Smart and Sustainable Ice Cream Making Through Edge Machine Learning

Filippo Tabanelli, Simon Dahdal, *Graduate Student Member, IEEE*, Nicolas Belletti, Elena Bellodi, Franck Ngatcha, Roberto Lazzarini, Cesare Stefanelli, *Member, IEEE*, and Mauro Tortonesi, *Member, IEEE*

***Abstract*—The manufacturing process of frozen dairy desserts, such as ice cream and gelato, is very sensitive to human errors in ingredient preparation: even minor variations in the ingredient mix can lead to quality issues and material waste. To become more sustainable, next generation ice cream making machines need to implement intelligent and adaptive processes that are both efficient and forgiving of human mistakes in mixture preparations. Toward that goal, we developed Hard-O-Tronic AI-driven (HOT-AI), a novel edge AI solution specifically designed for Carpigiani's ice cream making machines. Leveraging the innovative multimilestone classification methodology, HOT-AI performs inference at multiple stages—or milestones—during the ice cream making process, with increasing accuracy over time. This enables HOT-AI to take corrective actions by adapting the preparation process accordingly, thus improving batch-to-batch uniformity, minimizing ingredient waste, and enhancing production efficiency, cost-effectiveness, and sustainability. HOT-AI has been successfully validated under real production conditions, and its large-scale implementation is planned across Carpigiani Group machines.**

***Index Terms*—Adaptive manufacturing processes, edge AI, industry 5.0, real-time quality control, zero defect manufacturing, zero waste manufacturing.**

I. Introduction

FROZEN dairy desserts, such as ice cream and gelato, require a delicate balance of ingredients and processing to achieve their desired texture and quality. In fact, these products are typically consumed in a partially frozen state, relying on a combination of mix composition, processing conditions, and precise temperature control during freezing. Their preparation, therefore, is a challenging task that does not leave room for mistakes—either at the equipment or at the operator level.

More specifically, the manufacturing process for both ice cream and gelato generally involves three key steps: mix preparation, whipping, and freezing [1]. Within this process, freezing plays a critical role in determining the final product's quality, taste, and yield. In the freezing phase, the ice cream mix is vigorously agitated to incorporate air and to control the size of ice crystals that form. Agitation is crucial for achieving a smooth, creamy texture and the desired overrun (the increase in volume due to air incorporation). Traditional ice cream and gelato production often relies on batch freezers. These machines typically consist of a jacketed cylinder with a rotating dasher. The cylinder is cooled by a refrigerant, while the dasher scrapes the ice forming on the inner surface and mixes the product to incorporate air. This combined action of cooling and agitation is essential for achieving the desired texture. The freezing process in batch freezers requires precise temperature control, typically within a narrow range between -11°C and -8°C.

Built to handle intensive, heavy-duty operations, ice cream making machines are engineered to perform reliably for over a decade, demonstrating exceptional robustness and longevity even under demanding operational conditions. The need to deliver batches of ice cream ever more efficiently and in an ever shorter amount of time has evolved these machines toward the implementation of a reference, or default, production process optimized for speed. While machines typically come with a relatively wide range of pre-installed processes that can be selected, and often can be programmed to add even more, operators significantly appreciate the speed of the default process and seldom depart from it.

However, the default process is very sensitive to imbalances in the mix composition, which can lead to a compromised quality of the final product. For example, an excess of fats or sugars can affect the freezing rate and ice crystal formation, while a deficiency of stabilizers can lead to a less homogeneous texture. Unfortunately, quality degradation is either completely opaque to the operator or shows up too late in the process for the operator to take corrective actions. As a result, the high sensitivity of the default ice cream making preparation process makes it particularly prone to operator mistakes, which can lead to waste generation.

The ever growing attention towards sustainability, and in particular *zero defect manufacturing* [2] and *zero waste manufacturing* [3], raises the need to improve ice cream making machines so that they can be both efficient and forgiving of human mistakes in mixture preparations. Addressing this major challenge calls for the development of a new generation of *ice cream machines capable of automatically detecting imbalances in the ingredient mixture composition in real time and of adapting the preparation process accordingly*.

Recent Industry 5.0 applications have achieved remarkable results in transforming manufacturing processes through data-driven and AI-based approaches [4], [5]. However, most of

Received 18 August 2025; revised 18 December 2025; accepted 21 December 2025. Paper no. TII-25-5181. *(Corresponding author: Simon Dahdal.)*

Filippo Tabanelli, Simon Dahdal, Nicolas Belletti, Elena Bellodi, Cesare Stefanelli, and Mauro Tortonesi are with the University of Ferrara, 44121 Ferrara, Italy (e-mail: filippo.tabanelli@unife.it; simon.dahdal@unife.it; nicolas.belletti@unife.it; elena.bellodi@unife.it; cesare.stefanelli@unife.it; mauro.tortonesi@unife.it).

Franck Ngatcha and Roberto Lazzarini are with the Carpigiani Group, 40011 Bologna, Italy.

Digital Object Identifier 10.1109/TII.2025.3649155

these solutions so far have relied on centralized, and often cloud-based approaches [6], [7], [8], [9], [10], that are unsuited as a building block for the implementation of adaptive ice cream preparation processes. In fact, the highly distributed locations of ice cream making machines and the strict constraints for the decision making, in terms of low latency and high autonomy levels required to take effective corrective measures in case of improper loading with unbalanced ingredients, call instead for an edge-based approach. Unfortunately, industrial edge applications, especially those leveraging advanced AI solutions, have to deal with a significant set of challenges due to computational, memory, and bandwidth constraints [11], [12], [13], [14], [15], [16], [17], [18]. This often prevents the reuse of methodologies and tools commonly adopted in centralized applications, forcing to develop ad hoc approaches.

In this article, we introduce "*Hard-O-Tronic AI-driven (HOT-AI),*" a machine learning (ML)-powered solution specifically designed for Carpigiani's gelato machines capable of automatically detecting imbalances in the ingredient mixture and of taking real-time corrective actions by modifying the gelato preparation recipe accordingly. HOT-AI continuously monitors the ice cream preparation process by analyzing the time series data collected from the wide range of sensors installed on the machine and feeds it to a multiclass classifier that can detect if the mixture is imbalanced—and how (too much sugar, not enough cream, etc). If an unbalanced condition is detected, the classifier will notify the decision making component, which will then adapt the machine's processing settings to mitigate the impact of any errors and ensure the quality of the final product. For instance, if the system detects an excess of sugars, it can suggest increasing the agitation speed or reducing the freezing temperature, in order to obtain a final product with the desired characteristics.

To allow prompt corrective actions, HOT-AI performs inference at multiple stages—or milestones—during the ice cream making process, with increasing accuracy over time, allowing for corrective actions to preserve the product's quality. To enable this, we developed an innovative ML methodology, which we called *multimilestone classification*, based on five ML modules (one convolutional neural networks (CNN) and four fully connected (FC) neural network] designed to perform mixture classification at four milestones during the ice cream making process (respectively at 150, 200, 250, and 300 s after its beginning) and to be jointly trained for maximum performance. Our multimilestone solution presents important advantages compared to approaches based on multiple fully independent classifiers or on a single variable-length input classifier. In fact, multimilestone classification is not just capable of reaching increased accuracy levels but also lower inference times—thereby providing higher responsiveness—and smaller model sizes—thereby requiring less resource expenditures throughout the ML model lifecycle (running and updates).

