DiTEC: Digital Twin for Evolutionary Changes in Water Distribution Networks

Victoria Degeler[1] [0000−0001−7054−3770], Mostafa Hadadian[2] [0000−0003−4675−9111] *, Erkan Karabulut[1] [0000−0003−2710−7951], Alexander Lazovik[2] [0000−0002−6337−1840], Hester van het Loo[3], Andrés Tello[2] [0000−0003−1962−1440], and Huy Truong[2] [0009−0006−9079−321X]

[1] Informatics Institute, University of Amsterdam, Netherlands
{v.o.degeler, e.karabulut}@uva.nl
[2] Bernoulli Institute, University of Groningen, Netherlands
{m.hadadian, a.lazovik, andres.tello, h.truong}@rug.nl
[3] Vitens NV, Netherlands
{hester.vanhetloo}@vitens.nl

Abstract. Conventional digital twins (DT) for critical infrastructures are widely used to model and simulate the system's state. But fundamental environment changes bring challenges for DT adaptation to new conditions, leading to a progressively decreasing correspondence of the DT to its physical counterpart. This paper introduces the DiTEC system, a Digital Twin for Evolutionary Changes in Water Distribution Networks (WDN). This framework combines novel techniques, including semantic rule learning, graph neural network-based state estimation, and adaptive model selection, to ensure that changes are adequately detected, processed and the DT is updated to the new state. The DiTEC system is tested on the Dutch Oosterbeek region WDN, with results showing the superiority of the approach compared to traditional methods.

Keywords: Digital Twin · Water Distribution Networks · Rule Learning · Change Adaptation · State Estimation · Adaptive Model Selection.

1 Introduction

Conventional digital twins (DT) create a representation model of the physical world, recreating its diverse aspects [31]. The main DT requirement is to synchronize the state of a physical counterpart with its virtual representation, using internet of things (IoT) sensors and actuators. The synchronization link is continuously maintained, allowing to experiment, simulate and optimize the properties of the physical twin (PT) at any time. The real-time bi-directional communication link with the PT is one of the main distinguishing features of the Digital Twin paradigm from earlier concepts of digital simulation models.

Critical infrastructures, such as Water Distribution Networks (WDNs), exist in a changing environment. The ability of the DT to adapt to changes is

* Except the first author, authors are listed in the alphabetical order.

important to ensure the longevity of the created DT model, whether it stems from the data values (e.g. drought resulting in unusually low water levels), changing data distribution (e.g. changing yearly weather patterns), new physical structures or conditions (e.g. maintaining a piping system or adding a new neighborhood). Indeed, the most common reason for low accuracy in production-scale machine learning (ML) systems is that they no longer correspond to the data they were trained on, leading to model decay known as concept drift [3].

In WDNs, to make sure that clean and safe water is delivered timely to all locations, the extensive physical network of pipelines is controlled with valves, pumps, boosters, and reservoirs to direct the flow of water. The configuration of these assets must be carefully managed. Elaborate mathematical simulation systems, such as EPANET [29] and WNTR [22], that are based on the physical properties of water flows are currently employed to simulate the state of WDNs. But these mathematical simulation systems are limited, as scenarios for simulation must be manually defined and for that, the state of the physical world must be fully known. Any environmental change, such as a water leak, changed consumption demand, or worn-out valve, negatively affects the simulation accuracy, or completely invalidates it. Similarly, any network changes (e.g. in the pipeline topology, maintenance activities) require a costly and slow process of model recalibration and validation. Careful consideration by a human expert must be provided to choose the specific calibration and validation scenarios. Besides physical inspections of the network, the way to realize that the simulation is obsolete is by comparing simulated values to the real-time sensor data. If they diverge, it means the simulation's configuration is wrong and a new model needs to be selected. Unfortunately, there is no straightforward way to understand what this new model must be just by looking at the sensor data. Therefore, intelligent state estimation models, such as the one described later in this paper, must be employed to ensure that the correct state of the environment and new configuration for the simulation model can be found.

We present a use case of engineering a DT for WDNs, introducing the DiTEC system, a Digital Twin for Evolutionary Changes in WDNs. We argue that such a DT must contain a number of additional characteristics to ensure that it can withstand environmental change without any loss of its usefulness and precision. The main contributions of this work include creating a new change-resistant DT framework, using the ontological data representation to learn a set of rules that clearly describe the behavior of the system and can change with time, showing that graph neural networks can be used for state estimation given the minimum available sensor information, and unfolding the full architectural process of adapting the DT to a change, including the change detection, running what-if scenarios, and adaptive new model selection.

2 Related Work

Critical infrastructures are among the main application areas of DTs due to their higher need for protection and management. Recent examples include DT implementations from a cyber-security point of view to simulate and mitigate potential cyber threats [24], for improving the performance of edge AI devices in terms of computation and network delay with a smart task allocation mechanism [9], and to increase the resilience of the critical infrastructure to unforeseen events [6]. Two recent DT implementations for large cities' WDNs are in Valencia [8] and Lisbon [27]. Both studies provide a set of WDN DT requirements, example applications such as leak detection and localization, and highlight the use of AI techniques. These existing systems overlook the increasing usage of semantic technologies in DTs [19], which are used to model DT systems and data, facilitate semantic interoperability, and enable semantic inference.

With the integration of cyber-physical units, the DT solution can offer more evolutionary services, encompassing real-time monitoring, deeper analysis, and decision-making. Specifically, the solution leverages such units to estimate state variables of a physical ecosystem, such as electrical vehicles [26], power grids [5], and traffic networks [23]. Nevertheless, challenges persist as these variables often suffer from incompleteness and unreliability due to the data explosion, noise transmission, and the lag between the deployment of cyber-physical units and the extensive scale of such systems.

To mitigate the issue, two influential approaches are model-based and data-driven state estimations. The model-based approach relies on mathematical models empowered by differential equations and conservative physical laws to compute more reliable state variables under known conditions [22, 29]. Conversely, the data-driven approach employs ML algorithms and optimization techniques on sensor flow to infer states beyond initial settings [15]. In our study, we seamlessly integrate both approaches to create a hybrid solution that accurately estimates states in either historical data or unforeseen scenarios.

In cases where we have several models addressing the same task, the challenge lies in selecting the most appropriate model for the current situation [28]. Some work in the service-oriented computing domain addresses it by adaptively selecting services for tasks [36], where services and models can be deemed equivalent. But their focus is on optimizing non-functional requirements, such as response time, rather than functional requirements, like the accuracy of models or other performance measures. In contrast, our focus is on optimizing functional requirements while respecting non-functional ones. Our approach is tailored to capture the probabilistic nature of ML solutions.

