IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 3, 2021, accepted August 26, 2021, date of publication September 7, 2021, date of current version September 17, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3111113

Neural Network Techniques for Detecting Intra-Domestic Water Leaks of Different Magnitude

RICCARDO ZESE, ELENA BELLODI, CHIARA LUCIANI, AND STEFANO ALVISI
Department of Engineering, University of Ferrara, 44122 Ferrara, Italy

Corresponding author: Riccardo Zese (riccardo.zese@unife.it)

This work was supported in part by the Regione Emilia-Romagna within the context of the POR FESR 2014-2020 Project "Green Smart Technology for Water (GST4Water)."

ABSTRACT Nowadays, water leak control at different levels is a necessary tool for sustainable water resource management. Research shows that more than one third of the world's drinking water is lost during its transfer to users, and that leakages on users' properties vary between 2 and 13% of total residential water demand, are very frequent and difficult to detect. Thanks to the advances in Internet of Things solutions for smart metering devices, it is possible to gather household water consumption information with high spatial and temporal resolution and to analyse them. This article applies several supervised Machine Learning (ML) techniques for the automatic detection of leakages of different magnitudes - even smaller than the meter sensitivity – in pipes within the dwelling, by using data collected by smart meters installed at the connection of users to the distribution network in an Italian town. The results obtained are compared with the performance of an "empirical algorithm" previously presented by the authors, able to automatically identify leakages by checking if the hourly flow rate is never zero during the whole day, but not able to distinguish the size of the leakages. Experimental results over about 40,500 records show that ML techniques significantly improve the detection performance both in discriminating between presence and absence of leakages and in discriminating different-size leakages.

INDEX TERMS Machine learning, predictive analytics, smart water metering, small leakage detection.

I. INTRODUCTION

In recent years, ensuring a balance between water demand and availability is one of the most important issues that governments and authorities must face. Around the world, many countries adopt management policies that encourage the adoption of strategies aimed at conserving and safeguarding water resources [1]. In this context, water leak control at different levels is a necessary tool for sustainable water resource management. Research conducted by [2] shows that more than one third of the world's drinking water is lost during its transfer to users. Other studies [3], [4] estimate that leakages on users' properties vary between 2 and 13% of total residential water demand. Until recently, most of the efforts have focused on leaks in water distribution networks or district metered areas (DMA). However, nowadays, thanks to the advances in Internet of Things (IoT) solutions for smart metering devices [5], based on new generation communication protocols, it is possible to gather household water consumption information with high spatial and temporal resolution.

Several solutions have been developed at the infrastructure level, with the objective of building data collection infrastructures to monitor water consumption at the urban consumer level [6], [7]. Thus, these technologies represent powerful tools for supporting water sustainability and are increasingly being used by Water Utilities (WU) to implement their management strategies [8].

Less attention has received the analysis of data collected by smart metering devices, sometimes gathered with very high frequency, for the automatic detection of leakages. Detection goes from large leakages due to broken pipes, to medium-sized leakages due to the faulty operation of plumbing systems and sanitary appliances, to small leakages

The associate editor coordinating the review of this manuscript and approving it for publication was Wai-Keung Fung.

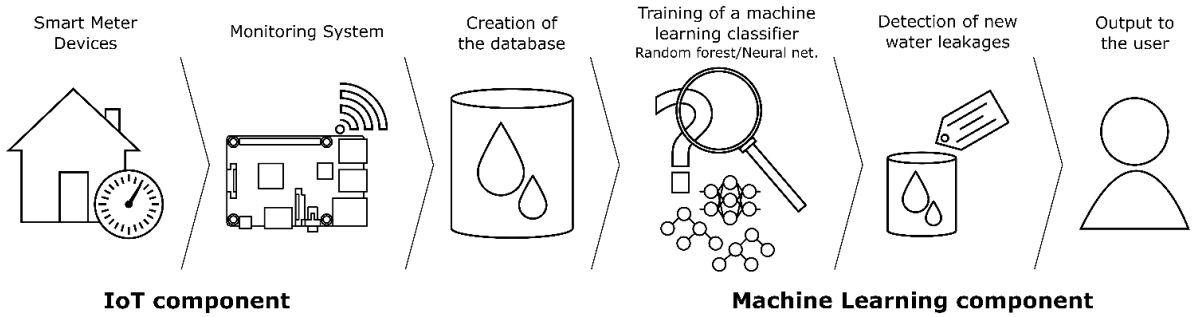


FIGURE 1. Workflow from the acquisition of water consumption data to the detection of water leakages.

due to fitting dripping. In this work we applied several supervised Machine Learning (ML) techniques - Random Forest and Neural Networks - *for the automatic detection of leakages of different magnitudes in pipes within the dwelling* and compared the results with a third methodology presented in [9]. In particular, *our target is detecting leakages possibly smaller than the meter sensitivity, an issue that has not been faced in the literature yet*. Data were collected by smart meters installed at the connection of users to the distribution network of a DMA in the town of Gorino Ferrarese (Italy), as part of the research project GST4Water (Green Smart Technology for Water[1]). The smart meters collected only the flow rate of water in the pipe entering the house. The collected data were used to train a Random Forest and several topologies of Neural Networks.

Both the Random Forest (RF) and the Neural Network (NN) classifiers significantly improve the performance of [9] in discriminating between presence and absence of leakages in water consumption records, but, more importantly, NNs also show very high performance in discriminating different-size leakages, especially the small ones, achieving Accuracy, Precision and Recall in the range 92-96%, and Area under the PR and ROC curves between 97% and 99%.

The advantages of a Machine Learning based approach are manifold:

- no knowledge about factors influencing water consumption is required by experts to tune the system to improve the quality of the classification;
- no knowledge is required about householders' habits;
- RF and NN are designed using open-source frameworks, so the approach does not require any proprietary software;
- the resulting model, in particular NN, can be applied to new unlabelled water consumption records and can discriminate among leaks of different magnitudes without any human intervention;
- the resulting model (both RF and NN) is small enough to be stored also in the smart meter or used by a mobile app, facilitating the monitoring of the domestic water system by the users themselves;
- the approach can be generalized to any set of time series to detect unnecessary waste due to water leak within the house at an early stage and can further improve performance as more training data become available.

As far as we know, this is the first application of ML techniques to the problem of automatic leak detection at domestic level, that requires only household water consumption data and can detect leaks of different size, especially lower than the meter sensitivity.

The paper is organized as follows: Section II discusses Internet-of-Thing approaches to the problem of smart water metering, while Section III presents related work. Section IV illustrates the machine learning techniques applied for leak detection, and Section V gives the details of the experimental evaluation. Finally, Section VI discusses the overall results and Section VII concludes the paper.

II. IoT FOR SMART WATER METERING

Despite smart water metering being a relatively recent practice, there are several smart water meters available on the market. The first generation of smart water meters adopted low power short-range wireless protocols, such as Wireless M-Bus, that operates over the unlicensed spectrum (169 or 868 MHz in Europe). These meters were designed to operate in "Remote Meter Reading" systems, in which data collection can be performed without a dedicated networking infrastructure: operators equipped with portable receivers collect data in the proximity of the smart meters either in walk-by or drive-by mode. These systems do not allow either real-time or automated consumption monitoring.

