Graph Neural Networks for Pressure Estimation in Water Distribution Systems

Huy Truong[1]*, Andrés Tello[1]*, Alexander Lazovik[1], and Victoria Degeler[2]

[1]Bernoulli Institute, University of Groningen, Groningen, The Netherlands
[2]Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands

Key Points:

- Mathematical Simulation Tools and Graph Neural Networks were combined for pressure estimation in Water Distribution Networks.
- Random sensor placement during model training is a good strategy for robustness against unexpected sensors' location changes.
- Time-dependent patterns and Gaussian noise injection enable a realistic evaluation protocol for pressure estimation models.

*Both authors contributed equally to this work

Corresponding author: Andrés Tello, andres.tello@rug.nl

Corresponding author: Huy Truong, h.c.truong@rug.nl

Abstract

Pressure and flow estimation in Water Distribution Networks (WDNs) allows water management companies to optimize their control operations. For many years, mathematical simulation tools have been the most common approach to reconstructing an estimate of the WDNs hydraulics. However, pure physics-based simulations involve several challenges, e.g. partially observable data, high uncertainty, and extensive manual calibration. Thus, data-driven approaches have gained traction to overcome such limitations. In this work, we combine physics-based modeling and Graph Neural Networks (GNN), a data-driven approach, to address the pressure estimation problem. Our work has two main contributions. First, a training strategy that relies on random sensor placement making our GNN-based estimation model robust to unexpected sensor location changes. Second, a realistic evaluation protocol that considers real temporal patterns and noise injection to mimic the uncertainties intrinsic to real-world scenarios. As a result, a new state-of-the-art model, GATRes, for pressure estimation is available. Our model surpasses the performance of previous studies on several WDNs benchmarks, showing a reduction of absolute error of $\approx 40\%$ on average.

Plain Language Summary

Water management practitioners have resorted to mathematical simulation tools to reconstruct pressure, flow, and demand in order to improve their control operations. However, pure physics-based methods need to deal with partially observable data, high uncertainty, and extensive manual calibration. We combine physics-based modeling and Graph Neural Networks, a data-driven approach, to address the pressure estimation problem and overcome those limitations. Our work has two main contributions. First, a random sensor placement strategy makes our estimation model resilient to unexpected sensor location changes. Second, a realistic evaluation protocol that considers real temporal patterns and noise injection to mimic the uncertainties of real-world scenarios. As a result, a new state-of-the-art model, GATRes, for pressure estimation is available. Our model surpasses the performance of previous studies on several WDNs benchmarks, showing a reduction of absolute error of $\approx 40\%$ on average.

1 Introduction

State Estimation in Water Distribution Networks (WDNs) is a general problem that encompasses pressure and flow estimation, often using scarce and sparsely located sensor devices. WDNs management companies rely on such estimations for optimizing their operations. Knowing the state of the network at any given time enables water managers to perform real-time monitoring and control operations. The research community and practitioners working in this field have resorted for many years to the power of mathematical simulation tools to reconstruct an estimate of the system hydraulics (Kang & Lansey, 2009; Arsene & Gabrys, 2014; Todini et al., 2021; Menapace et al., 2018; Koşucu et al., 2022; Ruiz et al., 2022). However, pure physics-based simulation approaches have to overcome the challenges of (i) data scarcity which translates to partially observable systems, (ii) high uncertainty introduced by the large number of parameters to configure, unexpected changes in consumers' behavior reflected in uncertain demand patterns, and noisy sensor measurements, and (iii) extensive manual configuration for model calibration using metered data, which requires expert knowledge and usually hinders model re-usability in a different WDN (Wang et al., 2021; Ostfeld et al., 2012). The challenges associated with physics-based modeling of WDNs have motivated researchers to investigate the usage of data-driven approaches, or a combination of both, to address the state estimation problem (Meirelles et al., 2017; Lima et al., 2018).

Graph Neural Networks (GNNs) is a data-driven approach that has shown successful results in several estimation problems where data can be modeled as a graph. Since

WDNs can be naturally modeled as a graph, GNNs can exploit the relational inductive biases imposed by the graph topology. As a result, GNNs have also attracted the attention of researchers in the field of WDNs. For example, Tsiami and Makropoulos (2021) used temporal graph convolutional neural networks, a combination of Convolutional Neural Networks (CNNs) and GNNs, to extract temporal and spatial features simultaneously in a model to detect cyber-physical attacks in WDNs. Zanfei, Menapace, et al. (2022) used GNNs for implementing burst detection algorithms. GNNs are also used for integrated water network partitioning and dynamic district metered areas (Fu et al., 2022). In the context of Digital Twins of WDNs, a GNN-based model is used for Pump Speed-Based State Estimation (Bonilla et al., 2022). Water demand forecasting has been also addressed using GNNs (Zanfei, Brentan, et al., 2022). Metamodeling is another interesting area where GNNs have been leveraged for the estimation of pressures and flows (Lima et al., 2018; Zanfei et al., 2023; Kerimov et al., 2023). GNN-based models has been also used to solve problems related to water quality. For example, Z. Li et al. (2024a) propose GNNs for identifying contamination sources in water distribution systems, and for water quality prediction (Z. Li et al., 2024b). Other recent works on GNNs for pressure estimation are (Hajgató et al., 2021; Ashraf et al., 2023). The main differences of those approaches with ours are the training strategy and the evaluation protocol to assess the model performance. Our proposed techniques in this regard are more realistic and give the model the ability to adapt to unexpected changes.

In this work, we focus on pressure estimation by leveraging both physics-based simulation models and GNN-based data-driven approaches. We rely on a data generation method that leverages the EPANET simulation tool to overcome the lack of data required for model training. However, in our approach we include all dynamic parameters (e.g., reservoir total heads, tank levels, roughness coefficient) which were not considered in previous works. This contributes to data variety and avoids that uncertainties propagate due to model simplification errors (Du et al., 2018). Our main contributions are twofold. First, our GNN-based estimation model is robust to unexpected sensor's location changes due to the proposed training strategy that relies on random sensor placement. Second, our evaluation protocol considers real time-dependent patterns and additionally injects the uncertainties intrinsic to real-world scenarios. The outcome is a new state-of-the-art GNN-based model, GATRes, for pressure estimation in WDNs.

GATRes is able to reconstruct the junction pressures of Oosterbeek, a large-scale WDN in the Netherlands, with an average 1.94m absolute error, which represents an 8.57% improvement with respect to other models. Similarly, our model outperformed previous approaches on other WDNs benchmark datasets. The highest improvement was seen for C-Town WDN (Ostfeld et al., 2012) with an absolute error decrease of 52.36%, for Richmond (Van Zyl, 2001) an error deacrease of 5.31%, and 40.35% error decrease for L-Town (Vrachimis et al., 2022). In addition, our first attempt on model generalization shows that a multi-graph pre-training followed by fine-tuning helps to increase the model performance. The absolute error on Oosterbeek network was reduced by $\approx 2\%$ following our generalization strategy.

The remainder of this document is as follows. Section 2 presents the problem statement, describes the issues that need to be addressed by pressure reconstruction models and defines the criteria to assess the model capabilities. Section 3 depicts the related work in the field, narrowed to GNNs for node-level regression tasks and how previous work on GNN-based pressure estimation satisfies the criteria defined in the Section 2. The methodology is presented in Section 4, including the data generation process, a detailed description of our model architecture, and the details of the proposed approach for model training and evaluation. Section 5 describes the setup of the experimental phase. It includes a description of WDNs benchmark datasets used in this work, the base model configurations, and the evaluation metrics. Section 6 describes all the empirical evaluations of our approach. First, the experiments on the main use case of this study, Oosterbeek WDN,

are depicted. Then, the experiments towards model generalization are shown. Next, the performance of the proposed model on different benchmark WDNs is presented. This section concludes with an ablation study to identify the contribution of the different components of the model architecture. A discussion of the most salient findings are presented in Section 7. Finally, the conclusions are presented in Section 8.

2 Pressure estimation in water distribution networks

2.1 Problem statement

Hydraulic experts have managed WDNs using essential measurements such as flow, demand, and pressure. These measurements offer a comprehensive perspective of a WDN, forming a foundation for various supervisory tasks like forecasting (Iwakin & Moazeni, 2024), leak detection(Garðarsson et al., 2022), and operational control(Nerantzis et al., 2020). For this reason, this study focuses on addressing the fundamental challenge of approximating measurements across all nodal locations within water networks. In this investigation, we initially assume that the prior knowledge (i.e., historical data) is unavailable and network states are discretely recorded under typical conditions, disregarding unforeseen events like leaks, earthquakes, or fires. Additionally, we presume that measured states between two neighborhoods within the network exhibit a degree of similarity. These assumptions are crucial for employing a data-driven machine learning model trained from valid cases while avoiding corruption in the complexities of water networks.

A real-life WDN consists of diverse components such as tanks, valves, pumps, reservoirs, and thousands of customer junctions whose measurement states are crucial for management. In this study, we narrow the scope and favor pressure as the primary measurement due to the ease of meter installation and the more affordable price compared to flow ones (Zhou et al., 2019a). Nevertheless, these pressure sensors are limited in practice due to infrastructural limits and privacy concerns. Consequently, the gathered data, known as the pressure states of the junction nodes, are scarce. These states play crucial roles as samples in training data-driven approaches aligned with a machine-learning model that often requires a large amount of data. As a prerequisite, pressure states should be fully observable to minimize the estimation loss of the trained model in all customer positions throughout the water network. This contradiction poses a challenge in applying the machine-learning approach to solve pressure estimation tasks.

The application context is about what and when the trained model should be applied. Generally, a model is often associated with a unique water network and fixed sensors previously seen during training. Also, the training environment may exclude noisy, uncertain conditions that could affect the model's decision-making. In other words, these challenges result in worse model performance when faced with unfamiliar network topologies or uncertain situations. Consequently, model retraining is inevitable, albeit such training is an expensive and unsustainable approach. This concern enhances the necessity of the generalization ability of pressure estimation models, which needs to be addressed in prior research. Before addressing this research gap, we will first delve into the specific problem within water networks and lay out the criteria necessary for a robust pressure estimation model.

2.2 Partially-Observable Data and Realistic Model Evaluation

Water Distribution Networks domain is characterized by partial-observability due to the limited sensor coverage. This imposes an additional challenge because the reconstruction models need to be trained on fully-observable network operation snapshots. The common approach to overcome this limitation is to rely on mathematical hydraulic simulation tools, e.g. EPANET (Rossman, 1999), to generate full-views of the network op-

eration and use them for model training (Hajgató et al., 2021; Xing & Sela, 2022; Ashraf et al., 2023; Zhou et al., 2023).

Although the hydraulic simulations solve the lack of training data for the reconstruction models, the remaining challenge is how to create a valid and reliable evaluation protocol and the data used for it. Sampling from the data generated from the simulation models and splitting them into training and test sets is not enough. Ideally, the assumption behind machine learning models is that the training data is ruled by the exact same distribution of the data on which the model will be evaluated. However, having absolute control over the data generation process and meeting such a perfect match between both distributions is unrealistic and the assumption is violated under real-world conditions (Bickel et al., 2007; Hendrycks & Gimpel, 2017; Fang et al., 2022). Thus, the prediction models should be robust to distribution shifts between training and testing samples, i.e., be able to generalize to out-of-distribution (OOD) data (Farquhar & Gal, 2022).

We observed that the data distribution of pressures in the training and test sets created by the simulation models are identical, which is unnatural in practice. The density distributions of pressures in the training and test sets from different WDNs, generated by the Hydraulic Simulation tool EPANET, are shown in Figure 1. In this case, the simulation's dynamic parameters, e.g. reservoir total heads, junction demands, pump speed, were randomly adjusted for every run. As we described before, we consider only WDN states under normal conditions. Thus, all static parameters, e.g., node elevation, pipe length, pipe diameter, etc. are fixed. Each simulation run produces an arbitrary state of the network (snapshot) given a particular combination of input parameter's values. The algorithm ran until 50,000 stable network states were reached. Then, the data were randomly partitioned into training, validation, and test set in a 60:20:20 ratio, respectively. The plots were generated taking 2,000 snapshots at random from the training and test sets, where each snapshot represent the pressures at all junctions. Nonetheless, as evident from the image, the distributions of the training and test sets created by the hydraulic simulations are identical in all examples. In this case, evaluating the reconstruction models on data generated by the same algorithm is simply evaluating the ability of the model to reconstruct the signals already seen during the training process.

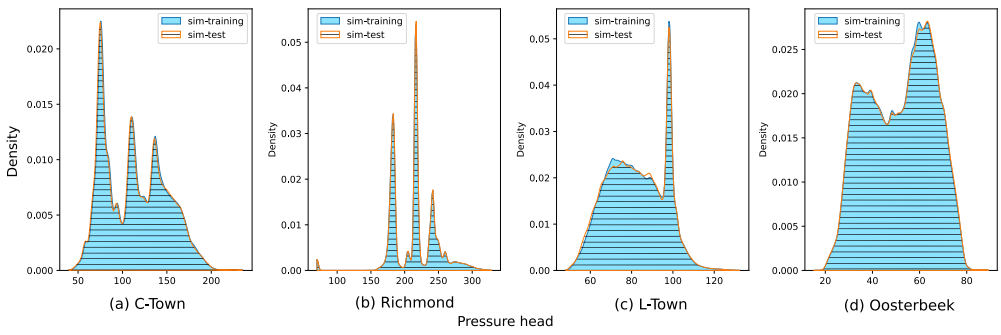

Figure 1. Density Distribution of training and test sets in C-Town, Richmond, L-Town and Oosterbeek WDNs, generated by the Hydraulic Simulation tool EPANET.

In our work we propose a realistic test set generation process that relies on time-based demand patterns. In addition, Gaussian noise is injected before the simulation to mimic the uncertainty intrinsic to real-world scenarios. Combining time-based demand patterns and noise injection allows to create realistic scenarios to evaluate the ability of

the models to generalize to OOD data, with visible differences in density distribution between training and test sets.

2.3 Criteria for model assessment

The out-of-distribution problem may originate from the uncertainty of measured sensors, the fluctuation in hydraulic parameters, and the diversity of water network topologies. Within the context of these factors, we delve into the limitations of physical models. First, the uncertainty from sensors is due to the gradual sensor coverage plan, maintenance period, and sensor malfunction. This uncertainty impacts the performance of physical models and often requires a human-involved recalibration process. Similarly, unforeseen fluctuations in hydraulic parameters can be triggered by changes in socioeconomic conditions, aging infrastructure, or unexpected events like pandemics. While existing techniques on top of the physical model, such as calibration and skeletonization, can mitigate these issues, they often require further human intervention. Additionally, encountering unseen water networks with different topologies inevitably leads to a redesign process that costs time and resources significantly. Furthermore, the physical model can only perform a proper simulation if all hydraulic parameters of all components in the new network are fully identified, posing a practical challenge and emphasizing the persistence of a generalization problem.

