 'falling down'.

Table 2. Computational load in Jetson Nano.

DL Model	Fall Risk	F1-Score *Fall Down*	F1-Score *Avg.*	Comp. Load
RGBI3D_16_112	low-level	90.32	82.65	10%
S3DG_8_224	mid-level	93.33	85.63	17%
S3DG_16_224	high-level	96.77	88.13	32%
Motion Detector	-	-	-	3%

To assess the generalization capabilities of the selected solutions, we also evaluated the distilled models training on the Indoor Action Dataset and testing on other public datasets. This allows us to test on unseen residents and rooms, and assess how the models perform under special circumstances like blur, haze, or bad lighting conditions. Evaluation is carried out by selecting the test samples from classes that match the activities that our solution is designed to classify. Specifically, the test set comprises 668 and 214 samples from Kinetics and Fall Dataset respectively. Table 3 shows that the alternative that re-analyzes critical situations reaches up to 90% F1-Score on the public Fall Detection dataset. For reference, we include in this table the evaluation performance obtained from a state-of-the-art Transformer-based architecture, UniFormerV2-L14 [38]. This model, with more than 350 million parameters only pre-trained in Kinetics-600, achieves very good evaluation results on the subset of the Kinetics test set. However, this architecture can only capture a subset of 5 out of 10 the indoor actions that our models are trained with. Note also how our solutions, with less than only 12 million parameters each and fine-tuned on the Indoor Action Dataset, offer reasonable levels of accuracy on datasets with artifacts or challenging scenarios as shown for Kinetics. Models that perform well on our dataset also demonstrate higher evaluation metric scores on the other datasets.

Table 3. Models tested on public datasets.

Model	Fine-Tuned	Distilled	Fall Detection [65] *F1-Score*	Kinetics-600 [31] *F1-Score*
UniFormerV2-L14 [38]	✗	✗	-	97.66%
RGBI3D_16_112	✓	✓	81.51%	64.53%
S3DG_8_224	✓	✓	76.28%	74.14%
S3DG_16_224	✓	✓	85.75%	78.95%
S3DG_64_196	✓	✓	89.95%	80.62%

Finally, Figure 9 shows the results using the Distilled RGBI3D_16_112 model for low-risk residents: for the fall, the model is not able to reach an acceptable confidence. However, the model for high-risk residents achieves 97% confidence for that sample.

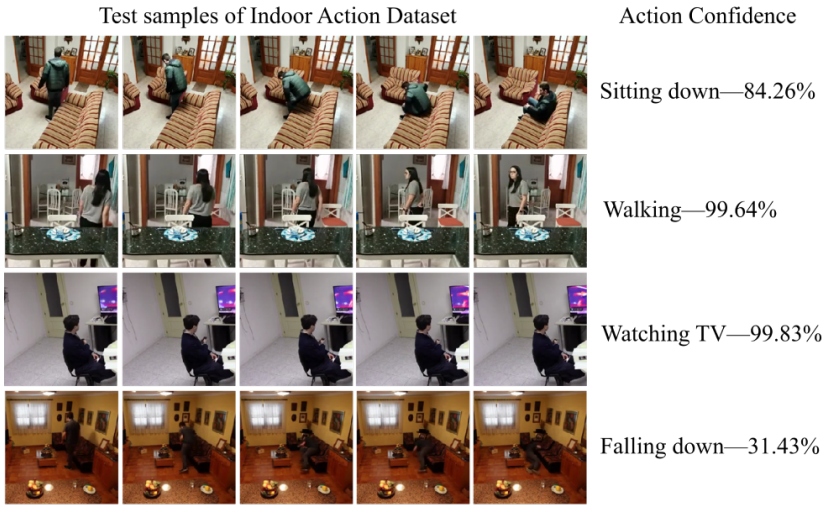

Figure 9. Prediction confidence for the Distilled RGBI3D_16_112 network over four test samples from Indoor Action Dataset: sitting down, walking, watching tv, and falling down. The Distilled RGBI3D_16_112 model is not able to classify the fall with good confidence. Therefore, a reconfiguration will be triggered to confirm what really occurred. Bear also in mind that, for this same sample, the model assigned for monitoring residents with high-level fall risk (Distilled S3DG_16_224) directly identifies that the person suffered a fall with a confidence that is higher than 97%.