II. Carpigiani State-of-The-Art and Challenges

Carpigiani is a world-leading company specializing in the production of ice cream machines and equipment for artisanal gelato and ice cream production, with 35% share of the global market and around 140 000 machines installed across the entire world. To optimize after sales assistance operations and maintain the high standards of performance and reliability expected from the brand, Carpigiani has implemented and adopted a reliable remote monitoring and diagnostic system: the *Teorema* e-maintenance platform [12], which enables centralized monitoring and diagnostics across deployed machines.

Teorema was designed to collect, process, and store a wide array of data types from the machines, providing a centralized approach to managing machine performance. Teorema provides centralized data collection and diagnostics through a cloud-based analytics infrastructure. Teorema facilitates real-time monitoring and diagnostics, allowing for proactive maintenance by detecting potential issues before they lead to equipment failures. This capability not only minimizes unplanned downtime but also extends the operational life of the machines by ensuring that maintenance is carried out precisely when needed.

Carpigiani's machines generate a large amount of data during operations. This data can be categorized into three primary types. *Technical data:* This includes metrics, such as temperatures, motor status, and mixture levels. *Operational data:* This refers to information about the completion of pasteurization cycles, alarm statuses, and other machine alerts. *Production data:* This encompasses information, such as the number of portions served, cycles performed, and hours of motor operation. Together, these data provide comprehensive and real-time insights into the status of a machine and its production processes.

However, the centralized approach adopted so successfully by Teorema would not be suited for the implementation of a data-driven automated human error correction system. In fact, effectively determining whether a mixture loaded by the operator is unbalanced and consequently taking corrective actions before it is too late in the preparation process represents a significant challenge in terms of system responsiveness—hardly compatible with a cloud-based approach. In addition, communication overhead is also an important factor. Carpigiani's gelato machines are deployed globally and operate in a wide variety of environments, from regions with robust 5G connectivity to areas with limited and unreliable connections. In regions with limited network connectivity, the machines rely on global system for mobile communications (GSM)/general packet radio service (GPRS), which presents high latency and bandwidth limitations. Finally, a centralized approach would present nonnegligible risks from the data privacy and security perspective, as it would require the transmission of potentially sensitive process-specific data (which Teorema currently does not consider) to the cloud.

Consequently, any effective implementation for an automated error correction function would need to adopt an edge AI approach, running ML-based inference tasks that assess the state of the ice cream production directly on the machine. Unfortunately, the edge AI approach presents several challenges: edge devices typically have limited computational power, memory, and energy resources, which can limit their ability to handle heavy ML models [19], [20], [21], [22]. Deploying models that are both lightweight and capable of performing real-time, low-latency inference is critical to ensuring system responsiveness. Achieving this requires careful optimization of the ML models to balance accuracy with computational efficiency. This ensures that the system delivers high-quality results without overburdening the edge hardware or necessitating expensive upgrades.

Realizing an edge AI system that can accurately identify an imbalanced ice cream mixture in time to take effective corrective

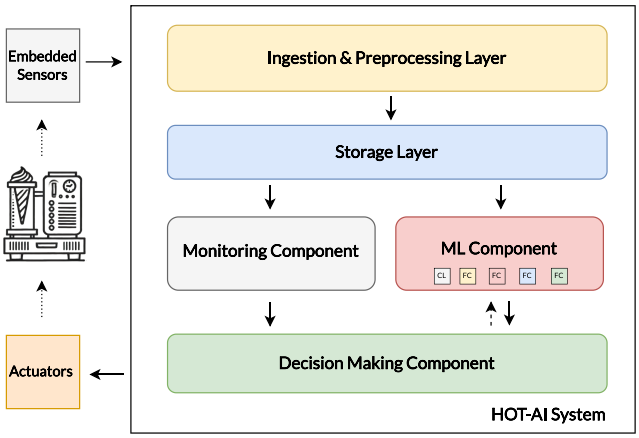

Fig. 1. Architecture of the HoT-AI System.

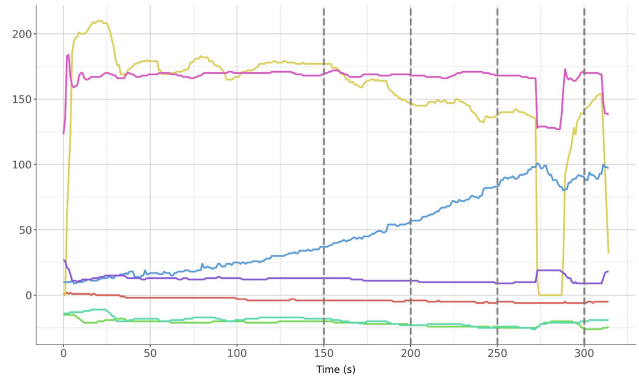

Fig. 2. Time series collected from the machine with 4 milestones at times +150, + 200, +250, and +300 s from the production starting point.

actions represents an ever bigger challenge. In fact, this means that the system must be capable of handling dynamic inputs and classifying in (soft) real-time the mixture's state at various points throughout the production cycle. This essentially rules out several mainstream designs for ML classifiers based on artificial neural networks (ANN), and calls for the development of innovative and ad hoc solutions.

III. HoT-AI System

HOT-AI is an intelligent and automated error correction system designed to operate without requiring any supervision from operators. It runs in (soft) real-time, monitoring the ice cream production process to check if the ingredient mixture loaded in the machine is unbalanced and adaptively adjusting the preparation process to avoid waste and maintain high product quality.

Fig. 1 provides a high-level overview of the HOT-AI architecture, depicting its closed-loop design and showing how on-machine sensing, edge inference, and adaptive actuation are seamlessly integrated throughout the production cycle. The *Monitoring* component plays a critical role: it is activated by the machine HMI subsystem when an operator starts the ice cream making process and oversees the HOT-AI system through the entire process, coordinating the interactions between all the components. The workflow begins with a continuous stream of data generated by the multitude of sensors (e.g., temperature, pressure, viscosity) integrated in the gelato machine. Raw sensor data are fed to the *Hard-O-Tronic (HOT)* system, a cutting-edge proprietary technology developed by Carpigiani that is capable of assessing the consistency and the quality of the ice cream preparation by analyzing raw sensor data with advanced heuristic algorithms. Raw sensor data enriched with data from HOT represents an information-rich data source, enabling a comprehensive analysis of the gelato mixture, which effectively allows to capture critical anomalies, such as the presence in defect or in excess of sugar, fat, water content, and other essential ingredients. These data are forwarded to the *data ingestion and preprocessing layer*, which is responsible for collecting, cleaning, and normalizing the data, preparing it for further analysis. Once processed, the data are stored in the *storage layer*, which archives all incoming information for future analysis.

