Representation Challenges. New Frontiers of AR and AI Research for Cultural Heritage and Innovative Design.

Efficient Deep Learning Models for Privacy-preserving People Counting on Low-resolution Infrared Arrays

Chen Xie, *Member, IEEE,* Francesco Daghero, *Member, IEEE,* Yukai Chen, *Member, IEEE,* Marco Castellano, Luca Gandolfi, Andrea Calimera, *Member, IEEE,* Enrico Macii, *Fellow, IEEE,* Massimo Poncino, *Fellow, IEEE,* and Daniele Jahier Pagliari *Member, IEEE*

***Abstract*—Ultra-low-resolution Infrared (IR) array sensors offer a low-cost, energy-efficient, and privacy-preserving solution for people counting, with applications such as occupancy monitoring and visitor flow analysis in private and public spaces. Previous work has shown that Deep Learning (DL) can yield superior performance on this task. However, the literature was missing an extensive comparative analysis of various efficient DL architectures for IR array-based people counting, that considers not only their accuracy, but also the cost of deploying them on memory- and energy-constrained Internet of Things (IoT) edge nodes. Such analysis is key for system designers, since it helps them select the most appropriate DL model given the constraints of their target hardware. In this work, we address this need by comparing 6 different DL architectures on a novel dataset composed of IR images collected from a commercial 8x8 array, which we made openly available. With a wide architectural exploration of each model type, we obtain a rich set of Pareto-optimal solutions, spanning cross-validated balanced accuracy scores in the 55.70-82.70% range. When deployed on a commercial Microcontroller (MCU) by STMicroelectronics, the STM32L4A6ZG, these models occupy 0.41-9.28kB of memory, and require 1.10-7.74ms per inference, while consuming 17.18-120.43 μJ of energy. Our models are significantly more accurate than a previous deterministic method (up to +39.9%), while being up to 3.53x faster and more energy efficient. So, our work serves also as a demonstration that DL can not only achieve higher accuracy, but also higher efficiency compared to classic algorithms for this type of task. Further, our models' accuracy is comparable to state-of-the-art DL solutions on similar resolution sensors, despite a much lower complexity. All our models enable continuous, real-time inference on a MCU-based IoT node, with years of autonomous operation without battery recharging.**

***Index Terms*—Infrared Sensors, People Counting, Edge Computing, Deep Learning, Microcontrollers, Energy Efficiency**

I. Introduction

DEEP learning (DL) has recently received attention in many Internet of Things (IoT) applications, ranging from embedded computer vision to time series forecasting, due to its remarkable predictive performance [1]–[5]. A direct execution of DL-based prediction tasks on *extreme-edge* IoT nodes such as smart sensors can provide unique benefits compared with traditional cloud-based approaches, by eliminating the need of transmitting large amounts of raw data through a wireless network link [1], [6], [7]. Specifically, on-device execution makes the IoT node responsive even in bad or no-connectivity conditions, with a predictable latency. Moreover, the only information (optionally) transmitted to the cloud is the aggregated output of the DL model, e.g., a class label. This is beneficial for confidentiality, as it reduces the risk of accidental or malicious leakage of sensitive raw data (e.g., images, audio, video, etc) [1], [6].

However, DL algorithms originally designed for the cloud are energy-hungry and require high computational complexity, far beyond the capacity of memory- and energy-constrained IoT nodes, which are typically based on battery-operated and resource-limited Microcontrollers (MCUs). Bridging this gap in order to successfully deploy DL applications at the extreme edge requires a thorough selection of the employed models and of the corresponding hyper-parameters [5].

Among the IoT applications that benefit from DL, people counting is increasingly popular due to its vast number of use cases in public safety, urban planning and commercial assistance [8]. Practical tasks range from monitoring the occupancy of indoor work spaces, museums and hospitals, to analysing the people flow statistics at the entrance of shops, supermarkets and other public places, to monitoring social distance violations or safety norms infringements especially in the context of the COVID-19 pandemic [9]–[11].

There exist a wide range of technical solutions based on IoT for people counting, mainly split into two categories: instrumented and uninstrumented [12]. The former approaches exploit the transceivers present in devices already owned by (or given to) users, such as smartphones, smartwatches, or tags [13]. However, these methods are heavily limited by voluntary participation and instrumental equipment, and are hard to apply in most real-world scenarios, especially in public places. On the other hand, uninstrumented solutions are free of the individuals' participation and rely on external sensors, such as proximity sensors, optical cameras, infrared arrays

C. Xie, F. Daghero, A. Calimera, M. Poncino and D. Jahier Pagliari are with the Department of Control and Computer Engineering, Politecnico di Torino, Turin, 10129, Italy, e-mail: name.first_surname@polito.it.

Y. Chen is with IMEC, Leuven, 3001, Belgium, e-mail: yukai.chen@imec.be.

M. Castellano and L. Gandolfi are with ST Microelectronics S.r.l., Cornaredo, 20010, Italy, e-mail:name.surname@st.com.

E. Macii is with the Interuniversity Department of Regional and Urban Studies and Planning, Politecnico di Torino, Turin, 10129, Italy, e-mail:enrico.macii@polito.it

Manuscript received January XX, XXXX; revised January XX, XXXX.

etc [12], [14]–[16]. Among these, infrared beam sensors and passive infrared sensors are inexpensive and simple to use, but rely on specific conditions such as object motion, and cannot easily distinguish multiple nearby people, which makes them often inaccurate [17]. As computer vision and video analysis techniques keep improving, vision-based people counting solutions are thus progressively replacing them. Most current vision-based approaches use optical cameras, processing each frame with a Machine Learning (ML) algorithm to recognize and locate individuals [18]–[20]. While effective, they face severe privacy issues, since sensitive details of individuals such as facial information and body morphology are also recorded and processed.

In this scenario, **low-resolution infrared (IR) array** sensors offer a promising alternative, with advantages in terms of low energy consumption, low cost and privacy preservation. The latter is due to the fact that IR arrays only detect body temperatures, and given their low spatial resolutions (typically 8x8 or 16x16 thermal pixels), they can only capture the rough body shapes, hiding all privacy-sensitive details of individuals. While other works have studied the combination of IR array sensors with DL models for people counting [21]–[24], they: i) target higher resolution arrays, which simplifies the task but results in higher cost, higher energy consumption, and lower privacy and ii) consider a single type of DL model.

In this work, we perform the first detailed exploration and comparison of multiple DL model families for people counting based on a *single, ultra-low-resolution (8x8)* IR array. We focus on *efficient* models, deployable on MCU-class platforms. The following is a summary of our main contributions:

- We compare multiple efficient DL models for predicting the people count based on data from a single 8x8 IR array. For each type of model, we perform an extensive architecture exploration, obtaining a rich set of Pareto-optimal solutions in terms of performance and complexity.
- Analyzing the results of our exploration, we derive some interesting guidelines on the best type of model to prefer based on the target accuracy range and cost metric (model size or operations count). Overall, our models span a 55.70%-82.70% range in balanced accuracy, with parameters and operation counts varying in 0.4k-2.4k and 2.9k-20k respectively. The best balanced accuracy is up to 39.9% higher than the one of a state-of-the-art deterministic algorithm [25], and comparable with previous DL solutions on similar resolution data [21].
- We deploy some of the found models on a commercial MCU by STMicroelectronics, the STM32L4A6ZG, obtaining model size, inference latency, and inference energy values ranging in 0.41-9.28kB, 1.10-7.74ms and 17.18-120.43μJ respectively. Our models are up to 3.53x faster and more energy efficient than [25], while also being significantly more accurate. Furthermore, all of them allow real-time inference at 10 frames per second with very low energy consumption, which would permit years of continuous operation without battery recharging.

The rest of the paper is structured as follows: Section II provides the background and overviews the related work on person counting applications based on IR sensors at edge. Section III presents a detailed description of the target dataset and of the various considered DL models, and describes the architecture exploration and deployment flow. Section IV reports the experimental results, and Section V concludes the paper.

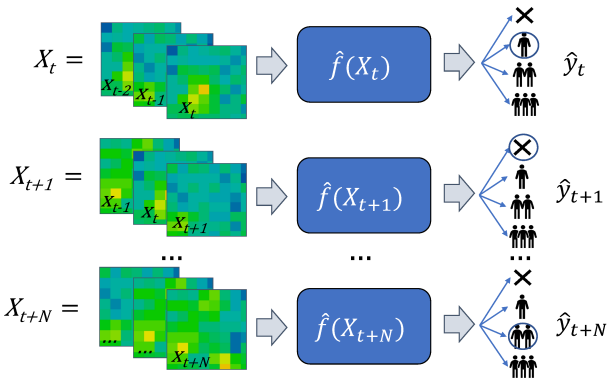


Fig. 1. People counting with IR array sensors: problem formulation. Depending on the work, the prediction function $\hat{f}(X)$ can be obtained either with a rule-based deterministic algorithm or learned from data using ML/DL, and the predicted person count $\hat{y}_t$ can be either a scalar or a class label.