5. Use Case: Monitoring Residents in a Long-Term Care Institution

In this section, we study the benefits of the proposed solution for monitoring residents in a long-term care institution. We focus our analysis on the contribution of the QRM tool for the design of a computationally efficient and accurate solution. We validate our edge-cloud system considering its deployment on the nursing home shown in Figure 1.

Our final objective is to minimize the number of processing nodes while maintaining the maximum possible accuracy for the activity monitoring. Our model optimization and the adaptive distributed processing allow multiple rooms to be monitored using a single edge node. Next, each room is analyzed using the DL model assigned to the risk of the resident (see again Figure 6). When inference confidence is low, runtime reconfiguration is activated, running the most accurate model (Distilled S3DG_64_196) to determine whether to trigger an alarm for a fall or dismiss it as a false positive. The adaptive load balancing tool running at the QRM module on the central server guarantees real-time processing without overloading any specific node.

For the scenario in Figure 1, we have eight rooms to be monitored: three with low-level risk residents, two with mid-level risk residents, and three with high-level fall risk residents. Taking into account the proposed optimization in Equation (1), let us discuss the different possible scenarios:

1. Non-distilled models are deployed. The system monitors every resident with the S3DG_64_168 model that offers similar recognition performance as the distilled model assigned for high-risk. However, running this non-distilled model on the node results in 72% of computational load. This configuration requires eight edge nodes (one node per room) to run this configuration.
2. Distilled models are deployed but no QRM module is integrated. All residents are monitored using the distilled S3DG_16_224 DL model and running this model on the node results in 32% computational load. This configuration requires three edge nodes.
3. Distilled models are deployed but no QRM module is integrated. All residents are monitored using the distilled RGBI3D_16_112 DL model and running this model on the node results in just 10% computational load. The configuration only requires one edge node.
4. The system uses distilled models and the QRM module is integrated. A different model is assigned to each resident according to their risk of falling (see Table 2). Only

two nodes are required to run this configuration, resulting in the best alternative considering the accuracy vs. required edge nodes trade-off.

In Table 4, we show a comparison of how optimal each system configuration is, again using the optimization in Equation (1). The table shows the scenarios along with their average accuracy (F1-Score), the number of edge processors they use, and the optimization value shown in the last column (lower values are better). We use α to weigh how reducing the total computational load consumed by the models impacts the optimization, and β to weigh the evaluation performance that the solution offers (in our case, $\alpha = 0.15$, $\beta = 5$).

Table 4. Analysis of system configuration in different scenarios.

Scenario	Distilled Models	QRM	F1-Score ↑	# Edge ↓	Optimization ↓
1	✗	✗	88.18	8	1.4549
2	✓	✗	88.13	3	0.9775
3	✓	✗	82.65	1	0.9875
4	✓	✓	85.45	2	**0.9675**

Bolded values represent the best-performing configuration. Arrows next to metrics indicate optimization direction: ↑ denotes that a higher value is better, while ↓ denotes that a lower value is better.

First, we observe how using distilled models is crucial to significantly improve the system optimal configuration (Scenario#1 vs. Scenario#2) by reducing the number of edge nodes from eight to only three, with very limited loss in evaluation performance. Finally, our proposal (Scenario#4) that uses distilled models that integrates the QRM tool and assigns a different model to each resident based on their risk of falling, leads to the best result in the optimization (minimum). This alternative offers the best evaluation performance vs. resource usage trade-off, requiring only two nodes that also guarantee real-time processing (see also Figure 10): edge_node#1 analyzes three low-level, one mid-level, and one high-level fall risk residents (amounting to 10 + 10 + 10 + 17 + 32 = 79% of computational load); and edge_node#2 takes care of video streams from one mid-level, and two high-level fall risk residents (17 + 32 + 32 = 81%). Eventually, if a new resident needs to be monitored, it will be assigned to edge_node#1, the node with the lowest workload.

Note that this adaptive distributed system is highly scalable. New rooms to be monitored can be easily added without the need of deploying more processing nodes or ad-hoc networks, until the edge processing nodes reach their maximum computational capacity.

