Clustering-Learning Approach to the Localization of Leaks in Water Distribution Networks

Luis Romero[1]; Joaquim Blesa, Ph.D., Dr.Eng.[2]; Vicenç Puig, Dr.Eng.[3]; and Gabriela Cembrano, Ph.D., Dr.Eng.[4]

Abstract: Leak detection and localization in water distribution networks (WDNs) is of great significance for water utilities. This paper proposes a leak localization method that requires hydraulic measurements and structural information of the network. It is composed by an image encoding procedure and a recursive clustering/learning approach. Image encoding is carried out using Gramian angular field (GAF) on pressure measurements to obtain images for the learning phase (for all possible leak scenarios). The recursive clustering/learning approach divides the considered region of the network into two sets of nodes using graph agglomerative clustering (GAC) and trains a deep neural network (DNN) to discern the location of each leak between the two possible clusters, using each one of them as inputs to future iterations of the process. The achieved set of DNNs is hierarchically organized to generate a classification tree. Actual measurements from a leak event occurred in a real network are used to assess the approach, comparing its performance with another state-of-the-art technique, and demonstrating the capability of the method to regulate the area of localization depending on the depth of the route through the tree.

Introduction

Nowadays water distribution networks (WDNs) are critical infrastructures in cities. Their efficient operation can reduce water losses, which are estimated to account for 30% of the total amount of distributed water (Puust et al. 2010), producing high associated operational costs, and environmental (Xu et al. 2014) and sanitary (LeChevallier et al. 2003) problems. Thus, leak detection and localization approaches are widely researched (see (Chan et al. 2018) for an extensive review). The different solutions can be classified considering several aspects. One of the most important classifications separates model-based approaches from data-driven methodologies.

Model-based techniques rely on the estimation of hydraulic dynamics using mathematical models (Savić et al. 2009). A leak localization approach using pressure sensors, proposed in Pérez et al. (2014), compares pressure disturbances caused by a leak with a fault signature matrix obtained by means of hydraulic model simulations. This sensitivity analysis is also exploited in Sophocleous et al. (2019), included into a two-phase approach that uses search-space reduction to decrease the number of decision variables and their range of values, and an optimization strategy that solves the detection and localization problem. Another solution, presented in Sanz et al. (2015), compares calibrated parameters with historical values to find changes produced by a leak.

However, whereas model-based methods can work effectively under ideal conditions, their performance is limited by the availability and accuracy of the mathematical models (Menapace et al. 2018). Hydraulic models may contain structural modeling errors, nodal demand uncertainties, and measurement noise (Blesa and Pérez 2018). Besides, the high computational cost and the uncertainty in parameter estimation hinder the application of these approaches. These drawbacks are gradually overcome by the appearance of machine learning and data-driven procedures, which are based on the mining of knowledge from the available data, gathered by installed sensor networks.

These methods typically use pressure sensors due to their lower cost in comparison to other metering devices. In Han et al. (2018), a two-phase strategy is used to estimate the complete state of the WDN from hydraulic heads at certain nodes by means of a Gauss-Newton belief propagation inference scheme and to decompose the network to locate the leak. A deep-learning (DL) framework is proposed in Zhou et al. (2019) to locate bursts using data from pressure sensors. More recently, Soldevila et al. (2020) proposed a leak localization method that interpolates the pressure at every node of the network from the measured values, comparing leak and leak-free scenarios to locate the leak and using Dempster-Shafer reasoning to deal with uncertainty.

Other methodologies deal with additional types of sensors, mostly flow meters. A data-driven method, proposed in Arifin et al. (2018), applies the concept of Kantorovich distance to detect and locate leaks from flow rates and pressure measurements. In Navarro et al. (2019), a real time leak localization method using time delay neural networks and flow/pressure measurements is presented.

[1]Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Carrer Llorens Artigas, 4-6, 08028 Barcelona, Spain (corresponding author). ORCID: https://orcid.org/0000-0002-4790-2031. Email: luis.romero.ben@upc.edu; lurobe_94@hotmail.com

[2]Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Carrer Llorens Artigas, 4-6, 08028 Barcelona, Spain; Supervision, Safety and Automatic Control Research Center (CS2AC) of the Universitat Politécnica de Catalunya, Campus de Terrassa, Gaia Bldg., Rambla Sant Nebridi, 22, 08222 Terrassa, Barcelona, Spain. ORCID: https://orcid.org/0000-0002-5626-3753. Email: joaquim.blesa@upc.edu

[3]Professor, Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Carrer Llorens Artigas, 4-6, 08028 Barcelona, Spain; Supervision, Safety and Automatic Control Research Center (CS2AC) of the Universitat Politécnica de Catalunya, Campus de Terrassa, Gaia Bldg., Rambla Sant Nebridi, 22, 08222 Terrassa, Barcelona, Spain. Email: vicenc.puig@upc.edu

[4]Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Carrer Llorens Artigas, 4-6, 08028 Barcelona, Spain. ORCID: https://orcid.org/0000-0003-1436-6022. Email: gabriela.cembrano@upc.edu

A combined artificial neural network method is presented in Pérez-Pérez et al. (2021) to perform leak diagnosis in pipes considering pressure and flow data.

Finally, other works deal with less common types of sensors. In Kang et al. (2017), data from accelerometers are used to feed a two-stage approach, based on a convolutional neural network (CNN)—support vector machine (SVM) architecture that detects leaks and a graph method based on virtual nodes for their location. Besides, Huang et al. (2020) presents an efficient multistage leak localization method that uses valve operations (VOs) to divide the demand metering area (DMA) into two regions and water balance analysis based on smart demand meters to locate the leaks.

This paper proposes a leak localization method based on the use of the network topology and pressure data from some inner nodes. The pressure information is encoded in the form of images, to exploit the power of classical image-classification DL techniques. In this way, the degree of freedom in the implementation of the method is higher, regarding the available software languages and platforms. The localization operation is performed by means of a recursive clustering/learning procedure. The structural information of the WDN is used by the clustering process to split a certain (sub) network two disjoint clusters of nodes, generating a set of binary labels. The hydraulic images, together with the derived labels, feed a training process which must produce a deep neural network (DNN) that indicates which one of the two generated clusters contains the leak. The recursivity affects the clustering stage, because these clusters are provided as inputs of the explained process for subsequent iterations. When all the generated DNNs are organized depending on the hierarchy of their associated (sub)network, a classification tree is produced.

In comparison with previous DL-related works like Javadiha et al. 2019)), a smart application of the image-encoding process allows to limit the size of the images to be fed to the DL algorithm, thus reducing the computational cost. Moreover, it is guaranteed that the set of possible candidates for the leak location is connected, due to the clustering stage dividing the (sub)network into connected components. This feature provides a secondary advantage: if a hydraulic simulator is not available to obtain the necessary pressure data for the training phase (hence using the strategy as a mixed model-based/data-driven approach), leak experiments can be performed over certain points of the real network after defining a set of subnetworks that will be the targets of the DNN classification tree.

Additionally, the hierarchical structure of this classification tree allows to tackle one of the main problems of learning-based leak localization methods: the similar effect of neighboring leaks in the hydraulic behavior of the WDN. When this similitude affects a group of nodes, it may become even impossible to discern the origin of the leak among them. The proposed recursive clustering/learning approach handles this behavior by retrieving information about the classification limitations (directly related to the leaks resemblance) from the training performance results at each level of the classification tree. Thus, the hierarchically organized DNNs of the trained tree are applied to new samples if and only if the training accuracy was high enough. In this way, an ad hoc solution for the studied WDN would be obtained, because it is adaptable to its characteristics and limitations from the leak localization point of view. This fact demonstrates the qualitative improvement at the leak localization solution that the proposed method offers in comparison with strategies that lack this flexibility and hence provide unreliable solutions if the sensors amount and/or precision are not appropriate.

Furthermore, some of the implemented techniques are adapted from their original design to efficiently and effectively tackle their specific tasks, hence implying novelty in their exploitation.