II. Background and Related Works

People counting based on visual data is typically formulated as an object recognition problem [34]. Several sensor types have been utilized to implement both single- or multi-sensor systems for this task [11], [35]. When considering this kind of sensor, the problem reduces to a classification or regression on image-like data, as shown in Fig. 1. Namely, at time instant t, and calling x_t the latest IR frame (i.e., "image") collected, the input to the recognition model is either a single frame $X_t = x_t$ or a *window* of consecutive frames $X = \{x_{t-W+1},...,x_t\}$, where W is the window size ($W = 3$ in the figure). The output is the predicted people count $\hat{y}_t = \hat{f}(X_t)$, obtained either as a continuous scalar, then rounded to the nearest integer (regression formulation), or as a categorical value corresponding to one in a set of possible counts (classification formulation). The input/output relationship $\hat{f}(X)$ can be either obtained with a deterministic rule-based algorithm or learned from a training dataset using ML/DL approaches.

A summary of the most relevant literature works on people counting with multi-pixel IR arrays is reported in Table I. In particular, we report the sensor model and resolution, its position, the target dataset, the counting algorithms, and the IoT device considered the in each work for deployment. In detail, prior works leverage both deterministic algorithms [11], [25]–[28], [30], classic ML models [32] or DL [21]–[24]. Among deterministic approaches, [11] implemented a novel real-time pattern recognition algorithm to process data sensed from doorway-mounted low-resolution IR array sensors to determine the number of people in a room. Similarly, [26] also takes advantage of a doorway-mounted sensor, combined with a body extraction and localization algorithm, and background determination. [27] proposed a similar lightweight deterministic solution based on a single array sensor positioned on a

TABLE I
STATE-OF-THE-ART PEOPLE COUNTING SOLUTIONS BASED ON INFRARED ARRAYS

Work	Sensor	Positioning	Dataset	Algorithm	Deployment Target
Perra et al. [11]	Grid EYE (8x8)	Door	Private	Deterministic	Z-Uno
Mohammadmoradi et al. [26]	Grid EYE (8x8)	Door	Private	Deterministic	Raspberry Pi Zero
Wang et al. [27]	MLX90641 (12x16)	Door	Private	Deterministic	ESP8266
Rabiee et al. [28]	Grid EYE (8x8)	Ceiling	Private/Nagoya-OMRON Dataset [29]	Deterministic	-
Singh et al. [30]	MLX90621 (16x4)	Ceiling/Side Wall	Private	Deterministic	Arduino Uno
Panasonic [25]	Grid EYE (8x8)	Ceiling	LINAIGE [31] (*)	Deterministic	STM32L4 (*)
Chidurala et al. [32]	Grid EYE (8x8) MLX90640 (32x24) Lepton (80x60)	Ceiling	Private	Naive Bayes KNN SVM RF	Raspberry Pi 3
Bouazizi et al. [21]	MLX90640 (32x24)	Ceiling	Private	CNN	Raspberry Pi 3
Gomez et al. [22]	Lepton (80x60)	Wall	Private	CNN	NXP LPC54102
Metwaly et al. [23]	MLX90640 (32x24)	Ceiling	Private	FNN CNN GRU	STM32F4/F7
Kraft et al. [24]	MLX90640 (32x24)	Ceiling	Thermo Presence [24]	CNN	Raspberry Pi 4
Xie et al. [10]	Grid EYE (8x8)	Ceiling	LINAIGE [31]	CNN (2 variants)	STM32L4
Xie et al. [33]	Grid EYE (8x8)	Ceiling	LINAIGE [31]	Wake-up Trigger + CNN	STM32L4
This Work	Grid EYE (8x8)	Ceiling	LINAIGE [31]	CNN (4 variants) CNN-LSTM CNN-TCN	STM32L4

(*) These entries refer to our deployment of the method described in [25].

door, to monitor trajectories of objects entering and exiting a room, and estimate the indoor people count accordingly. While interesting due to their use of a single, low-resolution sensor, these works solve a simplified and limited-scope version of the generic people counting problem. In fact, they only permit the counting of people entering/exiting a room through a doorway.

A more general deterministic method based on a ceiling-mounted sensor is described in [25]. This solution is based on the separation of moving thermal objects from the background by means of smoothing, linear interpolation and hot area labeling and clustering. After that, threshold-based human detection is performed on each labelled thermal object to determine if it corresponds to a person or not. The reference background image is updated regularly to automatically filter stationary warm objects.

Furthermore, multi-sensor deterministic solutions have also been explored. Specifically, [28] proposed a people flow counting algorithm to monitor occupancy in smart buildings. To achieve this goal, multiple low-resolution sensors are deployed in connection points between different building areas, in order to count the number of people moving across adjacent zones. The work of [30], instead, presents a framework to count people indoors based on two deterministic algorithms. Their method requires three 16x4 thermal sensors deployed at different locations, pointing to x, y, and z directions respectively.

Among classic ML works, [32] considers three ceiling-mounted IR arrays with different resolutions (8x8, 32x24, 80x60). It applies several preprocessing and feature extraction steps (active pixel and active frame detection, connected components analysis, statistical features), and then compares multiple classification algorithms for people counting. The considered algorithms are Naive Bayes, K-Nearest Neighbors (KNN), Support Vector Machines (SVM) and Random Forests (RFs). On a private dataset, they show that, for the lowest-resolution array, the best score is achieved with a RF.

Lastly, several DL-based solutions have been proposed. The authors of [21] use a Convolutional Neural Network (CNN) with 9 convolutional layers and 1 dense layer to process data from a ceiling-mounted, 32x24 pixels IR sensor to locate and count people indoors. Optionally, their proposed method allows the collection of lower-resolution samples (down to 8x6 pixels) to reduce sensor costs, thanks to the usage of a separate 8-layer CNN for frame upscaling. [22] developed a head detection and people counting algorithm for wall-mounted sensors, based on a small-sized CNN model, and targeting a limited-memory low-power platform deployment, but focusing on a relatively high-resolution 80x60 pixels array. [23] considered Feedforward Neural Networks (FNNs), CNNs and Gated Recurrent Units (GRU) for indoor occupancy estimation, based on ceiling-mounted 24x32 resolution IR arrays. The work of [24] also adopts a ceiling-mounted 24x32 resolution IR array, and leverages an encoder-decoder CNN architecture (a simplified version of U-Net) to reconstruct the position of people in the frame.

Most recently, in our previous work of [10], we applied, to our knowledge for the first time, a DL model directly to the output of an ultra-low-resolution (8x8) array. However, that work considered a simplified version of the people counting problem, where the goal was simply to detect if the area covered by the sensor contained 2 or more people, in the context of social distance monitoring to combat the spread of COVID-19. The same task variant was tackled also in [33], where an additional deterministic wake-up-trigger was used to avoid useless invocations to the CNN when no people are present in the frame, further reducing the energy consumption of the system.

All aforementioned data-driven (ML or DL) works suffer from important limitations: [32] and [21] only focus on deploying person counting on a high-end mobile Central Processing Unit (CPU), and they do not report detailed deployment

results in terms of memory occupation of the models, inference latency, and energy consumption. [22] and [23] focus on relatively high-resolution arrays, which are more costly and power-consuming, besides possibly allowing the identification of users, thus reducing privacy. [21] supports low-resolution sensors only through an auxiliary CNN model for frame upscaling, which contributes to the total inference complexity. Furthermore, the excellent results obtained by many of these works [22], [23], [32] are tainted by unfair data splitting, based on a random sampling at the level of individual frames or sliding windows. As explained in Sec. III, this unrealistically oversimplifies the task. The only work that performs a realistic data split at the session level is [21]. Lastly, as mentioned, [10] and [33] focus on a simplified task variant.

In this work, we study for the first time the application of DL methods to a people counting problem based on the output on a *single, ceiling-mounted, ultra-low-resolution IR array* (only 8x8 pixels). With an extensive architectural exploration of six families of efficient DL models, and many different hyper-parameters settings, we show that DL can not only provide significantly better counting performance compared to a deterministic algorithm, but also obtain benefits in terms of energy consumption, and latency.

III. Materials and Methods

A. Motivation

The goal of this work is to perform a detailed exploration and comparison of various DL model families for people counting based on a single, ultra-low-resolution (8x8) IR array. We focus on this setup due to its several practical advantages with respect to multi-sensor or higher-resolution alternatives, including better privacy preservation, lower overall system cost, and lower power consumption, especially for processing, as shown in our results of Sec. IV. In fact, intuitively, processing multiple and/or higher resolution images requires a higher number of operations, regardless of the specific algorithm employed, which is critical for ultra-low-power systems that need to operate for years on battery power.

As anticipated in Sec. I, the main motivation for this study is that, to our knowledge, such an extensive comparison of DL models has not been performed before for this particular task. Therefore, we believe that it serves two related purposes: on the one hand, it provides a useful guidance for system designers that want to use this kind of sensor, for selecting an appropriate family of DL models based on the required accuracy and on the hardware memory, latency and energy constraints; on the other hand, it serves as a practical demonstration of the fact that DL can not only achieve higher accuracy, but also higher efficiency, compared to a classic algorithm [25].