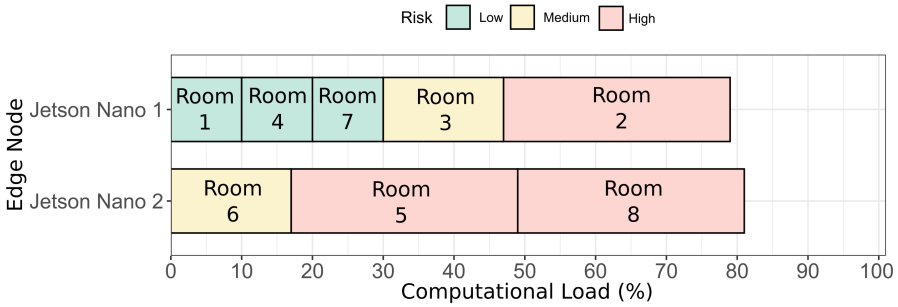

Figure 10. Computational load for the optimal configuration of the edge devices analyzing different residents of the nursing home in Figure 1. Edge_node#1 analyses residents from rooms 1, 2, 3, 4, and 7. Edge_node#2 processes the cameras from rooms 5, 6, and 8. The load balancing algorithm distributes processing among edge nodes dynamically. If a new room needs to be analyzed, it will be automatically assigned to the least loaded node. Bear in mind that the adaptive load balancing tool also continuously monitors the task allocation to equally distribute processing across nodes as described in Section 3.3.

6. Conclusions

In this work, we presented an optimized edge-cloud system for activity monitoring using knowledge distillation. We propose an efficient CPS for indoor action monitoring to be deployed in facilities such as nursing homes, providing better care for residents while reducing long-term care costs via edge intelligence.

Edge Computing enables efficient activity recognition, real-time execution, and privacy preservation. However, local processing devices are also very limited in resources. Therefore, it is crucial to optimize the DL architectures to embed them on the edge devices, enabling AI on them. We prove how the distillation of knowledge from more complex neural architectures improves accuracy of efficient models, while maintaining run-time operation. Also, it offers accuracy on par with models that require up to 10x more resources. Specifically, distilling from an ensemble of multimodal streams improves the F1-Score of one of the most computationally efficient models (Distilled RGBI3D_16_112) by 8.3% (reaching up to 82.65%) while performing inference at more than 250 fps.

The overall optimization of the system is addressed by the QRM tool that runs on the central server. Most importantly, the QRM tool that monitors node- and system-level qualities also carries out runtime reconfiguration to optimize the resources on the node and improve the recognition of falls for up to 4% (from 93.06% to 96.77%, a very relevant increase considering the critical nature of this action). In our case, the QRM also runs a resource-based adaptive load balancing algorithm that performs the runtime distribution of computation among the available nodes, preventing overloading while guaranteeing real-time operation.

Our edge-cloud system integrates the aforementioned components with the focus on leveraging an accurate and resource-efficient solution. In particular, the integration of these modules reduces up to 75% the computational budget required to deploy the system, and consequently also reducing the hardware cost: in our use case scenario, the initial system without QRM requires eight edge devices and the last one only two.

Moreover, we are releasing our Indoor Action Dataset in an effort to foster research for these indoor monitoring systems.

As future work, we plan to build a system that provides guidelines for activities and reports data to healthcare professionals based on the continuous monitoring of indoor actions for promoting healthy lifestyles.

Author Contributions: Conceptualization, D.D. and F.B.; methodology, F.B.; software, D.D.; validation, D.D. and E.M.O.; formal analysis, F.B.; investigation, D.D.; resources, E.R.; data curation, E.M.O.; writing—original draft preparation, D.D., E.R. and E.M.O.; writing, review and editing, F.B.; visualization, D.D.; supervision, F.B.; project administration, E.R.; funding acquisition, E.R. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Grant PID2022-141466OB-I00 funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU.

Data Availability Statement: The newly introduced Indoor Action Dataset includes recordings of several common scenes for indoor daily life activities. This manually curated dataset used for training and evaluating the Deep Learning architectures developed in this work is publicly available at https://github.com/DaniDeniz/IndoorActionDataset (accessed on 2 October 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AAL	Ambient Assisted Living
AI	Artificial Intelligence
CPS	Cyber–Physical System
DL	Deep Learning
EC	Edge Computing

EI	Edge Intelligence
HAR	Human Activity Recognition
IoT	Internet of Things
QRM	Quality and Resource Management
SoM	System-on-Module