The *multimilestone classifier* then performs inference on the relevant data in the storage layer, identifying potential anomalies caused by the unbalanced nature of the ingredient mixture loaded in the machine. More specifically, the component is designed to check the state of the mixture at four different stages (milestones) of the ice cream preparation process: 150, 200, 250, and 300 s, and to detect five different classes of mixture: *balanced, excess of cream, excess of sugar, insufficient sugar, excess of water*. Finally, note that the classifier is a pretrained ML model: HOT-AI does not implement any learning (or fine-tuning) phase on the ice cream making machine. The multimilestone classifier arguably represents the most important innovation within HOT-AI, and is discussed in depth in the Section IV.

In case an imbalanced mixture is detected, the multimilestone classifier notifies it to the *decision making* component, which decides whether to implement corrective actions. Toward that goal, the component considers the entire context of the ice cream making process, including the type of recipe selected by the operator, the current stage of the preparation process, etc. In fact, notifications of unbalanced mixtures received early in the ice cream preparation process might be slightly less accurate and could not require the immediate implementation of corrective actions. Predictions produced at early milestones are conservatively gated by the decision-making logic, which may consider class probabilities, confidence levels, and temporal consistency before enabling any adaptive actuation, thereby preventing premature or unstable process adjustments. If necessary, the decision making component sends instructions to the machine actuators, enabling precise calibration or adjustment of the gelato mixture process. This ensures the production remains optimized and that the final product consistently meets the highest quality standards.

IV. Multimilestone Classifier

Within HOT-AI, mixture classification must be performed at several stages during ice cream making, to have continuous feedback on the quality of the mixture throughout the process—with increasing levels of accuracy. As a result, the corresponding ML models must be capable of processing multivariate time series data with variable length, according to the time elapsed between the start of the gelato manufacturing process and the instant in which classification is actually executed, as illustrated in Fig. 2. In detail, Fig. 2 highlights the variable-length nature of the sensor time series and motivates the need for milestone-aware inference during production.

Input length variability presents a substantial challenge in model development, particularly those leveraging state-of-the-art solutions based on ANN. In fact, most ANN architectures include FC layers at the output stage, which typically require a fixed input size. Time series data is naturally sequential, making architectures, such as CNN and recurrent neural networks (RNN), suitable for this type of data. RNNs including variants like long short-term memory (LSTM) and gated recurrent unit (GRUs) maintain a hidden state across time steps, allowing them to process each sequence step-by-step. This sequential nature enables RNNs to handle variable-length inputs because they can keep track of temporal dependencies without requiring a fixed sequence length. CNNs apply filters over the input sequence, which enables them to capture localized patterns over time. The sliding nature of convolutional filters means that CNNs can process input sequences of various lengths during the feature extraction phase. Thus, the feature extraction layers in these networks can handle variable-length inputs with ease, allowing them to capture essential information regardless of input length.

However, while CNNs and RNNs provide flexibility in handling variable-length inputs during feature extraction, the FC layers commonly used in output layers present a challenge: by design, FC layers require fixed-size input vectors, as each neuron in the layer is connected to every output from the previous layer, making them incompatible with variable-length inputs. To address this issue, developers often employ preprocessing techniques that convert variable-length sequences into fixed-size representations, but this can lead to significant tradeoffs. One commonly used method to standardize input length is padding, which involves adding filler values to shorter sequences to make them the same length as the longest sequence. While padding does allow a single model to process time series data of varying lengths, it introduces challenges: the filler values lack meaningful information and may add noise to the model's learning process. This can degrade performance, as the model may learn patterns based on these artificial values rather than focusing on genuine trends in the data.

Developing multiple independent ML models, each designed to run at a specific stage (milestone) of the ice cream preparation process was an interesting possibility—but we ruled it out early in the design of HOT-AI. In fact, this approach allows each model to be tailored to the unique characteristics and features of the data at each milestone. In theory, this could simplify the handling of variable-length data since each model would focus on a fixed interval or specific time frame, potentially reducing the need for padding or complex preprocessing. However, deploying multiple models simultaneously, especially on edge devices, introduces significant computational challenges due to the resource limitations inherent to these devices, such as limited processing power, minimal memory and storage capacities, increase in energy consumption.

Another potentially interesting alternative would be to substitute the FC layer with a layer capable of handling variable-length time series data. This approach received a considerable amount of attention in scientific literature, leading to the development of several interesting methodologies and tools. For instance, global average pooling [23] is an operation that can replace the FC stages of a CNN by taking the mean value of every activation in each feature map, turning a $H \times W \times C$ tensor into a compact $1 \times 1 \times C$ vector that can be passed straight to a Softmax classifier, giving the model input-size invariance without adding extra parameters. Although this simplicity is attractive, the layer

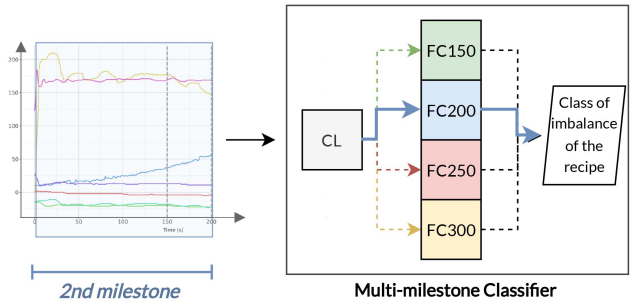

Fig. 3. Multimilestone classifier performing inference at time = +200 s from the beginning of the production process. The output is the class of imbalance, if any.

introduces some trade-offs: because every spatial location is averaged together, the network loses fine-grained positional cues. Also, having a global mean can also blur strong but sparse activations [24], reducing the discriminative power of successive layers. Spatial pyramid pooling (SPP) [25] is a technique used in computer vision for CNN to generate fixed-length representations from input images of arbitrary sizes by pooling features at multiple scales. It works by dividing the feature map into increasingly fine subregions and performing pooling operations within each of these divisions, effectively capturing multiscale spatial information. This method enables the network to process images without the need to resize them, preserving the original aspect ratio and content structure. Despite its advantages, SPP comes with several limitations. One of the main drawbacks is its increased computational complexity. Since SPP involves pooling across multiple levels of increasingly finer spatial bins, it adds a considerable computational burden, particularly for high-resolution inputs. In addition, the process of pooling across larger regions may lead to a potential loss of fine-grained spatial information. This can be problematic in applications requiring precise localization, as spatial details may be smoothed out or lost entirely. Another limitation is the increased model size and memory usage. Because the pooled features from each pyramid level are concatenated, the resulting feature representation can become quite large, leading to a greater number of connections with the subsequent FC layer, increasing memory consumption and potentially slowing down training and inference times, as shown in Section V. As a result, we decided to rule out this approach as well.

Instead, we decided to develop an innovative and *modular* solution based on the joint training of smaller modules that operate in an ensemble, that we called *multimilestone*. More specifically, our solution is based on five modules: a convolutional layer (CL) and four FC layers, corresponding to the number of milestones at which we intend to perform inference, as shown in Fig. 3. The modules are jointly trained.

The CL acts as a feature extractor: it applies convolutional filters that detect patterns and trends in the input data, capturing essential features that reveal the underlying structure of the time series, which are crucial for recognizing recipe imbalances. Fig. 3 illustrates how the FC layers perform the classification task at four distinct milestones by operating on the features extracted by the shared CL. Each FC layer is specialized for a specific inference point in the production process and predicts the state of the mixture (i.e., the presence and type of imbalance),

as denoted by the label "FC*M*," where M = 150, 200, 250, or 300 s.