3 Digital Twin System Architecture

In this section, the general architecture of the DiTEC system is discussed. We introduce and briefly discuss the main system modules and their interconnections. The next section describes these modules in detail, while Section 5 shows their validation on a real use case of a WDN in the Dutch region of Oosterbeek. The high-level architecture overview is presented in Figure 1.

The target **physical environment** that we aim to model consists of a closed system WDN. Normally, a WDN conceptually starts with the effluent of a

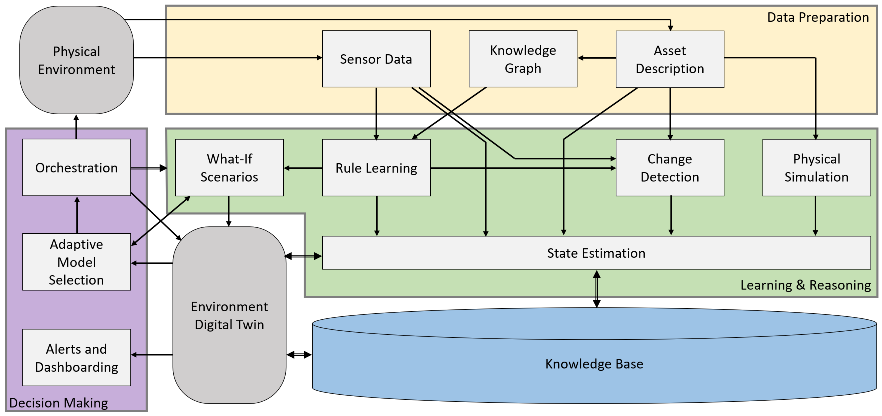

Fig. 1: The DiTEC system's main blocks

drinking water treatment plant or a reservoir. From there, the water is pumped into the WDN and, through a physical net of pipes, end at final distribution points, such as homes, factories, hospitals. When the system is not fully closed and contains a number of connections to the world outside of our modelling target, such as if we only model a separate district in a bigger water network, these connections must be rigorously modelled as well with a set of input/output variables. They can include other entities that affect the functioning of the network, e.g., weather, calendar of big events, maintenance schedule.

The static information about the environment is represented as a collection of **asset descriptions**. This description is domain-specific and contains the characteristics of the entities and the relations between them. For example, assets represent the existence of pipes, having as parameters pipe diameter, pipe length, pipe roughness, and being related to junctions, which are different entities, connecting several pipes together. Parameters can represent static values, such as pipe length, and dynamic values, such as water pressure. Representing the complexity of a domain through asset descriptions is not straightforward. Simple variable hierarchies and collections fail to capture complex relations between them, while tabular data may be suited for specific parts of the system, efficiently representing the system as a whole with tables is often hard due to the uniqueness and semantic richness of many system objects. On the other hand, using domain-specific ontologies and populating them with data from the exact system to obtain **knowledge graphs** (KG) gives an excellent solution to the complexity of data representation and allows keeping the needed semantic information in the DT. While static values can be adequately described with only asset descriptions, the dynamic real-time information about the environment is collected with sensors and is normally referred to as time series **sensor data**.

Physical simulations are often employed in systems where complex physical phenomena have well-defined mathematical formulations. In WDNs, existing physical simulation tools allow simulating water flows precisely and in detail, given the complete asset descriptions and demand patterns. But the requirement to know the exact physical state of a multitude of parameters is often impossible to achieve in practice, which limits the applicability of physical simulations. Therefore, DL-based **state estimation** models have proved their indispensability for being able to restore the exact state of unobservable parts of the environment from sensor data, without the need for additional calibration. Advanced state estimation mechanisms allow the full reconstruction of the original state of the environment, creating a complete **environment digital twin**. This DT uses the **knowledge base** to represent the current system's state, to store the historical states, and, using the simulations, and what-if scenarios, to glimpse into the possible future states. Capturing the system's behavior patterns in complex physical infrastructures presents an additional challenge. Semantic association **rule learning** is a mechanism of extracting logical associations between entities while taking into account their semantic relationships from KGs. Extracted rules represent clearly explainable human-understandable logical connections, expressed in high-level semantic notions.

The fundamental external conditions and the system's internal behavior change over time, representing data or concept drift, which affects learned rules, making some of them invalid, or introducing new ones. This can be captured with **change detection** mechanisms, by tracking the state of the system and comparing it against the learned rules. If a fundamental change is detected, the **what-if scenarios** can be employed to find the effect of the new situation on the current learning models. **Adaptive model selection** allows finding the best-performing model that corresponds to the current conditions. The DT is updated to the new model with **Orchestration**, which is also responsible for managing the necessary changes for execution in the environment while maintaining the system standards. Moreover, **Alerts and dashboarding** enable real-time issue detection and visualization for quick response and informed decision-making.

4 System Model

4.1 Data Preparation

Data preparation activities include processing diverse data sources to be available for further usage in our Digital Twin (DT) system model.

Asset Description. Assets involve all pieces of information related to the modelling counterpart, categorized into physical and virtual. Physical assets include constructed components of target systems, e.g. vehicle wheels, brakes, body, and engine, each associated with detailed information (model, brand, material, etc.). They contain relevant data about the physical environment, such as geographical locations, maintenance reports, and socioeconomic factors. The physical assets remain relatively unchanged throughout the PT lifecycle. Virtual assets comprise simulation-related parameters and generative products, exemplified by Building Information Modelling (BIM) and 3D models [25].

In WDNs, all informative data is primarily stored in a virtual asset, an input (INP) file, which describes the properties of infrastructural components. A WDN embodies an undirected graph. Nodes represent junctions – demand points, water sources like reservoirs, and extra supply storage like tanks, while edges symbolize connecting pipes, valves, and discharge pumps between two nodes. Each element also obtains its own set of sophisticated attributes, e.g. a pipe is associated with roughness, diameter, and length. Moreover, the INP file contains simulation-related parameters that allow practitioners to run synthetic scenarios, especially in unexpected scenarios such as pipe leaks, earthquakes, and fire hydrants [22]. Concretely, some critical parameters (e.g., junction demand, reservoir heads, and pump curves) expressed as dynamic patterns can be fed into a simulation tool to replicate a lively network in a fixed time. This benefits further analysis and understanding of the actual WDN. In light of this, the INP file, encompassing the overall network information and the attached simulation ability, has become a standard format widely used and shared across the research community.