B. Dataset

There exists several public datasets containing IR array thermal images. However, most of them have been collected by relatively high-resolution sensors from 160 x 120 to 640 x 480,

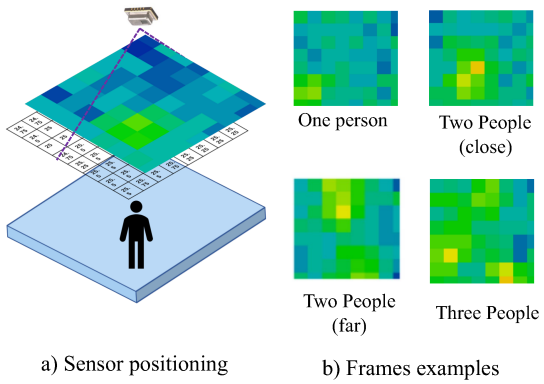


Fig. 2. Sensor mounting and example of the IR frames.

targeting applications such as pedestrian detection, and intelligent driving [36]–[39]. One public dataset containing low-resolution IR images is described in [40], and employs three wall-mounted sensors pointing in different directions, targeting human activity recognition tasks. Another low-resolution IR dataset containing 16x16 IR sensor arrays is described in [29], in this case for a ceiling-mounted sensor and specifically tailored for activity recognition. However, in this dataset at most one person appears in the frame, which deviates from the original people counting purpose. The dataset in [24] is instead dedicated to people counting applications, with up to 5 people in one frame. Each frame is annotated with the people's locations, which can be simply converted into counts, but the dataset is collected with a relatively high-resolution array (24x32 pixels). None of these datasets are suitable for experimenting on low-cost, energy-efficient people counting on ultra-low-resolution IR arrays. Indeed, as shown in Table I, most literature on this task uses privately-collected data.

Given this scenario, we collected and made openly available a new dataset called LINAIGE (Low-resolution INfrared-array data for AI on the edGE) [31]. LINAIGE targets specifically people counting and presence detection tasks in indoor environments, and its first version was described in our previous work of [10]. The dataset includes IR samples collected with a Panasonic Grid-EYE (AMG8833) sensor [25] outputting a 8 x 8 array, at 10 Frames Per Second (FPS). Each frame is associated with the corresponding people count label. During data collection, the sensor was ceiling-mounted as shown in Fig. 2a, and positioned in different indoor environments such as offices, laboratories and corridors, using a lens with a view angle of 60°. Volunteers passed in the view range of the sensor by walking, standing, running, etc, during a number of data collection sessions. Some examples of collected frames and corresponding people counts are shown in Fig. 2b. As detailed in [10], depending on the sensor height in different environments, the maximum distance between in-frame people varies in [1.53:2.04] m and the counting area is up to ≈ 2 m^2. People counting on larger areas can be simply achieved by combining the outputs of multiple sensors, appropriately positioned. With respect to the original dataset described in [10], this work is based on a new version with improved data quality. Namely, we removed the very rare frames with

$>$ 3 people (0.66% of the total), which were present only in one session, complicating the training and cross-validation of ML/DL models. Further, we also removed the shortest session (session 4 in [10]) which contained only 196 frames, i.e. around 20s worth of data, and unrealistically altered the recognition performance metrics. After these changes, the new dataset contains 25110 samples, split into 5 sessions. Each session is associated with a timestamp, environment name and room temperature.

IR frames have been labelled using a semi-automatic method: a data collection system based on a single-board computer named Raspberry Pi 3B has been set up, including both the IR sensor and an optical camera, pointing in the same direction and collecting synchronized frames. Optical frames have been then processed with a pre-trained object detection model (Mask R-CNN [41]) to automatically count the number of people in them, and associate the same count to the corresponding IR frame. The results have been double-checked by a human labeller to correct CNN mispredictions. Further, the human labeller also associated each frame with a binary *confidence* measure, which can be used to exclude frames for which it was difficult to assess the exact people count due to the imperfect alignment of the viewing angles between the IR sensor and the optical camera. More details on the labelling are found in [10].

In all experiments of this work, we excluded "hard-to-label" frames from training and testing, both for our method and for state-of-the-art comparisons. Moreover, in contrast to [10] where a simple per-session train/test split was used, here we adopt a per-session Cross Validation (CV) approach, to make our model evaluation independent from the characteristics of a specific test session. The cross validation strategy is shown in Table II. Given that Session 1 is significantly larger than all others (17958 frames versus a maximum of 2202 for other sessions, and 71% of the total data), we always kept it in the training set. Sessions 2, 3, 4 and 5 have been rotated as the test set in different iterations, with all other data in the training set, yielding 4 CV folds. This *leave-one-session-out* CV strategy ensures the fairness of model evaluation, by making sure that test frames correspond to a different environment, date-time, and room temperature setting compared to training frames. This is close to a realistic scenario, in which the system is likely to be tested in a different environment from where it was trained. In contrast, a purely random per-frame split would cause a leakage of information between training and testing, oversimplifying the problem.

C. Model Architectures

We considered six families of DL models to predict the people count in IR frames, exploring some of the key hyper-parameters of each. A graphic representation of all considered models is shown in Fig. 3.

1) Single-frame CNN: The first considered architecture is a simple CNN, which is known to be effective in many image-based pattern recognition tasks. The general template of the considered CNNs is shown in Figure 3a; it includes up to 2 Convolutional (Conv) layers with Rectified Linear Unit (ReLU) activation, 1 optional Max Pooling layer and up to 2 Fully Connected (FC) layers. The first FC layer has 64 hidden units and a ReLU activation, while the output layer has a number of neurons equal to the possible count "classes" (from 0 to 3 people, corresponding to 4 output neurons, in our experiments). Furthermore, compared to our previous work of [10], which focused on a simpler social distancing problem, we added Batch Normalization (BN) layers after each Conv layer to improve the classification performance. Utilizing this template as a starting point, a vast architecture exploration was performed, by eliminating/retaining layers which are enclosed in dashed boxes in Fig. 3a. Namely, we considered architectures with:

- 1 or 2 Conv layers, each followed by BN;
- 1 or 2 FC layers;
- 0 or 1 Max Pooling layers;

Besides varying the number of layers, we also explored the number of feature maps (i.e., *channels*) in each Conv layer, considering values in $\{8, 16, 32, 64\}$. Conv. and Pooling kernel sizes are fixed at 3x3 and 2x2 respectively. The input processed by this CNN model is a *single* IR array frame $X_t = x_t$, with a tensor shape (8, 8, 1). In total, we evaluate 48 different Single-frame CNN variants.

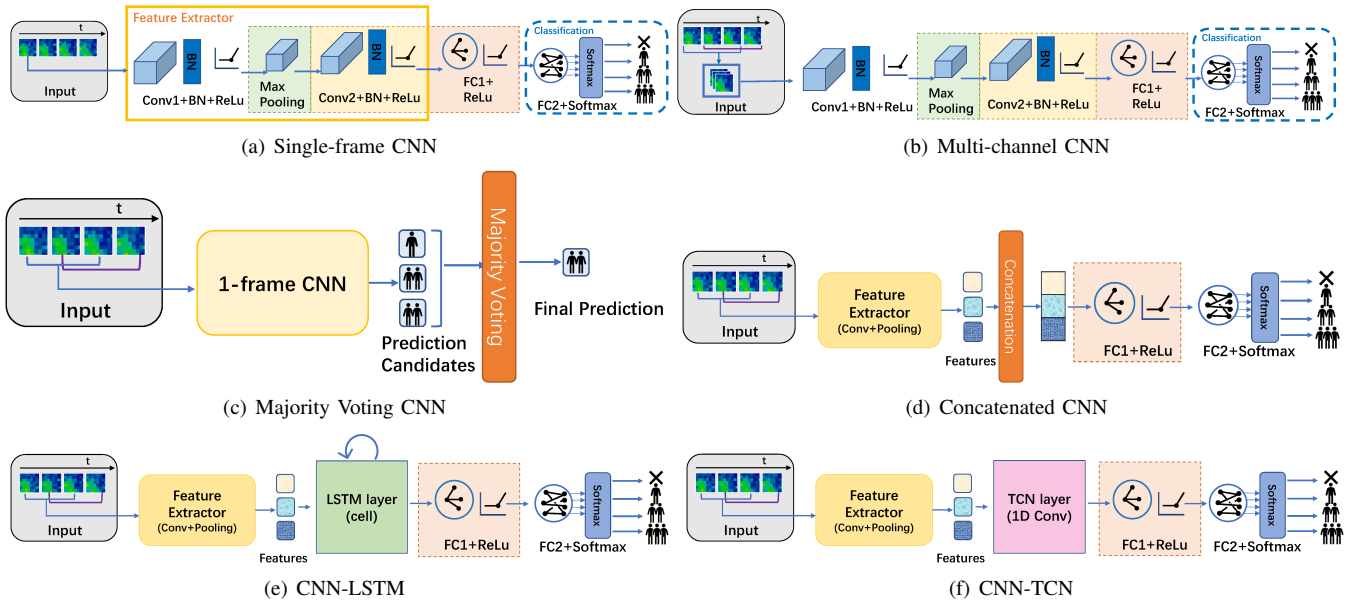
2) Multi-channel CNN: While the previous model considers a single IR array frame as input, all other models try to exploit the temporal information enclosed in a *sequence* of consecutive frames to improve the people counting accuracy. The rationale is that considering a sliding window of IR frames as input can reveal information on people movement, which in turn can improve the prediction accuracy in complex cases. For instance, Fig. 4 shows that a single hot area (highlighted by a purple box in the last frame) can be correctly associated with two people close to each other, rather than with a single person, by observing the movement of the two people (red and orange cycles) in preceding frames.