More recently, a second generation of smart water meters that leverage low-power long-range wireless protocols, such as LoRa (Long Range), which is designed for a wider communication range both in urban and extra-urban environments, hit the market [10], [11]. These meters were specifically designed for "Automatic Meter Reading" systems, in which data collection is fully automated: smart water meters periodically transmit their consumption information to gateway devices (such as Raspberry Pi 3) that gather the data from the

[1] https://www.gst4water.it

in-range smart meters and re-transmit it to the utility management typically using mobile communications (3G/LTE/4G). In both cases, the consumption data are forwarded to a Cloud platform, where they are available for processing by intelligent tasks of data analysis to train the machine learning model. The entire workflow is shown in Figure 1. Note that, given the rather small size of the trained models, they can be stored both at the level of the smart meter or at the cloud level. As studies concerning the *identification of leakages at user level* are still limited, an intelligent and possibly automated analysis of the collected consumption data in this field turns out to be of fundamental importance to pursue the goal of avoiding waste of water and of a more sustainable management of the water distribution network.

III. RELATED WORK

Several solutions have been proposed to identify household water leakages, by relying on very different techniques.

Approaches based on smart meters are the ones by [12], [13], who identify water leaks inside buildings through simple algorithms implemented within smart meter processing units. Nevertheless, these algorithms require specified input variables such as the maximum hourly flow rate and the minimum number of hours during which a flow rate of a certain entity may occur. Clearly, this aspect implies that leakages can only be identified with low Accuracy due to the high dependence of consumption on various factors such as the characteristics of the household, the habits and needs of users, and the characteristics of buildings (i.e., presence of irrigation). Therefore, before setting up each smart meter with *ad hoc* variables, it would be necessary to perform an accurate analysis of each user's typical consumption and of all the variables affecting water consumption, not necessarily known by managing authorities. Furthermore, these algorithms may fail to identify very small leaks, which is the goal of our work, usually ignored also by users as not visually detectable.

However, smart meters are not the only solution that can identify water leakages. In fact, some software exists such as Trace Wizard® [14], or those proposed by [15] and [16], designed to disaggregate water consumption into end-uses, including also household water leakages. However, as indicated by [8] and [16], the use of these approaches can be limited by several aspects. In fact, firstly, they require high temporal resolution data, in the order of the second (while we employ a 5-minute resolution). Furthermore, as indicated by [8], Trace Wizard®, prior to its application, requires a calibration step for the design of specific models that consider all factors affecting consumption of drinking water. Consequently, the water consumption information must be collected through audits and diaries, which obviously need the direct involvement of users. Once collected, data are passed to a decision tree algorithm, which applies a set of *if-then-else* rules by checking boundary conditions. The values considered by these conditions depend on the investigation made by the analyst, whose subjective interpretation can strongly condition the quality of the final output. On the other hand, the universal usability and compatibility of the tool by [15] is limited by the fact that the algorithms were trained with data originated from a specific water meter/data logger combination and the data set employed for the training of the proposed machine learning tool was obtained using Trace Wizard® software, which has limited capabilities for disaggregating overlapped consumption events [16]. A similar approach is also proposed by [17] that combines *if-then-else* rules with k-Nearest Neighbour on data from flow sensors with information such as the position of householders (at home or not). The flow data, gathered with high frequency and collected from sensors by a gateway, is stored and analysed in the cloud to detect large leaks. However, to properly work, this approach needs to be calibrated on the specific user, who must share the position through an application installed on the personal mobile.

All the previously mentioned approaches are based on flow measurements only. Other approaches have also been proposed in the technical literature based on other kinds of measurements, such as noise measurements. For example, [18] presented a system to detect leaks in a home environment on the basis of the sound produced by the water in the pipes. The identification of the leaks is made by a ''Shazam-like'' approach: a sensor records the sounds made by the water network and sends the records to a server which contains a database of sounds. The server extracts the fingerprint of the sound and compares it with those in the database to find the closest one, returning the label of the sound. This approach clearly needs to install audio sensors in the home water system with an internet connection. The detection quality highly depends on the quality of the database in the server, which must contain recording for pipes of different size, different material, and different flow.

Finally, Machine Learning techniques were applied to water leakage detection in the water distribution system (WDS), too. For instance, in [19] a convolutional NN has been coupled with a Support Vector Machine system to detect and localize leaks in a WDS. The same problem is considered by [20], which uses a fully connected NN to localize a leak in a pipeline. They consider a single pipe, of which the inlet and outlet flow and the substance passing through are known. These data can be used to approximate the changes of pressure and flow in the pipe in presence or absence of one or more leaks, also approximating their positions. This approach is explicitly tailored for the WDS and must be built on the specific pipe.

In [21], [22] a NN is used to detect bursts in the Yorkshire Water's Keighley distribution system using time series containing measurements of flow, pressure and possibly opacity of the water to forecast the next state of the distribution system in terms of a conditional density function. The results reported in the paper show an Accuracy of 75% for leak detection.

With respect to the detection of leakages inside homes, the scenarios above mentioned tackle different issues and analyse different kinds of data. Thus, a direct comparison

between our technique and those works would not be significant. In fact, in our scenario, we collect the water consumed by each household, thanks to a sensor positioned upstream the house network. Thus, we cannot resort neither on residual flows between two sensors, nor on sound or pressure data.

IV. MACHINE LEARNING FOR DETECTION OF DIFFERENT-SIZE LEAKAGES

The collected records available in the cloud, representing household water consumptions measured at the inlet point, were labelled:

1) firstly, with a class variable indicating the absence (label equal to 0) or presence (label equal to 1) of a leak;
2) secondly, with a class variable indicating the magnitude of the leak: small (<=1 L/h), medium-sized (1 to 10 L/h), large (>=10 L/h); we Recall that small leaks are mainly due to fitting dripping, medium-sized leaks to faulty sanitary appliances, large leaks to damage to pipes.

Pre-processing is detailed in Section V-A. The two datasets thus obtained are suitable for the application of supervised machine learning algorithms, with the goal of learning models able to detect whether new water consumption records will contain a leak and of which magnitude. In the first case, a binary classifier can be learnt that discriminates between the negative class (label 0) and the positive class (label 1); in the second case a multiclass classifier can be built so that it is able to find, in particular, *small leakages at domestic level*. Among the classification techniques, we considered two well-known and extensively used approaches: (1) Random Forest (RF), known for its good computational performance and scalability, and (2) Neural Networks (NN), which have proven to be very effective at modelling correlations among many features. Random forest was trained on a Linux machine equipped with an Intel(R) Core(TM) i7-8565U CPU @ 1.80(1.99)GHz and 16 GB of RAM, while neural networks on a Linux machine with 1 IBM POWER9 AC922 @ 2.6(3.1) GHz and 16 GB of RAM.

A. RANDOM FOREST

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive Accuracy and control overfitting [23]. We used the implementation available in the WEKA[2] workbench (version 3.8.5) for machine learning [24]. Given a training set X, learning a random forest in WEKA involves the following steps [25]: (1) Bootstrap samples B_i for every tree t_i are drawn by randomly selecting pairs of points with replacement from X until the sizes of B_i and X are equal; (2) a random subset of features (attributes) are selected for each B_i and used for the training of tree t_i in the forest; (3) an information gain metric is used to grow unpruned decision trees; and (4) the final classification result is the most popular of the individual tree predictions.