Time Series Sensor Data is used to obtain the latest state of the PT. This data provides information about the system components or its environment. The data source can be physical or virtual sensors, or external sources, such as weather forecast channels. It is coupled with a timestamp, which refers to the time of measurement or the value obtained from a source. Time series data is the main data source for a majority of operations performed in a DT including monitoring, detection, and reasoning tasks. In WDNs, water flow, pressure, height in water tanks, and conductivity are the major time series data sources.

Knowledge Graphs. Semantic technologies such as ontologies and knowledge graphs (KG) are being increasingly used in DTs, for semantic interoperability, semantic reasoning, and system/data modeling [19]. An ontology in this context describes physical and digital entities and their relations in the system and its environment in a formal way. KGs represent instances of those entities and relations based on the underlying ontology. We argue that KGs can be further utilized in learning and reasoning tasks, especially when combined with time series data. In our system model, each time series data source has a representation in the KG, hence providing context to time series data simply by looking at their adjacent nodes and relations on the graph.

4.2 Learning and Reasoning

Learning and reasoning tasks are used to better understand the current state of the system, and create actionable insights to be used in decision making.

Physical Simulations serve as a crucial tool for conducting in-depth analysis and manual calibration by domain experts of physical entities, such as complex infrastructures or objects post-disaster. These entities pose challenges in monitoring and further exploration. Conventional simulation of WDNs offers an overview of the virtual water network, allowing experts to gather insights under diverse conditions, such as customer demands, nodal pressure, and pipe velocity. The virtual replica is constructed with initial parameters derived from the asset description, then obtaining the measurements with a hydraulic simulation tool, e.g., EPANET [29] and WNTR [22]. DT can simulate and replicate rare incidents such as leaks, pipe bursts, and sensor malfunctions with a minimal cost.

Although physical models have been very influential, they have several drawbacks including calibration requirements, restricted parameters, and limited generalizability. These models need a manual calibration process to maintain data consistency with the physical counterpart, but the process relies on domain expertise. Even a minor discrepancy in the configuration file can lead to significant divergence in simulation outcomes, while unknown parameters can lead to a simulation failure. Physical simulations are robust in adjusting dynamic factors (e.g. patterns) but inflexible in varying static ones (e.g. nodal elevation, tank diameter, valve settings). Also, dynamic factors are often tied to privacy and security concerns, making them unavailable for access in public use [33]. Such issues not only hinder the ability of a DT to replicate rarely-seen scenarios but also indicate a lag of development in reproducibility and credibility.

Alternatively, physical simulations can be leveraged to generate a comprehensive sizeable dataset of scenarios derived from available WDNs and diverse configurations to train powerful data-driven models, which require extensive data inputs. The dataset effectively alleviates concerns about security and safety, as all data reside in a synthetic space. We have shared our dataset to offer a favorable method for comparing WDN research outputs [32].

State Estimation provides a complete view of the status of a system at any given time. In the context of WDNs, the state comprises the values of demand, pressure, and flow for every junction and pipe in the system. Usually, only a limited number of sparsely located sensors are installed in a WDN. Thus, the state at all other locations needs to be estimated in order to be able to optimize the operation of the network [33]. This facilitates near-real-time control and monitoring tasks to ensure the healthiness of the WDN. Conventionally, state estimation is performed using physics-based mathematical simulation tools. Then, a manual calibration process is used to match the estimated values with the known ones provided by the sensors. In the case of DTs, we propose a DL data-driven approach. The simulated data described in the previous section serves as input to train the state estimation models. The data is generated by a physics simulation without incorporating real sensor data, and is drawn from a uniform distribution. During inference, the DL data-driven model uses the known values from real sensors to estimate the nodal state of the entire WDN.

We propose that state estimation models can be trained using the simple, yet very effective, *masking* technique [14]. It consists of hiding part of the input data from the model and learning to predict those missing parts. The values are "removed" from the input using a binary vector mask $m = \{m_1, m_2, ..., m_k\}$, where $m_i \in \{0, 1\}$, whose elements are sampled from a binomial distribution. Thus, P values will be hidden from the model with probability p, and Q values will be shown to the model with probability $q = 1 - p$. Then, the models learn to predict the missing values using gradient-based optimization methods [21].

To assess the validity of the state estimation models the evaluation strategies should consider ground-truth data provided by the sensors and take into account the uncertainties of real-world scenarios [33]. The data created with the physics-based calibrated models becomes the ground truth for model testing, as those computed states match the sensors' measurements. Such models do not include uncertainties and resemble network operations under normal conditions. Then, after applying random masking for every snapshot in the calibrated data, we use the trained model to reconstruct the WDN states. We can statistically measure the performance of the state estimation models under normal conditions, by repeating this process N times with a different mask. While this approach gives us a good indicator of model performance, the uncertainties intrinsic to real-life scenarios may significantly affect the results. Hence, instead of using the calibrated data as it is, Gaussian noise is added to the input parameters of the simulation before creating the testing data. Then, applying a different mask for every snapshot and repeating this process N times allows us to measure the performance of the state estimation models under uncertain conditions.

Finally, state estimation models should be designed with three important capabilities to allow DTs to fully address the challenges inherent to their physical counterparts: *generalizability*, *adaptability*, and *robustness* [33]. *Generalizability* equips the models with the ability to make predictions for any WDN topology, even on a completely new and unseen network during the model training phase. *Adaptability* refers to the ability of the model to maintain its predictive performance under unexpected changes. The changes here are associated with the sensors' locations. Sensors can be added or removed for maintenance, planned extension of the network, or sensors' malfunction. The state estimation models should be able to deliver the same performance in the presence of these changes. *Robustness* to changes in the data. In real-life scenarios, the observations can change due to unexpected circumstances, e.g., unexpected changes in demand due to COVID-19, discrepancies between simulated and real data, and noisy data due to sensors' malfunction. The models should be robust to these changes and keep the performance under unexpected conditions.

Rule Learning. It is the process of learning commonalities in a given dataset in the form of logical statements. Rules represent *'expected'* working conditions of the physical system. One specific type of rule learning that we focus on is learning associations in implication form such as $X \rightarrow Y$, meaning *'if X, then Y'*. The *antecedent* X refers to a set of statements, and the *consequent* Y refers to a single statement. In the case of DTs, rules can be learned from both time series data and KGs at the same time [17, 18], hence, the literals are based on semantic properties in the KG as well as time series data.

An example rule learned from time series data only in the WDN domain is as follows: *'if sensor1 measures a value in range R, then sensor2 must measure a value in range R2'*. With semantic properties from the KG, more generalizable and explainable rules can be learned: *'if a water flow sensor placed in a pipe P with roughness > A1 measures a value in range R, then a water pressure sensor placed in a junction J connected to P measures a value in range R2'*. The rules in the first form are specific to *sensor1* and *sensor2*. The rules in the second form no longer correspond to individual sensors and are more generically applicable and explainable due to their semantic properties. Rules also allow for easy integration of the domain knowledge into the system via KGs.