Methodology

The proposed approach requires considering a set of assumptions, that should be fulfilled to proceed with the method application:

1. Pressure sensors are installed in a set of inner nodes. Besides, topological information about the network must be available, i.e., the junctions connectivity and the pipes length.
2. Records of pressure measurements are available for all possible leak cases. Because this requirement may not be practical, these records could be obtained using a hydraulic simulator and/or performing artificial leak experiments over a set of locations of the network. However, these records are only needed at the off-line training stage.
3. Due to the previous requirement, only single-leak scenarios are considered. The analysis of multileak episodes would require data about every combination of leaks. Besides, the leaks are supposed to appear at the network nodes, as usually considered in the literature (Blesa and Pérez 2018; Sophocleous et al. 2019).
4. The leak detection process is handled by an external method so that the proposed localization technique is applied after a detection event occurs. A typical approach to detect the presence of leaks consists of tracking changes in the night consumptions (Puust et al. 2010).

The following subsections describe the details of the methodology components and then the general procedure thereof.

Hydraulic Data Encoding

The installation of pressure sensors allows to collect hydraulic information of the network state. These measurements are converted into hydraulic heads by adding the elevation of each junction, because these values are important to determine the availability of water service. The hydraulic heads are stored in data vectors (with a component for each sensor) that can be gathered at each time interval, considering the sampling time of the measuring devices.

Besides, as mentioned above, information about the widest possible range of leak scenarios is required, either from a hydraulic simulator or from artificial leak experiments at the real network. Concretely, considering the importance of the size of a leak in its effect on the WDN behavior, the strategy that must be followed to handle different leak sizes must be adapted to the hydraulic information source:

1. On the one hand, if a hydraulic simulator is available, different leak scenarios can be derived considering an estimated leak rate value, which can be obtained from an external leak detection method, because most of these approaches handle the estimation of the leak size.
2. On the other hand, proper artificial leak experiments could be performed over the real WDN, considering that water utilities are usually interested in a concrete range of leak sizes that depends on the operational characteristics of their network.

The achieved hydraulic measurements can be arranged in a multidimensional matrix $\boldsymbol{H}$ that stores the gathered data vectors: the value h_{ij}^{l} would represent the hydraulic head at time instant i of sensor j at leak scenario l, i.e., the leak is located at node l. The total number of time steps, sensors and possible leak scenarios will be referred to as M, N, and P respectively. To highlight the time dependency, the notation h_{ij}^{l} is henceforth substituted by $h_{j}^{l}(i)$. Thus, a single data vector from the multidimensional matrix $\boldsymbol{H}$ can be referred to as:

$$\boldsymbol{h}^{l}(i) = [h_{1}^{l}(i), h_{2}^{l}(i), \ldots, h_{j}^{l}(i), \ldots, h_{N}^{l}(i)] \tag{1}$$

where each vector $\boldsymbol{h}^{l}(i)$ corresponds to a collection of the hydraulic heads of the N sensors for time instant i and leak scenario l.

The purpose of the data encoding process consists of converting the dataset $\boldsymbol{H}$ composed by $M \cdot P$ vectors $\boldsymbol{h}^l(i) \in \mathbb{R}^N$ into an image set $\boldsymbol{L}$ comprised by $M \cdot P$ images $L^l(i) \in \mathbb{R}^{N \times N}$.

The application of this process, despite the increase in the required memory space due to the use of images, is justified by two main advantages:

1. In this way, the training stage can be tackled by means of standard DL techniques for image classification, and hence a wide variety of tools are available for this task.
2. This procedure is handled by the Gramian angular field (GAF) method, presented in Wang and Oates (2015) (concretely, the Gramian summation angular field approach is considered). It not only converts vectors into images, but also extracts correlations among the components of the vector, enhancing the information provided to the learning stage.

However, the GAF technique has been adjusted to fulfil the requirements of a leak localization approach: the novelty of its application lies in the selection of the vectors that are converted into images. Despite its original design for the imaging of time-series, for this work, GAF is applied to the hydraulic data vectors $\boldsymbol{h}^l(i)$. This is justified by the following key points:

1. A small number of measuring devices is available at most of the WDNs (Savić et al. 2009). The reduced length N of the hydraulic information vectors $\boldsymbol{h}^l(i)$ implies a low dimensionality of the generated images, greatly diminishing the computational cost of the learning phase. Thus, the major drawback of the original usage of GAF is removed, and the application of post-processing techniques to reduce the image size is not required.
2. The proposed selection of data vectors $\boldsymbol{h}^l(i)$ entails the training of the different DNNs with a wide variety of time instants, hence generalizing the neural network learning by inducing an independence of dynamical variables like the nodal demands.
3. Besides, considering the opposite situation, the selection of time-series as the data vectors for the imaging process (that is, the vectors $\boldsymbol{h}^l_j$ of $\boldsymbol{H}$), would imply some drawbacks:
   a. The selection of a suitable time window $T < M$ (to split the complete time-series into a set of time slots) would require analyzing the length of the smallest set of time instants whose information is rich enough to explain the leak location.
   b. The collection of T-dimensional time-series would be required to feed the leak localization algorithm. This implies the existence of a minimum necessary time lapse to achieve a single localization.

GAF exploits the angular perspective of the polar-encoded hydraulic information. Thus, each input vector is rescaled to a range of values between -1 and 1 to fit the image of the cosine function. The vector $\boldsymbol{h}^l(i)$ is processed as follows:

$$\tilde{h}^l_j(i) = \frac{2h^l_j(i) - (\max(\boldsymbol{h}^l(i)) + \min(\boldsymbol{h}^l(i)))}{\max(\boldsymbol{h}^l(i)) - \min(\boldsymbol{h}^l(i))} \tag{2}$$

$$\phi^l_j(i) = arc\cos(\tilde{h}^l_j(i)) \tag{3}$$

By considering the trigonometric sum between each pair of polar-encoded head values, the associated GAF image is generated:

$$\begin{aligned} L^l(i) &= [\cos(\phi^l_a(i) + \phi^l_b(i))] \\ &= (\tilde{\boldsymbol{h}}^l(i))' \cdot \tilde{\boldsymbol{h}}^l(i) - \left(\sqrt{\mathbf{1}_N - (\tilde{\boldsymbol{h}}^l(i))^2}\right)' \cdot \sqrt{\mathbf{1}_N - (\tilde{\boldsymbol{h}}^l(i))^2} \end{aligned} \tag{4}$$

where a and b are possible values of j, and $\mathbf{1}_N$ is the unit row vector of length N.

The resulting image $L^l(i)$ is a $N \times N$ matrix whose $a-b$ component encodes the relation between the heads at the sensorized nodes a and b. Each $L^l(i)$ is processed to scale its values in the range 0–255 to use the standard format of an image, achieving the rescaled dataset $\bar{\boldsymbol{L}}$.

Recursive Clustering/Learning

Once the image set $\bar{\boldsymbol{L}}$ has been generated, the recursive clustering/learning stage uses this information, and the WDN topology, to produce the previously mentioned classification tree.

Clustering Stage

To manage the topology of a network, let us model its structure by means of a graph represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $\mathcal{V}$ stands for the set of nodes/junctions, and $\mathcal{E}$ is the set of edges/pipes. A node is represented as $v_x \in \mathcal{V}$, whereas $e_{xy} = (v_x, v_y) \in \mathcal{E}$ denotes an edge connecting v_x and v_y. Nodes from $\mathcal{V}$ are said to be adjacent if they are connected by edges in $\mathcal{E}$. Each edge e_{xy} is associated with a weight w_{xy}, and hence a weighted adjacency matrix $W \in \mathbb{R}^{N \times N}$ can be constructed:

$$w_{xy} = \begin{cases} c_{xy}, & \text{if } e_{xy} \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where c_{xy} represents the length of the pipe connecting junctions v_x and v_y. An extended version of W that stores the relation among all the nodes in the network can be derived: a nonadjacent pair of nodes, i.e., $e_{xy} = (v_x, v_y) \notin \mathcal{E}$, is given an estimated weight $\hat{w}_{xy}$, computed as the sum of the weights of the edges composing the shortest path [see Festa (2006) for an extensive review about algorithms for its computation] from v_x to v_y, and this translates to the sum of the lengths of the pipes in this shortest path. Thus, it is a pairwise distance matrix, referred to as $W \in \mathbb{R}^{N \times N}$.