References

1. Organisation for Economic Co-operation and Development. Long-Term Care Resources and Utilisation: Long-Term Care Recipients 2020. 2020. Available online: https://stats.oecd.org/Index.aspx?DatasetCode=HEALTH_STAT (accessed on 14 February 2024).
2. Vinciguerra, S.; Vinciguerra, M. Smart devices and healthy aging. *Nutr. Healthy Aging* **2019**, *5*, 13–19. [CrossRef]
3. Buyl, R.; Beogo, I.; Fobelets, M.; Deletroz, C.; Van Landuyt, P.; Dequanter, S.; Gorus, E.; Bourbonnais, A.; Giguère, A.; Lechasseur, K.; et al. e-Health interventions for healthy aging: A systematic review. *Syst. Rev.* **2020**, *9*, 128. [CrossRef]
4. Bergen, G.; Stevens, M.R.; Burns, E.R. Falls and fall injuries among adults aged ≥ 65 years—United States, 2014. *Morb. Mortal. Wkly. Rep.* **2016**, *65*, 993–998. [CrossRef] [PubMed]
5. Yacchirema, D.; de Puga, J.S.; Palau, C.; Esteve, M. Fall detection system for elderly people using IoT and ensemble machine learning algorithm. *Pers. Ubiquitous Comput.* **2019**, *23*, 801–817. [CrossRef]
6. Ganesan, B.; Gowda, T.; Al-Jumaily, A.; Fong, K.; Meena, S.; Tong, R. Ambient assisted living technologies for older adults with cognitive and physical impairments: A review. *Eur. Rev. Med. Pharmacol. Sci.* **2019**, *23*, 10470–10481.
7. Kannus, P.; Sievänen, H.; Palvanen, M.; Järvinen, T.; Parkkari, J. Prevention of falls and consequent injuries in elderly people. *Lancet* **2005**, *366*, 1885–1893. [CrossRef]
8. Al-Naime, K.; Al-Anbuky, A.; Mawston, G. Internet of Things Gateway Edge for Movement Monitoring in a Smart Healthcare System. *Electronics* **2023**, *12*, 3449. [CrossRef]
9. Chui, K.T.; Gupta, B.B.; Liu, J.; Arya, V.; Nedjah, N.; Almomani, A.; Chaurasia, P. A Survey of Internet of Things and Cyber-Physical Systems: Standards, Algorithms, Applications, Security, Challenges, and Future Directions. *Information* **2023**, *14*, 388. [CrossRef]
10. Calderita, L.V.; Vega, A.; Barroso-Ramírez, S.; Bustos, P.; Núñez, P. Designing a cyber-physical system for ambient assisted living: A use-case analysis for social robot navigation in caregiving centers. *Sensors* **2020**, *20*, 4005. [CrossRef]
11. Deniz, D.; Isern, J.; Solanti, J.; Jääskeläinen, P.; Hnětynka, P.; Bulej, L.; Ros, E.; Barranco, F. Efficient reconfigurable CPS for monitoring the elderly at home via Deep Learning. *Eng. Appl. Artif. Intell.* 2022, *in press*.
12. Deniz, D.; Barranco, F.; Isern, J.; Ros, E. Reconfigurable cyber-physical system for lifestyle video-monitoring via deep learning. In Proceedings of the 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vienna, Austria, 8–11 September 2020; IEEE: New York, NY, USA, 2020; Volume 1, pp. 1705–1712.