Change Detection module utilizes semantic association rules to detect changes in the physical system that are not yet reflected in the DT. It constructs subsets of rules which are referred to as *'hypotheses'* that can point out certain types of changes in the system such as leak detection, malfunctioning components, or architectural changes such as adding or removing components. This is done by checking whether the rules in the hypothesis hold for a certain period of time. An example hypothesis is: *'if the pressure level rules R1-R3 that correspond to a junction J do not hold for an hour, then there may be a leakage in J'*. Expert input can also be incorporated to mark detected changes as valid or not. The validity rules and hypotheses are checked regularly as new data is received.

What-If Scenarios This component tries to answer the hypothetical questions of "What if *something* was different" [10]. It can be a part of the environment that is controllable by operators, or an uncontrollable factor like weather. It can be a change in the data distribution, signifying concept drift, e.g. a considerable average pressure level drop in a pipe junction. It can also be a change in the ML model, its configurations and hyper-parameters. What-if scenarios are generated to answer these questions either automatically or by experts in the loop. They are executed in ephemeral and isolated software environments[4], ephemeral for resource utilization purposes on the hardware infrastructure and isolated to allow proper experimentation without interference from other executions. The whole process is an optimization problem to find an optimal setting for the variables to utilize the objective function.

4.3 Decision making

Adaptive Model Selection. Both internal and external factors of our DT system are continuously changing, the real world may experience data or concept drift. New ML models are developed to handle data processing tasks, but it is not always clear which model performs better on which data. This component addresses the problem of selecting the optimal model for executing a specific task on specific data. The system collects information on the models' performance to the knowledge base. This information can be the result of executions of what-if scenarios, or running the models in the production environment. From the knowledge base we can extract the performance measures of models on different characteristics of data. Using this information, our system continuously evaluates the current models' performance and incoming data to adapt to changes. As part of the realization of our solution, we designed the KEEP algorithm to (Keep Errors down with Enhanced Persistence) [11]. It is a heuristic algorithm that assesses models' performance by examining a window of the K most recent data points, predicated on the understanding that the current situation is closely correlated with these points. We adopt a window approach to leverage the benefits of averaging mechanisms in oscillations between models. KEEP

[4] `https://ephemeralenvironments.io/`

compares the performance gap between the optimal model and the one currently in production, initiating a switch only when the gap exceeds a specific threshold.

Orchestration component is responsible for orchestrating the DT and PT environments, and the system's components. Based on the decisions from the adaptive model selection, it orchestrates the software models in the DT environment through cloud computing technologies like Kubernetes. Moreover, it implements control actions in the physical environment, either directly or indirectly through human intervention. Finally, it maintains the system components to uphold the service level agreements.

Alerts and Dashboarding are essential for the DT systems, enabling real-time monitoring from data entry to model outputs and triggering notifications for users or system action on detected issues. Integrated tools for alert management and data visualization ensure rapid issue identification and understanding of the system's status. This functionality ensures prompt responses to concerns and informed decision-making for system management.

5 Validation: WDN Digital Twin in Oosterbeek

Our DT model is validated on a WDN in Oosterbeek, a place in the east of the Netherlands, province Gelderland. First, we describe Oosterbeek in general, then describe in detail our implementation of the main DT modules for its WDN.

5.1 Oosterbeek's WDN description

The Oosterbeek's WDN is one of the many balance area managed by Vitens, the largest water company in the Netherlands. In 2022, Vitens supplied 1.89 million m^3 of drinking water in this area to approximately 30,200 residents. The Oosterbeek area is largely rural and wooded. It is situated north of the Lower Rhine River and west of Arnhem. The height varies from 9 meters above Amsterdam Ordnance Datum (AOD), close to the river, to 76 meters above AOD, further north. The highest water connection is at 67 meters above AOD. Most of the population lives in the villages of Oosterbeek (11,000 inhabitants), Renkum (9,100), Doorwerth (4,900), Heelsum (3,500) and Wolfheze (1,700). On the Lower Rhine, there is a paper mill as the largest water user in the region. The water is mainly produced by the production facility (PF) "Oosterbeek", with a maximum capacity of 11,856 m^3. This PF also has a supply function from its reservoir to parts of Arnhem. It has an onward delivery of the water originating from the PF "Fikkesdries", on the south side of the Lower Rhine. "Fikkesdries" is also a backup in the event of a disruption in the PF "Oosterbeek". The PF "Wageningseberg" is located just west of the area and can supply water during emergencies. The PF "La Cabine", north of Oosterbeek, supplies 10 to 25 m^3 per hour of drinking water to the area. In the network, pressure reducers are present to manage pressure in low-lying areas. The area has a reservoir of 13,322 m^3.

The natural landscape of the Oosterbeek region is sensitive to drought. A large population increase is not expected in the area, but longer periods of extreme weather, such as drought, can increase annual water consumption. It is expected that water resources in the deep subsurface will decrease[5]. All water PFs can have less water as a backup. If surplus water from the surrounding areas is needed in the future, water has to be pumped from far away. Energy consumption will increase. That is an undesirable development of the water company's climate footprint. To diminish water and energy consumption, importance is attached to insights into the water distribution in the balance area, with focus on water which is produced, but not delivered to the customers. The aim is to minimize disruptive and latent leaks and pumping water unnecessarily.

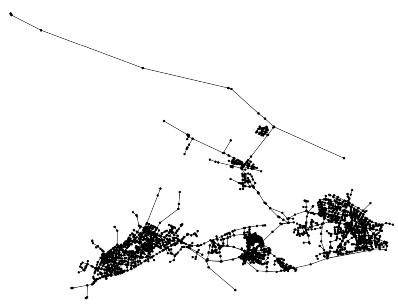

Fig. 2: Oosterbeek's water network graph

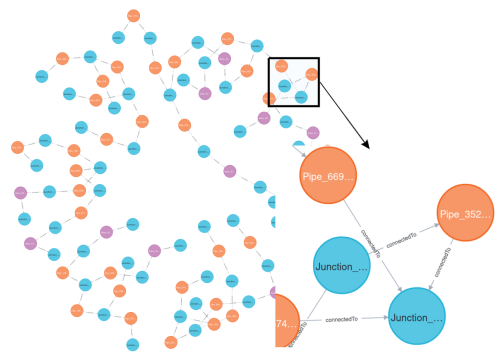

Fig. 3: Partial knowledge graph of Oosterbeek WDN.