The structural information stored in W is exploited by the graph agglomerative clustering (GAC) method, developed in Zhang et al. (2013) to segment the nodes of a graph into clusters. Basically, the GAC approach performs an operation composed of three main phases:

1. A k-nearest-neighbors (k-NN) graph [see the introduction in Wang et al. (2012) for a formal definition] is generated from the provided pairwise distance matrix W by means of the computation of an asymmetric weighted adjacency matrix.
2. A set of small clusters is computed considering the *weakly connected components* (Pemmaraju and Skiena 2003) of the k-NN graph with a low neighborhood size k.
3. Those clusters are iteratively merged into larger ones until the settled number of clusters is reached, selecting the two clusters with the highest affinity at each iteration.

In this work, *graph average linkage* (Hastie et al. 2009) is chosen as the agglomerative clustering algorithm that handles the affinity comparison. It uses the edge weights of the k-NN graph as the similarity metric, averaging the values between clusters.

Once the clustering procedure is performed, two disjoint sets of nodes are obtained, i.e., $\mathcal{Z}_1 \subseteq \mathcal{V}$ and $\mathcal{Z}_2 \subseteq \mathcal{V}$ holding that $\mathcal{Z}_1 \cap \mathcal{Z}_2 = \emptyset$ and $\mathcal{Z}_1 \cup \mathcal{Z}_2 = \mathcal{V}$. Assigning a different label to each subset, a label or target vector $\mathbf{t} \in \mathbb{R}^N$ can be derived:

$$t_x = \begin{cases} 1, & \text{if } v_x \in \mathcal{Z}_1 \\ 2, & \text{if } v_x \in \mathcal{Z}_2 \end{cases} \quad \forall\ x = 1, \ldots, N \tag{6}$$

The GAC process, as explained for graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ (corresponding to the complete WDN), is also applicable to graph $\mathcal{G}_{\mathcal{Z}} = (\mathcal{Z}, \mathcal{E}_{\mathcal{Z}})$ (associated with a certain subset of nodes $\mathcal{Z}$), hence

allowing the recursiveness of the clustering strategy. Similarly, the pairwise distance matrix and the targets vector would be $\hat{W}_{\mathcal{Z}} \in \mathbb{R}^{|\mathcal{Z}|\times|\mathcal{Z}|}$ and $\mathbf{t}_{\mathcal{Z}} \in \mathbb{R}^{|\mathcal{Z}|}$, respectively.

Several aspects about the implementation of the clustering strategy must highlighted due to their importance regarding the requirements of the leak localization method:

1. Each clustering operation produces two clusters due to the binary classification approach.
2. The pairwise distance matrix $\hat{W}$ associates a higher cost to distance nodes than to close nodes.
3. The value of parameter k directly influences the generation of the required k-NN graph by setting the size of the considered neighborhoods. For each clustering operation, depending on the cardinality $|\mathcal{Z}|$ of the node set of graph $\mathcal{G}_{\mathcal{Z}}$, it is computed as follows:

$$k_{\mathcal{Z}} = randi(|\mathcal{Z}| - o) + o - 1 \tag{7}$$

where $randi(X)$ is a function generating a pseudorandom integer from $1 - X$, and o is an offset, which must be set for the studied network, to discard possible values that are too close to 1 and X.

Learning Stage

The above-described procedure provides the last ingredient for the learning phase. On the one hand, the complete dataset of images $\bar{\boldsymbol{L}}$, composed of $M \cdot P$ samples of size $N \times N$, is obtained from the image encoding process. However, only $M \cdot |\mathcal{Z}|$ images would be applicable if considering an arbitrary iteration of the recursive clustering/learning approach that operates over $\mathcal{G}_{\mathcal{Z}}$. Furthermore, considering that only M_{tr} time instants out of the M available ones are used for training purposes, the final number of training images corresponds to $M_{tr} \cdot |\mathcal{Z}|$.

On the other hand, a vector of labels $\boldsymbol{t}_{\mathcal{Z}}$ was obtained from the clustering strategy. It must be repeated M_{tr} times to produce $\boldsymbol{t}_{\mathcal{Z}}^{ext} \in \mathbb{R}^{M_{tr}\cdot|\mathcal{Z}|}$, which matches the length of the samples set.

Apart from the described training target set, the validation and testing sets must be generated. Whereas the training set is employed during the learning phase to update the DNN parameters, the validation set is used periodically in the same phase to assess the DNN accuracy. Finally, the testing set is employed to evaluate the final performance of the DNN, once trained. Their lengths would be $M_{val} \cdot |\mathcal{Z}|$ and $M_{test} \cdot |\mathcal{Z}|$, respectively, with $M_{val}, M_{test} < M$. These sets are used to train, validate, and test DNNs as shown in Fig. 1.

The procedure is designed to be complete enough to extract features from the images and to use them to learn how to classify those images, and simple enough to reduce the computational cost of the training process. Additionally, the design is conceived to be general enough to be applicable to different networks, because the only configuration parameter that is related to the studied WDN is the number of sensors. The DL structure is composed by a set of three layers, associated with different roles during the learning process, namely:

1. Convolutional layer (*C* in Fig. 1): It applies sliding convolutional filters to its input, extracting key features that allow to assign a label to the analyzed image (Murphy 2016).
2. Batch normalization layer (*BN* in Fig. 1): It normalizes its inputs (outputs of the convolutional layer) over each input channel (there is one channel per filter in the associated convolution) and over a concrete minibatch, i.e., the set of samples processed before updating the model parameters. This technique speeds up the training and reduces the dependence on the initialization of the DNN. Besides, it adds a regularization effect due to the partition of the dataset in minibatches, and the associated parameter update only occurring after a complete mini-batch is analyzed (Ioffe and Szegedy 2015).
3. ReLu layer: The ReLu layer applies a nonlinear activation function to its inputs. It consists of a threshold operation, setting to zero any negative value (Agarap 2018).

After the instances of the presented set of layers, a fully connected network (FCN) is applied. It multiplies its input by a weight matrix and adds a bias vector, combining the features learned by the previous layers to identify larger patterns. Finally, a *softmax* layer (Bishop 2006) is used to compute the conditional probability of each one of the classes for the analyzed sample.

Procedure Overview

The above-described elements and processes pursue the goal of the generation and subsequent application of a classification tree, formed by the hierarchical organization of the trained DNNs.

Classification Tree Generation

The generation of the classification tree is an off-line operation, carried out with precollected measurements and the available network topology, as summarized in Fig. 2.

The algorithm starts with a preprocessing stage, where the graph associated to the WDN structure is extracted, and the hydraulic measurements at the available historical dataset of leaks are converted into an image set, by means of the described image encoding process.

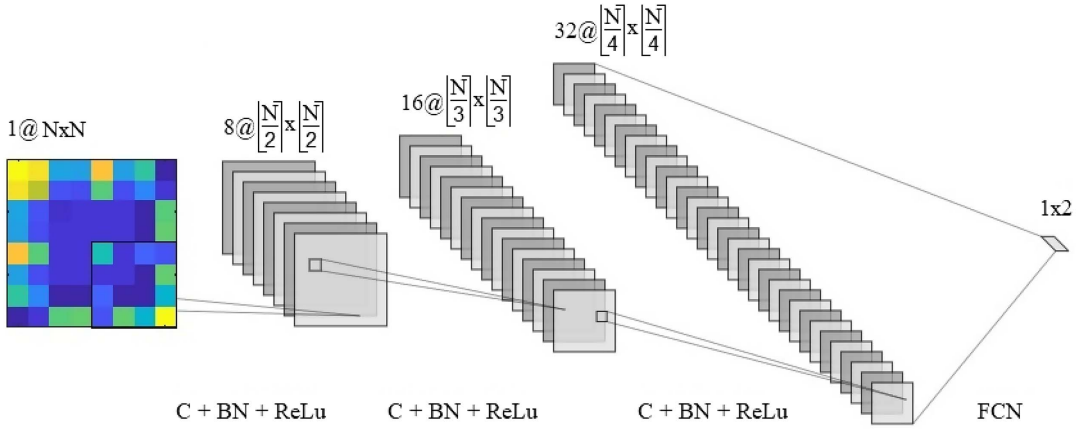

Fig. 1. Deep neural network structure.