Some previous studies have proposed utilizing more efficient data-driven machine-learning models (discussed in Section 3) to tackle the challenges above. However, it is critical to note that these works have often overlooked some of these problems. This oversight essentially motivates us to have a list of criteria that indicate the favorable capabilities of a data-driven model addressing the pressure estimation task on WDNs. We then propose the criteria of generalizability, adaptability, and robustness as follows.

($\mathbb{C}$1) Generalizability: The model is able to perform the pressure estimation task across diverse WDNs, regardless of their topology, even when encountering networks not seen during training. This is an important aspect of generalizability, making it more useful in practice.

($\mathbb{C}$2) Adaptability: The model should efficiently maintain its estimation capabilities in diverse contextual circumstances, typically involving the positional variation of sensor measurements. Sensor-related events like periodic maintenance, gradual coverage plans, and malfunctions can affect changes in sensor locations within the network, leading to exhaustive retraining for conventional machine-learning models in severe cases. Therefore, the criterion favoring model flexibility is essential to uphold the model performance during the network's life cycle.

($\mathbb{C}$3) Robustness: This criterion considers the model's robustness against inherent distortion in the wild. The reasons involve noise in observations, data transmission, and parameter discrepancy between simulated and actual environments. Notably, this uncertainty has been overlooked in existing approaches, as testing data was often evaluated in a noise-free, laboratory environment.

In the following section, the provided list is crucial in assessing the state-of-the-art methods for the pressure estimation task. Accordingly, we outline their accomplishments and limitations before presenting our solution that purposely fulfills all specified criteria.

3 Related Work

3.1 GNNs for node-level regression task

Most of the GNNs have originated from a feed-forward neural network involving multiple hidden layers of interconnected neurons. Each layer performs a transformation of the input data and then passes it through a series of mathematical operations. In terms of GNNs, the favored input data is the graph. As it can naturally represent an abstract level of a water distribution network, we purposely investigated existing GNN approaches for solving our core problem known as node-level regression.

Wu et al. categorized GNN based on their purposes into graph-level, link-level, and node-level tasks. These categories indicate the versatility and primary focus of GNN to provide outcomes across various domains. For instance, graph-level and link-level have been employed in domains such as chemistry (Reiser et al., 2022), bioinformatics (Nguyen et al., 2020), and recommendation systems (Z. Chen et al., 2020). On the other hand, the node classification task has risen the prominence in the node-level category, demonstrating its dominance in practical fields such as physics (Shlomi et al., 2020) and finance (B. Xu et al., 2021).

As an influence of prevalent node classification, well-known GNN architectures have been developed to excel for this specific task (Defferrard et al., 2016; M. Chen et al., 2020; Veličković et al., 2018). This leads to the lack of attention to node regression tasks and causes ambiguity regarding the effectiveness of these popular GNNs in handling continuous values within the node-level regime. While early research has explored node-level regression in narrow domains(Derrow-Pinion et al., 2021a; Ying et al., 2018), comprehensive comparisons across these popular architectures, especially within the water sector, are lacking. Addressing this issue, we delve into exploring the capability of popular GNNs in tackling a fundamental challenge in node regression: state estimation.

3.2 State Estimation with GNNs

Early research integrated Graph Neural Networks (GNNs) into calibration processes (Zanfei et al., 2023). Along with the development, recent advancements have introduced calibration-free GNNs, showcasing notably superior performance compared to classical models in state estimation tasks (Hajgató et al., 2021). State estimation is a process of inferring the current state of a water distribution network. For example, it involves predicting the water pressure or flow rate at different nodes based on sensor measurements within the network. Its application is crucial for subsequent management tasks, including leak localization (Mücke et al., 2023), optimal control (Martínez et al., 2007), and cyber-attack detection (Taormina et al., 2018).

It is worth distinguishing state estimation from a related concept known as surrogate modeling, as both approaches may produce similar outputs, such as pressure, flow rate, and velocity (Kerimov et al., 2023). The key distinction lies in their objectives and input features. Surrogate models strive to replicate the behavior of a physical simulation model across diverse input parameters—such as customer demand, nodal elevation, and pump head patterns. Conversely, state estimation focuses on reconstructing signals at unmeasured nodes based on existing ones and the network's topology. In other words, state estimation models have fewer requirements to provide outcomes, leading to beneficial convenience in the practical application. Expanding from this point, we supply a comprehensive list of representative works and assess them against the predefined criteria outlined in Section 2.3.

(Hajgató et al., 2021) is the first work that proposes to train a GNN model on a well-defined synthetic dataset. **(ℂ2)Adaptability** is satisfied because the authors trained the model on various snapshots concerning different sensor locations. On the other hand,

achieving **(ℂ3)Robustness** is vague as all reports were based on time-irrelevant and synthetic data. Also, the model cannot deal with the generalization problem due to the limitation of spectral-based GNNs in which the learned features extracted from Fourier space are closely associated with the corresponding network, yielding the lack of reusability (S. Zhang et al., 2019). For this reason, it fails to satisfy **(ℂ1)Generalizability**.

(Ashraf et al., 2023) improved the above work on historical data. In this case, working with a spatial-based GNN can help extract features from any encountering topology, so it is possible to satisfy **(ℂ1)Generalizability**. In addition, testing on noisy time-relevant data could be seen as an uncertainty consideration . However, this work heavily depends on historical data generated from a pure mathematical simulation engaging with "unchanged" dynamic parameters (e.g., customer demand patterns). In practice, this approach does not apply to the cases where those parameters are prone to error or unknown (Kumar et al., 2008). Hence, we consider that it weakly satisfies **(ℂ3)Robustness**. However, the authors fixed sensor positions during training, which could negatively affect the model observability to other regions in the WDN. For this reason, stacking very deep layers that increase model complexity is inevitable to ensure the information propagation from far-away neighbors to fixed sensors (Barceló et al., 2020). Additionally, retraining the model is mandatory whenever a new measurement is introduced, which has detrimental effects on its flexibility and scalability. Thus, the model violates **(ℂ2)Adaptability**.

Note that we exclude the heuristic-based methods as they do not consider the topology in decision-making. Also, several graph-related approaches (Kumar et al., 2008; Xing & Sela, 2022) have existed in this field. However, they accessed historical data, and neither attempted to solve the task in a generalized manner. Alternatively, we assume that prior knowledge (i.e., historical data) is unavailable. In this work, we delve into the capability of GNNs in a general case, in which the trained model can be applied to any WDN and any typical scenario.

4 Methodology

4.1 Water network as graph

A water distribution network is a complex infrastructure that provides safe and reliable access to clean water for individual usage. We define an immediate state of measurements recorded in a water network as a *snaphot* at a particular timestamp. A sequence of snapshots yields a scenario expressing a form of temporal correlation across *snaphots*. Due to initial assumptions, this temporal information is unavailable, leading us to consider a sampled snapshot as a representation of the entire scene for a particular network.

Mathematically, a *snapshot* is represented as a finite, homogeneous, and undirected graph $G = (\mathbf{X}, \mathbf{E}, \mathbf{A})$ that has N nodes and M edges. Edges represent pipes, valves, and pumps, while nodes can be junctions, reservoirs, and tanks. The nodal features are stored in the matrix $\mathbf{X} \in \mathbb{R}^{N \times d_{node}}$, where d_{node} is the nodal feature dimension. In this work, pressure is the unique node feature because it is recognized as the most vital stable factor in monitoring the WDN (Christodoulou et al., 2018) and aligns with prior research (Hajgató et al., 2021; Ashraf et al., 2023). Accordingly, we refer $\mathbf{X}$ to a pressure matrix, and the feature dimension d_{node} is fixed to 1.

$\mathbf{E} \in \mathbb{R}^{M \times d_{edge}}$ is an edge feature matrix, in which d_{edge} is the edge dimension. Depending on a particular model, we set d_{edge} to 0 if none of these edge attributes is used or 2, which indicates pipe lengths and diameters are supported. The node connection is represented in an adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $a_{ij} = 1$ means node i and j are connected by a link, and $a_{ij} = 0$ for otherwise.

Observing accurate pressure $\mathbf{X}$ for an entire water network is challenging due to partial observability. Hence, we rely on a physics-based simulation model to construct synthetic pressure as training samples for the model. Concretely, the simulation encompasses a variety of parameters, both static and dynamic. Static parameters, such as nodal elevation and pipe diameter, remain constant, while dynamic parameters, such as junction demands and tank settings, fluctuate over time. Then, it solves a hydraulic equation to estimate the pressure and flow at unknown nodes in a Water Distribution Network (Simpson & Elhay, 2011). For more details on solving hydraulic optimization, we refer the reader to the EPANET engine, which serves as our default mathematical simulation (Rossman, 1999).

Despite the usability of conventional simulations, they demand a manual calibration process to stay synchronized with the actual physical water network. Also, they suffer from the OOD problem mentioned in Section 2. In light of these limitations, we adopt a strategic alternative. In particular, we merely leverage a simulation model to generate synthetic samples for training our calibration-free model. The trained model can infer the pressure of any water network in the deployment.

4.2 Dataset creation

Throughout this paper, GNNs take WDN *snapshots* as inputs. In particular, each *snapshot* provides a global view of a WDN graph representing pressure values at an arbitrary time. Additionally, it contains topological information (e.g., node connectivity, and edge attributes) from a corresponding water network. We denote as a *clean snapshot* the one that describes an instantaneous pressure state without any hidden information. In contrast, a *masked snapshot* portrays a partially observable network in which the target feature known as pressure is mostly undetermined except for a small number of metered areas.

The conventional generation requires temporal patterns to create a set of *clean snapshots*. A pattern records a time series of a specific simulation parameter, such as customer demand or pump curve, in a fixed period. In other words, such a series is often recorded in ordinary scenarios. However, it is not guaranteed that these patterns include all events, and real-world data is highly volatile. For example, a model trained on data created from past patterns can fail to estimate the pressure of a WDN during the long-term COVID-19 pandemic due to the unexpected sudden change in water consumption that was never found in such patterns (Campos et al., 2021; Tiedmann et al., 2022). Furthermore, the number of available patterns is seldom provided or partially accessible due to privacy-related concerns, especially in public benchmark WDNs. For this reason, they are often repetitively overused in modeling large-scale water networks where the number of nodes is exponential compared to the required patterns. This significantly impacts dataset diversity and, therefore, limits the model capability to satisfy criteria **($\mathbb{C}$3) Robustness**.

We then scrutinize existing generation approaches that consider the characteristics of hydraulic parameter volatility and variability in Table 1. The underlying simulation (i.e., EPANET (Rossman, 1999)) still plays a crucial role in creating *clean snapshots* given an arbitrary set of parameters. Still, each approach has a specific selection and adjustment of dynamic parameters with respect to a design space. It is worth noting that we recognize the presence of other methods that intentionally distort topology or create unforeseen scenarios (such as leakage, earthquakes, etc.) (Menapace et al., 2020). However, such approaches contradict our initial assumptions, leading to their dismissal.

The conventional simulation method, EPANET (Rossman, 1999), operates time-dependently and relies primarily on fixed patterns. Excessive use of these patterns results in temporal correlations among snapshots, primarily due to their inherent seasonal

Table 1. The selection of dynamic parameters between conventional simulations and sampling-based generations. Parameters marked with a check can exhibit varying values, whereas others remain constant throughout the generation process. Note that the dynamic parameter selection also depends on component availability in a particular network and dataset creation stability to prevent abnormal results.

Dataset Creation	Reservoir Total Heads	Junction Demand	Pump Speed	Pump Status	Tank Levels	Valve Settings	Valve Status	Pipe Roughness
(Rossman, 1999)	✓	✓	✓[a]					
(Hajgató et al., 2020)		✓	✓	✓	✓			
Our	✓	✓	✓	✓	✓	✓	✓	✓

[a] *Pump speed* is implicitly adjusted by a *pump curve* pattern.

factors. This issue becomes inevitable, especially in large-scale networks, where numerous unmeasurable nodes require pattern assignments to complete a simulation process. Consequently, this leads to information leakage between snapshots within the same scenario (see after-splitting data distribution in training and testing sets in Figure 1).

Alternatively, (Hajgató et al., 2020) eliminate time patterns and consider a single snapshot as an instantaneous scenario. This way is more delicate to provide more observations for data-hungry models. However, (Hajgató et al., 2020) focus only on pump optimization, so half of the listed parameters remain untouched.

Both available generations assume the remaining parameters are deterministic and unchanged. Nevertheless, these parameters (e.g., pipe roughness) can be critical factors affecting the simulation result (Zanfei et al., 2023). Thus, neglecting any of these parameters can restrict the model in learning representations of WDN snapshots.

Intuitively, we consider a comprehensive modification of all dynamic parameters as a data augmentation to ensure the simulation quality and address the generalization problem. Our main objective is to design a sufficient search space to provide different pressure views from flexible sensor positions. This approach helps alleviate the data-hungry issue when training deep learning models and benefits model robustness thanks to the augmented data space (Cubuk et al., 2020).

In particular, we adopt a brute-force approach to explore the full range of available dynamic parameters. To ensure the simulation quality, we exclude parameter sets that generate pressure ranges surpassing practical limits, within the range of $[0\,m,\ 151\,m]$ (Paez & Filion, 2017). Subsequently, our generation takes these sets of dynamic parameters, an unchanged static set, and the topology of a particular water network to generate a single snapshot using the conventional simulation (refer to Figure 2). Note that it only performs a single simulation step that removes the essence of temporal patterns. This process outcome is a set of distinct immediate pressure states, which are more versatile and independent in time. In contrast to the classical usage, this approach eliminates the temporal correlation concern and leverages all dynamic parameters to generate training samples designed to cover the entire input space.