The training phase was controlled by the following settings:

- *P 100*: size of each bag, as a percentage of the training set size; the default value of 100 was kept;
- *I 100*: number of iterations (i.e., the number of trees in the random forest);
- num-slots 1: number of execution slots (thread) for constructing the ensemble. The default 1 means no parallelism;
- *K 0*: sets the number of randomly chosen attributes;
- *M 1*: the minimum number of instances per leaf;
- *V 0.001*: minimum numeric class variance proportion of train variance for split (it was kept the default value);
- *S 1*: seed for random number generator (it was kept the default value).

This combination of values achieved the best results in terms of classification.

B. NEURAL NETWORKS

In the last decades, the use of Neural Networks has become one of the most effective approaches to solve *classification* tasks. This is principally due to their capability to identify and model complex interactions among the entities to be classified. The first architectures proposed were called Fully Connected (Deep) Neural Networks, Artificial Neural Networks, or Multilayer Perceptron [26], [27], but have been later extended in order to define more complex models, that may be completely different.

Among them, Convolutional Neural Networks (CNNs), first theorized by [28] who applied them to define his LeNet5 [29], have come to the fore to solve problems where data present spatial and/or topological structure. Hence, they are specialized for processing data that have a known, grid-like topology, such as time series [30] and images [31]. CNNs are built using convolutional layers, performing convolutions on the input data, and extracting features from it, typically followed by some fully connected layers to carry out the classification by considering only the extracted features instead of the entire input data. The convolution operation considers the input and a kernel, combining them to compute a feature map, or simply feature. In practical scenarios the input and the kernel functions are tensors, i.e., multidimensional arrays, such as a 2D grid of pixels of a grayscale image or a 1D vector of a time series. The result of a convolution operation represents the features extracted from the input. In CNNs, convolution operations are often followed by a second operation, called pooling. This is used to reduce a feature map by summarizing its information. This summarization can be performed by, e.g., averaging neighbour values or extracting from them the maximum value. The use of convolution and pooling operations allows the extraction of the important features contained in the input data x, representing them by smaller tensors. These features can be used to help

[2]https://www.cs.waikato.ac.nz/ml/weka/

classification, or the recognition of certain patterns contained in the data, irrespectively of their position or scale. Indeed, CNNs' final layers are classical fully connected layers that perform classification on the features extracted by the previous layers.

In our case, input data are sequences of real values representing the households' daily water consumption. When classification is binary, usually one of the two classes is considered as the positive class while the other as the negative one. In this case, the final layer of the neural network has a single output neuron, which adopts as activation function the sigmoid function $\sigma(z) = \frac{1}{1+e^{-z}}$.

This function converts z into a value in [0, 1]: in this case, this value is interpreted as the probability $p(y = 1|x; \theta)$, where $y = \sigma(z)$, x is the input and θ the weights of the NN used to compute z; this is equivalent to the probability of the input belonging to the positive class 1.

This approach is known as logistic regression [26]. When classification is multiclass (with C classes), the last layer of the network used for binary classification is replaced with one having C neurons, one for each class.

A successful approach to reducing the variance of neural network models is to train multiple networks instead of a single one, and to combine either the predictions or the weights from these models. This is called ''ensemble learning'' and not only reduces the variance of predictions, but it can also result in predictions that are better than any single model.

1) SELECTION OF THE NETWORK TOPOLOGY

The performance of neural networks critically depends on identifying a proper architecture and good values for the hyper-parameters of the network. In order to find the best network, different architectures were tested on the data set for binary classification by varying several hyper-parameters. TensorFlow framework version 2.0.0 with CUDA version 10.0 was used. TensorFlow is an open-source machine learning platform written in Python, originally developed by Google Brain, and later made publicly available under Apache License 2.0 at the end of 2015 [32].

The following network topologies were evaluated:

- Fully Connected Networks (FCN), by varying the number of layers, of neurons per layer and the Dropout probability;
- Convolutional Neural Networks, by varying the number of convolutional layers, the number of feature maps and the kernel size per convolutional layer. In this case kernels are mono-dimensional, so given n, kernel size is $n \times 1$. After the convolutional layers, Global Average Pooling (GAP) followed by Dropout with varying probability was added. Finally, a varying number of FC layers, a different number of neurons per layer and a variable Dropout probability were tested;
- Recurrent Neural Networks (RNN), usually used to analyse sequences such as written sentences, by varying the number, the output size, and the direction (unidirectional or bidirectional) of Long-Short Term Memory (LSTM) layers [33]. Every LSTM layer used the tanh activation function, and Dropout with a varying probability. For the final FC layers, a varying number of layers, neurons and Dropout probability value was tested;
- Hybrid Convolutional-Recurrent Neural Networks (CNN-RNN), characterized by a set of convolutional layers, a sequence of LSTM layers, and final FC layers for classification. All hyper-parameters cited above were varied and tested.

The different networks were constructed with randomly selected hyper-parameters, a common approach in which values in a given range are randomly chosen in order to cover most of the search space. The hyper-parameters ranges are shown in Table 1. In particular, when Dropout probability is set to 0.0 it means that Dropout is not applied.

TABLE 1. Hyper-Parameter ranges (in square brackets) taken into account during random search of the best neural network architecture. Curly brackets are used for sets of values.

Hyper-parameter	Range
Number of FC Layers	[4, 10]
Number of Neurons per FC Layer	[8, 300]
Dropout Probability	[0.0, 0.8]
Number of Convolutional Layers	{3, 6, 9}
Number of Feature Maps	[5, 200]
Kernel Size	[10, 100]
Number of Unidirectional LSTM Layers	[0, 10]
Number of Bidirectional LSTM Layers	[0, 10]
LSTM Layer Output Size	[10, 50]

The number of convolutional layers was set to only three possible values, i.e., {3, 6, 9}, as each set of three layers contains the first and the third layer with a varying kernel size, and the second one with a kernel size 1×1 and the number of feature maps set to 10. Convolutional layers with kernel size 1×1 were introduced by [34] for the Google Inception network, with the purposes of reducing the number of filters, aggregating them pixel by pixel, and applying ReLU on the aggregated filters, hence speeding up training and making it more stable. We noticed that the removal of the layers with 1×1 convolution or the addition of further convolutional layers degraded the performance: in fact, during the training phase, these networks entered an overfitting regime from the first epochs. Kernel size and number of feature maps were randomly selected so that the first one decreased layer after layer while the second one increased: to do so, after each

TABLE 2. Best results achieved for each type of neural network architecture in terms of precision (Pr), recall (R) and F-Measure (F). The hyper-parameters of the best network for each architecture are abbreviated as: #CL = number of convolutional layers, KS = kernel size in order of layer, #FM = number of feature maps, #RL = number of LSTM unidirectional and bidirectional layers, #ROS = LSTM output size, #FCL = number of fully connected layers, #NS = number of neurons per layer, DP = dropout probability in bold the best results.

	Hyper-parameters of the best network								Performance metrics		
Architecture	#CL	KS	#FM	#RL	#ROS	#FCL	#NS	DP	Pr	R	F
FC		-	-	-	-	6	8	0.5	0.791	0.779	0.785
CNN	6	47,1,24,20,1,9	22,10,50,61,10,155	-	-	4	43	0.2	**0.874**	**0.805**	**0.838**
RNN	-	-	-	3,1	20	4	64	0.6	0.593	0.757	0.665
CNN-RNN	6	71,1,29,20,1,18	31,10,35,46,10,180	0,2	11	3	20	0.6	0.593	0.757	0.665

random selection, the hyper-parameter range was changed accordingly.