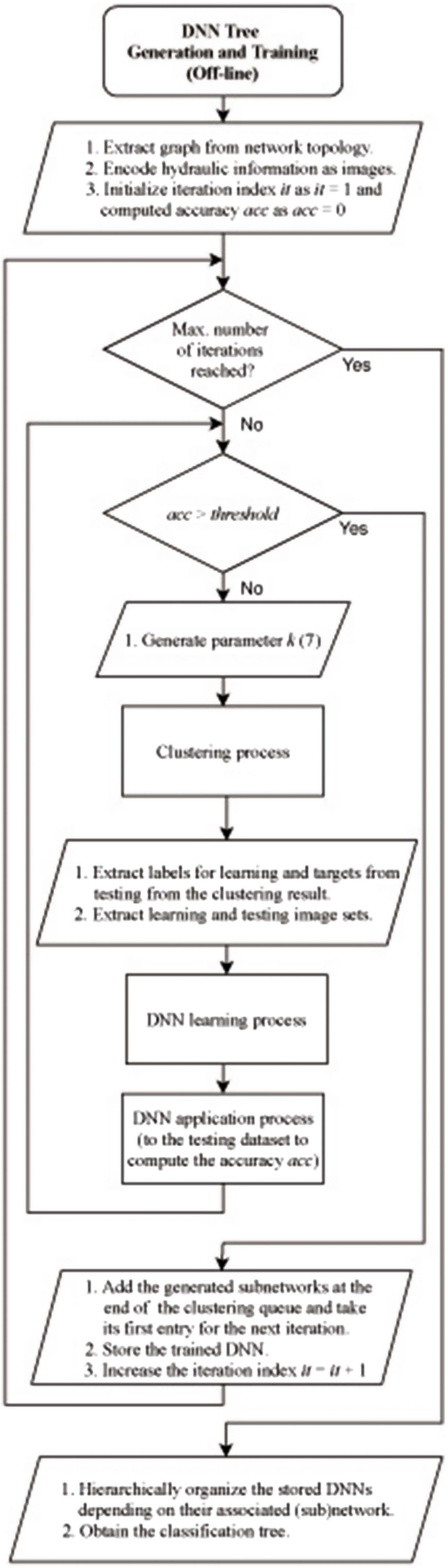

Fig. 2. Flow chart of the methodology steps for the learning (off-line) stage.

Then, the recursive clustering/learning operation is executed until the number of iterations reaches a certain limit. Each iteration is repeated until the testing accuracy of the considered DNN reaches a predefined threshold. Retraining may be performed to check the suitability of the k parameter of the clustering algorithm, which is initially random to explore the range of possible values while pursuing an effective performance of the DNN. For every generated k, the corresponding (sub)network is clustered and the labels are obtained. Taking into account that the image samples are already available and divided into learning (including training and validation) and testing sets, the learning process is performed. If the testing accuracy is good enough, the current iteration will be finalized by adding the subnetworks generated during the clustering at the bottom of a queue that stores the subnetworks to be clustered in subsequent iterations.

Once the process is finished, all the trained DNNs are organized depending on their corresponding (sub)network, obtaining the classification tree: the result of the application of a certain DNN of the tree indicates the next DNN to be used, continuously functioning in this way from the first DNN, which operates over the complete WDN, to the desired depth.

The classification tree generation scheme has an additional advantage: the testing accuracy at the different levels of depth of the tree allows to gain crucial knowledge about the limitations of the leak localization process, regarding issues like the WDN structure and the sensorization properties, i.e., the amount, distribution, and precision of the sensors. All the gathered information about these limitations allows the operator to decide a proper depth for the application of the tree.

Classification Tree Application

This procedure is represented by the flow chart of Fig. 3.

The application of the classification tree is carried out online, and hence hydraulic measurements are converted into an image that is provided to the trained classification tree in real-time, to locate an occurring leak once it has been detected. The depth to traverse through the tree is decided considering the gained knowledge about the limitations of the leak localization process.

First, the top DNN of the tree is fed with the generated image, obtaining a label which indicates the subnetwork where the leak is located from the two possible clusters. This label indicates the next DNN to apply, and the process is repeated until the localization depth is reached. Then, the application of the final DNN provides the set of candidates where the leak can be located.

Case Study

The application of the presented methodology is illustrated by means of a case study. It corresponds to a district metering area (DMA) [see Savić and Ferrari (2014) for extensive information about DMAs] from a real network referred to as E-Town. It is graphically represented in Fig. 4.

The considered DMA is formed by 125 pipes and 120 inner nodes, of which eight are equipped with pressure sensors (stars at Fig. 4). The water supplied to the area is obtained from a single water inlet (square at Fig. 4.). The location of a real leak, that will be studied to show the methodology performance, is indicated with a hexagram at Fig. 4.

Hydraulic Data Encoding for E-Town

To illustrate the image encoding process, Figs. 5 and 6, respectively, show the GAF images associated with eight nodes of E-Town and their locations, at a certain time instant.

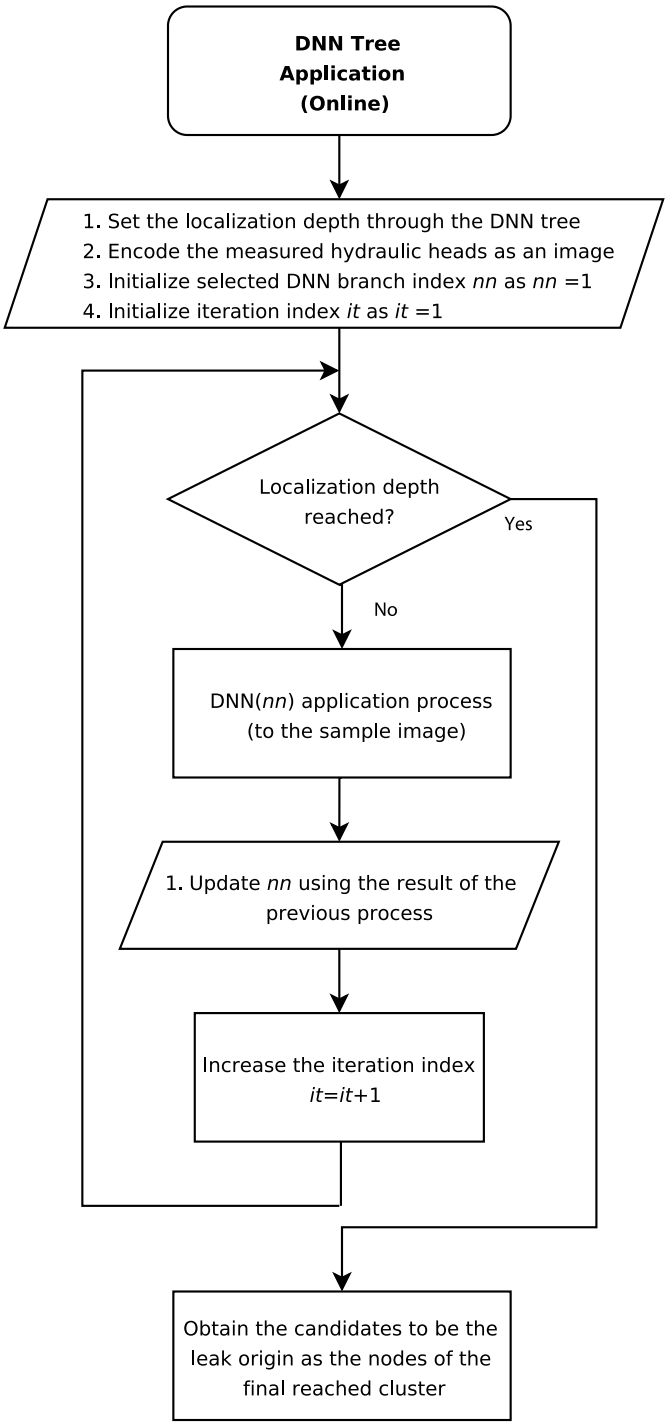

Fig. 3. Flow chart of the methodology steps for the application (online) stage.