4.3 Model Architecture

The model is expected to learn a graph representation from existing known signals to estimate unknown pressures. In addition, for accurately estimating the pressure at a distant location from sensors, it is crucial to have an approach to gather the (inferred) measurements from the metered nodes and intermediate nodes to the distant neighbor efficiently. We first recap Message Passing Neural Networks (MPNN) (Gilmer et al., 2017), the generic framework for spatial GNNs. Then, we discuss Graph Attention Network (GAT)

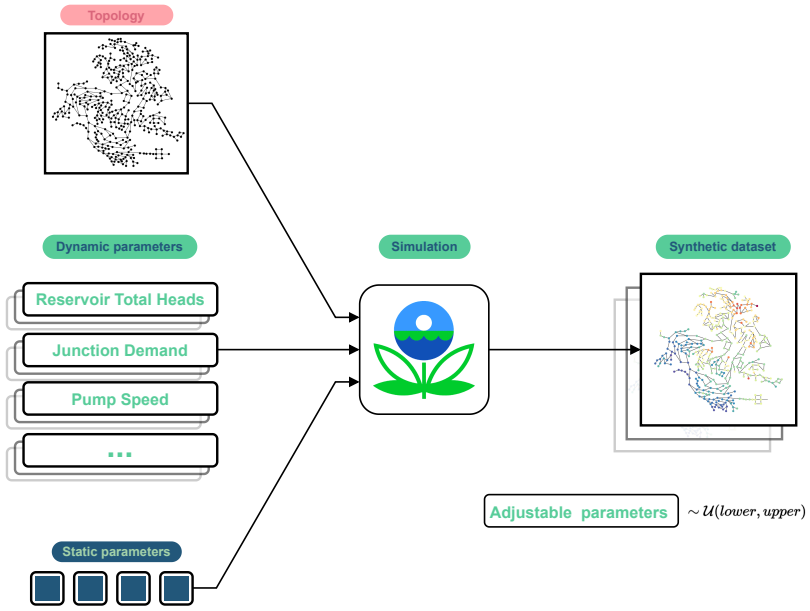

Figure 2. Our data generation approach. Dynamic parameters are sampled from a uniform distribution and passed to the mathematical simulation with static values and WDN topology. The result is a synthetic dataset containing legit snapshots whose pressure range should be close to reality.

(Veličković et al., 2018) as one of our fundamental components. In light of this, we propose *GATRes* as a principal block and devise the overall architecture illustrated in Figure 3.

4.3.1 Preliminaries

Considering a GNN as a series of stacked layers, MPNN describes a specific layer to transform previous representations to successive ones using message propagation. We omit the layer index for simplicity and denote representations of a target node i as x_i. Noticeably, the first representations are input features known as pressure values. Then, the output representations of a consecutive layer are computed as follows:

$$z_i = UPDATE\left(x_i, \bigoplus_{j \in \mathcal{N}(i)} MSG(x_j)\right) \tag{1}$$

where z_i is the corresponding output of the target node i, $\mathcal{N}(i)$ denotes the 1-hop neighbors, *MSG* and *UPDATE* are differential functions describing messages received from neighbors and the way to update that information concerning its previous representations respectively. $\bigoplus$ is a differentiable, permutation-invariant function, ensuring the gradient flow backward for model optimization and addressing concerns related to node ordering (Gilmer et al., 2017). This function plays a critical role in aggregating neighbor messages into the target one.

Depending on the task-specific purpose, numerous ways exist to define the message aggregator, such as mean, max, sum (K. Xu et al., 2019), or Multilayer Perceptron (Zeng et al., 2020). Ideally, $\bigoplus$ is designed to propagate messages from surrounding nodes in a sparse fashion, which only matters to non-zero values. Thus, this scheme efficiently scales

when dealing with enormous graphs and economically saves the memory allocation budget.

Next, we explain *GAT* in view of a target node i. Concretely, *GAT* focuses on the intermediate representation relationship between the target node i and one of its 1-hop neighbors j. If a node pairs with itself, it forms a self-attention relationship. Hence, we strategically establish a virtual self-loop link in every node to put weights between itself representations compared to the aggregated ones from the neighborhood. Mathematically, we can rewrite the *GAT* formula according to Equation 1 as:

$$z_i = \Big\Vert_{h}^{H} \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij}^{h} \mathbf{\Theta} x_j = GAT(x_i) \tag{2}$$

where H is the number of attention heads, $||$ is a concatenation operator, $\mathbf{\Theta} \in \mathbb{R}^{d_{in} \times d_{out}}$ is the layer weight matrix with d_{in} and d_{out} that are the input and output representation dimensions, respectively. Specifically, a summation is chosen as the *UPDATE* function to aggregate nodal messages. For each node, its *MSG* function is represented by multiplying α with the linear combination of learnable weights $\mathbf{\Theta}$ and nodal input x_j. In addition, α is referred to as the attention coefficient and is computed as:

$$\alpha_{ij} = softmax(\sigma(a^T[\mathbf{\Theta} x_i || \mathbf{\Theta} x_j])) \tag{3}$$

where $softmax(x) = \frac{e^{x_i}}{\sum_{j \in \mathcal{N}(i) \cup \{i\}} e^{x_j}}$ is used to compute the important score between the target node i and a neighbor j. Before calculating *softmax*, the concatenation of both nodal representations is then parameterized by learnable weights $a \in \mathbb{R}^{2d_{out}}$ and passed through a non-linear activation function $\sigma(.)$ (e.g., *ReLU*, *GELU*, or *LeakyRELU* (B. Xu et al., 2015)).

Inspired from (Vaswani et al., 2017), *GAT* leverages multiple attention heads to perform parallel computation and produce diverse linear views. In the original work, those head views should be joined to "merge" all-in-one representations, thanks to a linear layer mapping the concatenated heads. However, the conventional approach (Veličković et al., 2018) is to stack numerous concatenated *GAT* layers sequentially (hence, without any head joint) except for the last layer, where a final mean view is computed but only for the final logit in a classification task. The postponed head joining could preserve irrelevant views in the consecutive layer that double the detrimental effects of the nodal feature sparsity due to the high masking rate in an unsupervised setting. In other words, irrelevant head views quickly saturate the impact of final nodal presentations and accelerate the smoothing process (that is, oversmoothing (D. Chen et al., 2020)). Extra propagation layers are helpless because they worsen the situation. Thus, we hypothesize that merging head views could complete the original design and suppress unrelated information. Intuitively, it raises a question of whether to linearly transform the concatenation head view as in (Vaswani et al., 2017) or merely take an average of head representations.

4.3.2 GATRes

Alternatively, we empirically propose using an additional GAT layer to evaluate head views generated from the previous one. We name this approach as **GAT** with **Res**idual Connections (GATRes). Mathematically, we define our *GATRes* as follows:

$$z_i = x_i + \frac{1}{|\mathcal{N}(i)+1|} \sum_{j \in \mathcal{N}(i) \cup \{i\}} GAT(GAT(x_j; \alpha, \mathbf{\Theta}); \beta, \Psi) \tag{4}$$

where, attention coefficients $\alpha \in \mathbb{R}^{N \times H}$ computed in Equation 3 and learnable weight matrix $\mathbf{\Theta} \in \mathbb{R}^{d_{in} \times H d_{out}}$ belong to the first GAT. Identically, $\beta \in \mathbb{R}^{\mathbb{N}}$ and $\mathbf{\Psi} \in \mathbb{R}^{H d_{out} \times d_{out}}$ are from the second GAT but different in shape.

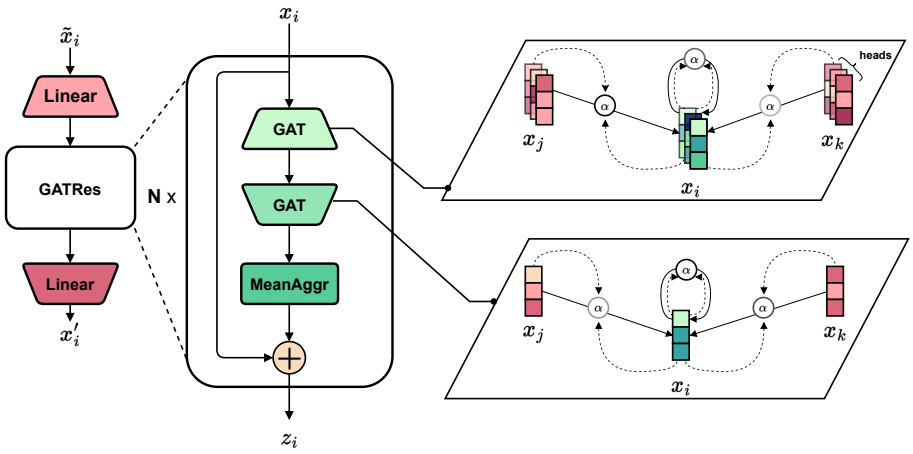

Figure 3. ***GATRes* architecture.** The left image indicates the overall architecture consists of two linear layers interleaving with *GATRes* blocks. The middle figure illustrates the abstract view in each block. The right-side ones explain the message aggregation mechanism between neighbor nodes.

As in the middle image in Figure 3, we feed the intermediate input x_i to two GAT layers sequentially. For a target node i, the inner $GAT : \mathbb{R}^{N \times d_{in}} \rightarrow \mathbb{R}^{N \times H d_{out}}$ devises multi-head views and weighs them among its 1-hop neighbors. In other words, it additionally enriches the diversity of multi-head views using the message aggregation from surrounding nodes.

Then, the outer $GAT : \mathbb{R}^{N \times H d_{out}} \rightarrow \mathbb{R}^{N \times d_{out}}$ creates a bottleneck in the feature dimension and, again, reweights the target representation considering the ones of its neighbors. Note that the second GAT has exactly one head to transform all previous heads into a consistent view. We call this process a *squeezing technique*. Since most initial features are noise or zeros, squeezing can reduce the sparsity duplication in feature space caused by head concatenation and, therefore, benefits second attention among nodal pairs. Moreover, we consider the distribution of pressure values in the neighborhood, so we empirically apply a mean aggregator to the current representations (K. Xu et al., 2019). Afterward, we use a non-parametric residual connection with the intermediate input x_i that allows depth extension and diminishes the overfitting problem (He et al., 2016).

As in Figure 3, the overall structure is a stack of numerous *GATRes* blocks. As each block considers 1-hop neighborhoods, stacking multi-blocks allows message propagation to faraway neighbors in the graph. Before message propagation layers, we employ a shared-weight linear transformation to project the masked input nodal features to higher-dimensional space. The details of masked inputs will be explained in the following subsection. We refer to the first linear layer as the steaming layer, which is well-known in computer vision tasks (Dosovitskiy et al., 2021; Tan & Le, 2019). After message propagation, the final linear layer acts as a decoder to project higher-dimensional representations back to the original dimension (i.e., $d_{node} = 1$). The end-to-end model will then output an immediate snapshot in which all pressure values at any junctions are recovered.

4.4 Model training

This section introduces our solution for addressing the pressure estimation task. We begin by outlining a general training scheme applicable to any GNN model. Figure 4 depicts this scheme, leveraging synthetic datasets from the previous stage for training. Following this, we detail an approach to evaluate the trained model using time-relevant data.

4.4.1 Training details

To begin with, we sample a binary mask vector $m = \{m_1, m_2, ..., m_N\}$ where each $m_i \in \{0, 1\}$. We then construct a feature subset of the masked node $\tilde{\mathbf{X}} \subset \mathbf{X}$ in which its element $\tilde{x}_i$ is denoted as:

$$\tilde{x}_i = \begin{cases} 0 & m_i = 1 \\ x_i & m_i = 0 \end{cases} \tag{5}$$

There exist various masking strategies, such as learnable [MASK] tokens, feature permutation, arbitrary vector substitution, and mixup nodal features, which could be helpful for future work (H. Zhang et al., 2018). In this work, we opt for a simple masking approach: replacing the node features with zeros in the masked positions to create the masked $\tilde{\mathbf{X}}$ (Equation 5). We then formalize the pressure estimation as follows:

$$\mathbf{X}' = f_{GNN}(\tilde{\mathbf{X}}, \mathbf{E}, \mathbf{A}; \mathbf{\Theta}) \tag{6}$$

where $f_{GNN} : \mathbb{R}^{N \times d_{node}} \times \mathbb{R}^{N \times N} \times \mathbb{R}^{M \times d_{edge}} \mapsto \mathbb{R}^{N \times d_{node}}$ is a generic GNN function that takes partial-observable feature matrix $\tilde{\mathbf{X}}$, the topology $\mathbf{A}$, and edge attributes $\mathbf{E}$ as inputs and yields reconstructed features $\mathbf{X}'$ characterized by model weights $\mathbf{\Theta}$. The key idea is to find the optimal weights that satisfy the minimum error between predicted and ground truth nodal features. Mathematically, the objective is formalized as follows:

$$\mathbf{\Theta}^* = \underset{\mathbf{\Theta}}{\operatorname{argmin}} \, \mathbf{L}(\mathbf{X}', \mathbf{X}) \tag{7}$$

Empirically, we use the mean square error (MSE) as a default loss function $\mathbf{L}$ for *GATRes* because it yields the best result in our tests. In addition, inspired by BERT (Devlin et al., 2018), the loss is computed on masked positions. After computation, model weights $\mathbf{\Theta}$ are updated by the partial derivatives w.r.t the computed loss. For detail, we refer to gradient descent optimization techniques (Ruder, 2016; Kingma & Ba, 2014) . The training progress is then iterated with different masked features $\tilde{\mathbf{X}}$ derived from the original features $\mathbf{X}$ until the model convergence.

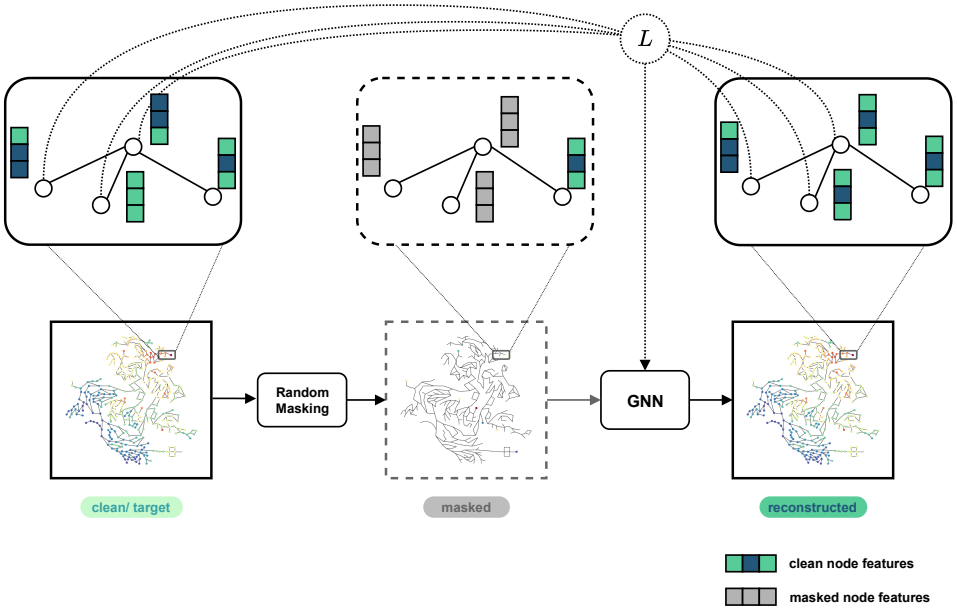

Figure 4. Graph Neural Network training scheme. Given clean snapshots, we mask out a significant number (95%) of node features. The remaining data is then sent into a GNN playing as an autoencoder to rebuild missing values with regard to graph properties (such as topology and edge attributes). GNN weights are updated using the loss derived by the predicted and ground truth values at masked places.