For each architecture, 10 neural networks were generated by randomly choosing the hyper-parameter values according to a normal probability distribution. The number of epochs was set to 1000, but thanks to early stopping this value was never reached. For each of the four architectures, the best network, with the highest values of Precision, Recall and F-Measure (see also section C. Performance Analysis), is reported in Table 2: it is easy to see that the best performing neural network is the convolutional one.

2) TRAINING OF THE CONVOLUTIONAL NEURAL NETWORK

Once chosen the topology of the NN, i.e., the convolutional one, a second random search on the best CNN was performed. The search used the values of the hyper-parameters shown in Table 2 for the best CNN (#CL, KS, #FM, #FCL, #NS, DP) as central values of new ranges to be used for random selection, 10% wide with respect to the original ranges shown in Table 1. Figure 2 shows the proposed architecture as a result of the 2-step hyper-parameter search mechanism. The detailed model layers are shown in Table 3.

Finally, the ensemble method via a 10-fold cross-validation was applied to obtain a binary or multiclass classification of the time series. The final model is obtained from the ten trained CNNs by averaging their weights by means of the Polyak-Ruppert averaging approach [35], [36].

Each of the ten CNNs was implemented with 11 layers, of which 6 convolutional, 2 of pooling and 3 fully connected. The first 8 layers perform 1D convolution and pooling to extract an increasing number of features of decreasing size. The first layer uses a kernel 50×1 to extract 10 feature maps, followed by a 1×1 convolution and a 20×1 convolution extracting respectively 10 and 50 feature maps. Then, max pooling is performed using a 3×1 grid reducing the size of the feature maps extracted so far. A second step of 20×1 and 1×1 convolution is performed, extracting respectively 70 and 10 feature maps. One last convolutional layer returns 160 features by means of a 10×1 convolution. These are passed through global average pooling returning a tensor of 160 real values, each representing a specific feature. Finally, three fully connected layers classify the examples using the extracted features: the first two layers are composed of 30 neurons, while the third one is the output layer with one or four neurons according to the type of classification. The ReLU activation function [37] is used on all the layers except the final one, where the sigmoid function is used to return the predicted class. The final network has a total of 103,432 weights to train.

Dropout was used after each fully connected layer and after the global average pooling operation. Training was done by the Adam optimizer [38] and guided by both Precision and Recall.

Early stopping was used with 15 epochs of patience, i.e., the number of epochs that produced a model with no improvement in the loss, after which training was interrupted. Moreover, the learning rate, with a starting value of 0.01, was reduced by a factor 0.1 every time the model did not improve for 5 consecutive epochs, i.e., every time the model encountered a plateau in the loss function. This, called learning rate decay, was done to fine-tune the weights and avoid at the same time to escape the plateau and climb back the loss function.

As for multiclass classification, the architecture used is the same except for the last layer of the network, which was modified by introducing 4 neurons (as shown in Figure 2), one for each class, and replacing the sigmoid function with the softmax function, which acts similarly to the sigmoid for multiclass classification.

C. PERFORMANCE ANALYSIS

The performance evaluation of an ensemble model is based on several metrics derivable from the confusion matrix, that can be built for any kind of classification; Table 4 shows its structure for a binary classification problem.

By comparing the predicted class, i.e., the classification returned by the model on the test set, and the relative actual class, one can quantify the water consumptions with actual presence of leaks (true positives or TP), those with

actual absence of leaks (true negatives or TN), those with non-existent leaks incorrectly found (false positives or FP) and, finally, those with actual leaks not found (false negatives or FN).

TP, TN, FP, FN are used to compute the following metrics, whose range is in [0, 1]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Recall = \frac{TP}{TP + FN} \quad Precision = \frac{TP}{TP + FP}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$Negative\ predictive\ val = \frac{TN}{TN + FN}$$

$$F\text{-}Measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

In particular, Accuracy represents the rate of records correctly classified by the model. However, when the dataset is unbalanced, one should prefer other metrics, i.e., Precision and Recall. The former quantifies the model's ability to avoid false positives, the latter the ability of avoiding false negatives. Finally, F-measure provides an overall indication of the balance between Precision and Recall: it is computed as their harmonic mean, which ensures that the result is close to the lowest value. Overall, the more all metrics take on values close to 1, the more effective the model is in performing classification.

Other metrics employed for performance evaluation are the areas under the Precision-Recall (PR) and the Receiver Operating Characteristics (ROC) curves [39], [40]. The PR curve plots the Precision against the Recall and is a useful measure of success of prediction when the classes are very imbalanced; it shows the trade-off between Precision and Recall for different thresholds (from 0 to 1). Analogously, the ROC curve plots the Recall against the FP Rate (FP divided by the total number of negatives) to graphically illustrate the performance of a classifier as its prediction threshold varies. The "Area Under the Curve" (AUC) measures the two-dimensional area underneath the entire ROC or PR curve from (0, 0) to (1, 1), and ranges in value from 0 (predictions 100% wrong) to 1 (predictions 100% correct).

TABLE 3. Layer parameters of the proposed CNN architecture.

Blocks	Layer (type & filter size)	# Feature maps	Activation function	Dropout
Input	Input Layer (Time series)	-	-	-
Feature learning	Conv-1 (50 × 1) Padding: same	10	ReLU	-
	Conv-2 (1 × 1)	10	ReLU	-
	Conv-3 (20 × 1)	50	ReLU	-
	Max Pooling (3 × 1)	50	-	-
	Conv-4 (20 × 1)	70	ReLU	-
	Conv-5 (1 × 1)	10	ReLU	-
	Conv-6 (10 × 1)	160	ReLU	-
Classification	Global Average Pooling	-	-	0.5
	FC-1	-	ReLU	0.2
	FC-2	-	ReLU	0.2
Output	FC-3	-	-	-

TABLE 4. Confusion matrix for a binary classification problem.

		Actual Class	
		Presence of water leak	Absence of water leak
Predicted class	Presence of water leak	TP	FP
	Absence of water leak	FN	TN

In the case of multiclass classification, Precision, Recall and F-measure must be computed separately for each class and then averaged. There are two possible approaches: Micro-average and Macro-average. The former computes the metrics for each class using "one-vs-all" methodology: given a class - considered as the positive one - it reduces the computation to a binary classification where all the other classes are considered as negative. The latter computes the metrics for each class considering each class separately. In both cases, once computed the metrics values for each class, these are averaged to obtain the final value of the metrics [41]. Note that Macro-average does not take into account class imbalance, so Micro-average is more suitable when considering unbalanced data sets.