As mentioned above, the approach requires hydraulic information of the complete range of possible leaks. In this work, EPANET 2 (Rossman 2000) is used to simulate the necessary leak scenarios. Because only eight sensors are installed, the associated images have a size of 8×8. A pixel ij of the image encodes the relation between components i and j of the data vector.

The majority of GAF images presented in Fig. 5 exhibit clear differences among them due to the effect of the leaks on the readings of the different sensors. These differences are exploited by the DL algorithm to discern the leak location between the possible clusters.

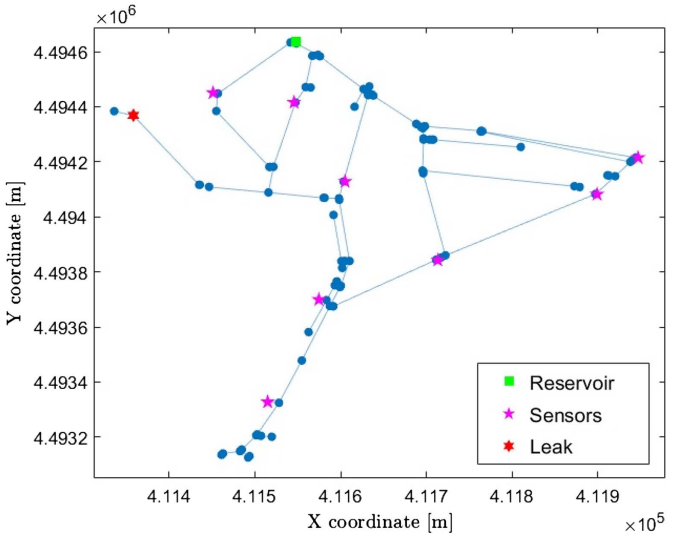

Fig. 4. Schematic representation of E-Town.

However, in the case of nodes 39 [Fig. 5(c)] and 58 [Fig. 5(f)], which correspond to rather close junctions, little differences appear between their images. Both facts illustrate the suitability of an approach that allows to regulate the localization area: the location at node-level is nearly impossible, due to the extremely slight differences of the corresponding leak effects. Hence, a trade-off between location accuracy and area must be considered while applying the classification tree to optimize the methodology performance.

Recursive Clustering/Learning for E-Town

The iterative application of the clustering process up to a third level of depth is shown at Fig. 7.

The produced clusters clearly arise from a topological perspective, partitioning the (sub)networks at links that connect distant nodes. However, it is interesting to remark the specific case of subnetworks 1.1 and 1.2.1 (referring to the numbering in Fig. 7, which are split into two sets that are very different in size. This effect is produced by the explained retraining policy, which repeats the learning stage of the corresponding iteration, modifying the k parameter of the clustering, in search for a testing accuracy that reaches the desired threshold (a 95% in the cases at Fig. 7). Thus, the clustering depends on the network structure, but it is also indirectly affected by the leak information, because the learning phase also seeks to improve the localization.

For the learning stage, the DL architecture has been implemented using the MATLAB® Deep-Learning Toolbox. The learning process is configured by means of the following parameters:

1. Solver: It specifies the algorithm to be used. Stochastic gradient descent with momentum (Qian 1999) is selected, keeping the default momentum value of 0.9. In this way, the oscillations along the path of the steepest descent towards the optimal point are reduced.
2. Learning rate: It is configured by means of four parameters, whose values are tuned and refined to maximize the classification accuracy:
   a. Initial learning rate: It is the value at the start of the training, and it is set to 0.01.
   b. Learning rate schedule: It is set to decrease the learning rate piecewise during the training, after a certain number of epochs, by multiplying it by a factor. This is interesting to refine the learning process at the final steps.

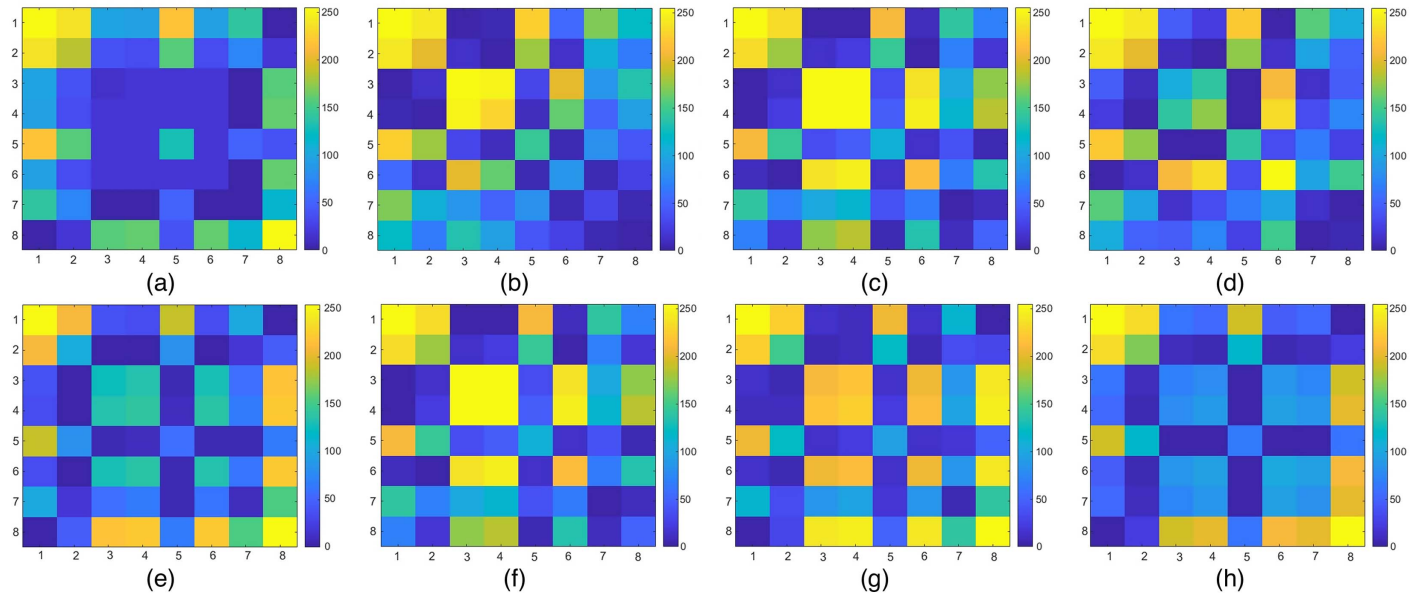

Fig. 5. A comparison among the GAF images associated to eight different leak events is presented. In appearing order: (a) Node 4; (b) Node 29; (c) Node 39; (d) Node 41; (e) Node 43; (f) Node 58; (g) Node 85; and (h) Node 94.