13. Isern, J.; Barranco, F.; Deniz, D.; Lesonen, J.; Hannuksela, J.; Carrillo, R.R. Reconfigurable cyber-physical system for critical infrastructure protection in smart cities via smart video-surveillance. *Pattern Recognit. Lett.* **2020**, *140*, 303–309. [CrossRef]
14. Sau, C.; Rinaldi, C.; Pomante, L.; Palumbo, F.; Valente, G.; Fanni, T.; Martinez, M.; van der Linden, F.; Basten, T.; Geilen, M.; et al. Design and management of image processing pipelines within CPS: Acquired experience towards the end of the FitOptiVis ECSEL Project. *Microprocess. Microsyst.* **2021**, *87*, 104350. [CrossRef]
15. Maheshwari, S.; Raychaudhuri, D.; Seskar, I.; Bronzino, F. Scalability and performance evaluation of edge cloud systems for latency constrained applications. In Proceedings of the 2018 IEEE/ACM Symposium on Edge Computing (SEC), Seattle, WA, USA, 25–27 October 2018; IEEE: New York, NY, USA, 2018; pp. 286–299.
16. Faliagka, E.; Skarmintzos, V.; Panagiotou, C.; Syrimpeis, V.; Antonopoulos, C.P.; Voros, N. Leveraging Edge Computing ML Model Implementation and IoT Paradigm towards Reliable Postoperative Rehabilitation Monitoring. *Electronics* **2023**, *12*, 3375. [CrossRef]
17. Cao, K.; Hu, S.; Shi, Y.; Colombo, A.W.; Karnouskos, S.; Li, X. A survey on edge and edge-cloud computing assisted cyber-physical systems. *IEEE Trans. Ind. Inform.* **2021**, *17*, 7806–7819. [CrossRef]
18. Shekhar, S.; Gokhale, A. Dynamic resource management across cloud-edge resources for performance-sensitive applications. In Proceedings of the 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Madrid, Spain, 14–17 May 2017; IEEE: New York, NY, USA, 2017; pp. 707–710.
19. Patel, A.D.; Shah, J.H. Performance analysis of supervised machine learning algorithms to recognize human activity in ambient assisted living environment. In Proceedings of the 2019 IEEE 16th India Council International Conference (INDICON), Rajkot, India, 13–15 December 2019; IEEE: New York, NY, USA, 2019; pp. 1–4.
20. Ardito, C.; Di Noia, T.; Di Sciascio, E.; Lofú, D.; Mallardi, G.; Pomo, C.; Vitulano, F. Towards a trustworthy patient home-care thanks to an edge-node infrastructure. In Proceedings of the HCSE 2020—8th IFIP WG 13.2 International Working Conference, Eindhoven, The Netherlands, 30 November–2 December 2020; Springer: New York, NY, USA, 2020; pp. 181–189.
21. Dang, L.M.; Min, K.; Wang, H.; Piran, M.J.; Lee, C.H.; Moon, H. Sensor-based and vision-based human activity recognition: A comprehensive survey. *Pattern Recognit.* **2020**, *108*, 107561. [CrossRef]
22. Qiu, S.; Fan, T.; Jiang, J.; Wang, Z.; Wang, Y.; Xu, J.; Sun, T.; Jiang, N. A novel two-level interactive action recognition model based on inertial data fusion. *Inf. Sci.* **2023**, *633*, 264–279. [CrossRef]