This design was motivated by both technical and operational constraints. Technically, using a shared CL allows the model to learn a unified and generalizable representation of time-series data extracted from the gelato making machine sensors. This shared layer maximizes computational efficiency by avoiding redundant feature extraction across different models and significantly reduces the model's overall memory footprint, which is essential for resource-constrained edge devices. Operationally, each FC layer is specialized for a specific milestone in the production cycle. At inference time, the system activates only the FC layer corresponding to the current milestone. This selective activation minimizes computational load and reduces latency, making the solution suitable for real-time deployment without sacrificing responsiveness or performance. The result is a system that strikes an effective balance between specialization—through milestone-specific FC layers, and computational efficiency—through shared early layers.

Building on this foundation, the multimilestone inference strategy achieves an optimal tradeoff between model size, inference timeliness, and prediction accuracy. By continuously refining its predictions with newly collected sensor data, the system enhances its ability to detect subtle anomalies in the mixture composition. This streaming approach allows the multimilestone classifier to maintain a more accurate and evolving understanding of the process dynamics. Crucially, this architecture empowers the system to implement corrective actions promptly upon detecting imbalances, preventing minor deviations from escalating and ensuring the highest final product quality. By continuously leveraging inference at multiple milestones, the classifier dynamically updates and refines its predictions, progressively improving performance as additional sensor information becomes available.

V. Experimental Validation

This section first summarizes the dataset and experimental setup used to validate the proposed multimilestone classifier, and then presents the training and tuning methodology and performance test results. The dataset was collected from operational Carpigiani gelato machines and consists of multivariate time series sampled at 1 Hz from on-machine sensors monitoring key process variables, including temperature, pressure, viscosity, and motor-related signals, and enriched with features derived from the proprietary HOT system. Each time series corresponds to a complete production cycle of a single recipe and is labeled according to five mixture classes: a balanced recipe, excess of cream (Cream+), excess of sugar (Sugar+), insufficient sugar (Sugar-), and excess of water (Water+). The full dataset contains 103 production cycles and was split into training & tuning, and test sets using a 70% /30% split, ensuring that the same recipes were consistently assigned to the same subset across all experiments.

A. Training and Tuning Processes

In detail, the training of the multimilestone classifier required collecting test runs from Carpigiani machines while in operation and generating a training and test set from them. Carpigiani machines provide sensor readings every second and produce time series with variable lengths ranging from 250 s to 450 s, reflecting the different durations of the production cycles. In order to generate the dataset, we first decided to investigate the most appropriate time series length. Therefore, we generated seven different dataset versions, each one composed of 103 time series of varying lengths (150, 200, 250, 300, 350, 400, and 450 s). Table I quantifies the tradeoff between generated and truncated data across different time series lengths, motivating the selection of the 300 s dataset.

TABLE I
VERSIONS OF THE DATASET PRELIMINARILY TESTED FOR THE TRAINING OF THE MULTIMILESTONE CLASSIFIER

Dataset ID	TS length (s)	Data generated	Data truncated
D150	150	0.00%	49.27%
D200	200	0.41%	32.65%
D250	250	1.81%	17.00%
D300	300	6.81%	5.41%
D350	350	16.74%	1.45%
D400	400	26.39%	0.44%
D450	450	34.29%	0.00%

Each version contains time series of seven different lengths, 150–450 s, and has a different percentage of generated and truncated data. Each version of the dataset is identified by the "*DTime series length*" ID (where D means dataset). TS means time series.

TABLE II
CLASSES OF IMBALANCE: BALANCED RECIPE, EXCESS OF CREAM (CREAM+), EXCESS OF SUGAR (SUGAR+), REDUCTION IN SUGAR (SUGAR-) AND EXCESS OF WATER (WATER+)

Class of imbalance	Balanced	Cream+	Sugar+	Sugar-	Water+
Percentage	28%	24%	14%	10%	24%

The percentages indicate the proportion of each class in the D300 dataset.

Each dataset has a different percentage of data, either generated with padding (noise) to standardize sequence lengths, or truncated (lost): for time series shorter than the predefined length, padding was applied by replicating the last element until the series reached the target length. For time series longer than the predefined length, truncation was used to remove excess elements. Thanks to this analysis, we identified the *D300* dataset, composed of time series of length 300 s, as the one minimizing both generated and truncated data, so we decided to use that next. Data was also cleaned by removing outliers and applying normalization to standardize input features.

Every time series in the dataset corresponds to a production cycle of a single recipe. Table II summarizes the five mixture imbalance classes considered, reflecting realistic operator-induced deviations in industrial production. To label time series (recipes) as balanced or unbalanced, we referred to the five classes: *balanced recipe* represents the standard, correct recipe; *Cream+* indicates an increased cream content of $+[300–400\%]$ compared to the balanced recipe, *Sugar+* an increased sugar level of $+[33–100\%]$; *Sugar-* a reduction of $-[33–67\%]$, and *Water+* an increased water content of $+[10–20\%]$. These imbalances were intentionally introduced in the test runs to mimic potential operator errors or variances in production conditions.

The *D300* dataset was split into a training set of 71 time series (70%) and a test set (30%) of 32 time series. On such a split we trained and tested various architectures: CNNs, RNNs, GRUs, LSTMs, and hybrid combinations, given their demonstrated strengths in handling time series. After thorough experimentation, we ultimately opted for a CNN, due to its superior performance and computational efficiency, which aligned well

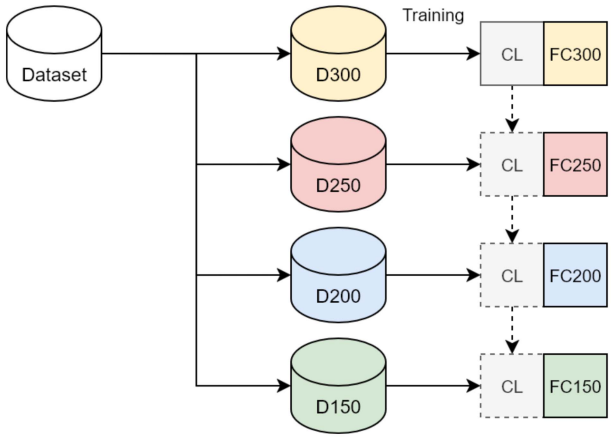

Fig. 4. Overview of the training phase of the multimilestone classifier highlighting the transfer learning approach. The *D300* dataset is used for learning a CNN, composed of a CL and a FC layer, for performing classification at milestone = 300 s. The other three datasets are used for learning a specialized FC layer for performing classification at milestones M = 150 s, 200 s, 250 s, while the CL is kept unmodified.

with the resource constraints of edge devices. The training of the CNN allowed us to learn robust feature extraction patterns, essential for classifying time series.