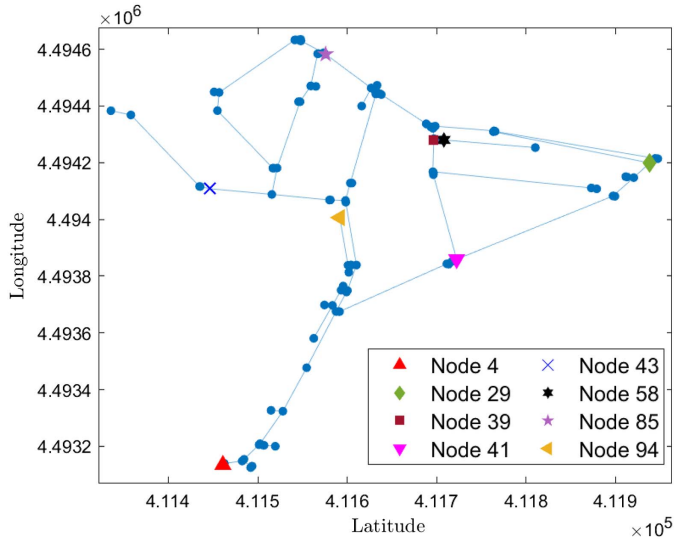

Fig. 6. Location of the considered nodes for the GAF demonstration.

c. Learning rate dropping period: It corresponds to the required number of epochs to drop the learning rate. It is settled to eight epochs, to maintain each learning rate value a minimum number of iterations.
d. Learning rate dropping factor: It is the multiplicative factor that affects the learning rate every learning rate dropping period, and it is configured to 0.9.

3. Convolutional filter size: It settles the length of the square filters at the convolutional layers. These lengths are different for each one of the three instances of the set of layers presented in the explanation of Fig. 1, and they depend on the number of sensors N to facilitate the adaptation to different networks: $[\lfloor N/2 \rfloor \lfloor N/3 \rfloor \lfloor N/4 \rfloor]$.
4. Number of filters: It corresponds to the number of filters at each one of the convolutional layers: [8, 16, 32].

The rest of the available parameters are maintained as default. The image/label set is shuffled before every epoch to improve the regularizing effect of the batch normalization, as the activation of a concrete image during training depends on the rest of the images of the minibatch. The division of the dataset at each learning operation includes a 75% for training, a 15% for validation and a 10% for testing purposes. The different neural networks composing the classification tree are trained by means of a NVIDIA GeForce MX130 GPU and a Intel® Core™ i7-8565U processor.

Results

A real leak event was evaluated with the trained classification tree to evaluate the goodness of the solution. The leak is estimated to have a size of 1.15 l/s, and hence the DNNs were trained with hydraulic data obtained using EPANET2 and considering this leak size. The dataset of the real leak pressure measurements is composed of 720 entries (that is, 24 h with a sampling period of 2 min) which are affected by different sources of uncertainty due to their real nature. Therefore, the EPANET2 scenarios were generated considering a 5% of uncertainty at the user water demands and the pipe roughness. Moreover, to consider the measurement noise in the sensors, the data from the hydraulic simulator were truncated to emulate a sensor precision of ± 0.01 m. The location of the leak in E-Town is indicated in Fig. 4. The application of the classification tree to the real leak data is graphically presented in Fig. 8. The encircled subnetworks are the selected ones by the localization algorithm at each level of the tree. The black cross marks the location of the real leak.

The information from the figure can be completed by these numeric results:

1. Considering the levels from (a) to (c) (the letters refer to the coding at Fig. 8), the localization is properly performed, as the leaky node is located in the highlighted area. This area is composed of a total of 34 nodes, with a mean pipe distance of 576.41 m from these nodes to the leaky one. Moreover, eight clusters are formed at this level of depth of the tree, i.e., eight

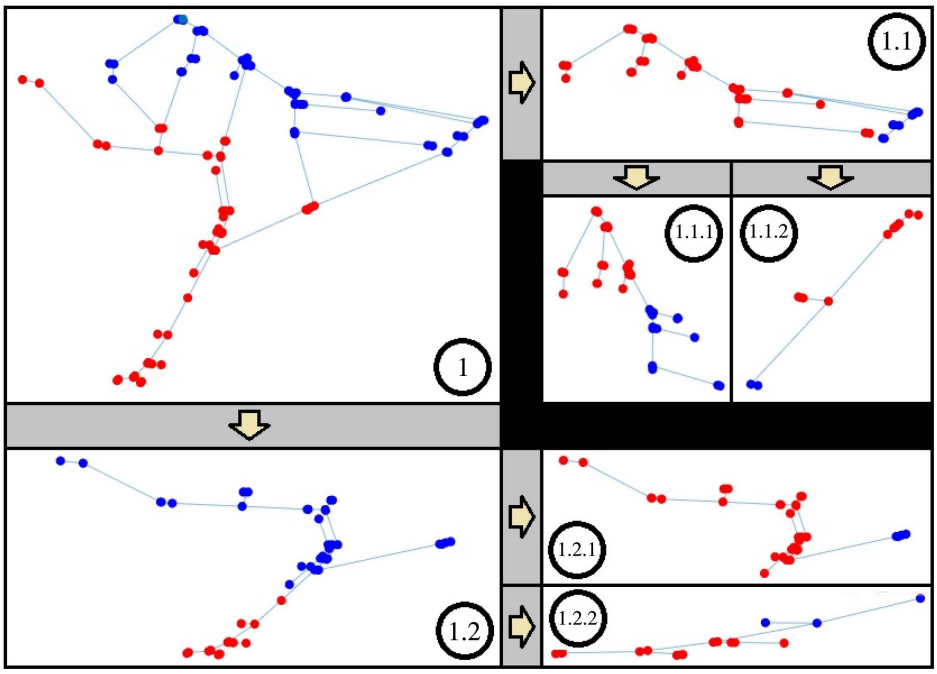

Fig. 7. Clustering results for E-Town until eight clusters are achieved.

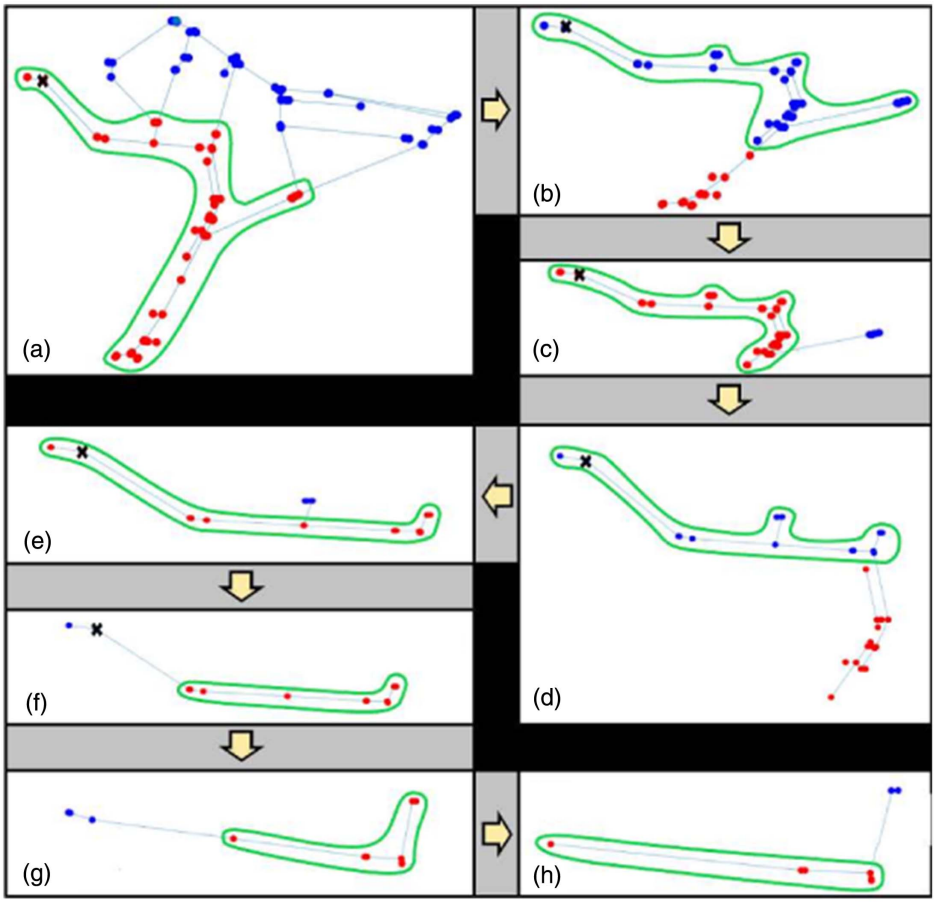

Fig. 8. Graphical summary of the localization process performance: (a) complete network; (b) subnetwork of the cluster selected from (*a*); (c) subnetwork of the cluster selected form (*b*); (d) subnetwork of the cluster selected form (*c*); (e) subnetwork of the cluster selected form (*d*); (f) subnetwork of the cluster selected form (*e*); (g) subnetwork of the cluster selected form (*f*); and (h) subnetwork of the cluster selected form (*g*).

possible outcomes of the algorithm if the tree is traversed until this depth.