4.4.2 Testing details

In testing, the test graphs can be represented as $\mathbf{G}_{test} = (\tilde{\mathbf{X}}_{test}, \mathbf{E}_{test}, \mathbf{A}_{test})$. By default, topology and edge attributes are retained as in training while $\tilde{\mathbf{X}}_{test}$ is varied and its data distribution is undetermined. For the generalization problem, $\mathbf{G}_{test}$ can differ from the training graphs in any property. In other words, the model should be able to estimate pressure on an unseen topology and an unknown data distribution.

When the temporal dimension is involved, the test graph at a particular time t is denoted as $\mathbf{G}^t_{test}$. As the designed model takes only one snapshot as input, we feed temporal $\mathbf{G}^t_{test}$ into the trained GNN model sequentially and individually. In other words, previously inferred outcomes do not impact any reconstruction result within a testing scenario. These scenarios are considered real-world situations associated with inherent uncertainty due to dynamic factors (Zhou et al., 2023). To reflect this uncertainty in our testing, we adopt a distortion method on junction demands during the testing phase, drawing inspiration from (Zhou et al., 2023; Mücke et al., 2023). Specifically, we outline two strategies as follows:

Clean test. We consider an assumption of no uncertainty, ensuring that baseline models observe clear, precisely calibrated pressure. As these models treat each snapshot as an independent sample, we gauge the model adaptability across different sensor configurations using a distinct mask for each snapshot in every trial. The statistical outcomes from 100 trials, encompassing diverse metered locations, illustrate the model performance under typical conditions.

Noisy test. Following (Zhou et al., 2023), we inject Gaussian noise into junction demands before processing the simulation and then pair each outcome snapshot with a random mask for a test case. Pipe roughness, another considered param-

eter, is intentionally excluded due to its inconsistency across different headloss formulas used for each water network. Notably, the Hazen-Williams and Darcy-Weisbach formulas are sensitive to pipe roughness, whereas the Chezy-Manning formula remains unaffected by this parameter (Klise et al., 2018). In the interest of generalization, we have opted to exclude this parameter. Nevertheless, we set a tougher noise for junction demands beyond the original tests. The new noisy test involves the mean and standard deviation of 10% and 100% of the initial demands, respectively. We run 100 test cases and report statistical findings.

5 Experiment settings

5.1 Datasets

The main use case in this study was performed using a private large-scale WDN in The Netherlands in the area of Oosterbeek. The network comprises 5855 junctions and 6188 pipes. Figure 5(a) shows the topology and the pressures at the nodes from some random snapshot of Oosterbeek WDN.

We also used four publicly available WDNs benchmarks, namely Anytown (Walski et al., 1987), C-Town (Ostfeld et al., 2012), L-Town (Vrachimis et al., 2022), and Richmond (Van Zyl, 2001) to provide a baseline for evaluation and reproducibility of our work . Finally, in the experiments related to model generalization, we used two additional public datasets, Ky13 (Hernadez et al., 2016) and an anonymized WDN called "Large" (Sitzenfrei et al., 2023). The WDNs used in this study vary in size and structure, ranging from small and medium size to large-scale networks like "Large" and Oosterbeek, as can be seen in Figure 5. Table 2 shows the main characteristics of each network.

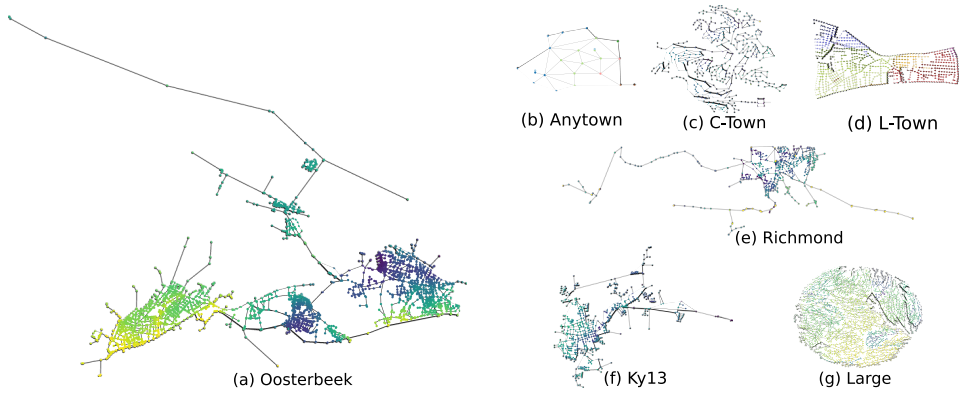

Figure 5. Water Distribution Networks used in this study.

Table 2. Properties of WDNs used in this study.

WDNs:	Oosterbeek	Anytown	C-Town	L-Town	Richmond	Ky13	Large
junctions	5855	22	388	785	865	775	3557
pipes	6188	41	429	909	949	915	4021

5.2 Baseline models settings

Generally, two goals dictate the baseline selection. Section 3 mentions the first goal: to try out popular GNN architectures on a node-level regression problem. The aim is to achieve acceptable errors when the models are tested on data points that are from an unknown "nature" distribution. The second goal is to explore existing frameworks, from synthesized data querying and model training to evaluation phases. We aim to establish a reliable benchmarking framework for pressure estimation tasks or problems related to water distribution networks. In other words, the model that performs better in our tests should be more useful in practical applications.

For this purpose, we compare our *GATRes* architecture to popular GNNs, including *GCNii* (M. Chen et al., 2020) and *GAT* (Veličković et al., 2018). In addition to this, *GraphConWat* (GCW) (Hajgató et al., 2021) and *mGCN* (Ashraf et al., 2023), which are dominant approaches in solving pressure estimation using GNN with sparse information, are also considered in our comparison. Table 3 summarizes the model settings.

Table 3. Baseline settings.

	GCNii	GAT	GCW	GCW tuned (ours)	mGCN	GATRes small (ours)	GATRes large (ours)
#blocks	64	10	4	4	45	15	25
#hidden.channels	32	{32,64}	{120,60,30}	32	{98,196}	{32,64}	{128,256}
Coefficient K	-	-	{240,120,20,1}	{24,12,10,1}	-	-	-
Edge Attribute	binary	binary	binary	binary	pipe.len, pipe.dia[a]	binary	binary
Norm Type	znorm	znorm	minmax	znorm	minmax	znorm	znorm
Loss	MSE	MSE	MSE	MSE	MAE	MSE	MSE
Learning Rate	3e-4	3e-4	3e-4	3e-4	1e-5	5e-4	5e-4
Weight Decay	1e-6	1e-6	1e-6	1e-6	0	1e-6	1e-6

[a] Pipe lengths and pipe diameters. They are static parameters gathered from the corresponding Water Distribution Network.

GATRes-small with hyperparameters is the optimal version after the optimization process, which will be carefully explained in the latter section. To study the impact of the model size, we also introduce *GATRes-large* scaling close to *mGCN* in terms of the number of parameters.

GAT remained at a shallow depth to prevent the oversmoothing problem (D. Chen et al., 2020). Precisely, neighbor features encoded by a too-deep GNN converged to indistinguishable embeddings that harm the model performance. Empirically, we balance the trade-off between performance and efficiency for each model to select the appropriate hyperparameters.

In *GraphConvWat* models, we detached binary masks from the input features as these masks did not improve the model performance, which aligns with the findings in (Ashraf et al., 2023). Furthermore, *GraphConvWat tuned* is a lightweight version in which the degrees of the Chebyshev polynomial K_i are set to smaller values to reduce complexity and work surprisingly well in our experiments.

Training *mGCN* slightly diverged from its original work in sensor positions. Concretely, (Ashraf et al., 2023) trained *mGCN* using fixed sensors with extensive historical data. However, as we explained in Section 4.4, this data was inaccessible throughout training. Therefore, we fed different random masks into the model in each epoch. Considering a synthetic dataset, the model had an opportunity to capture meaningful patterns in various sensor positions. Additionally, *mGCN* incorporated static edge attributes such as pipe length and diameter in its final decision-making process.

We then provide an overview of other hyperparameters, applicable to any mentioned model. Each model was trained in a fixed number of iterations, so-called epochs, throughout the entire training set. In addition, the whole dataset, including the training set, was pre-processed by z-score transformation (z-norm) or min-max feature scaling. Within an iteration, the model was sequentially fed data batches whose size determined the number of training examples. Both epochs and batch size were detailed in each specific experiment. Following each iteration, we computed the loss, typically Mean Squared Error (MSE) or MAE, comparing predicted outputs to ground truth. Subsequently, this loss guided the optimizer algorithm in updating the model's weights. The optimizer's behavior was influenced by additional hyperparameters, including the learning rate, regularization (defaulting to L2), and weight decay, which penalized model weights to mitigate overfitting. For comprehensive definitions, we recommend referring to (Biehl, 2023).

5.3 Evaluation metrics

The most common evaluation metrics used for assessing the performance of regression models are Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) (Zhao et al., 2019; Derrow-Pinion et al., 2021b; Jiang & Luo, 2022). Following the insights from Legates and McCabe Jr. (1999), the models should be evaluated using both, relative and absolute error metrics. Thus, our model is evaluated using MAE and MAPE. Additionally, we included the Nash and Sutcliffe Coefficient of Efficiency (NSE), widely used to evaluate the performance of hydrologic models (Legates & McCabe Jr., 1999). Finally, we used an accuracy metric defined as the ratio of positive predictions over the total number of predicted values. The positive predictions are those that deviates at most a certain threshold (δ_{thresh}) from the true value. Thus, the evaluation metrics used in this work are defined as follows:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{8}$$

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{y_i} \tag{9}$$

$$\text{NSE} = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2} \tag{10}$$

$$Acc(@\delta_{\text{thresh}}) = \frac{1}{N}\sum_{i=1}^{N} positive_i \; ; \quad positive = \begin{cases} 1, & \text{if } |y_i - \hat{y}_i| \leq \delta_{\text{thresh}} * y_i \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

where $\boldsymbol{y}$ denotes the true values, $\hat{\boldsymbol{y}}$ denotes the predicted values, $\overline{\boldsymbol{y}}$ is the mean of the true values, and $\boldsymbol{N}$ is the number of values to predict.

6 Experiments

6.1 Baseline comparison on Oosterbeek WDN

In this experiment, we investigated the proposed model performance against GNN variants on a large-scale WDN benchmark called Oosterbeek. Specifically, given the topology and hydraulic-related parameters, our dataset generation provided 10,000 synthetic snapshots divided into 6000, 2000, and 2000 for training, validation, and testing sets, respectively. However, as we discussed in Section 2.2, these synthetic sets might not reflect real-world scenarios. Therefore, we merely used them to keep track of model learning during the training process.

Alternatively, we performed the comparison on the Oosterbeek dataset recorded every five minutes for 24 hours. We relied on mathematical simulation to produce reproducible results that resembled real-world conditions. The topology and predefined parameters set by hydraulic experts under a calibration process made them valid for our analysis. As a result, we considered the simulated outcomes of time-relevant data as ground truths.

Table 4. Model comparison in the clean test performed on 24-hour Oosterbeek WDN at 95% masking rate.

Model	#Milion Params (↓)	MAE(↓)	MAPE(↓)	NSE(↑)	Acc(@0.1)(↑)
GCNii (M. Chen et al., 2020)	0.65	6.357±0.0197	0.2147±0.0008	-0.0137±0.0061	38.48±0.1351
GAT (Veličković et al., 2018)	0.35	3.726±0.0120	0.1287±0.0008	0.3276±0.0037	73.52±0.0900
GraphConvWat (Hajgató et al., 2021)	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 (Ashraf et al., 2023)	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

Table 5. Model comparison in the noisy test performed on 24-hour Oosterbeek WDN at 95% masking rate.

Model	#Milion Params (↓)	MAE(↓)	MAPE(↓)	NSE(↑)	Acc(@0.1)(↑)
GCNii (M. Chen et al., 2020)	0.65	6.696±0.0838	0.2484±0.0552	-0.1064±0.0266	36.02±0.4684
GAT (Veličković et al., 2018)	0.35	4.397±0.3052	0.2112±0.0767	0.1490±0.1153	66.98±1.6290
GraphConvWat (Hajgató et al., 2021)	0.92	3.611±0.1234	0.1551±0.0376	0.5877±0.0370	62.99±1.1600
GraphConvWat-tuned	**0.23**	2.347±0.0252	0.0963±0.0363	0.749±0.0086	81.09±0.3877
mGCN (Ashraf et al., 2023)	2.48	2.188±0.0558	0.0948±0.0155	0.6993±0.0213	82.83±0.4199
GATRes-small (ours)	0.66	**1.964**±0.0301	0.0802±0.0458	**0.778**±0.0113	**86.56**±0.2826
GATRes-large (ours)	1.67	2.115±0.0503	**0.0799**±0.0207	0.7417±0.0140	83.43±0.5044

Regarding the experimental setting, we trained all models in 500 epochs with a batch size of 8 for a fair comparison among the baseline models. Early Stopping was applied to suppress training if the validation error had no improvements in 100 steps. We used Adam optimizer (Kingma & Ba, 2014) and set the default masking rate at 95%, leaving only 5% of nodes unmasked.

For evaluation, we tested the baseline models on the 24-hour Oosterbeek, repeating the process 100 times. The mean and standard deviation of the results are presented in Table 4. Unless otherwise specified, the default is a clean test in our experiments. The results of the noisy test are given in Table 5.