The first and simplest mechanism that we considered to process multiple IR frames consists in feeding them to a CNN as *different input channels*. Specifically, calling W the length of the sliding window, a tensor with shape (8, 8, W) is formed by stacking IR frames $X_t = \{x_{t-W+1},...,x_t\}$ along the channels axis. The tensor is then associated with the people count label of the *last* frame y_t for training and testing. These inputs and outputs are also used for all the other multi-frame architectures described in the following. The template of the Multi-channel CNN model is shown in Fig. 3b. We explored the same hyper-parameters settings considered for Single-frame CNNs in terms of the number of layers, and the number of Conv channels. In addition, we also varied the window size W in $\{3, 5, 7, 9\}$. This exploration is interesting because, intuitively, with a too-short window the advantages of accessing past frames are limited, whereas a too-long window will provide useless information (too far in the past), while increasing the time and memory complexity of the first Conv layer.

3) Majority Voting CNN: Majority voting is a simple yet effective ensemble learning approach that takes advantage of multiple classification results to generate final predictions with lower variance [42]. In recent years, several literature

TABLE II
DATASET STATISTICS AND CROSS VALIDATION STRATEGY.

Train Fold						Test Fold					
Session	Sample N.	People Counts Statistics [%]				Session	Sample N.	People Counts Statistics [%]			
		0	1	2	3			0	1	2	3
1, 3, 4, 5	23529	26.07	43.49	23.61	6.83	2	1581	14.86	30.68	54.46	0
1, 2, 4, 5	23591	22.37	44.03	26.84	6.77	3	1519	71.89	21.72	5.66	0.72
1, 2, 3, 5	22908	25.3	41.85	26.17	6.67	4	2202	26.02	51.27	19.16	3.54
1, 2, 3, 4	23260	24.69	43.02	26.08	6.20	5	1850	33.78	38.38	18.92	8.92



(a) Single-frame CNN (b) Multi-channel CNN

(c) Majority Voting CNN (d) Concatenated CNN

(e) CNN-LSTM (f) CNN-TCN

Fig. 3. Model Architectures considered in this work

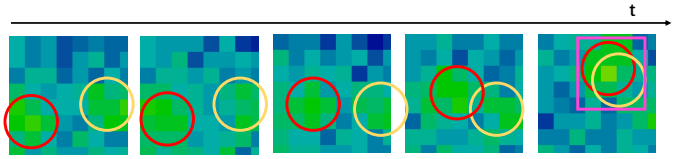


Fig. 4. Example of the IR frames sequence corresponding to 2 people moving close to each other.

works have applied this technique, using either different classifiers [43], multiple instances of the same model trained differently [44] or a single trained model fed with different inputs [45]. In our work, we follow the latter approach, applying majority voting (i.e., *mode inference*) to the W predictions obtained by executing a Single-frame CNN on each frame of the sliding window. A high-level scheme of this solution is shown in Fig. 3c. The clear advantage of this technique, from the point of view of edge inference, is that it requires approximately the *same memory* as a single-frame CNN, while possibly improving the prediction accuracy by filtering-out occasional mispredictions. The latency/energy cost for inference, instead, is roughly W times higher than that of a single-frame model. We consider again W values in $\{3, 5, 7, 9\}$. Note that majority voting requires an odd window size; thus, for fairness of comparison, all other multi-frame architectures have been tested only with odd W values.

To make our architectural exploration tractable, we consider the majority-voting models obtained using *Pareto-optimal Single-frame CNNs* as individual predictors. More specifically, we apply majority voting on top of all single-frame CNNs found in the Pareto front in terms of people counting accuracy versus model size or versus number of operations.

4) Concatenated CNNs: While the main advantage of majority voting is that it does not require extra trainable parameters, its main drawback is that it cannot assign different *importance* to the various IR frames in the sliding window. Intuitively, more recent frames should be given more importance to determine the people count, especially with large W. Although this could be approached by a *weighted voting* mechanism, such solution requires a difficult hand-tuning of the weights assigned to each frame. Thus, our next considered DL model exploits a feature concatenation approach to overcome this limitation [46], [47]. Specifically, as illustrated in Fig. 3d, each frame of the sliding window is individually fed into a feature extractor module to extract time-independent features. Then, all outputs are flattened and concatenated into a unique feature vector, and further processed by two FC layers to generate the final prediction. In this way, the training process can automatically assign appropriate weights to different frames' features. We consider the Conv and Pooling layer configurations (i.e., the part highlighted by an orange box in Fig. 3a) found in each Pareto-Optimal Single-frame model as possible feature extractors for concatenated CNNs. Furthermore, besides exploring the usual 4 values of

W, we also vary the number of neurons in the first FC layer in {8, 16, 32, 64}. Altogether, given N Pareto-optimal feature extractors, we evaluate a total of 4*4*N Concatenated CNNs.

5) CNN-LSTM: The next multi-frame model explicitly considers the time dependency between frames, replacing the simple feature concatenation with a Long-Short Term Memory (LSTM) cell. Several works have considered CNN-LSTM models to combine spatial and temporal information [48]–[50]. The closest work to ours is [29], which applied a CNN-LSTM for human activity recognition based on a 16x16 IR array. These works have demonstrated the remarkable performance achieved by CNN-LSTMs. However, LSTM cells are less hardware-friendly than CNNs [51] (see Sec. III-C6). Therefore, it is interesting to compare this model with other architectures, considering the trade-off between complexity and performance.

Our CNN-LSTM template is shown in Fig. 3e. The W feature extractor outputs are flattened and fed to the LSTM cell sequentially. One or two FC layers are then connected to the last hidden state produced by the LSTM to generate the output prediction. We apply the same feature extractors selection strategy illustrated above for Concatendated CNNs. Moreover, we vary W as before, and we also explore the number of hidden units in the LSTM cell, with values in {8, 16, 32, 64}. Again, given N Pareto-optimal feature extractors, a total of 4*4*N CNN-LSTM architectures are evaluated.

6) CNN-TCN: The last type of model considered is based on Temporal Convolutional Networks (TCN) [51] which have recently emerged as a more hardware-friendly alternative to LSTMs, and have been applied to several edge-relevant tasks [5], [52]. TCNs are simply 1D CNNs, with the peculiarity of using *causal* convolution, which is appropriate for time-series processing. Compared to LSTMs, these networks exhibit more data reuse and are more resilient to integer quantization, both of which are advantageous for edge deployment [51]. Therefore, our last architectural template is built by combining the outputs of the usual 2D CNN feature extractors applied to single IR frames with a single TCN layer, as shown in Fig. 3f. The TCN output is then flattened and fed to 1 or 2 FC layers to generate a prediction. We fix the 1D Conv kernel size at 3x1, and the dilation at 1. Besides varying W as in previous models, we explore the number of output channels of the TCN layer, considering values in {8, 16, 32, 64}. Therefore, with N feature extractors, also in this case we explore 4*4*N architectures.

D. Training and Deployment Flow

All models are trained with the leave-one-session-out CV strategy described in Sec. III-B. At first, we perform a standard floating point model training with Keras/TensorFlow 2.0 [53], for a maximum of 500 epochs per fold. We optimize a categorical cross-entropy loss function using the ADAM optimizer, with an initial learning rate of 10^{-3}. A learning rate reduction of factor 0.3 is applied when the training loss is stagnating, with a patience of 5 epochs. Early-stopping is applied after 10 non-improving epochs. Given the strong class imbalance of the LINAIGE dataset (see Table II), we apply class-dependent weights to the loss during training, which are computed as the inverse of the class frequencies.

After this initial floating point training, we quantize the parameters, inputs, outputs, and intermediate activations of the resulting models to 8-bit integers, using the TensorFlow Model Optimization (TFMOT) API. This step is important to further reduce the memory occupation, latency, and energy consumption of the models, when deployed on constrained MCU-based IoT nodes [54]. We then apply quantization-aware training (QAT) [55] to recover the accuracy drop due to quantization as much as possible. We use the same training protocol described above, with the only two differences that the initial learning rate is set to 5×10^{-4} and the learning rate scheduling and early stopping patience values are set to 10 and 20 epochs respectively. Note that the QAT of LSTM cells is not supported by the TFMOT API yet. Therefore, the CNN-LSTM models are directly deployed in floating point to the MCU. This turns out to be a major practical limitation of CNN-LSTMs.