2. At level (*d*), the localization remains correct, reducing the localization region to 16 nodes and the mean pipe distance to 357.02 m. There are 16 final clusters at this level of depth.
3. Progressing to level (*e*), the correct localization diminishes the number of nodes to 13 and the mean pipe distance to 335.18 m, whereas the number of final clusters adds up to 29.
4. However, at level (*f*), the localization turns out to be incorrect, as the leaky node is not considered in the localization area, which is formed by 11 nodes. The mean pipe distance and the number of final clusters get increased to 393.62 m and 52, respectively.
5. Considering levels from (*g*) to (*h*), the incorrect localization continues increasing both the mean pipe distance and the number of final clusters to 418.98 m and 85, respectively, whereas six nodes are included at the achieved localization region.
6. Finally, considering the single-node level (which implies the existence of 120 clusters, one per node), the pipe distance from the reached node to the leaky one is 442.01 m.

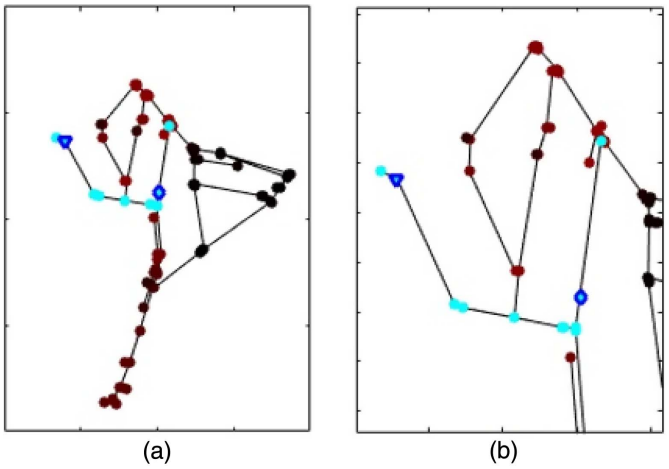

Fig. 9. Result of the application of a model-based approach to the leak localization of the real leak: (a) correlations map—the lighter the color of the node, the higher the correlation, and hence, the higher the probability of the node to be the origin of the leak; and (b) zoom of the most probable leak area.

Discussion

Several aspects can be discussed about the presented results:

1. The localization operation works properly until the (f)-level is reached, referring to both Fig. 8 and presented numerical results. Therefore, branch (e) of the tree would correspond to the localization area limit regarding the region where the real leak is located.
2. The mean pipe distance is increased by the amount of nodes at early stages of the tree, and the large separation between nodes of interest. For example, the pipe distance between the leaky node and its neighbor from the right is 267.25 m.
3. Regarding the final number of nodes at level (e), the search area is successfully reduced to approximately 10% of the total network. The value of 29 final clusters at this level shows that the clusters of the region of the leak are quite big in comparison with the rest of the network. The heterogeneity of the density of nodes per cluster at the different regions is produced by the network structure and the retraining approach.

The performance of the proposed localization approach for the considered real leak can be compared with that of a well-known model-based methodology, based on a fault sensitivity analysis of the residuals (Casillas et al. 2013). The application of this strategy is represented in Fig. 9.

The light-coloured nodes of this figure and the dark-coloured ones at level (e) of Fig. 8 correspond to a similar region of the network, and therefore the localization result of the proposed methodology can match the degree of performance of a model-based state-of-the-art technique. Furthermore, the proposed method produces an even more accurate result:

1. Regarding the exact localization, the model-based strategy indicates a node that is 501 m away from the real leak, whereas the proposed approach finds a node that is 442 m away.
2. About the selected area of localization, the proposed methodology produces a more precise result due to the exclusion of outliers, whereas the model-based method highlights one.

Therefore, the aforementioned model-based method is outperformed at both the leak candidate set selection and the node-level localization by the proposed methodology. Regarding these two aspects, it is important to focus on the selected area of localization because it is directly related to one of the novelties of the proposed approach in comparison with previous methods: the hierarchical organization of the classification tree. The differences between training data (which in this case is obtained from a simulator) and real measurements may introduce errors at the node-level localization, but the classification tree can be traversed backwards to search the leak at a larger set of nodes, which are also guaranteed to be in the same area of the network due to the topology-based clustering. This characteristic entails an advantage in comparison with other methodologies: for example, in the case of the compared model-based approach, there exists an outlier in the final set of most probable leak origins (the light-coloured point in Fig. 9 that is located out of the branch where the leak occurs) that hinders the localization of the real location of the leak.

Conclusions

This article presents a leak localization approach based on image encoding, graph-based clustering and deep-learning. It uses information about the different possible leak events at a WDN, converted into images through a data encoding process, to produce a classification tree by means of a recursive clustering/learning approach. The techniques selected to perform these operations are configured and applied in customized ways to fit the necessities of the leak localization task.

The application of the methodology allows to naturally gain knowledge about the leak localization limitations due to the sensor accuracy, placement, and number, and network characteristics. Accordingly, the best leak localization area can be selected, as the classification tree can be traversed from the first clustering of the complete network to any pretrained intermediate state until the single-node level is reached. Besides, this localization-area-based approach guarantees the selection of congregated nodes due to the exploitation of the structural and topological information.

The method has been applied to the case of a leak at a real network, using data from the actual WDN. The results are compared with the performance of another state-of-the-art technique, showing improvements regarding both the leak localization area definition and the node-level operation.

Some tasks remain open regarding the research in this field. On the one hand, a review of the different components of the

methodology can be performed to find even more suitable elements to substitute for the current ones. On the other hand, regarding the selected methods, their configuration and calibration can be improved by means of their inputs and parameters. Furthermore, extra studies about the effect of uncertainty and inaccuracies at the hydraulic model, sensor measurements, and network characteristics can be performed to analyze the sensitivity of the method to those aspects. Finally, it could be interesting to test the methodology considering that the hydraulic model of the WDN is not available, that is, working together with a water utility to perform several artificial leak experiments to gather data for the training of the DNNs.

Data Availability Statement

All data, models, or code generated or used during the study are proprietary or confidential in nature and may only be provided with restrictions. Specifically, there is a hard restriction in the sharing of information about the studied network: features, location, the hydraulic information from the real leak event. The associated codes could be shared upon reasonable request.

Acknowledgments

The authors want to thank the Spanish national project "DEOCS (DPI2016-76493-C3-3-R)" project (which is finished nowadays) by its continuation: "L-BEST Project (PID2020-115905RB-C21) funded by MCIN/ AEI /10.13039/501100011033" and the Spanish State Research Agency through the María de Maeztu Seal of Excellence to IRI (MDM-2016-0656). Joaquim Blesa acknowledges the support from the Serra Húnter program.

References

Agarap, A. 2018. "Deep learning using rectified linear units (relu)." Preprint, submitted March 22, 2018. http://arxiv.org/1803.08375.

Arifin, B., Z. Li, S. Shaha, G. Meyer, and A. Colin. 2018. "A novel data-driven leak detection and localization algorithm using the Kantorovich distance." *Comput. Chem. Eng.* 108 (Jan): 300–313. https://doi.org/10.1016/j.compchemeng.2017.09.022.

Bishop, C. 2006. "*Pattern recognition and machine learning*." In *Probability distributions—The exponential family.* New York: Springer.