Table 4 shows that *GATRes-small* achieved accurate junction pressure reconstruction with a MAPE of 7% and a MAE of 1.93m, even with a sparse masking ratio of 95%. Notably, the testing data were time-sensitive and originated from an unfamiliar distribution our models were not exposed to during training. The good results on snapshot-based models suggest that in case temporal data is not available, snapshot-based models seem to be good alternatives.

In addition, we assessed the efficiency of baseline models in the Oosterbeek experiment using an Nvidia RTX 3060 Laptop GPU for inference only. The results are presented in Table 6.

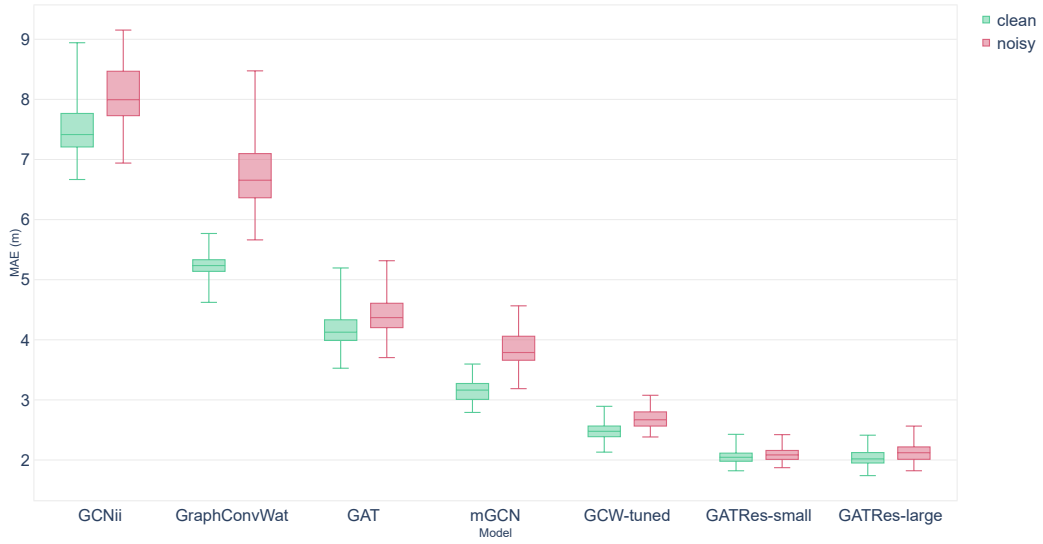

Figure 6. Baseline Mean Absolute Errors measured for a single snapshot under both clean and noisy conditions.

In this evaluation, we measured throughput, which counts the number of processed snapshots in a second. This metric can demonstrate the efficiency of baselines in terms of large-scale matter that demands continuous processing of massive data streams from sensors.

Notably, lightweight models such as *GAT* and *GraphConvWat-tuned* achieved the highest throughput, with our *GATRes-small* model ranking third. When we consider both efficiency and performance in Table 6, *GATRes-small* is a balanced option as this model delivers the best result while maintaining sufficient efficiency, a critical factor in saving computation resources and ensuring the sustainability of the environment.

Table 6. Throughput comparison in the clean test performed on 24-hour Oosterbeek WDN at 95% masking rate.

Model	Throughput(↑) (Snapshots per second)
GCNii (M. Chen et al., 2020)	663.80
GAT (Veličković et al., 2018)	2320.37
GraphConvWat (Hajgató et al., 2021)	90.39
GraphConvWat-tuned	2026.65
mGCN (Ashraf et al., 2023)	44.94
GATRes-small (ours)	749.38
GATRes-large (ours)	31.21

Our next focus was on analyzing the baseline robustness. Precisely, we assessed each baseline on clean and noisy tests using an individual snapshot with 100 randomly initialized masks. Our primary objective was to measure the model's robustness in these contrasting scenarios. A superior model should exhibit minimal error discrepancy between them. As illustrated in Figure 6, both versions of *GATRes* consistently maintained similar error levels even under conditions of high uncertainty. In contrast, other mod-

els exhibited a noticeable gap in their results when transitioning from clean to noisy environments.

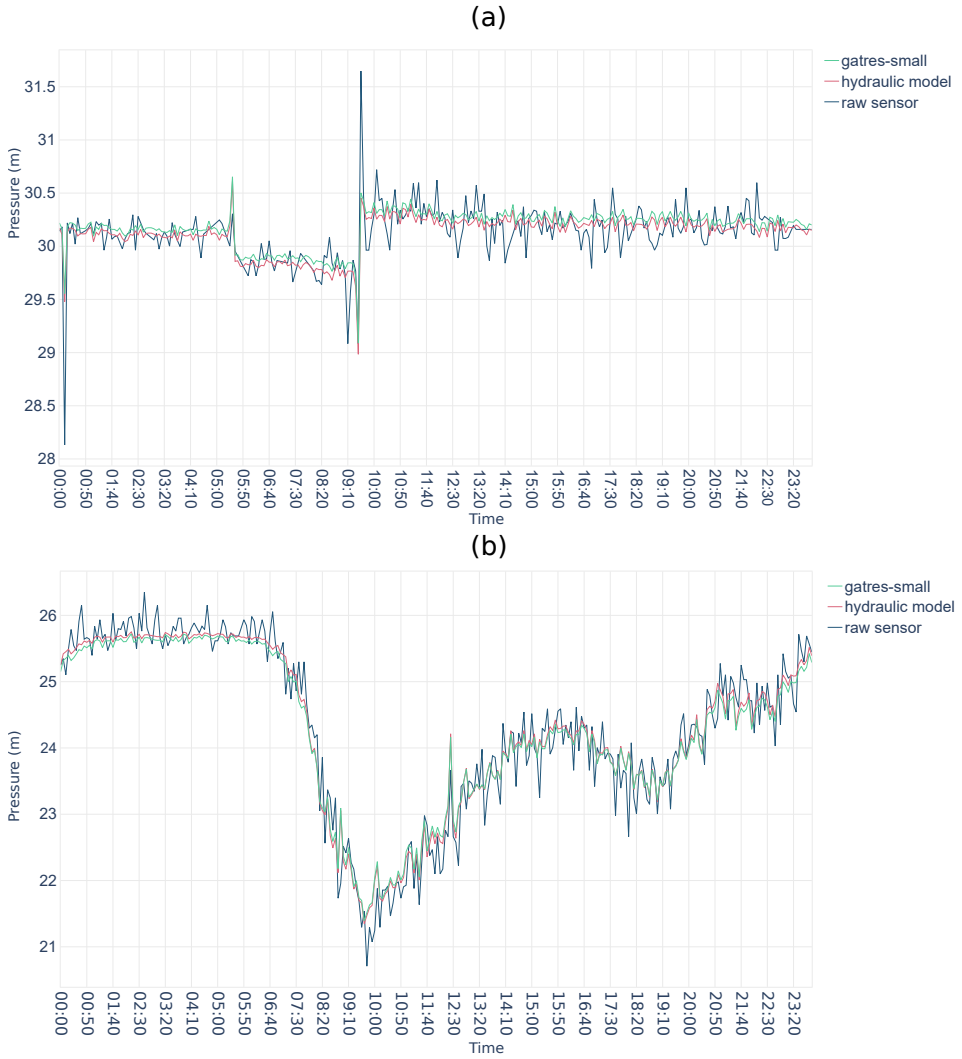

Figure 7. Pressure estimates derived from a hydraulic simulation model and our GATRes-small, validated against real sensor data from points (a) and (b) within the Oosterbeek WDN.

Finally, we conducted a detailed analysis of our top-performing model, *GATRes-small*, based on the evaluations conducted earlier. In this analysis, we intentionally masked sensor locations to observe model inference on those nodes. Figure 7 illustrates time series data from the predictions of *GATRes-small*, a well-calibrated simulation, and actual meter readings. As expected, *GATRes* closely mirrors the behavior of the hydraulic simulation. Although a slight difference exists between them, both time series were bounded in the range of actual measurements.

6.2 Generalization

This set of experiments is the first attempt aimed to achieve generalization capabilities of our model. We want to evaluate whether training a model on different topolo-

gies simultaneously can equip the model with generalization capabilities. For this, we chose three different WDNs, whose topologies vary in structure and size. We trained *GATRes-small* on L-Town, Ky13 and "Large" WDNs simultaneously. This model is named as *Multi-Graph model*. This model was trained with the *ReduceLROnPlateau* learning rate scheduler from Pytorch library. The scheduler reduces the learned rate if the model does not improve for a certain number of epochs. The initial learning rate was set to $5e$-3 and it is reduced by a factor of 0.1 if the validation loss does not improve for 30 consecutive epochs. The batch size was 16, and the model was trained for 500 epochs. Then, to assess the generalization capabilities of the pre-trained model we executed two different experiments: zero-shot inference, and transfer learning with fine-tuning.

Zero-shot inference implies evaluating the pre-trained *Multi-Graph model* on a fully unseen topology. Hence, we used the pre-trained model, directly, to reconstruct the pressures of our use case Oosterbeek, a WDN topology not seen during training.

The second experiment is transfer learning with fine-tuning. Transfer learning is a technique motivated by the fact that humans use previously learned knowledge to solve new tasks faster or better (Pan & Yang, 2009). Hence, the learned weights by a model trained on some particular network(s) can be transferred to train and improve (fine-tune) the prediction capabilities of a model on a new, previously unseen, WDN. In our work, the pre-trained model is the *Multi-Graph model*, trained on L-Town, Ky13 and "Large", and the fine-tuned model has as the target the Oosterbeek WDN. Usually, during fine-tuning, the top layers of the pre-trained model are frozen and reused as feature extractors for the target data. We empirically found that unfreezing the entire model and re-training all layers produce better results. Thus, we initialized the weights of the target model with those of the pre-trained one. Then, we reduced the learning rate for training the target model to avoid completely changing the pre-trained weights during fine-tuning. The learning rate was reduced from $5e$-3 in the source model to $1e$-4 during fine-tuning. The target model was trained with a batch size of 8 for 200 epochs. Fine-tuning can help practitioners reduce the implementation time of a predictive model for a completely new and unseen WDN. The more WDNs are added to the Multi-Graph model, the broader and more diverse training data. This can reduce the amount of samples (snapshots) required for training. Analyzing the number of snapshots with respect to the number of WDNs included in the training procedure is an interesting path for future work.

The results of both experiments are shown in Table 7. They reflect those of (Yosinski et al., 2014) who also found that combining transfer learning with fine-tuning shows better performance than a model trained directly on a target dataset.

Table 7. Generalization evaluation on 24-hour Oosterbeek WDN. Results of GATRes-small model, trained directly of Oosterbeek, are compared against those produced by the zero-shot and transfer learning with fine-tuning experiments.

	MAE (↓)	MAPE (↓)	NSE (↑)
GATRes-small	1.9370 ±0.0074	0.0703 ±0.0005	0.7773 ±0.0025
Multi-Graph (Zero-Shot)	3.0597 ±0.0074	0.0998 ±0.0005	0.5700 ±0.0045
Fine-tuned	**1.9097** ±0.0076	**0.0695** ±0.0005	**0.7980** ±0.0030

6.3 The effect of masking ratios

To explore the model capability, we investigated the *GATRes-small* on myriad masking rates. Identical to the previous experiment, the model was trained on the synthetic dataset generated from our algorithm and performed a clean test on the 24-hour data. Both were devised from the Oosterbeek WDN. In addition, each *GATRes-small* corre-

sponding to a specific mask rate was trained within 200 epochs with the default settings. For convenience, we replaced the model name with fixed masking rates in this experiment.

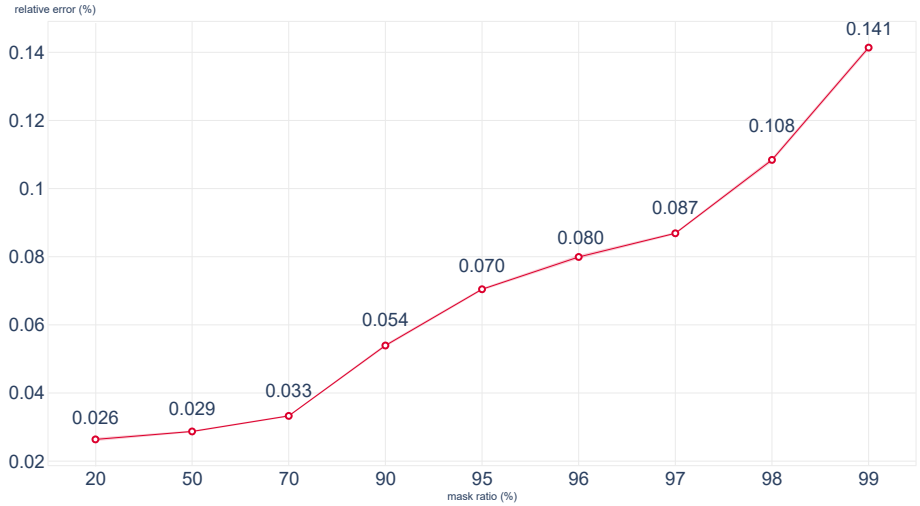

Figure 8. Relative errors (MAPE) for nodal pressure on different masking ratios(lower is better).

Figure 8 shows the influence of the masking ratio on the proposed model. Each ratio indicates a specific probability of missing nodal features (i.e., the pressure signals in a snapshot graph). Due to the sensor density being exceptionally sparse in real-world scenarios, the typical benchmark of 95%, commonly found in previous studies, is deficient in reflecting this practical issue. Therefore, we report errors occurring in all cases with lower and more extreme ratios that exceed the standard. As expected, we observe a decreasing trend with increasing masking rates. This pattern also appears across other metrics in Table 8.

Table 8. Detailed performance of *GATRes-small* on different masking ratios.

Mask ratio(%)	MAE(↓)	MAPE(↓)	NSE(↑)	Acc(@0.1)(↑)
20	0.610	0.0265	0.9599	0.9760
50	0.798	0.0286	0.9372	0.9654
70	0.900	0.0331	0.9301	0.9586
90	1.457	0.0544	0.8603	0.9167
95	1.939	0.0703	0.7770	0.8746
96	2.213	0.0800	0.7185	0.8467
97	2.415	0.0867	0.7059	0.8148
98	3.075	0.1091	0.5648	0.7465
99	4.087	0.1414	0.3424	0.6396

We conducted an additional investigation into the discrepancy between the masking ratios of train-test pairs. Our approach involved evaluating a trained model on the

24-hour Oosterbeek with various masking rates rather than just the specific rate initially trained. Through this exploration, we found that the best model of a specific testing masking rate was unnecessary to be trained on this rate. Table 9 showed this phenomenon in extreme ratios. Considering the trade-off between sparsity and performance, the model trained with a 97% masking rate demonstrated a Pareto-optimal solution. In addition, it surprisingly achieved the best results in extremely sparse test rates (i.e., $> 98\%$). This means that at most 3% of the total nodes would be sufficient for a quality model to monitor the Oosterbeek WDN– a large-scale network. Further analysis is highly recommended for WDN authorities to balance the trade-off between efficiency and measurement resources.