Fig. 4 clarifies the transfer learning strategy used to reuse a single feature extractor while specializing classifiers for different inference milestones. In detail, the CNN trained on the *D300* dataset is composed of a CL and a FC layer, indicated respectively by "CL" and "FC300." This network is capable of performing classification of new recipes at time +300 s. However, as our goal was to create a multimilestone classifier able to perform inference not only at +300 s, but also at earlier stages (+150 s, +200 s, +250 s) as explained in Section III, the next experimental step consisted of training three new FC layers on different datasets, characterized by shorter time series, in particular *D250*, *D200*, and *D150*, where the time series length corresponds to the milestones relevant to us. These last three datasets were also split into training/test sets in the same way as *D300*, dividing the same recipes between the two sets according to a 70/30% split.

Note that, rather than training a new CNN from scratch on each dataset, we reused the pretrained convolutional module from the CNN learnt from *D300*. This layer is "frozen," meaning it is not updated when training the FC layers, as it already retains the feature extraction capabilities learned from the longer time series. By reusing the convolutional module, the model remains computationally efficient, as only the FC layers—which are more sensitive to input length variations—require additional training. The overall approach is highlighted in Fig. 4, where the "CL" module is depicted with dashed lines to indicate that it is not modified. This technique takes inspiration from transfer learning [9], where a model is initially trained on a source domain D_s and then adapted to a specific target domain D_t by fine-tuning its parameters and weights, provided that the domains are closely related. This approach allows us to preserve the consistency of the learnt feature representations from the original dataset while adjusting the final classification stage for varying time series lengths through targeted fine-tuning.

TABLE III
ACCURACY (ACC) AND F1-SCORE ON TRAINING AND TEST SETS FOR THE TWO TRAINING EXPERIMENTS: D_{300} TO D_{150} AND D_{150} TO D_{300}

(a) Performance Metrics for the transition from D_{300} to D_{150}.

Layer	Acc. Training Set	Acc. Test Set	F1 Score
FC300	98.59%	**96.88%**	**0.9775**
FC250	97.18%	**96.88%**	**0.9775**
FC200	94.37%	**93.75%**	**0.9525**
FC150	92.95%	**90.63%**	**0.9214**

(b) Performance Metrics for the transition from D_{150} to D_{300}.

Layer	Acc. Training Set	Acc. Test Set	F1 Score
FC150	94.37%	81.25%	0.8341
FC200	87.32%	84.38%	0.8584
FC250	98.59%	84.38%	0.8571
FC300	98.59%	87.50%	0.8795

Bold values highlight the best test set performance.

Note that a feature hierarchy can be identified going from the smallest to the largest of the four datasets, i.e.,

$$F_{\text{ts}}(D150) \subset F_{\text{ts}}(D200) \subset F_{\text{ts}}(D250) \subset F_{\text{ts}}(D300)$$

where $F_{\text{ts}}(D)$ represents the set of features derived from the time series (ts) of dataset D. This means, for example, that the *D150* dataset captures only a fraction of the features present in the *D200* dataset, and so on. This hierarchical relationship enables efficient knowledge transfer from the longest time series *D300*, capturing the full feature set, down to shorter time series datasets.

To validate this theoretical approach, we empirically tested two different training strategies: a "progressive" approach (150 to 300) versus a "reverse" (300 to 150) one. Table III shows that training from longer to shorter sequences yields more robust feature representations and higher test accuracy. The progressive approach (starting with shorter sequences and moving to longer ones) resulted in less robust feature extraction and lower overall accuracy when the model was later tested on longer time series. This method limits the model's ability to generalize when presented with more complex, extended time series, as was trained on shorter sequences.

When training the CL and the FC layers, hyperparameter optimization was performed using Optuna,[1] a highly flexible and library-agnostic tool that can integrate seamlessly with any machine learning or deep learning framework. We optimized hyperparameters such as the number of convolutional filters, kernel sizes, number of neurons in each FC layer, learning rate, and dropout rates. This rigorous search allowed us to fine-tune the architecture for maximum performance, ensuring the model's performance met the real-time requirements of the gelato production process while remaining lightweight enough for deployment on edge devices. Table IV reports the Optuna-optimized architectural configurations for each layer of the classifier.

Finally, we compared our resulting multimilestone classifier with a more generic, input-length-agnostic alternative that integrates an SPP layer. This architecture consists of a CL layer followed by an SPP layer and a FC classifier—i.e., CL $\rightarrow$ SPP $\rightarrow$ FC. The SPP layer is designed to handle variable-length time

[1] https://optuna.org/

TABLE IV
ARCHITECTURE DETAILS FOR THE OPTIMIZED CL AND FC LAYERS IN OUR MULTIMILESTONE CLASSIFIER

Layer	Configuration
CL	Conv1d (In: 7, Out: 128, Kernel: 6, Stride: 1) MaxPool1d (Kernel: 4, Stride: 4) Conv1d (In: 128, Out: 64, Kernel: 4, Stride: 1), MaxPool1d (4, 4)
FC300	Flatten → Linear (1088 → 74) → ReLU → Dropout (p=0.2) Linear (74 → 86) → ReLU → Dropout (p=0.2) Linear (86 → 5)
FC250	Flatten → Linear (896 → 31) → ReLU → Dropout (p=0.2) Linear (31 → 62) → ReLU → Dropout (p=0.2) Linear (62 → 140) → ReLU → Dropout (p=0.2) Linear (140 → 5)
FC200	Flatten → Linear (704 → 128) → ReLU → Dropout (p=0.2) Linear (128 → 42) → ReLU → Dropout (p=0.2) Linear (42 → 26) → ReLU → Dropout (p=0.2) Linear (26 → 5)
FC150	Flatten → Linear (512 → 96) → ReLU → Dropout (p=0.2) Linear (96 → 35) → ReLU → Dropout (p=0.2) Linear (35 → 5)

TABLE V
COMPARISON IN TERMS OF ACCURACY (ACC.) AND AVERAGE INFERENCE TIME BETWEEN THE MULTIMILESTONE CLASSIFIER AND THE SPP-BASED ARCHITECTURE

Milestone	Train acc. [%]		Test acc. [%]		Avg infer. time [s]	
	Ours	SPP	Ours	SPP	Ours	SPP
300	98.59	89.00	96.88	85.00	0.0714	0.159
250	97.18	87.00	96.88	85.00	0.0713	0.162
200	94.37	85.00	93.75	85.00	0.0705	0.167
150	92.95	86.00	90.63	82.00	0.0594	0.174
Mean	**95.77**	86.75	**94.54**	84.25	**0.068**	0.166

In bold the highest mean values.

series inputs by producing a fixed-length output vector, thereby avoiding the need for padding or truncation. This way, we would replace the milestone-specific FC layers used in our architecture with a single FC classifier operating on the SPP-transformed features. This network was trained on combined datasets of time series of different lengths, which ranged from 150 s to 300 s. To ensure a fair comparison, we used the same time series of the previous experiments. The training and optimization procedures were kept consistent; however, we introduced the *Output Size* of the SPP layer as a tunable hyperparameter, providing the layer with a flexible search space to achieve optimal performance. After the optimization process, Optuna found that an output size of pooling equal to 91×91 offered the best performance, preserving the temporal information for an accurate classification.