V. EXPERIMENTAL EVALUATION

Our case study relates to the water supply system of Gorino Ferrarese, a village located in the province of Ferrara (northern Italy) covering an area of about 3 km^2. This water distribution network, managed by CADF S.p.A., supplies 293 users of which 276 are residential and 17 are attributable to public services and commercial or tourist activities. In June 2016, the water managing authority carried out a

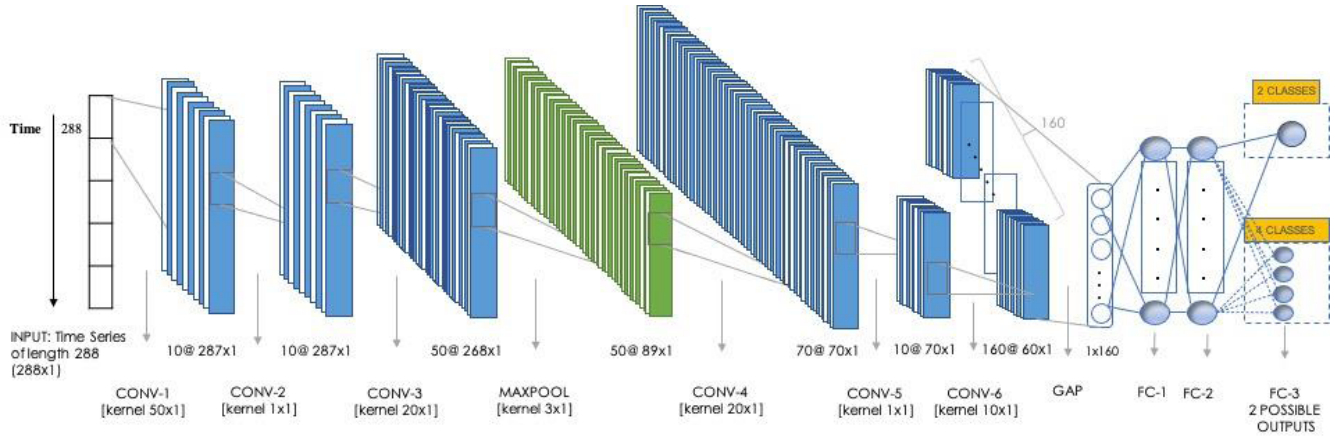


FIGURE 2. Proposed CNN architecture.

replacement campaign of traditional mechanical counters with electromagnetic smart meters (Sensus iPerl). This type of smart meter supplies a R800 metrology that ensures an accurate measurement starting with variations of the volume required by the user of 1 L in a specified time interval. Moreover, these smart meters can identify water leaks either when the flow rate is at least 25 L/h for at least 6 hours or when the flow rate is at least 3750 L/h for at least 4 hours.

Thanks to the ability of the new smart meters to also perform the Data Logging function, it was possible to undertake data collection campaigns using a RMR system in walk-by mode. In particular, time series of the cumulative volumes of water consumption by each user were collected from June 29th, 2016, to January 7th, 2017, at 5-minute time step and from January 8th, 2017, to January 9th, 2018, at 1 hour time step. In this study was made use of the time series collected at the 5-minute time step, for a total of more than 15,000,000 records.

The dataset is available as a comma-separated value file (CSV) of 32, 5 MBs at https://github.com/rzese/Intra-Domestic-Water-Leaks-Dataset.git.

A. DATA PRE-PROCESSING

The database was subjected to a preliminary cleaning step that removed all data associated with commercial users (Figure 3a), as these are characterized by a different consumption behavior than domestic users (Figure 3b) and are highly dependent on the type of commercial activity. In addition, the consumption time series of some residential users were excluded because they were characterized by negative consumption values, due to false logging caused by a set of faulty meters provided by the producer.

The resulting database contains the 5-minute time series of the cumulative volume of 211 residential users, for a total of 11,667,456 records. Subsequently, the 5-minute time series of cumulative volumes were converted into 5-minute time series of flow rates for each user.

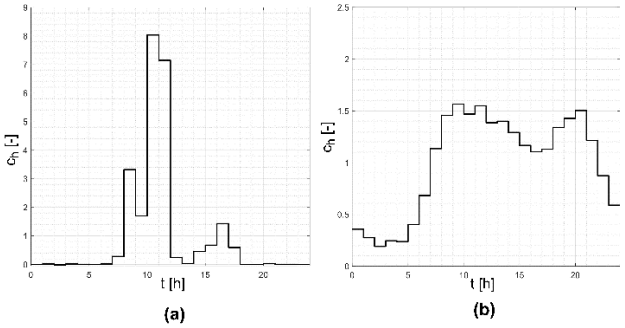


FIGURE 3. Pattern of hourly consumption coefficients: (a) commercial user (aquaculture cooperative) and (b) residential user.

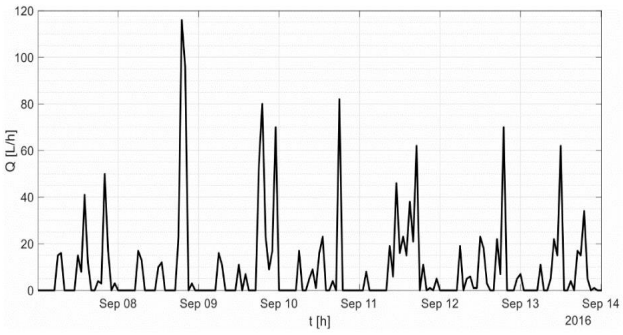


FIGURE 4. Example of the hourly water consumption pattern of a domestic user.

Both types of time series were examined by an expert with the goal to characterise the consumption behaviour of each individual user. In general, most residential users show a standard consumption behaviour, in which larger consumptions, mainly occurring in the daytime, alternate with lower (or even zero) ones, especially during the night (Figure 4 shows this variation during a week).

Unusual consumption behaviour was spotted when residential users were affected by the presence of water leaks inside their homes. Regardless of the cause, while large water

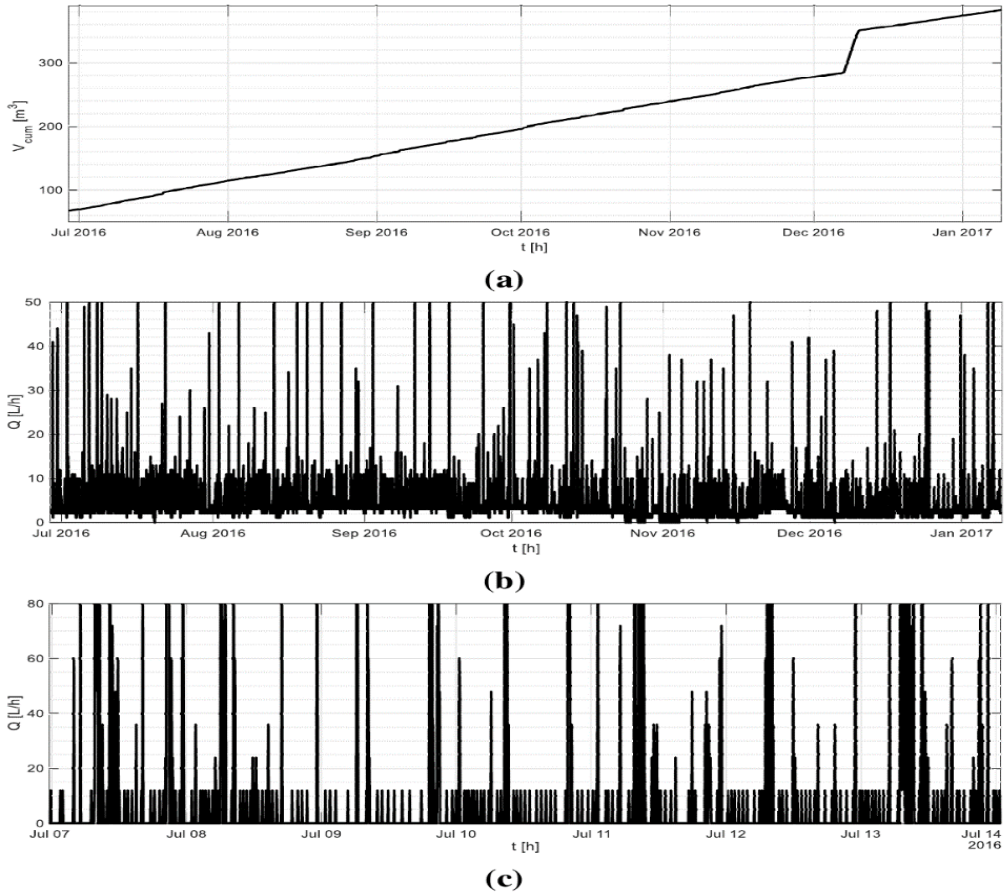


FIGURE 5. Time series of: (a) the hourly cumulative volume in case of a large leak, (b) the hourly flow rate in case of medium leak, and (c) the 5-minute flow rate in case of a small leak.

leaks can be easily identified by analysing the cumulative volume time series (Figure 5a), characterized by an abnormal step in the cumulative time series of the consumed water volume, smaller leaks require the observation of the flow rate trend. In more detail, medium-sized leaks were identified by analysing the hourly flow rate trend (Figure 5b), while the small ones were detected by examining the 5-minute time series of the flow rate (Figure 5c), as they involve a ''sawtooth behaviour'' where null values of the flow rate alternate with values of 12 L/h (i.e., variations of 1 L in a 5-minute time interval).