In modelling, Oosterbeek WDN, shown in Figure 2, consists of 5,855 nodes and 6,188 links. These nodes comprise 5 reservoirs supplying drinking water to 5,850 junctions, representing residential and industrial zones. The network also includes 5,157 pipes and 1,031 valves as connection links. Each component derives its attributes from either internal infrastructure data, Asset Description, or external sources. For instance, nodal elevation data is sourced from public geographic information systems like AHN and USGS[6]. Dynamic properties, such as customer demand patterns, are calculated by using average water consumption patterns, which are adjusted based on local characteristics. Large industrial consumers have meter readings and their patterns are added to the model individually. Due to infrastructure constraints, 16 unique demand patterns are shared across individual junctions, albeit with varying scaling factors to maintain the diversity. Lastly, 9 sensors, equipped throughout Oosterbeek, provide crucial insights into pressure and flow information, essential for calibrating the virtual replica before deployment in downstream applications.

[5] https://publications.deltares.nl/11209219_hoofdrapport.pdf
[6] www.ahn.nl/hoogtegegevens, www.usgs.gov

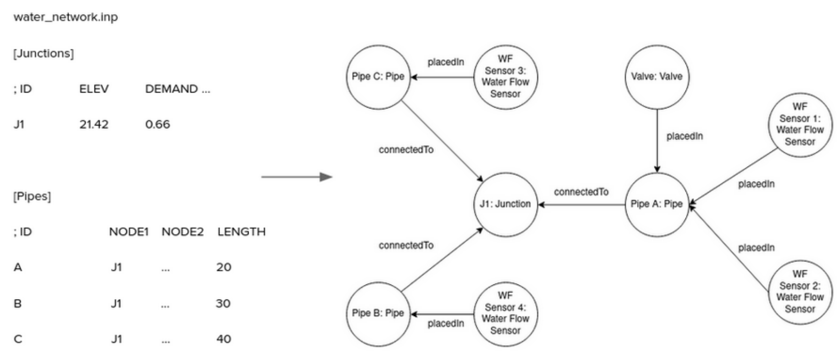

Fig. 4: An example knowledge graph construction process for WDNs.

5.2 Data Preparation

Rule learning, change detection, and what-if scenarios use semantic information provided by the KG. KG construction uses the domain ontology and asset descriptions. At the moment, there is no standard domain ontology for WDNs. A Large Language Models (LLMs)-based methodology to develop a WDN ontology is in progress. Research shows that LLMs can automatize one or multiple steps of ontology construction [2] processes which reduce the required manual effort and potentially make ontologies more accessible in various domains. Besides a domain ontology, data schemas can be used in KG construction [30]. For Oosterbeek, we used EPYNET's[7] class hierarchy to construct a KG, as shown in Figure 4. Asset descriptions, given inside the INP file, are parsed using WNTR [22]. A node is created per asset, and nodes are linked as described in the asset descriptions, in this case, pipes connect other assets. Lastly, a node is created per sensor and attached to the corresponding node in which the sensor is placed. Part of the Oosterbeek KG is given in Figure 3. The KG is a property graph and stored inside a Neo4j[8] database. The red nodes represent junctions, light brown nodes represent pipes, purple nodes represent valves, and blue nodes represent sensors in the KG. In total, there are over 12.000 nodes in the Oosterbeek KG.

5.3 Learning and Reasoning

State Estimation. Implementing a DL data-driven model, particularly for state estimation on WDNs, demands to address three critical factors: input modelling, model architecture, and evaluation protocol. A good input representation can accelerate the model training and benefit the inference when encountering unseen data points. When selecting a feasible model architecture that fits the state estimation problem, numerous families of DL models exist, such as convolution, recurrent, and self-attention networks. But our study opts

[7] https://github.com/Vitens/epynet
[8] https://neo4j.com/

Table 1: Model comparison in 24-hour Oosterbeek WDN with a 95% masking rate, adopted from [33].

Model	#Milion Params (↓)	MAE(↓)	MAPE(↓)	NSE(↑)	Acc(@0.1)(↑)
GCNii [7]	0.65	6.357±0.0197	0.2147±0.0008	-0.0137±0.0061	38.48±0.1351
GAT [34]	0.35	3.726±0.0120	0.1287±0.0008	0.3276±0.0037	73.52±0.0900
GraphConvWat [12]	0.92	3.067±0.0077	0.1160±0.0004	0.6938±0.0020	69.92±0.1205
GraphConvWat-tuned	**0.23**	2.293±0.0087	0.0821±0.0005	0.7518±0.0024	83.03±0.1025
mGCN [1]	2.48	2.111±0.0085	0.0806±0.0003	0.7100±0.0030	84.05±0.0693
GATRes-small (ours)	0.66	**1.937**±0.0074	**0.0703**±0.0005	0.7773±0.0025	**87.48**±0.0761
GATRes-large (ours)	1.67	2.020±0.0132	0.0711±0.0003	**0.7864**±0.0031	84.33±0.1347

for Graph Neural Network (GNN) due to the inherent focus on graph data which naturally expresses the structure of water networks. Model evaluation needs to be explicit and well-designed, so we introduce a testing approach for models trained largely or entirely on a synthetic dataset.

In input modelling, the state of a WDN at any timestep is represented as an undirected graph, whose nodes denote reservoirs, junctions, or tanks, whilst edges indicate links, pumps, or valves. In the scope of this study, we focus on solving the univariate state estimation task and the target nodal feature is pressure. Nevertheless, the framework can be extended to another measure (e.g., demand or pressure head) or estimate multiple variables simultaneously.

After conversion, a dataset containing numerous network states represented as graphs can be input into a DL-based model. Yet, even with all available sensor data samples from existing WDNs, such an amount is still insufficient for training a high-quality deep model. To mitigate the problem, physical simulation is employed to generate synthetic states, with a crucial assumption that these states are obtained in supposedly common situations. Specifically, we alter potential dynamic parameters with diverse configurations beyond the capability of a conventional model, still within the allowed boundary. This approach alleviates the issue of data scarcity in training deep models and diversifies the dataset.

Having acquired training data from the previous step, we establish a semi-supervised training scheme for the deep model. The input nodal feature (pressure), is disturbed by hiding several nodes with a masking ratio up to 95%. The deep model leverages the remaining available nodes (sensors) to estimate the pressure values of masked nodes. Errors derived from predicted and ground truth, guide the model to update its weights. For this, we employ the GATRes [33], a specialized Graph Autoencoder model tailored for state estimation in WDNs.