Blesa, J., and R. Pérez. 2018. "Modelling uncertainty for leak localization in water networks." *IFAC-PapersOnLine* 51 (24): 730–735. https://doi.org/10.1016/j.ifacol.2018.09.656.

Casillas, M., L. Garza-Castañón, and V. Puig. 2013. "Model-based leak detection and location in water distribution networks considering an extended-horizon analysis of pressure sensitivities." *J. Hydroinf.* 16 (3): 649–670. https://doi.org/10.2166/hydro.2013.019.

Chan, T. K., C. S. Chin, and X. Zhong. 2018. "Review of current technologies and proposed intelligent methodologies for water distributed network leakage detection." *IEEE Access* 6 (Dec): 78846–78867. https://doi.org/10.1109/ACCESS.2018.2885444.

Festa, P. 2006. "Shortest path algorithms." In *Handbook of optimization in telecommunications*, edited by M. Resende and P. Pardalos, 185–210. Boston: Springer.

Han, Q., R. Eguchi, S. Mehrotra, and N. Venkatasubramanian. 2018. "Enabling state estimation for fault identification in water distribution systems under large disasters." In *Proc., IEEE 37th Symp. on Reliable Distributed Systems (SRDS)*, 161–170. New York: IEEE. https://doi.org/10.1109/SRDS.2018.00027.

Hastie, T., R. Tibshirani, and J. Friedman. 2009. *The elements of statistical learning. Data mining, inference, and prediction*. Berlin: Springer.

Huang, Y., F. Zheng, Z. Kapelan, D. Savić, H.-F. Duan, and Q. Zhang. 2020. "Efficient leak localization in water distribution systems using multistage optimal valve operations and smart demand metering." *Water Resour. Res.* 56 (10): e2020WR028285. https://doi.org/10.1029/2020WR028285.

Ioffe, S., and C. Szegedy. 2015. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." Preprint, submitted February 11, 2015. https://arxiv.org/abs/1502.03167.

Javadiha, M., J. Blesa, A. Soldevila, and V. Puig. 2019. "Leak localization in water distribution networks using deep learning." In *Proc., 2019 6th Int. Conf. on Control, Decision, and Information Technologies (CoDIT)*, 1426–1431. New York: IEEE. https://doi.org/10.1109/CoDIT.2019.8820627.

Kang, J., Y.-J. Park, J. Lee, S.-H. Wang, and D.-S. Eom. 2017. "Novel leakage detection by ensemble CNN-SVM and graph-based localization in water distribution systems." *IEEE Trans. Ind. Electron.* 65 (5): 4279–4289. https://doi.org/10.1109/TIE.2017.2764861.

LeChevallier, M. W., R. W. Gullick, M. R. Karim, M. Friedman, and J. E. Funk. 2003. "The potential for health risks from intrusion of contaminants into the distribution system from pressure transients." *J. Water Health* 1 (1): 3–14. https://doi.org/10.2166/wh.2003.0002.

Menapace, A., D. Avesani, M. Righetti, A. Bellin, and G. Pisaturo. 2018. "Uniformly distributed demand EPANET extension." *Water Resour. Manage.* 32 (6): 2165–2180. https://doi.org/10.1007/s11269-018-1924-6.

Murphy, J. 2016. *An overview of convolutional neural network architectures for deep learning*. Plymouth, MA: Microway.

Navarro, A., O. Begovich, J. Delgado-Aguiñaga, and J. Sánchez. 2019. "Real time leak isolation in pipelines based on a time delay neural network." In *Proc., 2019 IEEE Int. Autumn Meeting on Power, Electronics and Computing (ROPEC)*, 1–6. New York: IEEE. https://doi.org/10.1109/ROPEC48299.2019.9057112.

Pemmaraju, S., and S. Skiena. 2003. "*Computational discrete mathematics: Combinatorics and graph theory with mathematica®.*" In *Strong and weak connectivity—Connectivity—Properties of graphs*, edited by M. Resende and P. Pardalos. Cambridge, UK: Cambridge University Press.

Pérez, R., G. Sanz, V. Puig, J. Quevedo, M. À. Cugueró-Escofet, F. Nejjari, J. Meseguer, G. Cembrano, J. M. Mirats-Tur, and R. Sarrate. 2014. "Leak localization in water networks: A model-based methodology using pressure sensors applied to a real network in Barcelona." *IEEE Control Syst.* 34 (4): 24–36. https://doi.org/10.1109/MCS.2014.2320336.

Pérez-Pérez, E., F. López-Estrada, G. Valencia-Palomo, L. Torres, V. Puig, and J. Mina-Antonio. 2021. "Leak diagnosis in pipelines using a combined artificial neural network approach." *Control Eng. Pract.* 107 (Feb): 104677. https://doi.org/10.1016/j.conengprac.2020.104677.

Puust, R., Z. Kapelan, D. Savić, and T. Koppel. 2010. "A review of methods for leakage management in pipe networks." *Urban Water J.* 7 (1): 25–45. https://doi.org/10.1080/15730621003610878.

Qian, N. 1999. "On the momentum term in gradient descent learning algorithms." *Neural Networks* 12 (1): 145–151. https://doi.org/10.1016/S0893-6080(98)00116-6.

Rossman, L. A. 2000. *EPANET 2 user's manual*. Cincinnati: National Risk Management Research Laboratory.

Sanz, G., R. Pérez, Z. Kapelan, and D. Savić. 2015. "Leak detection and localization through demand components calibration." *J. Water Resour. Plann. Manage.* 142 (2): 04015057. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000592.

Savić, D., and G. Ferrari. 2014. "Design and performance of district metering areas in water distribution systems." *Procedia Eng.* 89: 1136–1143. https://doi.org/10.1016/j.proeng.2014.11.236.

Savić, D. A., Z. S. Kapelan, and P. M. Jonkergouw. 2009. "Quo vadis water distribution model calibration?" *Urban Water J.* 6 (1): 3–22. https://doi.org/10.1080/15730620802613380.

Soldevila, A., J. Blesa, T. N. Jensen, S. Tornil-Sin, R. M. Fernández-Cant, and V. Puig. 2020. "Leak localization method for water-distribution networks using a data-driven model and Dempster-Shafer reasoning." *IEEE Trans. Control Syst. Technol.* 29 (3): 937–948. https://doi.org/10.1109/TCST.2020.2982349.

Sophocleous, S., D. Savić, and Z. Kapelan. 2019. "Leak localization in a real water distribution network based on search-space reduction." *J. Water Resour. Plann. Manage.* 145 (7): 04019024. https://doi.org/10.1061/(ASCE)WR.1943-5452.0001079.

Wang, D., Y. Zheng, and J. Cao. 2012. "Parallel construction of approximate KNN graph." In *Proc., 2012 11th Int. Symposium on Distributed Computing and Applications to Business, Engineering & Science*, 22–26. New York: IEEE. https://doi.org/10.1109/DCABES.2012.87.

Wang, Z., and T. Oates. 2015. "Imaging time-series to improve classification and imputation." In *Proc., 24th Int. Joint Conf. on Artificial Intelligence*, 3939–3945. Buenos Aires, Argentina: IJCAI.

Xu, Q., R. Liu, Q. Chen, and R. Li. 2014. "Review on water leakage control in distribution networks and the associated environmental benefits." *J. Environ. Sci.* 26 (5): 955–961. https://doi.org/10.1016/S1001-0742(13)60569-0.

Zhang, W., D. Zhao, and X. Wang. 2013. "Agglomerative clustering via maximum incremental path integral." *Pattern Recognit.* 46 (11): 3056–3065. https://doi.org/10.1016/j.patcog.2013.04.013.

Zhou, X., Z. Tang, W. Xu, F. Meng, X. Chu, K. Xin, and G. Fu. 2019. "Deep learning identifies accurate burst locations in water distribution networks." *Water Res.* 166 (Dec): 115058. https://doi.org/10.1016/j.watres.2019.115058.