Table 9. Confusion matrix of relative mean errors (MAPE) between different train and test masking ratios(lower is better). Bold and underline are used to highlight the best and second-best results for a specific test mask, respectively.

test mask(%)	train mask (%)				
	95	96	97	98	99
95	**0.0702**	0.0723	0.0725	0.0772	0.0814
96	**0.0766**	0.0797	0.0784	0.0843	0.0882
97	**0.0858**	0.0901	0.0870	0.0934	0.0970
98	0.1031	0.1077	**0.1018**	0.1090	0.1109
99	0.1454	0.1494	**0.1388**	0.1450	0.1414

6.4 Baseline comparison on benchmark WDNs

In this set of experiments we compared the performance of *GATRes-small* against two state-of-the-art baseline models, GraphConvWat (Hajgató et al., 2021) and mGCN (Ashraf et al., 2023). We evaluated the three models on four benchmark WDNs: Anytown, C-Town, Richmond, and L-Town, described in Section 5.1. The experiments were conducted following the method proposed by (Hajgató et al., 2020) to ensure a fair comparison. Specifically, 1,000, 10,000, and 20,000 snapshots were generated for the C-Town, L-Town, and Richmond WDNs, respectively. Then, the datasets were split into training, validation, and test sets in a 6:2:2 ratio.

We used the same experimental settings as proposed in the baseline approaches to guarantee a fair comparison. Thus, in all experiments the models are trained for 2,000 epochs with early stopping, using the Adam gradient-based optimization algorithm (Kingma & Ba, 2014). The GraphConvWat model training was stopped if the validation loss did not improve for 50 consecutive epochs. In the case of mGCN, the training was stopped if no improvement is seen after 250 epochs. Likewise mGCN, our model training is stopped after 250 epochs if no improvement is observed. In all cases, it is considered an improvement when the validation loss decreases at least by *1e-6*.

The evaluation of the models' performance on each WDN, with the exception of Anytown, was using data that included realistic demand patterns per node. In the case of C-Town and Richmond, the WDN snapshots for evaluation were created using a 24-hour demand pattern time series sampled at 5 minutes interval. L-Town evaluation snapshots were created using a 1-week demand pattern time series sampled at 5 minutes interval. Table 10 shows the results of the performance comparison of ten runs per WDN, and then the mean and standard deviation are reported. In order to make the test deterministic we hard coded 10 different seeds, one for each run. Then, the same seed was

used for each run, for all three models, to ensure a fair comparison of the results. As can be seen from the table, our model *GATRes-small* achieves the lowest MAE in all WDNs and the lowest MAPE in all networks but Richmond. Likewise, *GATRes-small* achieves the higher NSE in all WDNs but Richmond.

Table 10. Models performance comparison on 24-hour demand pattern time series data.

WDN	Metrics	Models		
		GraphConvWat	mGCN	GATRes-small (ours)
C-Town	MAE (↓)	14.8860 ±0.1418	19.9138 ±0.0948	**9.4860** ±0.1822
	MAPE (↓)	0.1028 ±0.0009	0.1318 ±0.0005	**0.0690** ±0.0010
	NSE (↑)	0.7870 ±0.0046	0.6310 ±0.0030	**0.8480** ±0.0075
Richmond	MAE (↓)	4.3501 ±0.0170	2.9690 ±0.0283	**2.8114** ±0.0899
	MAPE (↓)	0.0196 ±0.0001	**0.0128** ±0.0002	0.0133 ±0.0005
	NSE (↑)	0.9500 ±0.0000	**0.9630** ±0.0046	0.9390 ±0.0030
L-Town	MAE (↓)	3.4505 ±0.0129	1.5928 ±0.0050	**0.9501** ±0.0086
	MAPE (↓)	0.0611 ±0.0002	0.0305 ±0.0001	**0.0157** ±0.0002
	NSE (↑)	0.5040 ±0.0049	0.8000 ±0.0000	**0.9000** ±0.0000

One limitation of previous approaches is the evaluation of model performance on unrealistic data, i.e., an exact copy of the training data distribution (Section 2.2). In previous approaches, the snapshots representing random WDN states used for training, validation and test are created by the same algorithm. Consequently, the distribution of the data used for testing is a fidelity copy of the data used for training. However, in practice, the distribution of the real data differs from the data used for training the reconstruction models, as explained in Section 2.2. Therefore, it is important that the models adapt to circumvent such uncertainties. Previous approaches achieve impressive performance when tested on replicas of the training data (see Table 11), but the performance drop is evident when they are evaluated on a realistic scenario (see Table 10).

The density distributions of training and test sets in C-Town, Richmond and L-Town WDNs are shown in Figure 9. It is clear that the distributions of the training and testing data created by the same algorithm (mathematical simulation) are identical, while the distribution of the test data with a demand pattern greatly differs from the one used during training. This shows the ability of *GATRes* to adapt to the changes that occur in real life scenarios. It also shows that other models achieve better results only when evaluated on fidelity copies of the training data, caused by overfitting due to the large model complexity of the previous approaches. The density distribution plot in Figure 9(b) explains the good performance of mGCN on Richmond WDN in terms of MAPE and NSE (Table 10), the time-based demand pattern test dataset has a similar distribution than the one used for training.

6.5 Ablation study

The ablation study presented in this section evaluates the importance of the different components of *GATRes* model architecture, and the effect of their removal or alteration on performance. In every run, a specific component is removed or altered, and *GATRes* is restored to its original version before a new change is made. The different variants used in the ablation study are the following:

Table 11. Models performance comparison on synthetic sampling-based snapshots following (Hajgató et al., 2020) approach.

WDN	Metrics	Models		
		GraphConvWat	mGCN	GATRes-small (ours)
Anytown	MAE (↓)	5.1044 ±0.0714	3.9460 ±0.0642	**3.9245** ±0.1056
	MAPE (↓)	0.0654 ±0.0012	0.0497 ±0.0009	**0.0491** ±0.0012
	NSE (↑)	0.7440 ±0.0049	**0.8020** ±0.0075	0.7980 ±0.0189
C-Town	MAE (↓)	4.1619 ±0.0170	**1.6963** ±0.0133	1.8928 ±0.0149
	MAPE (↓)	0.0354 ±0.0001	**0.0148** ±0.0001	0.0169 ±0.0001
	NSE (↑)	0.9640 ±0.0049	**0.9900** ±0.0000	**0.9900** ±0.0000
Richmond	MAE (↓)	2.3999 ±0.0069	**0.6363** ±0.0061	1.5979 ±0.0106
	MAPE (↓)	0.0110 ±0.0000	**0.0029** ±0.0000	0.0080 ±0.0001
	NSE (↑)	0.9805 ±0.0003	**0.9900** ±0.0000	0.9750 ±0.0009
L-Town	MAE (↓)	1.2970 ±0.0036	**0.2441** ±0.0014	0.4930 ±0.0028
	MAPE (↓)	0.0159 ±0.0000	**0.0030** ±0.0000	0.0061 ±0.0000
	NSE (↑)	0.9700 ±0.0000	**1.0000** ±0.0000	**1.0000** ±0.0000

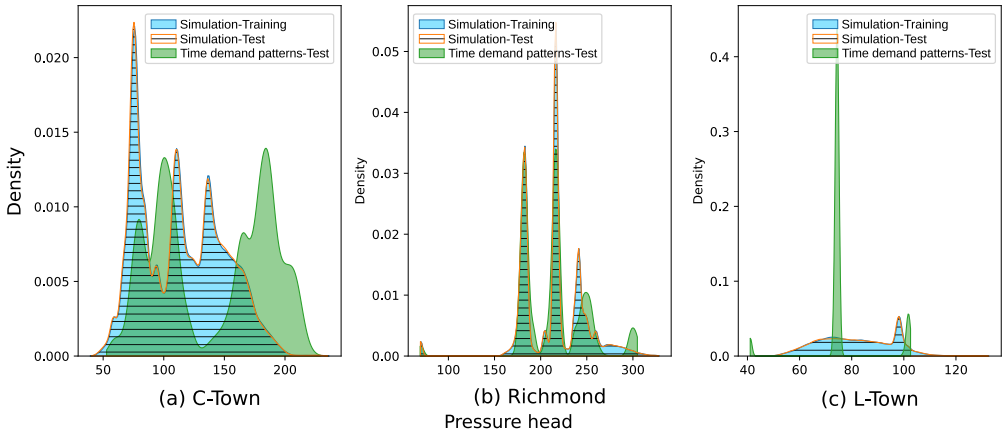

Figure 9. Comparison of density distributions of synthetic and time-based datasets in C-Town, Richmond and L-Town WDNs. The Simulation-training and Simulation-test curves represent the density of pressure heads created without including demand patterns. On the contrary, Time demand patterns-test represent the data created using the demand-patterns time series.

Without Residual Connections (woResCon). The residual connections used within each *GATRes* Block are removed.

Without Mean Aggregation (woMeanAggr). The Mean Aggregation applied after the second convolution within each *GATRes* Block is removed and the residual connection is added to the output of the second convolution.

Without Residual Connection and Without Mean Aggregation (woResConwoAggr). Both, the residual connection and the mean aggregation are removed from the *GATRes* Block.

Mean Aggregation Outside the Block (MeanAggrOut). Instead of applying a mean aggregation within each block, it is applied only once in the forward pass, after the output of the last *GATRes* Block and before the final Linear layer.

These experiments were performed by training *GATRes-small* on the C-Town WDN. As can be seen in Table 12, removing the Residual Connections produced the highest negative impact on model performance for all metrics.

Table 12. Ablation study of *GATRes* evaluated on C-Town 24-hour time series data.

Variants	MAE (↓)	MAPE (↓)	NSE (↑)
GATRes-small	**09.4860** ±0.1822	0.0690 ±0.001	**0.8480** ±0.0075
woMeanAggr	09.7479 ±0.1444	**0.0686** ±0.0009	0.8473 ±0.0046
woResCon-woAggr	10.0934 ±0.1644	0.0735 ±0.0010	0.8150 ±0.0062
MeanAggrOut	10.3735 ±0.1251	0.0735 ±0.0006	0.8333 ±0.0058
woResCon	11.5362 ±0.1697	0.0815 ±0.0008	0.7694 ±0.0089

7 Discussion

In this section, we generally discuss our findings and technical changes that affect our model in estimating pressures on Water Distribution Networks. We first review changes that made *GATRes* versions outperform other baselines and their limitations. Then, we discuss the role of synthetic data and the relationship between hydraulic simulation and data-driven models. Finally, we address the question of generalizability in the context of our research.

7.1 General findings and limitation

We first remark that our *GATRes* satisfies the predefined criteria: **(ℂ1)Generalizability** by *GATRes* being a spatial-based GNN approach that has topology awareness in its decision, **(ℂ2)Adaptability** by random masking that dynamically changes sensor positions and myriad contextual snapshots from our data generation tool, and **(ℂ3)Robustness** by effectively evaluating the model on unseen time-relevant data with respect to uncertainty conditions.

In addition, *GATRes* achieved pressure reconstruction with an average relative error of 7% and an absolute error of 1.93 water column meters on a 95% masking rate (see Table 4). We attribute its success primarily to the fundamental blocks and training strategy. These blocks update the weights of connections using nodal features and, therefore, relax the original topology in a given Water Distribution Network. This relaxation provides robustness and generalizability to *GATRes* in uncertain conditions and across diverse network topologies, which may vary in size, headloss formula, and component configurations. Furthermore, *GATRes* utilizes a random sensor replacement strategy, eliminating the need for time-consuming retraining when a new sensor is introduced in the future. For these reasons, both blocks within the architecture and training strategy sharpen a *GATRes* as a highly reusable and sustainable solution for predicting pressures in numerous Water Distribution Networks.

The significance of the improvement in performance of our model with respect to previous approaches was assessed using the Wilcoxon signed-rank and the paired t-test tests. The results obtained for C-Town, Richmond, and L-Town benchmark datasets show to be statistically significant with a value $p \ll 0.05$, and comparable to the performance of mGCN model only for Richmond dataset. In terms of applicability, previous studies found a relationship between leakage and pressure, and suggested that proper pressure management is crucial for reducing water loss (Zhou et al., 2019b; Y. Li et al., 2022). Therefore, having a model that improves the pressure estimation, and taking into account that such improvement is statistically significant, the applicability of our model for solv-

ing problems related to leak detection and localization is worth to consider as a future work.

However, it is essential to acknowledge the limitations when *GATRes* comes to scale. The limit becomes apparent when comparing the larger and smaller versions of *GATRes* in Tables 4 and 5. The larger *GATRes* eventually reaches a saturation point of performance and is surpassed by its smaller counterpart. The same finding is available in both GATConvWat variants. They likely originate from inherent issues in graph neural networks, such as over-smoothing and over-squashing, where nodes tend to propagate redundant information excessively (Di Giovanni et al., 2023). While *GATRes* employs residual connections that partially mitigate the over-smoothing and mainly contribute to the model performance, as shown in Section 6.5, they are unable to eliminate this phenomenon completely (Kipf & Welling, 2017). To address these issues in the future, potential solutions may include exploring graph rewiring strategies and subgraph sampling techniques.

7.2 Benefit of synthetic data

Incorporating generated snapshots into the training of deep models not only enhances their capabilities but also highlights the value of synthetic data, especially in situations where dynamic data is limited or sensor placements change. These common issues have been found in many public benchmarks, such as five reviewed water networks in Section 5.1 due to the missing historical patterns and privacy issues. They have made reproducibility a persistent challenge in water network research.

As a solution, our data generation tool extends the limits in approaching these public networks without confidential matters. For practical purposes, the synthesized training set could involve as many cases as possible, reducing the risk of long-term incidents that may not have occurred in historical records. Thus, it boosts model robustness when dealing with unforeseen scenarios.