GATRes and other GNN-based models are evaluated on data from the Oosterbeek WDN recorded at 5-minute intervals over a 24-hour period, with only 5% of observable sensor nodes. Results from physical simulations served as ground truth for this experiment. GATRes and the baseline models require only a few sensors and the network topology to estimate the pressure state of the entire WDN, while a conventional simulation need not only sensor values but also numerous relevant parameters to output a similar result. The used metrics include mean relative percentage error (MAPE), mean absolute error (MAE), Nash-Scliffed efficiency (NSE), and a customized accuracy determined by the ratio of positive predictions within a defined threshold δ ($\delta = 10\%$) from the true values (Acc@0.1). As illustrated in Table 1, GATRes consistently outperforms other GNNs across all metrics, demonstrating its superior performance.

Rule Learning A DL-based rule learning algorithm [18] is run on Oosterbeek WDN. The algorithm consists of a pipeline of operations illustrated in Figure 5. It uses the Oosterbeek KG, time series sensor data, and a binding that tells the algorithm how the sensors are represented in the KG. The output is a set of logical association rules as first-order horn clauses.

The first step of the algorithm is to enrich time series sensor data with semantic properties from the graph. The properties can be based on the item that the sensor is placed in, as well as its neighbors. Next, semantically enriched time series data is transformed into vectors by applying one-hot encoding. The obtained set of vectors is then passed to an under-complete denoising autoencoder (AE) [35] for training. A neural representation of the input data is created after the training process. The final step is to extract associations between input features of the trained AE. This is done by creating a set of *test vectors* which are essentially vectors of the same length as used in the training with marked features. Suppose a forward run on the trained AE with marked test vectors results in the successful reconstruction of other features. In that case, we say that the marked features imply successfully reconstructed features.

Example. Let there be 2 features $pipe\_diameter = \{a, b\}$ and $water\_flow = \{c, d, e\}$, referring to the diameter of a pipe and the water flow in the same pipe, with $\{a, b, c, d, e\}$ being numerical intervals such as $a = [10, 20)$. To test whether $pipe\_diameter$ being a implies any of the values of the $water\_flow$ features, a test vector $t = [1, 0, 0.33, 0.33, 0, 33]$ with 100% probability for $pipe\_diameter(a)$, 0% probability for $pipe\_diameter(b)$, and equal 0.33 probabilities for the $water\_flow$ values is created. Assume that a similarity threshold of 80% is preset, and a forward run on a trained AE with the created test vector t produced $[0.94, 0.06, 0.03, 0.1, 0.87]$ vector. Since the value that corresponds to $water\_flow(e)$ (0.87) is bigger than the preset threshold, we conclude that $pipe\_diameter(a) \rightarrow water\_flow(e)$.

A sample learned semantic association rule learned from the actual Oosterbeek WDN looks as follows: *'if a water pressure in a reservoir with a single pipe connected to it is in between 0-54 water column meter, then the water flow the connected pipe must be 0-25 CMH.'*

The state-of-the-art association rule mining (ARM) suffers from big high-dimensional data [20], which can be the case in a DT as we utilize both sensor data and semantics from KGs for rule learning. Table 2 shows a concise rule quality (support, confidence, lift, leverage and Zhang's metric, further explained in [18]) and execution time comparison of our approach with an optimizaton-based [20] ARM algorithm named Harris' Hawks Optimization (HHO) [16] and an exhaustive ARM algorithm named FP-Growth [13]. The results show that the

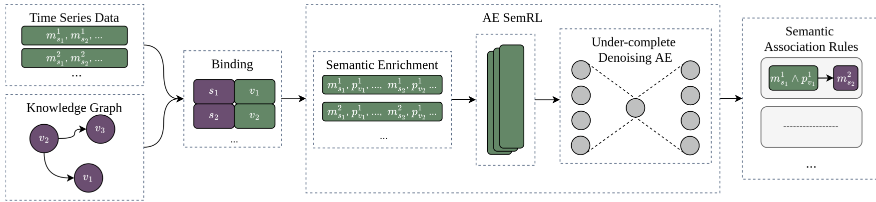

Fig. 5: AE-based rule learning pipeline from [18].

AE-based approach is capable of learning high-quality association rules hundreds of times faster than the state-of-the-art ARM [18].

Change Detection The change detection module is currently under development and this section summarizes the primary idea. We observed that not all the semantic association rules learned from the Oosterbeek use case are directly usable in practice for detecting unforeseen events. As an example, water flow sensors are only placed inside Pipes, and in the semantic association rules, we see that every range of water flow measurements is associated with the node-type Pipe. However, this does not provide valuable information for detecting changes in the system. Therefore, the first step is to eliminate obvious associations from the learned set of rules. Second, inferring semantic properties is not useful for change detection. For instance, *'a water pipe with a certain flow rate must have the diameter X'* would not be an actionable rule. The second step is to filter out rules with semantic properties in the consequent side of a rule and keep only the sensor measurements in the consequent.

After the initial filtering, there are two major research directions to investigate. The first is to check for correlations between subsets of rules and historical changes such as past pipe leakages. Similar to the AE-based approach, this can be done by training a neural network with the learned set of rules with historical changes and then looking for associations between them. The second is to find similarities between semantic association rules and grouping them together. After that, the goal is to link grouped rules to the presence or absence of certain types of changes in a certain part of a WDN.

What-If Scenarios are used to tweak the WDN parameters to find an optimal setting. We design scenarios to answer questions like "What if we open/close

Table 2: A rule quality comparison of the AE-based approach (max_antecedent=1, similarity_threshold=0.8), FP-Growth (min_sup=0.1, min_conf=0.8) and optimization-based HHO (init_population=100, max_evaluations=50000) algorithms.

Algorithm	Number of Rules	Execution Time (s)	Support	Confidence	Lift	Leverage	Zhang's Metric
FP-Growth	4130	107.863	0.229	0.95	1.96	0.551	0.066
HHO	21596	143	0.084	0.852	1.013	0.002	0
AE-based (ours)	774	0.427	0.4	0.911	1	0	0.435

a valve?" or "What if we change a pump pressure settings?". Then based on the results of these experiments, domain experts can perform educated actions. In addition, we utilize these scenarios to answer questions like "What if another model was in the production software environment?" or "What if we had executed this model on historical data?" This approach helps to get a better understanding of our models by contributing to the knowledge base. Thereby, it enhances the decision of the adaptive model selection component.

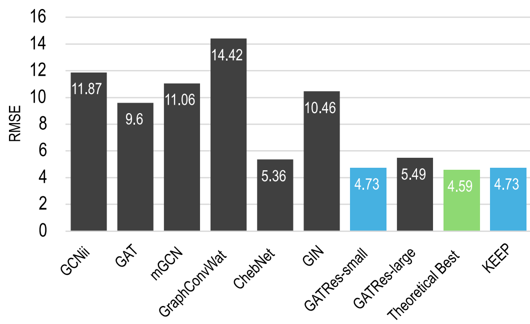

Fig. 6: Models, KEEP, and optimal performances comparison.