B. Testing Results

Table V compares our solution against the SPP-based architecture, showing that the latter reaches a lower accuracy and a higher inference time. Also, a great difference lies in the file size, as the SPP model occupies 2.7 MB, whereas ours requires only 1.24 MB. This advantage is a direct consequence of the Optuna-driven architectural optimization of the proposed model, which yields compact milestone-specific classifiers with a reduced number of trainable parameters and minimal activation dimensionality, whereas the SPP-based approach relies on high-dimensional pooled representations that increase both parameter count and per-inference computational cost. This significant reduction in memory footprint is crucial for deployment on resource-constrained edge devices, such as those used in Carpigiani machines. Overall, these results demonstrate that the proposed multimilestone classifier not only outperforms the SPP-based model in terms of accuracy and inference speed but is also substantially more lightweight and efficient, making it a more suitable and scalable solution for real-time intelligent applications.

TABLE VI
AVERAGE FILE SIZE FOR EVERY MODULE OF THE MULTIMILESTONE CLASSIFIER

	CL	FC300	FC250	FC200	FC150	Total
File Size (KB)	152	344	156	382	210	1240

For the final deployment of our platform, a Raspberry Pi 4 with 2 GB of RAM was used. This choice was driven by its cost-effectiveness, energy efficiency, and adequate processing power to handle the real-time data processing and classification tasks required by the system. Table VI highlights the compact size of each model component, confirming the suitability of the approach for resource-constrained edge hardware. The total model size (1.24 MB) corresponds to the sum of the individual file sizes for each layer, making it a very lightweight solution to operate on the Raspberry Pi. The average inference times shown in Table V are less than a tenth of a second, ensuring rapid response for real-time applications.

VI. RELATED WORK

Recent research has demonstrated significant advancements in applyingML to industrial processes, particularly in the areas of smart maintenance and quality control within Industry 4.0 and 5.0 frameworks. A comprehensive survey on machine learning-based smart maintenance solutions by Frankó et al. [26] highlighted the increasing trend of integrating advanced ML techniques into industrial operations to enhance reliability, accuracy, and timeliness in predictive maintenance and quality control.

One of the key areas of focus is anomaly detection (AD), which has shown great promise in identifying irregularities in real-time, ensuring that faults are detected early and corrected before they lead to operational failures. For example, Bellavista et al. [27] implemented a window-based statistical data processing method, followed by the application of an isolation forest algorithm to detect anomalies in working machines. Their work demonstrated the ability to continuously monitor industrial processes and identify anomalies with high precision. Similarly, Colombi et al. [28] employed autoencoders to create a representational latent space of processing windows passing from the time series domain to an embedding domain, applying various ML-based AD algorithms. They compared their methods with AD approaches based solely on reconstruction errors, achieving improved performance.

While many industrial analytics applications have relatively loose constraints in terms of latency and bandwidth, and are thus compatible with centralized cloud-based approaches, an ever growing set of innovative services present more challenging

requirements that call instead for edge computing architectures [20], [21]. HOT-AI falls well within the industrial edge AI research avenue, and represents an innovative large-scale application outside the shop floor.

Recently, many applications have started considering a broader perspective and turning their attention towards compute continuum approaches. The compute continuum is a new paradigm that attempts to unify edge, fog and cloud computing into a single distributed environment where resources can be seamlessly allocated from edge devices to the cloud [29], [30]. The capability to allocate and manage workloads across the continuum in a homogenous fashion facilitates and foster the exploration of various tradeoffs between cloud and edge deployments, with potentially significant benefits for many applications—particularly ML-based ones.

Related to this, Ebiaredoh-Mienye et al. [31] employed an information-gain–based filter feature selection method combined with a cost-sensitive AdaBoost classifier for chronic kidney disease prediction. The approach reduces the feature space while explicitly handling class imbalance during learning. Performance is evaluated against standard machine learning classifiers using the reduced feature set. Ebiaredoh-Mienye et al. [32] proposed a hybrid medical diagnosis framework that integrates an enhanced sparse autoencoder for unsupervised feature learning with an optimized Softmax regression classifier, specifically targeting the challenges of imbalanced medical datasets. Unlike conventional SAEs that enforce sparsity on neuron activations, their method regularizes network weights, leading to more stable feature representations and improved classification performance. Similar Ebiaredoh-Mienye et al. [33] introduced an unsupervised feature learning approach based on a stacked sparse autoencoder for credit card default prediction. The learned latent representations are used to train downstream classifiers, improving robustness on dynamic and imbalanced data. Performance is compared against models trained on raw features and prior machine learning approaches Relatively, Esenogho et al. [34] proposed a neural network ensemble framework using LSTM networks as base learners within an AdaBoost scheme for credit card fraud detection. To address class imbalance, a hybrid resampling strategy combining SMOTE and edited nearest neighbor was employed. Obaido et al. [35] applied multiple machine learning classifiers, including decision tree, logistic regression, support vector machine (SVM), random forest, AdaBoost, and XGBoost, for hepatitis B diagnosis. Model interpretability is incorporated using shapley additive explanations to explain and visualize individual feature contributions, emphasizing transparent prediction analysis alongside comparative model performance evaluation. In a related work in the cybersecurity domain by Kumari et al. [36] presented an intrusion detection framework that integrates the gravitational search algorithm for feature selection with SMOTE-IPF for class imbalance handling. Machine learning classifiers, particularly random forest, are employed for attack classification.

Within this research avenue, the optimization of the deployment of ML models across the industrial compute continuum, to achieve good tradeoffs between accuracy and latency, represents a key area of investigation. Studies, such as [37], examine the impact of edge offloading, where certain tasks are processed locally while others are offloaded to the cloud. Other works, such as [38], focus on the latency introduced by deploying compressed and reduced versions of ML and deep learning models on edge servers. Adopting a similar approach, which instead investigates the tradeoff between energy consumption and latency, Zhang et al. [39] have investigated methods to optimize this balance, ensuring that real-time responses are maintained without excessively draining the energy resources of edge devices. However, these studies often overlook the effects of AI service offloading, where part of the processing is done remotely, and its impact on the overall performance of the system. Another interesting area of investigation focuses on improving efficiency by optimizing bandwidth usage across the continuum. Toward that goal, recent approaches, such as deep compressed sensing, and the use of auto-encoders [19], [40] have been proposed. These methods compress data for more efficient transmission, but this comes at the cost of increased computational demands on edge devices, which can pose significant challenges in resource-limited environments. With the ongoing integration of Teorema and HOT-AI, Carpigiani's analytics platform is rapidly aligning with the compute continuum paradigm. In fact, we are currently investigating innovative approaches that leverage compute continuum methodologies and tools to implement the distributed fine-tuning of the HOT-AI ML modules, exploiting potentials for transfer learning while at the same time respecting data privacy concerns. This could allow HOT-AI to achieve a highly contextual behavior, and consequently even higher accuracy levels.