7.3 Relationship between hydraulic simulations and *GATRes*

Yet, an intriguing question arises: Can we replace traditional mathematical simulations with data-driven models like *GATRes*? Conventional simulation bridges the interaction between hydraulic experts and the Water Distribution Network in water management. Such an interaction should be preserved in the design or analysis phase. In the deployment, especially for Digital Twin or water systems on big data, pressure estimate models often deal with heavy computation and require a low response time (Pesantez et al., 2022). In this case, *GATRes* and GNN variants can be alternative approaches due to their competitive results and impressive throughput (see Table 6). However, these deep models may involve the risk of over-relaxation of energy conservation laws and other constraints within the actual networks. The risk is often minimal in pure physics-based simulations.

Accordingly, these simulations still play a critical role in data synthesis as they define a valid boundary for newly created models thanks to their generated training samples and testing environments. When fast computation is required, *GATRes* is a good alternative to estimate the pressure of a large WDN given unlimited sensor streams. In the future, it is potential to focus on physics-inspired models that can regularize *GATRes* to preserve fundamental physical laws and yield more confident results.

7.4 Generalization

GATRes is able to generalize to previously unseen WDNs by design given the ability of spatial methods (e.g. *GAT*) to generalize across graphs (Bronstein et al., 2017). On the contrary, previous works that rely on spectral approaches suffer from the gen-

eralization problem because their convolutions (e.g. ChebNet) depend on the eigen-functions of the Laplacian matrix of a particular graph (S. Zhang et al., 2019).

GATRes trained on multiple WDNs simultaneously produced a MAE of 3.06m and a MAPE of 9.98%, on average, at zero-shot inference on 24-hour Oosterbeek WDN. These results shows a promising direction given that pressure estimation was performed on a completely different, previously unseen, WDN and they do not deviate significantly from those obtained with the GATRes-small model. The zero-shot inference offers immediate monitoring of a target WDN given a sufficiently pre-trained model, eliminating the need for data generation and model training specific to each WDN. These results can be improved by adding more WDNs in the pre-training data and exploring self-supervised pre-text tasks like nodal degree estimation, link predictions, and shortest paths. Moreover, the fine-tuned model, on the target dataset Oosterbeek, produced a reduction in MAE of 1.41% with respect to the model trained directly and only on Oosterbeek.

The results of our first attempts towards generalization (see Table 7) show that our approach is worth further exploration. GNN models fail to generalize when the local structures of the graphs in the training data differ from the local structures in the test data (Yehudai et al., 2021). In our case, the term "local structures" does not refer only to topology but also to water system components. By training the model with several WDNs and myriad scenarios, we can better prepare for unforeseen configurations and potentially include or approximate them within our dataset. This can mitigate the impact of the heterogeneity in terms of number and type of system components. Then, a possible explanation of the generalization capabilities of *GATRes* is the training on several WDNs simultaneously. Using graphs that differ in size and structure, for training, allows *GATRes* to learn a richer set of local structures that may be present in the target WDN.

Despite these promising results, several questions remain unanswered. For example, how to choose the right WDNs in order to enrich the training data in terms of local structures' diversity? How to design a pre-trained task that can effectively capture the local-level patterns and extrapolate those to unseen larger graphs? How to train a GNN-based foundation model, in the Water Management Domain, that can be applied on different downstream tasks on any WDN topology? All these questions open paths for future research directions.

8 Conclusions

In this work we presented a hybrid, physics-based and data-driven, approach to address the problem of state estimation in WDNs. We leveraged mathematical simulation tools and GNNs to reconstruct the missing pressures at 95% of the junctions in the network, from only 5% of them seen during training. We also tested our approach on more extreme cases of sensor sparcity, reaching up to 99% masking rate. The outcome of our work is a new state-of-the-art for pressure estimation in WDNs, with two main contributions. First, a training strategy that relies on random sensor placement making our GNN-based estimation model robust to unexpected sensor location changes. Second, a realistic evaluation protocol that considers real temporal patterns and noise injection to mimic the uncertainties intrinsic to real-world scenarios.

Our model was evaluated on a large-scale network, in The Netherlands, showing a clear improvement over previous approaches. *GATRes* obtained an average MAE of 1.94m, which represents an 8.57% improvement with respect to other models. Similarly, it showed an average reduction of MAE of ≈40% on other WDN benchmarks with respect to previous approaches. We attribute the high performance of *GATRes* to its building blocks and training strategy. These blocks relax the original topology leveraging nodal features to re-weight the connections by means of an attention mechanism. Despite its

success, there are still some aspects that demand further exploration. On the one hand, while the residual connections mitigate the over-smoothing problem, inherent to GNNs, the phenomenon is not completely removed. Therefore, other techniques such as graph rewiring and subgraph sampling would be a fruitful area for further work. On the other hand, our multi-graph pre-training strategy is a promising direction towards model generalization and transferability in the WDN domain. Nonetheless, further research is needed to explore the connection between network topologies and pre-training tasks, as well as the applications of state estimation models such as pipe burst identification, leak localization, and anomaly detection.

Open Research

Data Availability Statement

In this section, we provide an overview of the publicly available benchmark water distribution networks and libraries that were employed in our study. Specifically, three networks, including Anytown (Walski et al., 1987), C-Town (Ostfeld et al., 2012), and Richmond (Van Zyl, 2001), are included in (Hajgató et al., 2021). The L-town dataset is referenced in the paper by (Vrachimis et al., 2022), while the Ky13 benchmark (Hernadez et al., 2016) can be readily obtained via a free download on `https://www.uky.edu/WDST/database.html`. The "Large" network is referenced in the "Availability of Data and Materials" section in (Sitzenfrei et al., 2023). The Oosterbeek water network is not publicly available, as it is provided under confidentiality by the water provider Vitens.

Software Availability Statement

The data generation tool was constructed using the Epynet wrapper, a third party library, publicly available on `https://github.com/Vitens/epynet`, and is licensed under Apache-2.0. We also leveraged the WNTR library (Klise et al., 2018) and Ray version 2.3.1 (Moritz et al., 2018) in our implementation.

Our datasets are organized in the *zarr* format, a file storage structure created using the Zarr-Python package version 2.14.2 (Miles et al., 2020), which is licensed under MIT. These datasets were employed in training both baseline models and *GATRes* variants using PyTorch (Paszke et al., 2019) and PyTorch Geometric (Fey & Lenssen, 2019). The dataset generation tool and *GATRes* model are publicly available on `https://github.com/DiTEC-project/gnn-pressure-estimation` (Truong et al., 2023).

Acknowledgments

This work is funded by the project DiTEC: Digital Twin for Evolutionary Changes in Water Networks (NWO 19454). We express our appreciation to Ton Blom and the Digital Twin group at Vitens, a Dutch drinking water company, for providing hydraulic knowledge and valuable data. Furthermore, we are grateful to Prof. A. Veldman for our insightful discussions. Also, we thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Hábrók high performance computing cluster. We also thank M. Hadadian, F. Blaauw and Researchable for discussions about the experiments platform.

References

Arsene, C. T., & Gabrys, B. (2014). Mixed simulation-state estimation of water distribution systems based on a least squares loop flows state estimator. *Applied Mathematical Modelling*, *38*(2), 599-619. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0307904X13004149` doi: https://doi.org/10.1016/j.apm.2013.06.012

Ashraf, I., Hermes, L., Artelt, A., & Hammer, B. (2023). Spatial graph convolution neural networks for water distribution systems. In *International symposium on intelligent data analysis* (pp. 29–41).

Barceló, P., Kostylev, E. V., Monet, M., Pérez, J., Reutter, J., & Silva, J. P. (2020). The logical expressiveness of graph neural networks. In *International conference on learning representations.* Retrieved from https://openreview.net/forum?id=r1lZ7AEKvB

Bickel, S., Brückner, M., & Scheffer, T. (2007). Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on machine learning* (pp. 81–88).

Biehl, M. (2023). *The shallow and the deep: A biased introduction to neural networks and old school machine learning.* University of Groningen Press.

Bonilla, C. A., Zanfei, A., Brentan, B., Montalvo, I., & Izquierdo, J. (2022). A digital twin of a water distribution system by using graph convolutional networks for pump speed-based state estimation. *Water*, *14*(4), 514.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, *34*(4), 18–42.

Campos, M. A. S., Carvalho, S. L., Melo, S. K., Gonçalves, G. B. F. R., dos Santos, J. R., Barros, R. L., ... Abreu Reis, R. P. (2021). Impact of the covid-19 pandemic on water consumption behaviour. *Water Supply*, *21*(8), 4058–4067.

Chen, D., Lin, Y., Li, W., Li, P., Zhou, J., & Sun, X. (2020, Apr.). Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(04), 3438-3445. Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/5747 doi: 10.1609/aaai.v34i04.5747

Chen, M., Wei, Z., Huang, Z., Ding, B., & Li, Y. (2020, 13–18 Jul). Simple and deep graph convolutional networks. In H. D. III & A. Singh (Eds.), *Proceedings of the 37th international conference on machine learning* (Vol. 119, pp. 1725–1735). PMLR. Retrieved from https://proceedings.mlr.press/v119/chen20v.html

Chen, Z., Wang, Y., Zhao, B., Cheng, J., Zhao, X., & Duan, Z. (2020). Knowledge graph completion: A review. *Ieee Access*, *8*, 192435–192456.

Christodoulou, S. E., Fragiadakis, M., Agathokleous, A., & Xanthos, S. (2018). Chapter 1 - introduction. In S. E. Christodoulou, M. Fragiadakis, A. Agathokleous, & S. Xanthos (Eds.), *Urban water distribution networks* (p. 1-20). Butterworth-Heinemann. Retrieved from https://www.sciencedirect.com/science/article/pii/B9780128136522000013 doi: https://doi.org/10.1016/B978-0-12-813652-2.00001-3

Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 18613–18624). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2020/file/d85b63ef0ccb114d0a3bb7b7d808028f-Paper.pdf

Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th international conference on neural information processing systems* (p. 3844–3852). Red Hook, NY, USA: Curran Associates Inc.

Derrow-Pinion, A., She, J., Wong, D., Lange, O., Hester, T., Perez, L., ... Velickovic, P. (2021a). Eta prediction with graph neural networks in google maps. In *Proceedings of the 30th acm international conference on information & knowledge management* (p. 3767–3776). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3459637.3481916 doi: 10.1145/3459637.3481916

Derrow-Pinion, A., She, J., Wong, D., Lange, O., Hester, T., Perez, L., ... others (2021b). Eta prediction with graph neural networks in google maps. In *Proceedings of the 30th acm international conference on information & knowledge management* (pp. 3767–3776).

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Di Giovanni, F., Giusti, L., Barbero, F., Luise, G., Lio, P., & Bronstein, M. M. (2023). On over-squashing in message passing neural networks: The impact of width, depth, and topology. In *International conference on machine learning* (pp. 7865–7885).

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *International conference on learning representations*. Retrieved from `https://openreview.net/forum?id=YicbFdNTTy`

Du, K., Ding, R.-y., Wang, Z.-h., Song, Z.-g., Xu, B.-f., Zhou, M., ... Zhang, J. (2018). Direct inversion algorithm for pipe resistance coefficient calibration of water distribution systems. *Journal of Water Resources Planning and Management*, *144*(7), 04018027.

Fang, Z., Li, Y., Lu, J., Dong, J., Han, B., & Liu, F. (2022). Is out-of-distribution detection learnable? *Advances in Neural Information Processing Systems*, *35*, 37199–37213.

Farquhar, S., & Gal, Y. (2022). What 'out-of-distribution' is and is not. In *Neurips ml safety workshop*. Retrieved from `https://openreview.net/forum?id=XCS_zBHQA2i`

Fey, M., & Lenssen, J. E. (2019). Fast graph representation learning with PyTorch Geometric [Software]. In *Iclr workshop on representation learning on graphs and manifolds*.

Fu, M., Rong, K., Huang, Y., Zhang, M., Zheng, L., Zheng, J., ... Yaseen, Z. M. (2022). Graph neural network for integrated water network partitioning and dynamic district metered areas. *Scientific Reports*, *12*(1), 19466.

Garðarsson, G. Ö., Boem, F., & Toni, L. (2022). Graph-based learning for leak detection and localisation in water distribution networks*. *IFAC-PapersOnLine*, *55*(6), 661-666. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2405896322005882` (11th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS 2022) doi: https://doi.org/10.1016/j.ifacol.2022.07.203

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *Proceedings of the 34th international conference on machine learning - volume 70* (p. 1263–1272). JMLR.org.

Hajgató, G., Gyires-Tóth, B., & Paál, G. (2021). *GraphConvWat*. `https://github.com/BME-SmartLab/GraphConvWat`. GitHub.

Hajgató, G., Gyires-Tóth, B., & Paál, G. (2021). Reconstructing nodal pressures in water distribution systems with graph neural networks. *arXiv preprint arXiv:2104.13619*.

Hajgató, G., Paál, G., & Gyires-Tóth, B. (2020). Deep reinforcement learning for real-time optimization of pumps in water distribution systems. *Journal of Water Resources Planning and Management*, *146*(11), 04020079. doi: 10.1061/(ASCE)WR.1943-5452.0001287

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 770–778).

Hendrycks, D., & Gimpel, K. (2017). A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th international conference on learning representations, ICLR 2017, toulon, france, april 24-*

26, 2017, conference track proceedings. OpenReview.net. Retrieved from https://openreview.net/forum?id=Hkg4TI9xl

Hernadez, E., Hoagland, S., & Ormsbee, L. (2016). Water distribution database for research applications [Dataset]. In *World environmental and water resources congress 2016* (p. 465-474). Retrieved from https://ascelibrary.org/doi/abs/10.1061/9780784479865.049 doi: 10.1061/9780784479865.049

Iwakin, O., & Moazeni, F. (2024). Improving urban water demand forecast using conformal prediction-based hybrid machine learning models. *Journal of Water Process Engineering*, *58*, 104721. Retrieved from https://www.sciencedirect.com/science/article/pii/S2214714423012412 doi: https://doi.org/10.1016/j.jwpe.2023.104721

Jiang, W., & Luo, J. (2022). Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications*, *207*, 117921.

Kang, D., & Lansey, K. (2009). Real-time demand estimation and confidence limit analysis for water distribution systems. *Journal of Hydraulic Engineering*, *135*(10), 825-837. Retrieved from https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29HY.1943-7900.0000086 doi: 10.1061/(ASCE)HY.1943-7900.0000086

Kerimov, B., Bentivoglio, R., Garzón, A., Isufi, E., Tscheikner-Gratl, F., Steffelbauer, D. B., & Taormina, R. (2023, 10). Assessing the performances and transferability of graph neural network metamodels for water distribution systems. *Journal of Hydroinformatics*, *25*(6), 2223-2234. Retrieved from https://doi.org/10.2166/hydro.2023.031 doi: 10.2166/hydro.2023.031

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International conference on learning representations.* Retrieved from https://openreview.net/forum?id=SJU4ayYgl

Klise, K. A., Murray, R., & Haxton, T. (2018). An overview of the water network tool for resilience (wntr). In *Proceedings of the 1st international wdsa/ccwi joint conference, kingston, ontario, canada.* Sandia National Lab.(SNL-NM), Albuquerque, NM (United States).