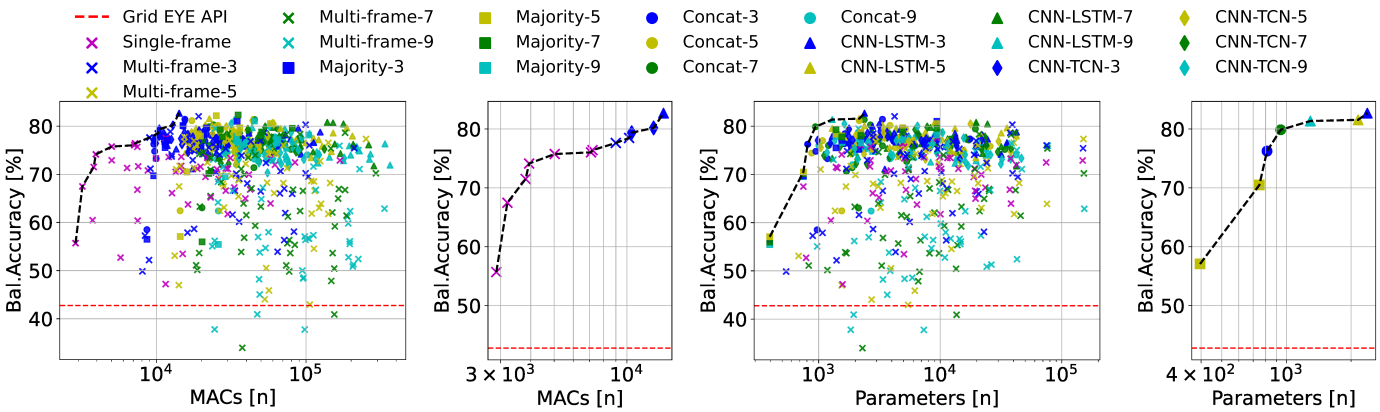
The trained and quantized models are then converted into TensorFlow Lite (TFLite) format [53]. Lastly, we utilize the X-CUBE-AI toolchain 7.2.0 [56] to convert the TFLite files into optimized C language implementations for our deployment target, i.e., the ultra-low-power STM32L4A6ZG MCU by ST Microelectronics, which is based on a 32-bit Cortex-M4 core [57]. The latency and energy results refer to the MCU running at 80MHz, with a supply voltage of 1.8V.

IV. Experimental Results

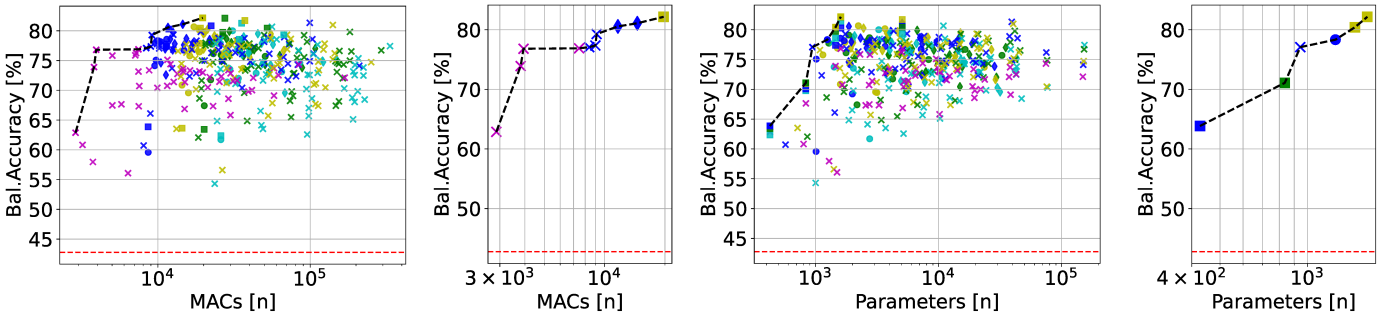
A. Setup

To evaluate the performance of our models, we mainly consider the Balanced Accuracy (Bal. Acc.) metric, i.e., the average of recall on each class. Compared to the standard accuracy (Acc.), i.e., the fraction of correct predictions, which we also report for completeness, the Bal. Acc. is more suitable for class-unbalanced datasets. Moreover, we also measure the F1-Score (F1), defined as the harmonic mean of precision and recall. Since ours is a multiclass problem, we compute the weighted average of the F1 on each class. Lastly, we also report the Mean Absolute Error (MAE) and the Mean Squared Error (MSE) between ground truth and predicted people counts. We consider MAE and MSE although our task is a classification, because they allow taking into account the *significance* of errors: e.g., for a frame with a ground truth people count of 3, a model that outputs 2 makes a "smaller" error compared to one that outputs 1. All metrics are reported as the mean $\pm$ standard deviation over the 4 CV folds, where each fold is weighted by the number of its test samples over the total test samples.

To estimate the hardware-independent computational complexity of each model, we consider the *number of parameters* as a proxy for model size, and the *number of Multiply-and-Accumulate (MAC)* operations, i.e., the dominant operations in DL inference, as a proxy for energy and latency. We then deploy on the target MCU a selection of Pareto-optimal models in the Bal. Acc. versus parameters and MACs planes. For deployed models, we derive the total memory occupation,



(a) Floating point architecture exploration results



(b) 8-bit quantized architecture exploration results

Fig. 5. Results in terms of balanced accuracy versus number of parameters and number of MAC operations for all considered models. All models (left), and isolated Pareto fronts (right).

as well as the total clock cycles, energy consumption and latency per prediction, from measurements on the real hardware, i.e, the STM32L4A6ZG MCU by ST Microelectronics. Concerning memory occupation, both the model size and the total Flash usage are measured. In particular, model size is obtained from the X-CUBE-AI toolchain when generating C code for our DL models, while Flash usage is evaluated using the STM32CubeIDE [58] during deployment. The CPU cycles per inference, which determine the total latency, are also measured using STM32CubeIDE, and in turn used to compute the energy consumption based on the average active power of the MCU from the datasheet. We consider the STM32L4A6ZG MCU working at 80MHz clock frequency, with 1.8V supply voltage [57]. The latency and energy consumption estimates for each architecture have been obtained running the models and the baselines 1000 times, and reported as the mean $\pm$ standard deviation, as shown in Table IV.

Our main baseline for comparison is [25], i.e., the only publicly available people counting solution based on a ceiling-mounted IR array with the same resolution sensor as ours. We compiled and executed the code of [25], written in C language, on our target MCU, using the same compilation flags of our models, and we tested it on the LINAIGE dataset. Furthermore, we also compare with [21]–[24], [32], although only qualitatively, since those works target different datasets and hardware platforms.

B. Architecture Exploration

Figure 5 shows the results of our architecture exploration. In particular, the top (bottom) graphs show the results before (after) 8-bit quantization. For each data precision and target cost metric (MACs or parameters), we report both the entire set of considered models (left), and a "zoom" on the Pareto frontier (right), highlighted by a black dashed line. The people counting performance is reported in terms of the average Bal. Acc. over the CV test folds. Each marker shape refers to one model type, whereas colors correspond to different sliding window sizes W. Note that LSTM-based models are not present in Fig.5b because their quantization is not supported by TFMOT. The performance of our comparison baseline of [25] is shown by a horizontal red line.

The complete graphs show the breadth of our architectural exploration, which includes models that span more than two orders of magnitude in terms of MACs (2.9k-364k) and parameters (0.4k-153k). When considering only Pareto-optimal models, the MACs range is 2.9k-14k for float models and 2.9k-20k for quantized models, while parameters vary in 0.4k-2.4k and 0.4k-1.6k respectively. The Bal. Acc. spanned by these models ranges in 55.70-82.70% for float models and 62.88-82.17% for quantized ones.

All Pareto-optimal models outperform the deterministic approach of [25], showing the benefit of data-driven methods for this task. Moreover, all 6 considered model families are present

in at least one Pareto front, demonstrating that focusing on a single architectural template would be sub-optimal. Single-frame CNNs are only achieving optimal trade-offs in the lowest end of the accuracy range. At the same time, models with $W = 9$ are rarely on the frontier, highlighting at the same time the importance of processing a sequence of IR frames to achieve high accuracy, and the fact that too long sequences stop providing useful information and lead to over-fitting. Lastly, comparing Fig. 5a and Fig. 5b, shows that quantization does not cause relevant accuracy drops, and rather yields a Bal. Acc. *increase* on most models, especially the lowest complexity ones. This is due to its well-known regularizing effect [55], which again helps to reduce overfitting.

Going more into the details of each chart, we note that each Pareto front is formed by a different combination of model types, showing that different architectures are preferable for optimizing the model size or the number of operations. Specifically, when considering the Bal. Acc. versus MAC graphs, Single-frame and Multi-channel CNNs (crosses in the charts) occupy most of the Pareto front, for both float and 8-bit models. In contrast, when considering the number of parameters as a cost metric, the front is mainly composed of Majority voting (squares) and Concatenated CNNs (circles). This is expected, since for $W > 1$, Multi-channel CNNs require additional MACs only in the first Conv layer, whereas all subsequent layers remain identical to the case of $W = 1$. In contrast, Majority-voting and Concatenated CNNs repeat the execution of the entire network, or feature extractor, on each frame, which makes the total MACs grow almost linearly with W. Therefore, these models "pay" the Bal. Acc. benefits deriving from a larger W with a much larger number of operations. Vice versa, since the weights used to process each IR frame are *shared*, the cost increase in terms of model size is lower. Specifically, it is near-zero for Majority-voting CNNs, and limited to the final FC layers for Concatenated CNNs. Accordingly, when considering the parameters as a cost metric, these models are able to outperform multi-channel CNNs and reach the Pareto frontier.

Predictably, the most complex models (CNN-LSTM and CNN-TCN) appear in the high-accuracy part of the Pareto curves. Namely, the most accurate floating point model is a CNN-LSTM, reaching 82.7% Bal. Acc. with $\approx$ 14k MACs and 2.38k parameters, whereas two CNN-TCN appear in the MACs-related Pareto front for quantized models, close to the top. However, in general, most instances of these two types of model suffer from over-fitting, achieving sub-optimal performance, while incurring a high cost in terms of MACs and parameters, as shown by the fact that they mostly occupy the right side of the complete charts. Overall, we can conclude that simple and efficient solutions to combine multiple IR frames (multi-channel, mode inference, and feature concatenation) are preferable for this relatively simple task and small dataset.