VII. Conclusion and Future Work

This work addresses an industrial informatics problem by closing the loop from on-machine sensing to edge-based intelligence and adaptive actuation in an industrial ice cream production process. We presented HOT-AI, a multimilestone edge machine learning system that performs on-device inference during production and dynamically adapts machine control parameters to compensate for ingredient imbalances. The proposed architecture achieves high classification accuracy (up to 96.9% at the final milestone) while maintaining low inference latency (≈70 ms) and a compact footprint (1.24 MB), making it suitable for deployment on edge industrial hardware. By enabling earlier and progressively more reliable detection of mixture deviations, HOT-AI supports faster and more precise corrective actions, improving batch uniformity and significantly reducing material waste in line with zero-defect and zero-waste manufacturing goals. Although the experimental validation focuses on a limited set of representative imbalance classes associated with the main ingredients (water, sugar, and cream), future work will extend the proposed approach to additional recipes and investigate the identification of previously unseen anomalies. By learning the temporal evolution of process dynamics from on-machine sensor signals rather than relying on explicit ingredient quantities, HOT-AI can be scaled to new recipe types through the introduction of additional labeled classes or by fine-tuning the milestone-specific classifiers, without requiring modifications to the overall system architecture. In parallel, future work will investigate distributed learning paradigms, such as federated learning, to derive more robust global models across the machine fleet, which can subsequently be fine-tuned to address machine-specific and operational requirements.

References

[1] H. D. Goff et al. *Ice Cream*. Cham, Switzerland: Springer Nature, 2025.

[2] P. K. Wan and T. L. Leirmo, "Human-centric zero-defect manufacturing: State-of-the-art review, perspectives, and challenges," *Comput. Ind.*, vol. 144, Jan. 2023, Art. no. 103792, doi: 10.1016/j.compind.2022.103792.

[3] S. Singh et al., "Towards zero waste manufacturing: A multidisciplinary review," *J. Cleaner Prod.*, vol. 168, pp. 1230–1243, Dec. 2017, doi: 10.1016/j.jclepro.2017.09.108.
[4] R. X. Gao et al., "Artificial intelligence in manufacturing: State of the art, perspectives, and future directions," *CIRP Ann.*, vol. 73, no. 2, pp. 723–749, 2024, doi: 10.1016/j.cirp.2024.04.101.
[5] R. Venanzi et al., "Collective intelligence-based service migration enabling zoom-in functionality within industry 5.0," *Internet of Things*, vol. 35, 2026, Art. no. 101830, doi: 10.1016/j.iot.2025.101830.
[6] G. Dimitrakopoulos et al., "Industry 5.0: Research areas and challenges with artificial intelligence and human acceptance," *IEEE Ind. Electron. Mag.*, vol. 18, no. 4, pp. 43–54, Dec. 2024, doi: 10.1109/MIE.2024.3387068.
[7] R. Venanzi et al., "Enabling adaptive analytics at the edge with the Bi-Rex Big Data platform," *Comput. Ind.*, vol. 147, 2023, Art. no. 103876, doi: 10.1016/j.compind.2023.103876.
[8] L. Ren, J. Dong, X. Wang, Z. Meng, L. Zhao, and M. J. Deen, "A data-driven Auto-CNN-LSTM prediction model for lithium-ion battery remaining useful life," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 3478–3487, May 2021, doi: 10.1109/TII.2020.3008223.
[9] S. Shao, S. McAleer, R. Yan, and P. Baldi, "Highly accurate machine fault diagnosis using deep transfer learning," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2446–2455, Apr. 2019, doi: 10.1109/TII.2018.2864759.
[10] S. Dahdal et al., "An MLOps framework for GAN-based fault detection in Bonfiglioli's EVO plant," *Infocommunications J.*, vol. 16, pp. 2–10, 2024, doi: 10.36244/ICJ.2024.2.1.
[11] A. Carvalho et al., "Edge computing applied to industrial machines," *Procedia Manuf.*, vol. 38, pp. 178–185, 2019, doi: 10.1016/j.promfg.2020.01.024.
[12] A. Corradi et al., "Smart appliances and RAMI 4.0: Management and servitization of ice cream machines," *IEEE Trans. Ind. Informat.*, vol. 15, no. 2, pp. 1007–1016, Feb. 2019, doi: 10.1109/TII.2018.2867643.
[13] M. Sharma, A. Tomar, and A. Hazra, "Edge computing for industry 5.0: Fundamental, applications, and research challenges," *IEEE Internet Things J.*, vol. 11, no. 11, pp. 19070–19093, Jun. 2024, doi: 10.1109/JIOT.2024.3359297.
[14] W. Xiang, K. Yu, F. Han, L. Fang, D. He, and Q. -L. Han, "Advanced manufacturing in industry 5.0: A survey of key enabling technologies and future trends," *IEEE Trans. Ind. Informat.*, vol. 20, no. 2, pp. 1055–1068, Feb. 2024, doi: 10.1109/TII.2023.3274224.
[15] J. Xu, A. Xu, L. Chen, Y. Chen, X. Liang, and B. Ai, "Deep reinforcement learning for RIS-aided secure mobile edge computing in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 20, no. 2, pp. 2455–2464, Feb. 2024, doi: 10.1109/TII.2023.3292968.
[16] A. Joshi et al., "Integration of Edge–AI into IoT–Cloud architecture for landslide monitoring and prediction," *IEEE Trans. Ind. Informat.*, vol. 20, no. 3, pp. 4246–4258, Mar. 2024, doi: 10.1109/TII.2023.3319671.
[17] J. Wang, S. Agarwal, D. P. Kanungo, and R. K. Panigrahi, "An efficient deep neural network for surface defect detection in industrial edge sensing," *IEEE Trans. Ind. Informat.*, vol. 21, no. 3, pp. 2560–2569, 2025, doi: 10.1109/TII.2024.3507954.
[18] D. Liang, H. Zhang, Q. Han, D. Yuan, and M. Zhang, "RasPiDets: A quasi-real-time defect detection method with end-edge-cloud collaboration," *IEEE Trans. Ind. Informat.*, vol. 21, no. 7, pp. 5525–5535, Jul. 2025, doi: 10.1109/TII.2025.3556031.
[19] A. M. Ghosh et al., "Edge-cloud computing for Internet of Things data analytics: Embedding intelligence in the edge with deep learning," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 2191–2200, Mar. 2021, doi: 10.1109/TII.2020.3008711.
[20] C. Savaglio et al., "Edge intelligence for industrial IoT: Opportunities and limitations," in *Proc. 5th Int. Conf. Ind. 4.0 Smart Manuf.*, 2024, pp. 397–405, doi: 10.1016/j.procs.2024.01.039.
[21] F. Foukalas and A. Tziouvaras, "Edge artificial intelligence for industrial Internet of Things applications: An industrial edge intelligence solution," *IEEE Ind. Electron. Mag.*, vol. 15, no. 2, pp. 28–36, Jun. 2021, doi: 10.1109/MIE.2020.3026837.
[22] S. Dahdal et al., "RoamML distributed continual learning: Adaptive and flexible data-driven response for disaster recovery operations," *J. Netw. Comput. Appl.*, vol. 243, 2025, Art. no. 104322, doi: 10.1016/j.jnca.2025.104322.
[23] M. Lin et al., "Network in network," in *Proc. 2nd Int. Conf. Learn. Representations*, 2014, doi: 10.48550/arXiv.1312.4400.
[24] B. Zhang et al., "AlphaMEX: A smarter global pooling method for convolutional neural networks," *Neurocomputing*, vol. 321, pp. 36–48, 2018, doi: 10.1016/j.neucom.2018.07.079.
[25] K. He et al., "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 346–361, doi: 10.1007/978-3-319-10578-923.
[26] A. Frankó et al., "Applied machine learning for IIoT and smart production—Methods to improve production quality, safety and sustainability," *Sensors*, vol. 22, no. 23, 2022, Art. no. 9148.
[27] P. Bellavista, S. Dahdal, L. Foschini, D. Tazzioli, M. Tortonesi, and R. Venanzi, "Kubernetes enhanced stateful service migration for ML-Driven applications in industry 4.0 scenarios," in *Proc. 2024 IEEE Annu. Congr. Artif. Intell. Things*, 2024, pp. 25–31, doi: 10.1109/AIoT63253.2024.00015.
[28] L. Colombi et al., "Embedding models for multivariate time series anomaly detection in industry 5.0," *Data Sci. Eng.*, Jun. 2025, doi: 10.1007/s41019-025-00295-w.
[29] M. Tortonesi, "The compute continuum: Trends and challenges," *Computer*, vol. 58, no. 03, pp. 105–108, 2025, doi: 10.1109/MC.2024.3520255.
[30] L. F. Bittencourt et al., "The computing continuum: Past, present, and future," *Comput. Sci. Rev.*, vol. 58, Nov. 2025, Art. no. 100782, doi: 10.1016/j.cosrev.2025.100782.
[31] S. A. Ebiaredoh-Mienye et al., "A machine learning method with filter-based feature selection for improved prediction of chronic kidney disease," *Bioengineering*, vol. 9, no. 8, 2022, Art. no. 350, doi: 10.3390/bioengineering9080350. [Online]. Available: https://www.mdpi.com/2306-5354/9/8/350
[32] S. A. Ebiaredoh-Mienye et al., "Integrating enhanced sparse autoencoder-based artificial neural network technique and softmax regression for medical diagnosis," *Electronics*, vol. 9, no. 11, 2020, Art. no. 1963, doi: 10.3390/electronics9111963. [Online]. Available: https://www.mdpi.com/2079-9292/9/11/1963
[33] S. A. Ebiaredoh-Mienye et al., "Artificial neural network technique for improving prediction of credit card default: A stacked sparse autoencoder approach," *Int. J. Elect. Comput. Eng.*, vol. 11, no. 5, Oct. 2021, Art. no. 4392, doi: 10.11591/ijece.v11i5.pp4392-4402.
[34] E. Esenogho, I. D. Mienye, T. G. Swart, K. Aruleba, and G. Obaido, "A neural network ensemble with feature engineering for improved credit card fraud detection," *IEEE Access*, vol. 10, pp. 16400–16407, 2022, doi: 10.1109/ACCESS.2022.3148298.
[35] G. Obaido et al., "An interpretable machine learning approach for hepatitis B diagnosis," *Appl. Sci.*, vol. 12, no. 21, 2022, Art. no. 11127, doi: 10.3390/app122111127. [Online]. Available: https://www.mdpi.com/2076-3417/12/21/11127
[36] M. Kumari et al., "An optimized IDS framework for Big Data environments: Integrating gravitational search and SMOTE-IPF data balancing for high-accuracy IDS," *SN Comput. Sci.*, vol. 6, no. 7, Aug. 2025, Art. no. 786, doi: 10.1007/s42979-025-04194-9.
[37] W. Sun, J. Liu, and Y. Yue, "AI-enhanced offloading in edge computing: When machine learning meets industrial IoT," *IEEE Netw.*, vol. 33, no. 5, pp. 68–74, Sep./Oct. 2019, doi: 10.1109/MNET.001.1800510.
[38] S. Wang et al., "When edge meets learning: Adaptive control for resource-constrained distributed machine learning," in *Proc. IEEE Conf. Comput. Commun.*, 2018, pp. 63–71, doi: 10.1109/INFOCOM.2018.8486403.
[39] G. Zhang, W. Zhang, Y. Cao, D. Li, and L. Wang, "Energy-delay tradeoff for dynamic offloading in mobile-edge computing system with energy harvesting devices," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4642–4655, Oct. 2018, doi: 10.1109/TII.2018.2843365.
[40] M. Zhang, H. Zhang, C. Zhang, and D. Yuan, "Communication-efficient quantized deep compressed sensing for edge-cloud collaborative industrial IoT networks," *IEEE Trans. Ind. Informat.*, vol. 19, no. 5, pp. 6613–6623, May 2023, doi: 10.1109/TII.2022.3202203.