Koşucu, M. M., Albay, E., & Demirel, M. C. (2022). Extending epanet hydraulic solver capacity with rigid water column global gradient algorithm. *Journal of Hydro-environment Research*, *42*, 31–43.

Kumar, S. M., Narasimhan, S., & Bhallamudi, S. M. (2008). State estimation in water distribution networks using graph-theoretic reduction strategy. *Journal of Water Resources Planning and Management*, *134*(5), 395-403. Retrieved from https://ascelibrary.org/doi/abs/10.1061/%28ASCE%290733-9496%282008%29134%3A5%28395%29 doi: 10.1061/(ASCE)0733-9496(2008)134:5(395)

Legates, D. R., & McCabe Jr., G. J. (1999). Evaluating the use of "goodness-of-fit" measures in hydrologic and hydroclimatic model validation. *Water Resources Research*, *35*(1), 233-241. Retrieved from https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/1998WR900018 doi: https://doi.org/10.1029/1998WR900018

Li, Y., Gao, J., Shen, C., Guan, Y., & Wang, W. (2022). Estimation of leak area-pressure relationship for cracks on water pipes using models based on linear-elastic fracture mechanics. *Water Research*, *221*, 118692.

Li, Z., Liu, H., Zhang, C., & Fu, G. (2024a). Gated graph neural networks for identifying contamination sources in water distribution systems. *Journal of Environmental Management*, *351*, 119806.

Li, Z., Liu, H., Zhang, C., & Fu, G. (2024b). Real-time water quality prediction in water distribution networks using graph neural networks with sparse monitoring data. *Water Research*, *250*, 121018.

Lima, G. M., Brentan, B. M., Manzi, D., & Luvizotto Jr, E. (2018). Metamodel for nodal pressure estimation at near real-time in water distribution systems using artificial neural networks. *Journal of Hydroinformatics*, *20*(2), 486–496.

Martínez, F., Hernández, V., Alonso, J. M., Rao, Z., & Alvisi, S. (2007, 01). Optimizing the operation of the Valencia water-distribution network. *Journal of Hydroinformatics*, *9*(1), 65-78. Retrieved from `https://doi.org/10.2166/hydro.2006.018` doi: 10.2166/hydro.2006.018

Meirelles, G., Manzi, D., Brentan, B., Goulart, T., & Luvizotto, E. (2017). Calibration model for water distribution network using pressures estimated by artificial neural networks. *Water Resources Management*, *31*, 4339–4351.

Menapace, A., Avesani, D., Righetti, M., Bellin, A., & Pisaturo, G. (2018). Uniformly distributed demand epanet extension. *Water resources management*, *32*, 2165–2180.

Menapace, A., Zanfei, A., Felicetti, M., Avesani, D., Righetti, M., & Gargano, R. (2020). Burst detection in water distribution systems: The issue of dataset collection. *Applied Sciences*, *10*(22). Retrieved from `https://www.mdpi.com/2076-3417/10/22/8219` doi: 10.3390/app10228219

Miles, A., Kirkham, J., Durant, M., Bourbeau, J., Onalan, T., Hamman, J., ... Schut, V. (2020). *zarr-developers/zarr-python: v2. 4.0* [Software]. doi: 10.5281/zenodo.3773450

Moritz, P., Nishihara, R., Wang, S., Tumanov, A., Liaw, R., Liang, E., ... Stoica, I. (2018). Ray: A distributed framework for emerging ai applications [Software]. In *Proceedings of the 13th usenix conference on operating systems design and implementation* (p. 561–577). USA: USENIX Association.

Mücke, N. T., Pandey, P., Jain, S., Bohté, S. M., & Oosterlee, C. W. (2023). A probabilistic digital twin for leak localization in water distribution networks using generative deep learning. *Sensors*, *23*(13). Retrieved from `https://www.mdpi.com/1424-8220/23/13/6179` doi: 10.3390/s23136179

Nerantzis, D., Pecci, F., & Stoianov, I. (2020). Optimal control of water distribution networks without storage. *European Journal of Operational Research*, *284*(1), 345-354. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0377221719310124` doi: https://doi.org/10.1016/j.ejor.2019.12.011

Nguyen, T., Le, H., Quinn, T. P., Nguyen, T., Le, T. D., & Venkatesh, S. (2020, 10). GraphDTA: predicting drug–target binding affinity with graph neural networks. *Bioinformatics*, *37*(8), 1140-1147. Retrieved from `https://doi.org/10.1093/bioinformatics/btaa921` doi: 10.1093/bioinformatics/btaa921

Ostfeld, A., Salomons, E., Ormsbee, L., Uber, J. G., Bros, C. M., Kalungi, P., ... others (2012). Battle of the water calibration networks [Dataset]. *Journal of water resources planning and management*, *138*(5), 523–532. doi: 10.1061/(ASCE)WR.1943-5452.0000191

Paez, D., & Filion, Y. (2017). Generation and validation of synthetic wds case studies using graph theory and reliability indexes. *Procedia Engineering*, *186*, 143-151. Retrieved from `https://www.sciencedirect.com/science/article/pii/S1877705817313693` (XVIII International Conference on Water Distribution Systems, WDSA2016) doi: https://doi.org/10.1016/j.proeng.2017.03.220

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345–1359.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... others (2019). Pytorch: An imperative style, high-performance deep learning library [Software]. *Advances in neural information processing systems*, *32*.

Pesantez, J. E., Alghamdi, F., Sabu, S., Mahinthakumar, G., & Berglund, E. Z. (2022). Using a digital twin to explore water infrastructure impacts during the covid-19 pandemic. *Sustainable Cities and Society*, *77*, 103520. Retrieved from `https://www.sciencedirect.com/science/article/pii/`

S2210670721007861 doi: https://doi.org/10.1016/j.scs.2021.103520

Reiser, P., Neubert, M., Eberhard, A., Torresi, L., Zhou, C., Shao, C., ... others (2022). Graph neural networks for materials science and chemistry. *Communications Materials*, *3*(1), 93.

Rossman, L. A. (1999). The epanet programmer's toolkit for analysis of water distribution systems. In *Wrpmd'99: Preparing for the 21st century* (pp. 1–10).

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Ruiz, E., Díaz, S., & González, J. (2022). Potential performance of hydraulic state estimation in water distribution networks. *Water Resources Management*, 1–18.

Shlomi, J., Battaglia, P., & Vlimant, J.-R. (2020, dec). Graph neural networks in particle physics. *Machine Learning: Science and Technology*, *2*(2), 021001. Retrieved from https://dx.doi.org/10.1088/2632-2153/abbf9a doi: 10.1088/2632-2153/abbf9a

Simpson, A., & Elhay, S. (2011). Jacobian matrix for solving water distribution system equations with the darcy-weisbach head-loss model. *Journal of Hydraulic Engineering*, *137*(6), 696-700. Retrieved from https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29HY.1943-7900.0000341 doi: 10.1061/(ASCE)HY.1943-7900.0000341

Sitzenfrei, R., Hajibabaei, M., Hesarkazzazi, S., & Diao, K. (2023). Dual graph characteristics of water distribution networks—how optimal are design solutions? [Dataset]. *Complex & Intelligent Systems*, *9*(1), 147–160. doi: 10.1007/s40747-022-00797-4

Tan, M., & Le, Q. (2019, 09–15 Jun). EfficientNet: Rethinking model scaling for convolutional neural networks. In K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (Vol. 97, pp. 6105–6114). PMLR. Retrieved from https://proceedings.mlr.press/v97/tan19a.html

Taormina, R., Galelli, S., Tippenhauer, N. O., Salomons, E., Ostfeld, A., Eliades, D. G., ... Ohar, Z. (2018). Battle of the attack detection algorithms: Disclosing cyber attacks on water distribution networks. *Journal of Water Resources Planning and Management*, *144*(8), 04018048. Retrieved from https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29WR.1943-5452.0000969 doi: 10.1061/(ASCE)WR.1943-5452.0000969

Tiedmann, H. R., Spearing, L. A., Sela, L., Kinney, K., Kirisits, M. J., Katz, L. E., ... Faust, K. M. (2022). Modeling in the covid-19 pandemic: Overcoming the water sector's data struggles to realize the potential of hydraulic models. *Journal of Water Resources Planning and Management*, *148*(6), 05022003. Retrieved from https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29WR.1943-5452.0001561 doi: 10.1061/(ASCE)WR.1943-5452.0001561

Todini, E., Santopietro, S., Gargano, R., & Rossman, L. A. (2021). Pressure flow–based algorithms for pressure-driven analysis of water distribution networks. *Journal of Water Resources Planning and Management*, *147*(8), 04021048.

Truong, H., Tello, A., Lazovik, A., & Degeler, V. (2023, November). *GATRes and Dataset generation tool* [Software]. Retrieved from https://github.com/DiTEC-project/gnn-pressure-estimation doi: 10.5281/zenodo.10159270

Tsiami, L., & Makropoulos, C. (2021). Cyber—physical attack detection in water distribution systems with temporal graph convolutional neural networks. *Water*, *13*(9), 1247.

Van Zyl, J. E. (2001). *A methodology for improved operational optimization of water distribution systems* (Dataset, University of Exeter UK). doi: 10.13140/RG.2.1.1117.7127

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. *Advances in neural informa-*

tion processing systems, *30*.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph Attention Networks. *International Conference on Learning Representations*. Retrieved from https://openreview.net/forum?id=rJXMpikCZ

Vrachimis, S. G., Eliades, D. G., Taormina, R., Kapelan, Z., Ostfeld, A., Liu, S., ... Polycarpou, M. M. (2022). Battle of the leakage detection and isolation methods [Dataset]. *Journal of Water Resources Planning and Management*, *148*(12), 04022068. doi: 10.1061/(ASCE)WR.1943-5452.0001601

Walski, T. M., Brill Jr, E. D., Gessler, J., Goulter, I. C., Jeppson, R. M., Lansey, K., ... others (1987). Battle of the network models: Epilogue [Dataset]. *Journal of Water Resources Planning and Management*, *113*(2), 191–203. doi: 10.1061/(ASCE)0733-9496(1987)113:2(191)

Wang, S., Taha, A. F., Gatsis, N., Sela, L., & Giacomoni, M. H. (2021). Probabilistic state estimation in water networks. *IEEE Transactions on Control Systems Technology*, *30*(2), 507–519.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4-24. doi: 10.1109/TNNLS.2020.2978386

Xing, L., & Sela, L. (2022). Graph neural networks for state estimation in water distribution systems: Application of supervised and semisupervised learning. *Journal of Water Resources Planning and Management*, *148*(5), 04022018. Retrieved from https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29WR.1943-5452.0001550 doi: 10.1061/(ASCE)WR.1943-5452.0001550

Xu, B., Shen, H., Sun, B., An, R., Cao, Q., & Cheng, X. (2021). Towards consumer loan fraud detection: Graph neural networks with role-constrained conditional random field. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 35, pp. 4537–4545).

Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.

Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). How powerful are graph neural networks? In *7th international conference on learning representations, ICLR 2019, new orleans, la, usa, may 6-9, 2019*. OpenReview.net. Retrieved from https://openreview.net/forum?id=ryGs6iA5Km

Yehudai, G., Fetaya, E., Meirom, E., Chechik, G., & Maron, H. (2021). From local structures to size generalization in graph neural networks. In *International conference on machine learning* (pp. 11975–11986).

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining* (p. 974–983). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3219819.3219890 doi: 10.1145/3219819.3219890

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? *Advances in neural information processing systems*, *27*.

Zanfei, A., Brentan, B. M., Menapace, A., Righetti, M., & Herrera, M. (2022). Graph convolutional recurrent neural networks for water demand forecasting. *Water Resources Research*, *58*(7), e2022WR032299.

Zanfei, A., Menapace, A., Brentan, B. M., Righetti, M., & Herrera, M. (2022). Novel approach for burst detection in water distribution systems based on graph neural networks. *Sustainable Cities and Society*, *86*, 104090.

Zanfei, A., Menapace, A., Brentan, B. M., Sitzenfrei, R., & Herrera, M. (2023). Shall we always use hydraulic models? a graph neural network metamodel for water system calibration and uncertainty assessment. *Water Research*, *242*, 120264. Retrieved from https://www.sciencedirect.com/science/article/

pii/S0043135423007005 doi: https://doi.org/10.1016/j.watres.2023.120264

Zeng, H., Zhou, H., Srivastava, A., Kannan, R., & Prasanna, V. (2020). Graphsaint: Graph sampling based inductive learning method. In *International conference on learning representations.* Retrieved from https://openreview.net/forum?id=BJe8pkHFwS

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). mixup: Beyond empirical risk minimization. In *International conference on learning representations.* Retrieved from https://openreview.net/forum?id=r1Ddp1-Rb

Zhang, S., Tong, H., Xu, J., & Maciejewski, R. (2019). Graph convolutional networks: a comprehensive review. *Computational Social Networks*, *6*(1), 1–23.

Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., ... Li, H. (2019). T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE transactions on intelligent transportation systems*, *21*(9), 3848–3858.

Zhou, X., Tang, Z., Xu, W., Meng, F., Chu, X., Xin, K., & Fu, G. (2019a). Deep learning identifies accurate burst locations in water distribution networks. *Water Research*, *166*, 115058. Retrieved from https://www.sciencedirect.com/science/article/pii/S0043135419308322 doi: https://doi.org/10.1016/j.watres.2019.115058

Zhou, X., Tang, Z., Xu, W., Meng, F., Chu, X., Xin, K., & Fu, G. (2019b). Deep learning identifies accurate burst locations in water distribution networks. *Water research*, *166*, 115058.

Zhou, X., Zhang, J., Guo, S., Liu, S., & Xin, K. (2023). A convenient and stable graph-based pressure estimation methodology for water distribution networks: Development and field validation. *Water Research*, *233*, 119747. Retrieved from https://www.sciencedirect.com/science/article/pii/S0043135423001823 doi: https://doi.org/10.1016/j.watres.2023.119747