Table III reports a summary of all considered DL models, highlighting the features and requirements of each type based on our results. The table reports only qualitative trends, since the exact numerical results could change for different sensors or datasets. Specifically, for each model type, we summarize our Pareto analysis on both floating point and quantized implementations, reporting: i) whether a given model is most effective for memory reduction or for latency/energy reduction, depending on whether it is found more frequently on the parameters or MACs Pareto frontier respectively (*Best For* column); ii) the accuracy range for which such model is preferable (*Bal. Acc. Target* column), which also implicitly defines the corresponding resource range (memory or latency/energy); iii) The maximum IR frames window length that yields Pareto-optimal results for that model family (*Max Input Win.*). Approximately, Low, Mid, and High Bal. Acc. ranges correspond to $< 75\%$, $75\% - 80\%$ and $> 80\%$ respectively. The table provides, at a glance, a general guidance for system designers. For instance, it shows that single-frame CNNs are a good choice when the objective is to obtain a fast and energy-efficient inference, and very high accuracy is not required. Similarly, it shows that Majority voting is a very effective solution for memory reduction, across the whole accuracy range, or that CNN-LSTMs are the only models for which a window length > 7 is useful for improving accuracy, etc.

TABLE III
SUMMARY OF THE CHARACTERISTICS OF THE CONSIDERED DL MODELS.

Model	Best For	Bal. Acc. Target	Max Input Win.
Single-frame	Latency/Energy	Low	1
Multi-frame	Latency/Energy	Mid	3
Majority	Memory	Whole Range	7
Concat	Memory	Mid	7
CNN-LSTM	Memory	High	9
CNN-TCN	Latency/Energy	High	3

C. Deployment

We have selected 5 floating point and 5 quantized architectures from the Pareto curves derived in Sec. IV-B to deploy on the target MCU. Namely, we deployed: i) the model achieving the best balanced accuracy (Top); ii) the smallest model overall (Size-L) and the one requiring the least number of MACs (MAC-L); iii) the smallest/fewest-MAC models that achieve a Bal. Acc. drop $< 5\%$ with respect to Top (Size-H/MAC-H).

Table IV shows the detailed deployment results for these architectures on the STM32L4A6ZG MCU. Quantized models are denoted with a "-Q" suffix. Besides people counting accuracy metrics, we also report the memory occupation, energy consumption and inference latency of each model. In particular, for what concerns memory, we report both the model size and the total occupied Flash, which also includes code size. The same quantities are also reported for [25] for comparison. The rightmost column summarizes the architecture of each deployed neural network. Namely, the symbols inside the square brackets indicate the model type, using the same marker shape of Fig. 5 (e.g., ▲ corresponds to a CNN-LSTM). The number in brackets corresponds to the value of W. Then, the sequence of layers in the model is encoded as follows: "Cn" corresponds to a Conv layer with n output channels, with implicit BatchNorm and ReLU, "FC" is a fully-connected layer, "P" a max. pooling layer, "Lm" a LSTM cell with hidden size m, and "Cat" a concatenation.

As shown, all quantized models, as well as most floating-point models (except Size-L and MAC-L) greatly outperform

TABLE IV
DETAILED EVALUATION AND DEPLOYMENT RESULTS OF SELECTED ARCHITECTURES.

Model	Bal. Acc. [%]	Acc. [%]	F1	MSE	MAE	Model Size [kB]	Tot. Mem. [kB]	Energy [µJ]	Latency [ms]	Architecture
Top	82.70±6.15	84.34±7.84	0.85±0.07	0.18±0.09	0.16±0.08	9.28	82.38	80.26±0.10	5.16±0.0064	[▲3] C8-P-C8-L16-FC
Size-H	76.25±5.54	78.13±9.08	0.79±0.08	0.24±0.09	0.23±0.09	2.97	68.73	54.96±0.01	3.53±0.0006	[●3] C8-P-C8-Cat-FC
MAC-H	77.62±5.98	78.04±8.18	0.80±0.07	0.27±0.11	0.24±0.09	5.7	42.95	29.25±0.01	1.88±0.0003	[×3] C8-P-C16-FC
Size-L	57.08±11.37	51.10±22.49	0.52±0.23	0.78±0.70	0.58±0.36	1.45	37.88	85.75±0.06	5.51±0.0036	[■5] C8-P-FC
MAC-L	55.70±11.86	50.35±21.31	0.51±0.21	0.81±0.68	0.59±0.35	1.45	37.59	17.18±0.01	1.10±0.0007	[×1] C8-P-FC
Top-Q	82.17±6.42	86.06±5.59	0.86±0.05	0.15±0.05	0.14±0.05	1.71	78.01	120.43±0.02	7.74±0.0010	[■5] C8-P-C8-FC-FC
Size-H-Q/MAC-H-Q	77.08±6.05	79.48±6.53	0.81±0.06	0.24±0.07	0.22±0.07	0.9	76.32	27.70±0.02	1.78±0.0010	[×3] C8-P-C8-FC
Size-L-Q	63.87±10.76	70.83±13.79	0.70±0.14	0.33±0.12	0.30±0.13	0.41	71.56	61.90±0.02	3.98±0.0010	[■3] C8-P-FC
MAC-L-Q	62.88±7.52	68.97±14.03	0.69±0.14	0.36±0.13	0.33±0.14	0.41	71.39	20.45±0.01	1.32±0.0007	[×1] C8-P-FC
[25]	42.77±14.50	57.54±11.50	0.56±0.12	0.61±0.21	0.49±0.14	-	20.07	60.34±0.005	3.88±0.0003	-

[25] in all considered accuracy metrics. In terms of balanced accuracy, our models outperform [25] by 20.1%-39.4% and 12.9-39.9% for integer and floating-point data representations respectively. Moreover, MAC-H and MAC-L in both implementations are faster and more energy efficient (from 2.06x to 3.51x) than [25], while still significantly outperforming it. For example, MAC-H-Q is 2.18x times faster and more energy efficient than [25], while also achieving +34.3% Bal. Acc., +21.9% Acc., 1.44x higher F1 Score, and 2.54x/2.22x lower MSE/MAE.

The model size of all selected architectures is extremely small, with the smallest one occupying only 0.408 KB. The total memory, instead, is larger than [25], but this is mostly due to the large code size of X-CUBE-AI libraries, which contributes to up to 97% of the Flash occupation. As shown in the table, the resulting memory depends on the types of layers present in the model (e.g., the "Top" floating point model requires more memory partly because of the additional inclusion of LSTM-related code). Further, quantized models have a larger code size compared to floating point ones on average, probably due to the more complex logic for handling scaling factors and re-quantization operations [55]. Nonetheless, all considered models can easily fit in memory-limited IoT nodes, requiring 37.6-82.4kB of Flash, which corresponds to 3.7%-8% of the 1MB available on the MCU considered for our experiments.

All our models also have a latency < 10ms , which is below the real-time constraint, considering the 10FPS acquisition rate of our target dataset. Furthermore, considering a small 1400mAh@3.7V battery, and ignoring non-idealities and conversion losses for simplicity, a model such as MAC-H-Q would be able to continuously run inferences at that frame rate for more than 2 years without recharging.

D. Comparison with state-of-the-art ML/DL Approaches

Table V compares our work with the most relevant Machine Learning and Deep Learning approaches for people counting with IR array sensors. Of course, the comparison is only qualitative, since most previous works have been tested on private datasets, and deployed on different hardware. In the table, besides the input frame size, we report the Acc., F1 and MSE scores when available (other metrics were not considered by previous works). All scores are directly taken from the original papers. We also report the model size and the number of operations (OPs) per inference, as two hardware-independent complexity metrics. For DL solutions, we approximate OPs with the number of MACs, and when either Size or OPs are not reported by the authors, we calculated them based on the layers' geometries. For [32], instead, estimating Size and OPs was not possible, since the authors did not report the number and maximum depth of the decision trees that compose their best-performing RF.

TABLE V
COMPARISON WITH THE STATE OF THE ART.

Result	Input	Acc. [%]	F1	MSE	Size [kB]	OPs
[21]	8x6	n.a.	0.88	n.a.	450	$34{\cdot}10^6$
[22]*,†	10x10	95.9	n.a.	n.a.	13.6	$117{\cdot}10^3$
[23]*	32x24	**98.9**	n.a.	**0.01**	400	$400{\cdot}10^3$
[32]*	8x8	94.6	0.95	n.a.	n.a.	n.a.
[24]§	32x24	94.1	n.a.	0.057	520.8	$25{\cdot}10^6$
Top-Q	8x8	86.1	0.86	0.15	**1.71**	$\mathbf{20{\cdot}10^3}$
Top-Q*	8x8	95.3	**0.95**	0.05	20.1	$80{\cdot}10^3$

(*) Train/test split based on random sampling, not per-session.
(§) Train/validation/test split based on sequences splitting in different locations, not per-session.
(†) Numbers refer to the processing of a single 10x10 sliding window. This approach also *localizes* people.