Overall, this manual analysis spotted 211 water leaks, most of them lasting more than one day: among them, 26 were large (>10 L/h), 101 were in the 1 - 10 L/h range (medium-sized), and 84 were small (<=1 L/h). The 5-minute time series of flow rates, relative to each user, were divided into records of 24 hours and each record was labelled:

1) with a two-value class variable to indicate the presence or absence of water leaks, for the binary classification task;
2) with a four-value class variable to indicate the presence of small, medium-sized, large leaks or no leak, for the multiclass classification task.

Each record representing the flow rates for a specific day (24 hours) and a specific user is a row in the data set and is referred to as an ''example'' in the following. The resulting data set for case (1) counted 40,428 examples, among which 30,744 were labelled as negative, i.e., absence of a leak, while 9,684 were labelled as positive, i.e., presence of a leak. The number of positive examples may appear in contrast with the number of water leaks spotted (211). However, as reported above, among the 211 leaks detected, most of them lasted more than 24 hours, therefore, in these cases, a single leak is used to label a set of positive examples. The resulting data set for case (2) counted 30,744 negatives and the positives divided into 3565 small leaks, 5511 medium-size leaks, 608 large leaks. As you can notice, the data sets are heavily unbalanced, with a positive/negative ratio of 0.31: almost 24% of the examples are positive, while 76% are negative. Both data sets were split in two parts, a training set containing 80% of the examples and a test set containing the remaining 20%; the training-test split was stratified in both cases. Therefore, each example in the data sets is a labelled time series of 288 values, equivalent to a 1D tensor corresponding to the water flow rates collected every 5 minutes in 24-hour windows from 06/29/2016 to 01/07/2017.

B. RANDOM FOREST RESULTS

Accuracy, Precision, Recall, F-Measure, AUC-PR, and AUC-ROC averaged over the 2 classes, returned by the model on the test set for binary classification, are shown in the overview Table 5 together with the results obtained by the convolutional neural networks, for the sake of comparison. Analogously, in Table 6 we report Accuracy, Precision, Recall, F-Measure, AUC-PR, and AUC-ROC averaged over the 4 classes, returned by the model on the test set for multiclass classification. Table 7 shows the confusion matrix obtained in this case.

Values closer to 1 for all metrics mean better performance.

In the binary case, the RF model (38Mb) was built in 96.90s while the time for testing the model on the test set was 1.60s.

In the multiclass case, the RF model (57Mb) was built in 98.75s while the time for testing the model on the test set was 1.74s.

C. NEURAL NETWORKS RESULTS

In Table 5 we report Accuracy, Precision, Recall, F-Measure, AUC-PR, and AUC-ROC achieved on the test set by the ensemble model built for binary classification.

In Table 6 we report the same metrics (micro-averaged) achieved on the test set by the ensemble model built for multiclass classification. Values closer to 1 for all metrics mean better performance. Table 8 reports the confusion matrix obtained in this case.

In the binary case, each network of the ensemble was trained in about 10 minutes on average and inference time, i.e., time for classifying an example, was about 7 ms.

In the multiclass case, each network of the ensemble was trained in 22 minutes on average, while inference time took about 0.372 ms.

The CNN models occupy a maximum of 2Mb.

TABLE 5. Results obtained by random forest (RF) and neural networks (CNN) in terms of accuracy (ACC), precision (PR), recall (R), F-Measure (F), AUC-PR and AUC-ROC on the test set used for binary classification in bold the best results.

	ACC	Pr	R	F	AUC-PR	AUC-ROC
RF	0.917	0.919	0.917	0.912	0.967	0.960
CNN	**0.957**	**0.925**	0.890	0.907	**0.986**	**0.988**

TABLE 6. Results obtained by random forest (RF) and neural networks (CNN) in terms of accuracy (ACC), precision (PR), recall (R), F-Measure (F), AUC-PR and AUC-ROC on the test set used for multiclass classification in bold the best results.

	ACC	Pr	R	F	AUC-PR	AUC-ROC
RF	0.879	0.857	0.879	0.843	0.913	0.950
CNN	**0.964**	**0.930**	**0.924**	**0.927**	**0.975**	**0.990**

TABLE 7. Confusion matrix for the multiclass classification task obtained with random forest. Along the diagonal (in bold) the number of examples correctly classified.

	Actual class			
Predicted class	No water leak	Small leak	Medium-sized leak	Large leak
No water leak	**6139**	12	37	1
Small leak	570	**19**	98	1
Medium-sized leak	225	0	**854**	18
Large leak	0	0	13	**99**

TABLE 8. Confusion matrix for the multiclass classification task obtained with the CNN. Along the diagonal (in bold) the number of examples correctly classified.

	Actual class			
Predicted class	No water leak	Small leak	Medium-sized leak	Large leak
No water leak	**6055**	104	28	2
Small leak	230	**351**	106	1
Medium-sized leak	12	64	**985**	36
Large leak	1	0	13	**98**

D. THE "EMPIRICAL ALGORITHM"

In [9] an "empirical algorithm" was proposed to automatically identify leakages at the individual user level by looking for non-consumption in certain periods of the day. Specifically, the algorithm was originally developed to be applied to hourly time series and classifies the consumption pattern as a correct operation of the plumbing systems and the hygienic and sanitary appliances if the hourly flow rate is equal to 0 at least once during the whole day. By contrast, it classifies as a probable presence of water leakage if the hourly flow rate is never zero during the whole day, so it behaves as a binary classifier. In this paper, in order to be compared with the ML approaches, the algorithm could be applied only on the dataset for binary classification.

The algorithm performance, shown in Table 9, was estimated by considering both the 5-minute time series of flow rates as recorded by the smart meters (first column), and by uniformly "spreading" aggregated information, such as the hourly volume (second column, as done in the original publication) and the bi-hourly volume (third column) required by the generic user, on 5-minute time intervals.

The latter method was applied in order to improve the ability of identifying small water leaks, which represents the main goal of the paper. By constructing time series of hourly averages and by turning these into 5-minute time series, we could provide a constant flow rate in a generic hourly or bi-hourly time interval, avoiding giving as input a 5-minute time series characterized by the sawtooth behaviour.

TABLE 9. Results in terms of accuracy, precision, recall, and F-Measure of the "empirical algorithm" on the test set for the binary classification task.