23. Hegde, N.; Bries, M.; Swibas, T.; Melanson, E.; Sazonov, E. Automatic recognition of activities of daily living utilizing insole-based and wrist-worn wearable sensors. *IEEE J. Biomed. Health Inform.* **2017**, *22*, 979–988. [CrossRef] [PubMed]
24. Mardanpour, M.; Sepahvand, M.; Abdali-Mohammadi, F.; Nikouei, M.; Sarabi, H. Human activity recognition based on multiple inertial sensors through feature-based knowledge distillation paradigm. *Inf. Sci.* **2023**, *640*, 119073. [CrossRef]
25. Chen, C.F.R.; Panda, R.; Ramakrishnan, K.; Feris, R.; Cohn, J.; Oliva, A.; Fan, Q. Deep analysis of cnn-based spatio-temporal representations for action recognition. In Proceedings of the IEEE Conference on CVPR, Nashville, TN, USA, 20–25 June 2021; pp. 6165–6175.
26. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge distillation: A survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [CrossRef]
27. Sepahvand, M.; Abdali-Mohammadi, F. A novel method for reducing arrhythmia classification from 12-lead ECG signals to single-lead ECG with minimal loss of accuracy through teacher-student knowledge distillation. *Inf. Sci.* **2022**, *593*, 64–77. [CrossRef]
28. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.
29. Sigurdsson, G.A.; Varol, G.; Wang, X.; Farhadi, A.; Laptev, I.; Gupta, A. Hollywood in homes: Crowdsourcing data collection for activity understanding. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: New York, NY, USA, 2016; pp. 510–526.
30. Challa, H.; Niu, N.; Johnson, R. Faulty requirements made valuable: On the role of data quality in deep learning. In Proceedings of the 2020 IEEE Seventh International Workshop on Artificial Intelligence for Requirements Engineering (AIRE), Zurich, Switzerland, 1 September 2020; IEEE: New York, NY, USA, 2020; pp. 61–69.
31. Carreira, J.; Zisserman, A. Quo vadis, action recognition? A new model and the kinetics dataset. In Proceedings of the IEEE Conference on CVPR, Honolulu, HI, USA, 21–26 July 2017; pp. 6299–6308.
32. Soomro, K.; Zamir, A.R.; Shah, M. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv* **2012**, arXiv:1212.0402.
33. Park, G.; Gu, B.; Heo, J.; Yi, S.; Han, J.; Park, J.; Min, H.; Piao, X.; Cho, Y.; Park, C.W.; et al. Adaptive load balancing mechanism for server cluster. In Proceedings of the 2006 International Conference on Computational Science and Its Applications, Glasgow, UK, 8–11 May 2006; Springer: New York, NY, USA, 2006; pp. 549–557.
34. Khan, S.; Nazir, B.; Khan, I.A.; Shamshirband, S.; Chronopoulos, A.T. Load balancing in grid computing: Taxonomy, trends and opportunities. *J. Netw. Comput. Appl.* **2017**, *88*, 99–111. [CrossRef]
35. Saleem, G.; Bajwa, U.I.; Raza, R.H. Toward human activity recognition: A survey. *Neural Comput. Appl.* **2023**, *35*, 4145–4182. [CrossRef]
36. Kong, Y.; Fu, Y. Human action recognition and prediction: A survey. *Int. J. Comput. Vis.* **2022**, *130*, 1366–1401. [CrossRef]
37. Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE Conference on CVPR, Boston, MA, USA, 7–12 June 2015; pp. 2625–2634.
38. Li, K.; Wang, Y.; He, Y.; Li, Y.; Wang, Y.; Wang, L.; Qiao, Y. UniFormerV2: Spatiotemporal Learning by Arming Image ViTs with Video UniFormer. *arXiv* **2022**, arXiv:2211.09552. [CrossRef]
39. Li, X.; Wang, L. ZeroI2V: Zero-Cost Adaptation of Pre-trained Transformers from Image to Video. *arXiv* **2023**, arXiv:2310.01324. [CrossRef]
40. Srivastava, S.; Sharma, G. OmniVec2 - A Novel Transformer Based Network for Large Scale Multimodal and Multitask Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, 16–22 June 2024; IEEE: New York, NY, USA, 2024; pp. 27402–27414. [CrossRef]
41. Wang, L.; Huang, B.; Zhao, Z.; Tong, Z.; He, Y.; Wang, Y.; Wang, Y.; Qiao, Y. Videomae v2: Scaling video masked autoencoders with dual masking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 14549–14560.
42. Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.A.; Serre, T. HMDB: A large video database for human motion recognition. In Proceedings of the IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, 6–13 November 2011; Metaxas, D.N., Quan, L., Sanfeliu, A., Gool, L.V., Eds.; IEEE Computer Society: New York, NY, USA, 2011; pp. 2556–2563. [CrossRef]
43. Xie, S.; Sun, C.; Huang, J.; Tu, Z.; Murphy, K. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8 September 2018; pp. 305–321.
44. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 448–456.
45. Chen, Z.; Cai, C.; Zheng, T.; Luo, J.; Xiong, J.; Wang, X. RF-Based Human Activity Recognition Using Signal Adapted Convolutional Neural Network. *IEEE Trans. Mob. Comput.* **2023**, *22*, 487–499. [CrossRef]
46. Li, G.; Ma, X.; Wang, X.; Yue, H.; Li, J.; Liu, L.; Feng, X.; Xue, J. Optimizing deep neural networks on intelligent edge accelerators via flexible-rate filter pruning. *J. Syst. Archit.* **2022**, *124*, 102431. [CrossRef]
47. Courbariaux, M.; Bengio, Y.; David, J.P. Binaryconnect: Training deep neural networks with binary weights during propagations. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 3123–3131.