Policy	RMSE
Always Latest	9.13
KEEP	6.18
Theoretical Best	5.98

Table 3: Evaluating selection strategies for progressive model integration.

5.4 Decision Making

Adaptive Model Selection. We conducted two sets of experiments based on a state estimation example for a 1-year scenario, during which we collected input and predicted the state every hour, resulting in 8,760 data points. The first involved having all models operational from the outset, while the second saw models being progressively developed and integrated into the system over a year. Figure 6 illustrates the Root Mean Square Error (RMSE) for the pool of eight models, the best theoretically achievable selection result, and the outcome of employing KEEP algorithm. GATRes-small emerges as the best-performing static model on average. In our case, KEEP reaches the same result. The theoretical best represents the optimal result a selection algorithm could achieve by selecting the best model for each data point, which in this case is marginally better (3%). Even though the difference is minor, we might not always know which of these eight models is superior right from the start. For example, we might have chosen GATRes-large or ChebNet, which could have resulted in approximately a 14% deficit. Our KEEP selection algorithm is a solution to this issue by achieving a result comparable to that of the best static model.

In a realistic scenario, we start with one model and continuously develop new ones, e.g. GATRes models stem from considerable effort and numerous attempts to refine them. To simulate this, we evenly distributed the development of a new model across every 1.5 months. We conducted our experiments on a random sample comprising 30% of all $8! = 40,320$ possible combinations of the models' order of introduction. Table 3 presents the outcomes of this experiment, which, in addition to KEEP and Theoretical Best, includes Always Latest selection policy, based on the established assumption that newer models are better. The results reveal a 35% improvement margin. Our KEEP selection algorithm successfully narrows this gap, coming within 3% of the best theoretical selection outcome.

6 Discussion

We presented the architecture of the DiTEC (Digital Twin for Evolutionary Changes in Water Networks) system, and discussed each module in detail. Main data sources to the DT are asset descriptions, incorporating domain knowledge via knowledge graphs, and real-time sensor data. We discussed the main learning and reasoning modules, including the conventional physical simulations, semantic rule learning to extract the patterns of the system's behavior, change detection methods and what-if scenarios to test different possibilities. The central state estimation module can use the information from these diverse sources to reconstruct the current system's state, even when sensor data is limited and large parts of the system are unobservable, and the physical simulation presents differences with the real-world due to unknown changed environmental conditions. The reasoning results are used to adaptively select the best available reasoning model, and execute control actions though orchestration module. Human-in-the-loop involvement is essential, so real-time alerts and full dashboarding of the system's state is an integral part of the presented DT. Our DT architecture is validated on the real use case of the Oosterbeek region, and it is shown that such an approach to creation of DTs can bring considerable benefits compared to the traditional DT models. Traditional DTs are more reliant on the pre-defined behavior domain knowledge, as opposed to learning it via semantic rule learning, and are less capable of withstanding a change due to being designed for the exact description of the world. Moreover, the ability of the GATres neural network to combine training on the real historical data with the diverse physical simulations allows learning on a wide range of conditions which provides precise state estimation results. Training with the addition of simulated data also means that confidential customer data remains protected and that errors in the database across the area have less impact on the final performance of the system. DiTEC is especially useful for operational simulations, where a lot of (training) data is available and many varying conditions require flexibility and robustness. Traditional DTs will continue to be useful in, for example, strategic considerations. They can calculate variants for the long-term water distribution prospects and investments to be made. Ultimately, adaptive algorithm selection can ensure a seamless transition from one twin to the other, as they both model different aspects of the same balance area which can have more value integrally.

Logical rules, especially with semantic properties, can provide valuable insights into how a PT works. Besides more classical approaches such as data mining and optimization-based methods, rules can also be extracted from neural representations of a given input data. Combined with semantics, rules learned from time series sensor data in DTs represent domain knowledge for which the

DT is created and can then be used to perform various tasks. There are 3 advantages of working with rules in DTs. The first is that Rules are explainable, which is particularly important when making high-stake automated decisions on the PT. Making inferences with rules is a simple comparison operation over the unseen data since they act as a lookup table. In DTs with a high number of data sources, rules can be highly efficient in detecting changes in the system. Incorporating domain knowledge into the rules, so in the inference phase, is straightforward. We have shown, domain knowledge can be represented as part of a KG that is based on a domain ontology, making both learning and inference phases to benefit from already existing domain knowledge.

The potential addition to the DT are *foundation models*, which are trained in an unsupervised fashion on broad large-scale data and adapted to solve a myriad of downstream specific tasks [4]. They gained momentum with the advent of LLMs, e.g., GPT-4, Gemini, and Llama 3. While their unprecedented success has been mainly exploited in the fields of natural language processing and computer vision, foundation models for WDNs are achievable by tailoring the existing technologies to the water domain. One starting point can be pre-training the GNN models to learn node degree distributions or centrality metrics based on shortest paths in the network. GNNs have shown generalization capabilities when trained on such tasks. Pre-trained foundation models can then be fine-tuned to solve a wide range of domain-specific tasks such as state estimation, demand forecasting, and leakage detection, among other problems.

7 Conclusions

In this paper we have introduced the DiTEC system and discussed the importance of engineering digital twins in a way that can withstand changes in the environment. We have shown the three stages of the DT framework: data preparation, learning and reasoning and decision making, and discussed the main capabilities and modules that each layer should have. We discussed automated domain-specific behavior extraction, achievable with semantic rule learning, dynamic GNN-based state estimation from minimal sensor data, and shown transitioning between models over time with adaptive model selection. Validation of the DiTEC system on the Oosterbeek region have proven its applicability in real-case scenarios. While the DiTEC system is engineered for the WDN systems, all of its main modules are applicable to general critical infrastructures and beyond, as specific domain knowledge is tightly incorporated into the data sources, such as asset description and the ontology for the KG creation. The DiTEC system presents significant advancements in the field of digital twins with its robust change-adaptation framework, ensuring long-term resilience and efficiency of the critical infrastructures management. Future work will focus on expanding and validating its applicability in diverse domains.

Acknowledgments. This work is funded by the project DiTEC: Digital Twin for Evolutionary Changes in Water Networks (NWO 19454).