We report two results for our work: the first one corresponds to the "Top-Q" network of Table IV, found using the described per-session CV approach. Additionally, since [22], [23], [32] use a purely random sampling method to separate training and test sets, we also report the best quantized results obtained with such kind of splitting. Precisely, we repeat the architecture search using a random 80%/20% train/test split, and report the average test set results over 4 iterations. Note that the resulting Top-Q* model has a different architecture from Top-Q. Namely, it is a quantized CNN-TCN model with the following structure: [♦9] C8-P-C32-TCN32-FC-FC, where TCN*o* refers to a TCN (1D Conv) layer with *o* output channels.

Our main reference for comparison among state-of-the-art DL methods is [21], which uses a per-session split and a similar input resolution. Compared to this work, we obtain a comparable F1 Score with our Top-Q, but since our classification model is significantly smaller, and we do not need an additional super-resolution network, we achieve a 263x reduction in size and 1700x fewer OPs. The work of [24] also uses a time-based data split, although simpler than ours: they assign to different data buckets the frames collected in the same location at different times. Their work achieves a higher Accuracy than our Top-Q (94.1% vs 86.1%) but this is

mostly due to the 12x higher resolution input. Furthermore, their model requires about 130k floating point parameters, resulting in a model size of 521kB, which is 304x more than that of Top-Q. Similarly, the number of OPs is in the order of millions, more than 1000x larger than Top-Q.

Since the dataset of [24] is publicly available, we also ran two additional experiments on it. First, we down-sampled the images to 8x8 resolution and excluded all samples with more than 3 people to fairly compare with LINAIGE. Then, we trained our "Top" model from Table IV using only the data from [24] and maintaining our training protocol. We obtained an accuracy of 72.6%, much lower than the one achieved by their model, but acceptable given the lower resolution of our inputs and the striking $>$ 1000x complexity reduction. Further, the dataset in [24] only contains $\approx$ 9k samples with less than 3 people, versus the $>$ 20k of LINAIGE. Thus, we also tried to use the down-sampled data from [24] to *augment* the LINAIGE training dataset in each CV fold. In this case, the "Top" model improves in all classification metrics on average (Acc. +4.8%, F1 +0.04, MSE -0.06, MAE -0.05) except for the Bal. Acc (-1.2%) with respect to pure LINAIGE training. This shows that, potentially, using a larger dataset could further improve the results achieved by our efficient DL models, especially the most complex architectures.

When considering a random data split, Top-Q* obtains slightly lower accuracy and higher MSE compared to [23], but uses a smaller-resolution input, and requires a 233x smaller model and 20x fewer inference operations. It also achieves comparable accuracy and F1 score with respect to the RF-based approach of [32]. Lastly, [22] uses a model smaller than Top-Q* to achieve a slightly higher accuracy on a 10x10 input. However, the inputs processed by [22] are patches extracted from a much higher resolution input (80x60), which is further upscaled to 120x90 and 160x120. All three versions of the image are then processed by the CNN in 10x10 sliding-window patches. Therefore, the total number of inference operations is huge for this solution (approximately $450{\cdot}10^6$ based on our calculations), which translates into very long latencies and high energy consumption. Indeed, the authors report a total latency of 63s and an energy of 2.2J, orders of magnitude higher than those achieved by our models. It must be underlined that [22] attempts not only to count people in the frame, but also to localize their heads, which is significantly different from our goal, and only possible due to their higher-resolution input. Indeed, the 95.9% accuracy reported in the table refers to head detection on a single 10x10 patch, whereas the final counting accuracy is just 53.7%.

In summary, these comparisons show that our proposed models achieve comparable counting accuracy with much lower complexity on average, compared to state-of-the-art solutions. This is particularly important for deployment at the IoT edge, where devices have very tight memory budgets, and extreme constraints in terms of energy consumption, being typically battery powered and expected to operate for years without recharging. The tiny and efficient DL models explored in this work could enable novel pervasive and privacy-preserving people counting solutions in environments where access to the power grid is not available, which would exclude most of the energy-hungry state-of-the-art solutions.

V. Conclusion

We have conducted the first systematic study on efficient DL architectures for person counting based on ultra-low-resolution IR arrays, obtaining a wide range of trade-offs between classification scores, memory occupation, latency and energy consumption, and showing that different types of DL models are preferable for different objectives. The resulting Pareto-optimal models obtain counting accuracy scores that are significantly higher than those of a publicly available deterministic solution [25] (up to 82.70% balanced accuracy vs 42.77%), and comparable with a state-of-the-art DL approach [21] (0.86 vs 0.88 F1-score), while reducing the latency and energy requirements by up to more than 2x with respect to the former, e.g. 1.78ms/27.70μJ vs 3.88ms/60.34μJ per inference at approximately $+34.3\%$ balanced accuracy for our method. Furthermore, our models enable continuous real-time inference ($<$ 10ms latency) on IoT edge devices based on MCUs, with years of autonomous operation, while requiring less than 100kB of memory.

Acknowledgment

This work has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 101007321. The JU receives support from the European Union's Horizon 2020 research and innovation programme and France, Belgium, Czech Republic, Germany, Italy, Sweden, Switzerland, Turkey.

References

[1] J. Chen *et al.*, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.

[2] B. Jiang *et al.*, "Wearable vision assistance system based on binocular sensors for visually impaired users," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1375–1383, 2019.

[3] K. Muhammad *et al.*, "Cost-effective video summarization using deep cnn with hierarchical weighted fusion for iot surveillance networks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4455–4463, 2020.

[4] A. Burrello *et al.*, "Bioformers: Embedding transformers for ultra-low power semg-based gesture recognition," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2022, pp. 1443–1448.

[5] M. Risso *et al.*, "Lightweight neural architecture search for temporal convolutional networks at the edge," *IEEE Transactions on Computers*, pp. 1–1, 2022.

[6] Z. Zhou *et al.*, "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738–1762, 2019.

[7] W. Shi *et al.*, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[8] Y.-L. Hou *et al.*, "People counting and human detection in a challenging situation," *IEEE transactions on systems, man, and cybernetics-part a: systems and humans*, vol. 41, no. 1, pp. 24–33, 2010.

[9] P.-R. Tsou *et al.*, "Counting people by using convolutional neural network and a pir array," in *2020 21st IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 2020, pp. 342–347.

[10] C. Xie *et al.*, "Privacy-preserving social distance monitoring on micro-controllers with low-resolution infrared sensors and cnns," in *Proceedings of the 2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, ser. ISCAS 2022. IEEE, 2022.

[11] C. Perra *et al.*, "Monitoring indoor people presence in buildings using low-cost infrared sensor array in doorways," *Sensors*, vol. 21, no. 12, p. 4062, 2021.

[12] C. Raghavachari *et al.*, "A comparative study of vision based human detection techniques in people counting applications," *Procedia Computer Science*, vol. 58, pp. 461–469, 2015.

[13] W. Xi *et al.*, "Electronic frog eye: Counting crowd using wifi," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 361–369.

[14] K. Hashimoto *et al.*, "People count system using multi-sensing application," in *Proceedings of International Solid State Sensors and Actuators Conference (Transducers' 97)*, vol. 2. IEEE, 1997, pp. 1291–1294.

[15] I. Udrea *et al.*, "New research on people counting and human detection," in *2021 13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*. IEEE, 2021, pp. 1–6.

[16] M. B. Shami *et al.*, "People counting in dense crowd images using sparse head detections," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 9, pp. 2627–2636, 2018.

[17] A. D. Shetty *et al.*, "Detection and tracking of a human using the infrared thermopile array sensor—"grid-eye"," in *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*. IEEE, 2017, pp. 1490–1495.

[18] S. Basalamah *et al.*, "Scale driven convolutional neural network model for people counting and localization in crowd scenes," *IEEE Access*, vol. 7, pp. 71 576–71 584, 2019.

[19] V. Nogueira *et al.*, "Retailnet: A deep learning approach for people counting and hot spots detection in retail stores," in *2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 2019, pp. 155–162.

[20] S. D. Khan *et al.*, "Person head detection based deep model for people counting in sports videos," in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2019, pp. 1–8.

[21] M. Bouazizi *et al.*, "Low-resolution infrared array sensor for counting and localizing people indoors: When low end technology meets cutting edge deep learning techniques," *Information*, vol. 13, no. 3, p. 132, 2022.

[22] A. Gomez *et al.*, "Thermal image-based cnn's for ultra-low power people recognition," in *Proceedings of the 15th ACM International Conference on Computing Frontiers*, 2018, pp. 326–331.

[23] A. Metwaly *et al.*, "Edge computing with embedded ai: Thermal image analysis for occupancy estimation in intelligent buildings," in *Proceedings of the INTelligent Embedded Systems Architectures and Applications Workshop 2019*, 2019, pp. 1–6.

[24] M. Kraft *et al.*, "Low-cost thermal camera-based counting occupancy meter facilitating energy saving in smart buildings," *Energies*, vol. 14, no. 15, 2021. [Online]. Available: https://www.mdpi.com/1996-1073/14/15/4542

[25] P. Industry, "Grid-eye application note on social distancing. people detection and tracking with ceiling mounted sensors," 2020.

[26] H. Mohammadmoradi *et al.*, "Measuring people-flow through doorways using easy-to-install ir array sensors," in *2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2017, pp. 35–43.

[27] H. Wang *et al.*, "A lightweight people counting approach for smart buildings," in *2021 13th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2021, pp. 1–5.