	5-min time series		
	as recorded by smart meters	spreading the hourly volume	spreading the bi-hourly volume
Accuracy	0.768	0.916	0.898
Recall	0.038	0.656	0.847
Precision	0.991	0.956	0.759
F-Measure	0.072	0.781	0.799

VI. DISCUSSION

The experimental evaluation presented in the previous Section aimed at testing different ML algorithms with the purpose of finding the best classifier both for binary leak detection (presence vs absence of a leak) and for multiclass leak detection: in the latter case, among the four classes of leaks considered, we were particularly interested in the ones under the meter sensitivity (that we called *small* for being lower than 1L/h), which are the most difficult to automatically detect.

Comparison between RF, CNNs and the empirical algorithm on the *binary classification task* highlights that convolutional neural networks better discriminate between presence and absence of water leakages inside homes, with a 95.7% Accuracy, a 92.5% Precision and areas under the PR and ROC curves near to 99%; Recall and F-measure are comparable for RF and CNN. RF comes second, with all performance metrics that are always above 91% (see Table 5). Finally, as Table 9 shows, the "empirical algorithm" of [9], applied to the 5-minute time series recorded by smart meters, is pretty inaccurate, since Recall and F-measure are close to 0, while Accuracy is close to the lower bound (0.76) obtainable with the data set used for binary classification. This is due to a large number of actual leakages not identified by the algorithm, occurring mainly when residential users are affected by leakages lower than 1 L/h. These are the hardest to be detected because they fall below the meter's sensitivity. In fact, the 5-minute time series are mainly characterized by a sawtooth behaviour where null values irregularly alternate with sporadic flow rate values of 12 L/h (equivalent to the volume of 1 L in a 5-minute interval), as shown in Figure 5c. This leads the empirical algorithm to confuse the trend of the time series with the user's ordinary consumption, where non-zero flow rates in discontinuous and typically brief time intervals alternate with long time periods of null consumption. In other terms, the sawtooth behaviour is not related to the user's water demand but is caused by the fact that the meter records a change in the cumulative volume after detecting the transition of a certain volume of water (in the case of Sensus iPerl equal to 1 L) in a fixed time interval.

Instead, the "empirical algorithm" is precise in classifying absence of leakages as true negatives (high Precision). This is due to the fact that during each generic 5-minute time interval a null flow rate is recorded. The performances of the algorithm increase if aggregated data are used as input and are more in line with - even though slightly worse than - those shown in [9], but it is worth Recalling that the data set considered in this paper is slightly different than in [9] where small leakages, lower than 1 L/hour, had not been considered. Considering the hourly volume required by the user spread on 5-minute intervals, the number of actual leakages not identified by the algorithm decreases, as shown by Recall and F-measure, as well as Accuracy which increases by about 20%. Considering the bi-hourly volume spread on 5-minute intervals, it is possible to observe a further improvement. However, in this case, a reduction in Accuracy is observed, due to an increase in the number of false positives.

As for the *multiclass classification task*, Table 6 demonstrates that CNNs overcome RF in distinguishing different-magnitude leakages under all performance metrics: RF ranges between 84.3% and 95%, while CNNs range between 92.4% and 99%. Also, the confusion matrices obtained by the two ML techniques in the multiclass classification task highlight (Tables 7 and 8) how RF poorly detects small leakages w.r.t. CNN, by confusing a lot of records without leak (570) or with medium-sized leaks (98) as small leaks. However, the classification of small leakages remains the hardest task for neural networks, too: Table 8 shows that the greatest difficulty is discriminating between absence of leak and small leaks. We also computed the Specificity and the Negative predictive value to measure the ability of the RF and NN models to reliably recognize non-leakages, in order to avoid false alarm rate of the leak detection algorithm. Also, in this case CNNs perform better with Specificity and Negative predictive value near 84%, while RF comes still second with Specificity and Negative predictive value near to 70%.

Firstly, it is worth noting that, although household water leakages can vary in size from tens of litres to less than a litre, the detection of the latter still may have a significant economic and environmental impact due to the very nature of the small leaks that, difficult in being discovered, can persist over time and represent a substantial amount of the final user's residential loss. Therefore, small leaks may have an impact in terms of environmental sustainability as well as larger ones. Our approach is specifically targeted to automatically detect leaks of different size, especially lower than the meter sensitivity.

Secondly, smart water meters are being increasingly deployed in aqueducts [6], [8] for real-time monitoring of

the flow rate in order to send warnings about eventual losses. The proposed approach can be extremely energy efficient for such battery powered devices, as neural network models are very small (a few Megabytes) and can be easily stored in them, directly allowing for small leakages control in the urban context as well.

VII. CONCLUSION

In this work we presented the application of several Machine Learning techniques to water consumption time series collected through an IoT infrastructure at user level in Gorino Ferrarese (Ferrara, Italy) based on smart meters, for the automatic detection of different-size water leaks within the households. The approach is specifically targeted to automatically detect leaks of different size, especially lower than the meter sensitivity. Convolutional Neural Network models demonstrate to be the best in detecting both the presence or absence of water leaks, but also in discriminating their class among small, medium-sized, and large. Both the models, given the daily water consumption, return whether there is a leak and, in this case, if the leak is small, medium, or large if multiclass classification is performed. Both the models also return a numeric value representing the probability of the prediction, the highest the probability the most accurate is the detection. Training and test phases are fast, and results show that the model is able to classify water leaks with Accuracy, Precision, Recall, F-measure, and area under the PR and ROC curves taking values ranging from 92% to 99%.

REFERENCES

[1] Y. Huang, F. Zheng, Z. Kapelan, D. Savic, H. Duan, and Q. Zhang, "Efficient leak localization in water distribution systems using multistage optimal valve operations and smart demand metering," *Water Resour. Res.*, vol. 56, no. 10, Oct. 2020, Art. no. e2020WR028285.

[2] M. Farley, G. Wyeth, Z. B. M. Ghazali, A. Istandar, S. Singh, N. Dijk, V. Raksakulthai, and E. Kirkwood, "A guide to understanding water losses," in *The Manager's Non-Revenue Water Handbook*. USA: United States Agency for International Development, 2008, pp. 1–110.

[3] A. Athuraliya and P. E. A. Roberts Brown, "Yarra valley future water-residential water use study," in *Summer*, vol. 2. Mitcham, VIC, USA: Yarra Valley Water Victoria, 2012.

[4] M. Heinrich, "Water use in Auckland households. Auckland water use study (AWUS) final report," Watercare Services, BRANZ Ltd, Judgeford, New Zealand, Tech. Rep. EC1356, 2008.

[5] J. Lloret, J. Tomas, A. Canovas, and L. Parra, "An integrated IoT architecture for smart metering," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 50–57, Dec. 2016.

[6] T. Boyle, D. Giurco, P. Mukheibir, A. Liu, C. Moy, and S. E. R. W. Stewart, "Intelligent metering for urban water: A review," *Water*, vol. 5, no. 3, pp. 1052–1081, 2013.

[7] M. E. Shafiee, A. Rasekh, L. Sela, and A. Preis, "Streaming smart meter data integration to enable dynamic demand assignment for real-time hydraulic simulation," *J. Water Resour. Planning Manage.*, vol. 146, no. 6, Jun. 2020, Art. no. 06020008.

[8] A. Cominola, M. Giuliani, D. Piga, A. Castelletti, and A. E. Rizzoli, "Benefits and challenges of using smart meters for advancing residential water demand modeling and management: A review," *Environ. Model. Softw.*, vol. 72, pp. 198–214, Oct. 2015.