References

1. Ashraf, I., Hermes, L., Artelt, A., Hammer, B.: Spatial graph convolution neural networks for water distribution systems. In: International Symposium on Intelligent Data Analysis. pp. 29–41. Springer (2023)
2. Babaei Giglou, H., D'Souza, J., Auer, S.: Llms4ol: Large language models for ontology learning. In: International Semantic Web Conference. pp. 408–427. Springer (2023)
3. Bayram, F., Ahmed, B.S., Kassler, A.: From concept drift to model degradation: An overview on performance-aware drift detectors. Knowledge-Based Systems **245**, 108632 (2022). `https://doi.org/10.1016/j.knosys.2022.108632`
4. Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., et al.: On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258 (2021)
5. Bretas, A., Bretas, N., London Jr, J.B., Carvalho, B.: Cyber-physical power systems state estimation. Elsevier (2021)
6. Brucherseifer, E., Winter, H., Mentges, A., Mühlhäuser, M., Hellmann, M.: Digital twin conceptual framework for improving critical infrastructure resilience. at-Automatisierungstechnik **69**(12), 1062–1080 (2021)
7. Chen, M., Wei, Z., Huang, Z., Ding, B., Li, Y.: Simple and deep graph convolutional networks. Machine Learning Research, vol. 119, pp. 1725–1735. PMLR (2020)
8. Conejos Fuertes, P., Martínez Alzamora, F., Hervás Carot, M., Alonso Campos, J.: Building and exploiting a digital twin for the management of drinking water distribution networks. Urban Water Journal **17**(8), 704–713 (2020)
9. Gai, K., Xiao, Q., Qiu, M., Zhang, G., Chen, J., Wei, Y., Zhang, Y.: Digital twin-enabled ai enhancement in smart critical infrastructures for 5g. ACM Transactions on Sensor Networks (TOSN) **18**(3), 1–20 (2022)
10. Hadadian Nejad Yousefi, M., Degeler, V., Lazovik, A.: Empowering machine learning development with service-oriented computing principles. Service-Oriented Computing pp. 24–44 (2023)
11. Hadadian Nejad Yousefi, M., Degeler, V., Lazovik, A.: Self-adaptive service selection for machine learning continuous delivery. In: 2024 IEEE International Conference on Web Services (ICWS) (2024)
12. Hajgató, G., Gyires-Tóth, B., Paál, G.: Reconstructing nodal pressures in water distribution systems with graph neural networks. preprint arXiv:2104.13619 (2021)
13. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. ACM sigmod record **29**(2), 1–12 (2000)
14. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: Proc. the IEEE-CVF conference on computer vision and pattern recognition. pp. 16000–16009 (2022)
15. He, L., Wen, K., Gong, J., Wu, C.: A multi-model ensemble digital twin solution for real-time unsteady flow state estimation of a pumping station. ISA Transactions **126**, 242–253 (2022). `https://doi.org/10.1016/j.isatra.2021.08.021`
16. Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H.: Harris hawks optimization: Algorithm and applications. Future generation computer systems **97**, 849–872 (2019)
17. Karabulut, E., Degeler, V., Groth, P.: Semantic association rule learning from time series data and knowledge graphs. In: Proc. Int. Workshop on Semantic Industrial Information Modelling (SemIIM). pp. 1–7 (2023)

18. Karabulut, E., Degeler, V., Groth, P.: Ae semrl: Learning semantic association rules with autoencoders. arXiv preprint arXiv:2403.18133 (2024)
19. Karabulut, E., Pileggi, S.F., Groth, P., Degeler, V.: Ontologies in digital twins: A systematic literature review. Future Generation Computer Systems **153**, 442–456 (2024). https://doi.org/10.1016/j.future.2023.12.013
20. Kaushik, M., Sharma, R., Fister Jr, I., Draheim, D.: Numerical association rule mining: a systematic literature review. arXiv preprint arXiv:2307.00662 (2023)
21. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
22. Klise, K.A., Murray, R., Haxton, T.: An overview of the water network tool for resilience (wntr). (2018)
23. Lai, J., Chen, Z., Zhu, J., Ma, W., Gan, L., Xie, S., Li, G.: Deep learning based traffic prediction method for digital twin network. Cognitive Computation **15**(5), 1748–1766 (2023)
24. Masi, M., Sellitto, G.P., Aranha, H., Pavleska, T.: Securing critical infrastructures with a cybersecurity digital twin. Software and Systems Modeling **22**(2), 689–707 (2023)
25. Pregnolato, M., Gunner, S., Voyagaki, E., De Risi, R., Carhart, N., Gavriel, G., Tully, P., Tryfonas, T., Macdonald, J., Taylor, C.: Towards civil engineering 4.0: Concept, workflow and application of digital twins for existing infrastructure. Automation in Construction **141**, 104421 (2022)
26. Qin, Y., Arunan, A., Yuen, C.: Digital twin for real-time li-ion battery state of health estimation with partially discharged cycling data. IEEE Transactions on Industrial Informatics **19**(5), 7247–7257 (2023)
27. Ramos, H.M., Morani, M.C., Carravetta, A., Fecarrotta, O., Adeyeye, K., López-Jiménez, P.A., Pérez-Sánchez, M.: New challenges towards smart systems' efficiency by digital twin in water distribution networks. Water **14**(8), 1304 (2022)
28. Rice, J.R.: The algorithm selection problem. In: Advances in computers, vol. 15, pp. 65–118. Elsevier (1976)
29. Rossman, L.A., et al.: Epanet 2: users manual (2000)
30. Tamašauskaitė, G., Groth, P.: Defining a knowledge graph development process through a systematic review. ACM Transactions on Software Engineering and Methodology **32**(1), 1–40 (2023)
31. Tello, A., Degeler, V.: Digital Twins: An enabler for digital transformation. In: The Digital Transformation handbook. Groningen Digital Business Centre (GDBC) (2022). https://doi.org/10.5281/zenodo.7647493
32. Tello, A., Truong, H., Lazovik, A., Degeler, V.: Large-scale multipurpose benchmark datasets for assessing data-driven deep learning approaches for water distribution networks. arXiv preprint arXiv:2404.15386 (2024)
33. Truong, H., Tello, A., Lazovik, A., Degeler, V.: Graph neural networks for pressure estimation in water distribution systems. Water Resources Research **60**(7), e2023WR036741 (2024). https://doi.org/10.1029/2023WR036741
34. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph Attention Networks. International Conference on Learning Representations (2018)
35. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th international conference on Machine learning. pp. 1096–1103 (2008)
36. Zhao, D., Zhou, Z., Hung, P.C.K., Deng, S., Xue, X., Gaaloul, W.: Ctl-based adaptive service composition in edge networks. IEEE Transactions on Services Computing **16**(2), 1051–1065 (2023)