[28] R. Rabiee *et al.*, "Multi-bernoulli tracking approach for occupancy monitoring of smart buildings using low-resolution infrared sensor array," *Remote Sensing*, vol. 13, no. 16, p. 3127, 2021.

[29] T. Kawashima *et al.*, "Action recognition from extremely low-resolution thermal image sequence," in *Proceedings of the 14th IEEE International Conference on Advanced Video and Signal Based Surveillance*, Aug. 2017, pp. 1–6.

[30] S. Singh *et al.*, "Non-intrusive presence detection and position tracking for multiple people using low-resolution thermal sensors," *Journal of Sensor and Actuator Networks*, vol. 8, no. 3, p. 40, 2019.

[31] C. Xie *et al.*, "Low-resolution infrared-array data for ai on the edge," 2022. [Online]. Available: https://www.kaggle.com/datasets/francescodaghero/linaige

[32] V. Chidurala *et al.*, "Occupancy estimation using thermal imaging sensors and machine learning algorithms," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 8627–8638, 2021.

[33] C. Xie *et al.*, "Energy-efficient and Privacy-aware Social Distance Monitoring with Low-resolution Infrared Sensors and Adaptive Inference," in *2022 17th Conference on Ph.D Research in Microelectronics and Electronics (PRIME)*, Jun. 2022, pp. 181–184.

[34] X. Liu *et al.*, "Detecting and counting people in surveillance applications," in *IEEE Conference on Advanced Video and Signal Based Surveillance, 2005.* IEEE, 2005, pp. 306–311.

[35] M. Stec *et al.*, "Multi-sensor-fusion system for people counting applications," in *2019 First International Conference on Societal Automation (SA)*. IEEE, 2019, pp. 1–4.

[36] D. Olmeda *et al.*, "Pedestrian detection in far infrared images," *Integrated Computer-Aided Engineering*, vol. 20, no. 4, pp. 347–360, 2013.

[37] FLIR, "Free flir thermal dataset for algorithm training," 2018.

[38] S. Hwang *et al.*, "Multispectral pedestrian detection: Benchmark dataset and baselines," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[39] R. E. Rivadeneira *et al.*, "Thermal image super-resolution challenge - pbvs 2020," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[40] Y. Karayaneva *et al.*, "Infrared human activity recognition dataset - coventry-2018," 2020. [Online]. Available: https://dx.doi.org/10.21227/baja-1j59

[41] K. He *et al.*, "Mask r-cnn," 2017. [Online]. Available: https://arxiv.org/abs/1703.06870

[42] L. Lam *et al.*, "Application of majority voting to pattern recognition: an analysis of its behavior and performance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 27, no. 5, pp. 553–568, 1997.

[43] D.-S. Lee, "Handprinted digit recognition: A comparison of algorithms," in *Proceedings of the Third International Workshop on Frontiers in Handwriting Recognition*, 1993, pp. 153–164.

[44] M. Amin-Naji *et al.*, "Cnns hard voting for multi-focus image fusion," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 4, pp. 1749–1769, 2020.

[45] A. Yazdizadeh *et al.*, "Ensemble convolutional neural networks for mode inference in smartphone travel survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 6, pp. 2232–2239, 2020.

[46] F. Demir *et al.*, "A new pyramidal concatenated cnn approach for environmental sound classification," *Applied Acoustics*, vol. 170, p. 107520, 2020.

[47] Q. Wu *et al.*, "Concatenate convolutional neural networks for non-intrusive load monitoring across complex background," *Energies*, vol. 12, no. 8, p. 1572, 2019.

[48] T.-Y. Kim *et al.*, "Predicting residential energy consumption using cnn-lstm neural networks," *Energy*, vol. 182, pp. 72–81, 2019.

[49] J. Zhao *et al.*, "Speech emotion recognition using deep 1d & 2d cnn lstm networks," *Biomedical signal processing and control*, vol. 47, pp. 312–323, 2019.

[50] V. Sciannameo *et al.*, "A deep learning approach for Spatio-Temporal forecasting of new cases and new hospital admissions of COVID-19 spread in Reggio Emilia, Northern Italy," *Journal of Biomedical Informatics*, vol. 132, p. 104132, Aug. 2022.

[51] C. Lea *et al.*, "Temporal Convolutional Networks: A Unified Approach to Action Segmentation," in *Computer Vision – ECCV 2016 Workshops*, ser. Lecture Notes in Computer Science, G. Hua *et al.*, Eds. Cham: Springer International Publishing, 2016, pp. 47–54.

[52] A. Burrello *et al.*, "Q-PPG: Energy-Efficient PPG-based Heart Rate Monitoring on Wearable Devices," *IEEE Transactions on Biomedical Circuits and Systems*, p. 1, 2021.

[53] M. Abadi *et al.*, "Tesorflow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: https://www.tensorflow.org/

[54] F. Daghero *et al.*, "Energy-efficient deep learning inference on edge devices," in *Hardware Accelerator Systems for Artificial Intelligence and Machine Learning*, ser. Advances in Computers, S. Kim *et al.*, Eds. Elsevier, 2021, vol. 122, ch. 8, pp. 247–301.

[55] B. Jacob *et al.*, "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, jun 2018.

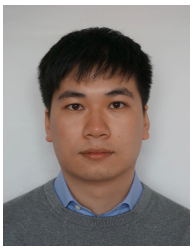
[56] STMicroelectronics, "X-CUBE-AI, AI expansion pack for STM32CubeMX," https://www.st.com/en/embedded-software/x-cube-ai.html.

[57] ——, "STM32L4A6ZG, Ultra-low-power Arm Cortex-M4 32-bit MCU," https://www.st.com/en/microcontrollers-microprocessors/stm32l4a6zg.html.

[58] ——, "STM32CubeIDE, Integrated Development Environment for STM32," https://www.st.com/en/development-tools/stm32cubeide.htm.

Chen Xie received the M.Sc degrees in Electronics Engineering at Politecnico di Torino in 2020. Since May 2020, she joined the EDA group in the Department of Control and Computer Engineering at Politecnico di Torino. Her main research interests concern energy-efficient implementations of machine learning algorithms and synthesis of smart sensors.

Francesco Daghero is a PhD student at Politecnico di Torino. He received a M.Sc. degree in computer engineering from Politecnico di Torino, Italy, in 2019. His research interests concern embedded machine learning and Industry 4.0.



Yukai Chen earned his M.Sc. and Ph.D. degrees in Computer Engineering from the Politecnico di Torino, Turin, Italy, in 2014 and 2018, respectively. He currently serves as a Senior Researcher at IMEC, where he contributes to the System and Technology Co-optimization Program. His primary focus is on system-level power and thermal management for High-Performance Computing Architectures. His research interests encompass design automation for non-functional property modeling, simulation, and optimization, with particular emphasis on energy-efficient design and design space exploration.

Marco Castellano received the Laurea degree from the Univ. of Pavia, Italy (2005) and in 2009 a Ph.D. in electrical engineering from Univ. of Pavia, Italy, in a joint research center supported by the Univ. of Pavia and STMicroelectronics. In 2008 he joined STMicroelectronics in Cornaredo (Italy) working in MEMS division as digital designer. His main fields of interest include complex gesture recognition algorithms implementation, FIFO, sensors, DSP and compensations design. Since 2016, he leads a team of digital experts working on co-design of controller and related software for custom low-power application design. He has authored several papers, conference contributions and patents on topics related to algorithms integration.

Luca Gandolfi received the Laurea degree from the Univ. of Pisa, Italy (2019). In 2019 he joined STMicroelectronics in Cornaredo (Italy) working in a R&D digital design team for the Analog MEMS and Sensor Group. His research interest is in the codesign of firmware and hardware for complex algorithms in sensor systems.

Andrea Calimera took the M.Sc. degree in Electronic Engineering and the Ph.D. degree in Computer Engineering from Politecnico di Torino. He is currently an Associate Professor of Computer Engineering at Politecnico di Torino. His research interests cover the areas of electronic design automation, with emphasis on optimization techniques for low-power and reliable integrated circuits, energy/quality management in embedded systems and portable applications, novel computing paradigms, and emerging technologies.

Enrico Macii is a Full Professor of Computer Engineering with the Politecnico di Torino, Torino, Italy. He holds a Laurea degree in electrical engineering from the Politecnico di Torino, a Laurea degree in computer science from the Università di Torino, Turin, and a PhD degree in computer engineering from the Politecnico di Torino. His research interests are in the design of digital electronic circuits and systems, with a particular emphasis on low-power consumption aspects energy efficiency, sustainable urban mobility, clean and intelligent manufacturing. He is a Fellow of the IEEE.

Massimo Poncino is a Full Professor of Computer Engineering with the Politecnico di Torino, Torino, Italy. His current research interests include various aspects of design automation of digital systems, with emphasis on the modeling and optimization of energy-efficient systems. He received a PhD in computer engineering and a Dr.Eng. in electrical engineering from Politecnico di Torino. He is a Fellow of the IEEE.

Daniele Jahier Pagliari received the M.Sc. and Ph.D. degrees in computer engineering from the Politecnico di Torino, Turin, Italy, in 2014 and 2018, respectively. He is currently an Assistant Professor with the Politecnico di Torino. His research interests are in the computer-aided design and optimization of digital circuits and systems, with a particular focus on energy-efficiency aspects and on emerging applications, such as machine learning at the edge.