[9] C. Luciani, F. Casellato, S. Alvisi, and M. Franchini, "Green smart technology for water (GST4Water): Water loss identification at user level by using smart metering systems," *Water*, vol. 11, no. 3, p. 405, Feb. 2019.

[10] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A study of LoRa: Long range & low power networks for the Internet of Things," *Sensors*, vol. 16, no. 9, p. 1466, 2016.

[11] R. Sanchez-Iborra, J. Sanchez-Gomez, J. Ballesta-Viñas, M. D. Cano, and A. F. Skarmeta, "Performance evaluation of LoRa considering scenario conditions," *Sensors*, vol. 18, no. 3, p. 772, 2018.

[12] T. Britton, G. Cole, R. Stewart, and D. Wiskar, "Smart metering as a tool for the remote diagnosis of leakage in residential households," *Water*, vol. 35, pp. 56–60, Sep. 2008.

[13] R. A. Stewart, R. Willis, D. Giurco, K. Panuwatwanich, and G. Capati, "Web-based knowledge management system: Linking smart metering to the future of urban water planning," *Austral. Planner*, vol. 47, no. 2, pp. 66–74, Jun. 2010.

[14] W. B. DeOreo, J. P. Heaney, and P. W. Mayer, "Flow trace analysis to access water use," *J. Amer. Water Works Assoc.*, vol. 88, no. 1, pp. 79–90, Jan. 1996.

[15] K. A. Nguyen, R. A. Stewart, and H. Zhang, "An intelligent pattern recognition model to automate the categorisation of residential water end-use events," *Environ. Model. Softw.*, vol. 47, pp. 108–127, Sep. 2013.

[16] L. Pastor-Jabaloyes, F. Arregui, and R. Cobacho, "Water end use disaggregation based on soft computing techniques," *Water*, vol. 10, no. 1, p. 46, Jan. 2018.

[17] H. Fuentes and D. Mauricio, "Smart water consumption measurement system for houses using IoT and cloud computing," *Environ. Monitor. Assessment*, vol. 192, no. 9, pp. 1–16, Sep. 2020.

[18] S. Seyoum, L. Alfonso, S. J. V. Andel, W. Koole, A. Groenewegen, and N. van de Giesen, "A shazam-like household water leakage detection method," *Proc. Eng.*, vol. 186, pp. 452–459, Jan. 2017.

[19] J. Kang, Y.-J. Park, J. Lee, S.-H. Wang, and D.-S. Eom, "Novel leakage detection by ensemble CNN-SVM and graph-based localization in water distribution systems," *IEEE Trans. Ind. Electron*, vol. 65, no. 5, pp. 4279–4289, May 2018.

[20] I. Barradas, L. E. Garza, R. Morales-Menendez, and A. Vargas-Martínez, "Leaks detection in a pipeline using artificial neural networks," in *Proc. CIARP*, 2009, pp. 637–644.

[21] S. R. Mounce, A. J. Day, A. S. Wood, A. Khan, P. D. Widdop, and J. Machell, "A neural network approach to burst detection," *Water Sci. Technol.*, vol. 45, nos. 4–5, pp. 237–246, Feb. 2002.

[22] S. R. Mounce and J. Machell, "Burst detection using hydraulic data from water distribution systems with artificial neural networks," *Urban Water J.*, vol. 3, no. 1, pp. 21–31, Mar. 2006.

[23] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[24] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd ed. San Mateo, CA, USA: Morgan Kaufmann, 2011.

[25] M. Amrehn, F. Mualla, E. Angelopoulou, S. Steidl, and A. Maier, "The random forest classifier in WEKA: Discussion and new developments for imbalanced data," 2018, *arXiv:1812.08102*. [Online]. Available: http://arxiv.org/abs/1812.08102

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[27] C. C. Aggarwal, *Neural Networks and Deep Learning—A Textbook*. Cham, Switzerland: Springer, 2018.

[28] Y. LeCun, "Generalization and network design strategies," *Connectionism Perspective*, vol. 19, pp. 143–155, Jun. 1989.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[30] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Müller, "Deep learning for time series classification: A review," *Data Mining Knowl. Discovery*, vol. 33, no. 4, pp. 917–963, Jul. 2019.

[31] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Comput.*, vol. 29, pp. 1–98, Jun. 2017.

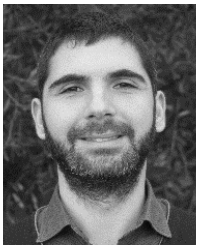
[32] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*. [Online]. Available: http://arxiv.org/abs/1603.04467

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[35] D. Ruppert, "Efficient estimators from a slowly convergent Robbins–Monro procedure," School Oper. Res. Ind. Eng., Cornell Univ., Ithaca, NY, USA, Tech. Rep. 781, 1988.

[36] B. T. Polyak, "New method of stochastic approximation type," *Automat. Rem. Contr.*, vol. 51, no. 7, pp. 937–946, 1990.
[37] R. H. Hahnloser, H. S. Seung, and J. J. Slotine, "Permitted and forbidden sets in symmetric threshold-linear networks," in *Proc. NIPS*, 2000, pp. 217–223.
[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015, pp. 1–15.
[39] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 233–240.
[40] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.
[41] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, 2009.



RICCARDO ZESE received the B.S. and M.S. degrees in computer engineering and the Ph.D. degree in computer science engineering from the University of Ferrara, Italy, in 2016.

Since 2019, he has been an Assistant Professor with the Department of Engineering, University of Ferrara, where he is a member of the Machine Learning Research Group, and has been involved in several European, national, and local research projects. He is the author of one book, and more than 60 articles. His research interests include machine learning (both neural and symbolic), artificial intelligence, probabilistic reasoning and learning, and semantic web technologies.

Dr. Zese was awarded with an honorable mention for the EurAI Distinguished Dissertation Award 2016, the Best Paper Award at the 7th edition of RR, held in Mannheim, in 2013, and several recognitions for his M.S. and Ph.D. thesis.

ELENA BELLODI received the Ph.D. degree in computer science engineering from the University of Ferrara, Italy, in 2009.

She is currently an Assistant Professor with the Department of Engineering, University of Ferrara, where she belongs to the Machine Learning Research Group. She has coauthored over 60 articles published in international venues in the machine learning research area, with particular reference to probabilistic logic programming, statistical relational learning, process mining, and semantic web.

Dr. Bellodi received several national and international recognitions for her Ph.D. thesis, the Best Paper Award at International Conference on Web Reasoning and Rule Systems (RR) 2013, and the Highly Commended Paper Award at International Conference on Knowledge Science, Engineering and Management (KSEM) 2010.

CHIARA LUCIANI received the B.S. and M.S. degrees in civil engineering from the University of Ferrara. She is currently a Research Fellow with the Department of Engineering, University of Ferrara.

STEFANO ALVISI is currently an Associate Professor with the University of Ferrara. He is an author of more than 100 articles, around 60 of these articles are in internationals (ISI) journals. He has been involved in several European, national, and local research projects. His main research interests include water demand modeling and forecasting, pipe network rehabilitation and leakage detection, and data driven techniques for real time flood forecasting models.

He is a member of the editorial boards of some international journals. He was granted with Certificates of Excellence to recognize outstanding achievements as the best paper nominees in the "Battle of Water Calibration Network (BWCN)," in 2010 and in the "Battle of Background Leakage Assessment for Water Networks (BBLAWN)," in 2014.