Filippo Tabanelli received the master's degree in computer engineering from University of Ferrara, Ferrara, Italy, in 2024.

He is a Research Assistant with the University of Ferrara, in the Distributed Systems Research Group, Ferrara, Italy, where he is actively involved in leveraging Big Data and Artificial Intelligence to address complex, real-world industrial challenges.

Simon Dahdal (Graduate Student Member, IEEE) is currently working toward the Ph.D. degree in engineering science with the Engineering Department, University of Ferrara, in the Distributed Systems Research Group, Ferrara, Italy.

His current research interests incude applied AI in industry 4.0 and human assistance and disaster recovery scenarios.

Nicolas Belletti received the master's degree in computer engineering from University of Ferrara, Ferrara, Italy, in 2024.

He is a Research Assistant with the University of Ferrara, in the Distributed Systems Research Group, Ferrara, Italy. His current research focuses on the application of Big Data and Machine Learning in industrial environments.

Elena Bellodi received the Ph.D. degree in engineering sciences from University of Ferrara, Ferrara, Italy, in 2013.

She is an Associate Professor with the Department of Engineering, University of Ferrara, Ferrara, Italy, where she leads the Statistical Relational Artificial Intelligence Laboratory. Her research interests include machine learning, probabilistic logic programming, statistical relational AI, process mining, and natural language processing.

Franck Ngatcha received the master's degree in computer engineering from University of Ferrara, Ferrara, Italy, in 2013.

He is currently a Computer Science Engineer and the Head of Carpigiani's Software and Electronics Development Division.

Roberto Lazzarini received the Ph.D. degree in engineering science form University of Ferrara, Ferrara, Italy, in 2015.

He serves as the CTO with Carpigiani—Ali Group, Bologna, Italy. He has coauthored many several international scientific publications and the coinventor of numerous patents, including 120 granted in the United States. He is also actively involved in standardization activities, serving as the Convenor of the CEN (Comité Européen de Normalisation) working group TC153/WG6 on "Artisan Ice Cream Machinery."

Cesare Stefanelli (Member, IEEE) received the Ph.D. degree in computer science engineering from University of Bologna, Bologna, Italy, in 1996.

He is a Full Professor, Lead of the Distributed Systems research group, and Director of the Engineering Department with the University of Ferrara, Ferrara, Italy. He has authored or coauthored more than 200 publaiction, holds several patents, and coordinates industrial research projects carried on in collaboration with world-leading companies.

Mauro Tortonesi (Member, IEEE) received the Ph.D. degree in engineering science from University of Ferrara, Ferrara, Italy, in 2006.

He is an Associate Professor and the Head of the Big Data and Compute Continuum Research Laboratory with the University of Ferrara, Ferrara, Italy. He has 130+ publications, holds several patents, and participates with several roles in a wide number of research projects in the distributed systems area, with particular reference to Compute Continuum, Big Data, and IoT solutions in industrial and military environments.