48. Yu, X.; Liu, T.; Wang, X.; Tao, D. On compressing deep models by low rank and sparse decomposition. In Proceedings of the IEEE Conference on CVPR, Honolulu, HI, USA, 21–26 July 2017; pp. 7370–7379.
49. Kwasniewska, A.; Szankin, M.; Ozga, M.; Wolfe, J.; Das, A.; Zajac, A.; Ruminski, J.; Rad, P. Deep learning optimization for edge devices: Analysis of training quantization parameters. In Proceedings of the IECON 2019—45th Annual Conf. of the IEEE Industrial Electronics Society, Lisbon, Portugal, 14–17 October 2019; IEEE: New York, NY, USA, 2019; Volume 1, pp. 96–101.
50. Tonellotto, N.; Gotta, A.; Nardini, F.M.; Gadler, D.; Silvestri, F. Neural network quantization in federated learning at the edge. *Inf. Sci.* **2021**, *575*, 417–436. [CrossRef]
51. Koc, W.W.; Chang, Y.T.; Yu, J.Y.; İk, T.U. Text-to-Speech with Model Compression on Edge Devices. In Proceedings of the 2021 22nd Asia-Pacific Network Operations and Management Symposium (APNOMS), Tainan, Taiwan, 8–10 September 2021; IEEE: New York, NY, USA, 2021; pp. 114–119.
52. Luo, H.; Chen, T.; Li, X.; Li, S.; Zhang, C.; Zhao, G.; Liu, X. KeepEdge: A Knowledge Distillation Empowered Edge Intelligence Framework for Visual Assisted Positioning in UAV Delivery. *IEEE Trans. Mob. Comput.* **2022**, *22*, 4729–4741. [CrossRef]
53. Yamazaki, M.; Mori, E. Rethinking Deconvolution for 2D Human Pose Estimation Light yet Accurate Model for Real-time Edge Computing. In Proceedings of the 2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021), Jodhpur, India, 15–18 December 2021; IEEE: New York, NY, USA, 2021; pp. 1–5.
54. Wang, Y.; Li, X.; Shi, M.; Xian, K.; Cao, Z. Knowledge distillation for fast and accurate monocular depth estimation on mobile devices. In Proceedings of the IEEE Conference on CVPR, Nashville, TN, USA, 20–25 June 2021; pp. 2457–2465.
55. Beyer, L.; Zhai, X.; Royer, A.; Markeeva, L.; Anil, R.; Kolesnikov, A. Knowledge distillation: A good teacher is patient and consistent. *arXiv* **2021**, arXiv:2106.05237.
56. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on CVPR, Salt Lake, UT, USA, 18–23 June 2018; pp. 4510–4520.
57. Stroud, J.; Ross, D.; Sun, C.; Deng, J.; Sukthankar, R. D3d: Distilled 3D networks for video action recognition. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Snowmass, CO, USA, 1–5 March 2020; pp. 625–634.
58. Kullback, S. *Information Theory and Statistics*; Dover Publications: New York, NY, USA, 1968.
59. Anwar, A.; Raychowdhury, A. Autonomous navigation via deep reinforcement learning for resource constraint edge nodes using transfer learning. *IEEE Access* **2020**, *8*, 26549–26560. [CrossRef]
60. Lahsen-Cherif, I.; Liu, H.; Lamy-Bergot, C. Real-Time Drone Anti-Collision Avoidance Systems: An Edge Artificial Intelligence Application. In Proceedings of the 2022 IEEE Radar Conference, New York, NY, USA, 21–25 March 2022; pp. 1–6.
61. Azizpour, M.; da Roza, F.; Bajcinca, N. End-to-End Autonomous Driving Controller Using Semantic Segmentation and Variational Autoencoder. In Proceedings of the 2020 7th International Conference on Control, Decision and Information Technologies (CoDIT), Prague, Czech Republic, 29 June–2 July 2020; IEEE: New York, NY, USA, 2020; Volume 1, pp. 1075–1080.
62. Yoshikawa, Y.; Lin, J.; Takeuchi, A. Stair actions: A video dataset of everyday home actions. *arXiv* **2018**, arXiv:1804.04326.
63. Monfort, M.; Andonian, A.; Zhou, B.; Ramakrishnan, K.; Bargal, S.A.; Yan, T.; Brown, L.; Fan, Q.; Gutfreund, D.; Vondrick, C.; et al. Moments in time dataset: One million videos for event understanding. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *42*, 502–508. [CrossRef]
64. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
65. Charfi, I.; Miteran, J.; Dubois, J.; Atri, M.; Tourki, R. Optimized spatio-temporal descriptors for real-time fall detection: Comparison of support vector machine and adaboost-based classification. *J. Electron. Imaging* **2013**, *22*, 041106. [CrossRef]