Towards transferable metamodels for water distribution systems with edge-based graph neural networks

Kerimov, Bulat; Taormina, Riccardo; Tscheikner-Gratl, Franz

DOI
10.1016/j.watres.2024.121933

Publication date
2024

Document Version
Final published version

Published in
Water Research

Citation (APA)
Kerimov, B., Taormina, R., & Tscheikner-Gratl, F. (2024). Towards transferable metamodels for water distribution systems with edge-based graph neural networks. *Water Research*, *261*, Article 121933. https://doi.org/10.1016/j.watres.2024.121933

Water Research 261 (2024) 121933

Contents lists available at ScienceDirect

Water Research

journal homepage: www.elsevier.com/locate/watres

Towards transferable metamodels for water distribution systems with edge-based graph neural networks

Bulat Kerimov [a,*], Riccardo Taormina [b,1], Franz Tscheikner-Gratl [a,1]

[a] *Department of Civil and Environmental Engineering, Norwegian University of Science and Technology, Trondheim, Norway*
[b] *Department of Water Management, Faculty of Civil Engineering and Geosciences, Delft University of Technology, Delft, The Netherlands*

ARTICLE INFO

Keywords:
Water networks
Artificial intelligence
Transfer learning
Deep learning
GRAPH Neural Networks
EPANET

ABSTRACT

Data-driven metamodels reproduce the input-output mapping of physics-based models while significantly reducing simulation times. Such techniques are widely used in the design, control, and optimization of water distribution systems. Recent research highlights the potential of metamodels based on Graph Neural Networks as they efficiently leverage graph-structured characteristics of water distribution systems. Furthermore, these metamodels possess inductive biases that facilitate generalization to unseen topologies. Transferable metamodels are particularly advantageous for problems that require an efficient evaluation of many alternative layouts or when training data is scarce. However, the transferability of metamodels based on GNNs remains limited, due to the lack of representation of physical processes that occur on edge level, i.e. pipes. To address this limitation, our work introduces Edge-Based Graph Neural Networks, which extend the set of inductive biases and represent link-level processes in more detail than traditional Graph Neural Networks. Such an architecture is theoretically related to the constraints of mass conservation at the junctions. To verify our approach, we test the suitability of the edge-based network to estimate pipe flowrates and nodal pressures emulating steady-state EPANET simulations. We first compare the effectiveness of the metamodels on several benchmark water distribution systems against Graph Neural Networks. Then, we explore transferability by evaluating the performance on unseen systems. For each configuration, we calculate model performance metrics, such as coefficient of determination and speed-up with respect to the original numerical model. Our results show that the proposed method captures the pipe-level physical processes more accurately than node-based models. When tested on unseen water networks with a similar distribution of demands, our model retains a good generalization performance with a coefficient of determination of up to 0.98 for flowrates and up to 0.95 for predicted heads. Further developments could include simultaneous derivation of pressures and flowrates.

1. Introduction

1.1. Metamodels of water distribution systems

Hydraulic models are essential for the design, management, and control of water distribution systems (WDS). These physics-based models, such as EPANET (Rossman, 2022), usually solve the mass and energy conservation equations to estimate the steady state of the system at any given time, i.e., the flow rates in every pipe and pressures at all nodes. These models utilize the information about network layout and static component settings (e.g. pipe and pumps parameters, and elevation) as well as hydraulical parameters (e.g., reservoir heads and demands) in order to provide reliable results. However, they can be too time intensive for certain applications that require a multitude of simulations, especially for large WDSs. That includes, but is not limited to the optimization of network design (Bi and Dandy, 2014), criticality assessment of the network parts (Meijer et al., 2021), and real-time control (Pasha and Lansey, 2014). This is of increasing importance as digitalization steadily revolutionizes the water sector (Makropoulos and Savić, 2019). In order to meet the necessary criteria for implementing a comprehensive digital twin of a WDS (Fuertes et al., 2020), utilities can resort to surrogate models, commonly referred to as metamodels (Garzón et al., 2022) to allow for timely results for the given tasks. These metamodels play a crucial role in integrating the hydraulic model with

* Corresponding author.
E-mail addresses: bulat.kerimov@ntnu.no (B. Kerimov), r.taormina@tudelft.nl (R. Taormina), franz.tscheikner-gratl@ntnu.no (F. Tscheikner-Gratl).
[1] These authors share senior authorship

https://doi.org/10.1016/j.watres.2024.121933
Received 5 November 2023; Received in revised form 7 June 2024; Accepted 11 June 2024
Available online 20 June 2024

various sources of information and employing advanced analytics, ultimately enabling efficient computations.

Metamodels are designed to decrease simulation time while maintaining comparability to the results of the original hydraulic model. Machine learning methods, such as artificial neural networks (ANN) successfully approximate the input-output relation of the physics-based model to obtain results in a fraction of the time while still retaining a sufficient accuracy (Bi and Dandy, 2014; Garzón et al., 2022). As such, ANN-based metamodels were used in near real-time modeling and extended period simulation based on pressure metrics (Lima et al., 2017). Similarly, these models were adopted to speed-up the water quality optimization with genetic algorithms (Broad et al., 2004). Although ANNs efficiently leverage network parameters such as elevation, pipe roughness, and length, they omit the topological information and the connectivity of the WDS. This information, however, is a vital element of an accurate approximation and an efficient model transfer between varying network layouts and configurations. Here we denote transferability as the capacity of a model to preserve predicting power for WDSs or the variations of a WDS that have not been seen during the training process (Levie et al., 2019; Neuman and Bramburger, 2023). This feature opens the way for solving a plethora of design problems, e.g., when the design of a new or expansion of the existing WDS requires looking at alternative topological options (McClymont et al., 2015; Mala-Jetmarova et al., 2018). In that regard, one would need to re-train a current metamodel for every considered layout, which defeats its purpose.

1.2. Graph neural networks in water distribution system analysis

A possible solution to the highlighted issue involves utilizing a graph-based representation of WDSs. The graph represents junctions, reservoirs, and storage tanks as nodes, while pipes, pumps, and valves are considered as edges. Graph neural networks (GNNs) can account for the non-euclidean structure of graph-based datasets (Isufi et al., 2022; Gama et al., 2020). GNNs extend Deep Learning with consideration of the graph topology by using, for example, the adjacency or the laplacian matrix. Additionally, GNNs can have permutation equivariant properties, which allow them to consider arbitrarily sized graphs (Isufi et al., 2022). These inductive biases preserve the additional information embedded in the graph structure and decrease the number of trainable parameters in the metamodel, which leads to more lightweight models (Kerimov et al., 2023).

By virtue of these qualities, we observe the increasing popularity of graph-based machine learning methods in various problems related to urban infrastructure (Donon et al., 2020) and water systems. For example, GNNs aided in the imputation of missing pipe diameters in a sewer system database (Belghaddar et al., 2021). Applications for WDS include burst detection (Zanfei et al., 2022), district metered area sectorization (Rong et al., 2021), and identification of cyberattacks (Tsiami and Makropoulos, 2021). Another work by Örn Garðarsson et al. (2022) introduced the ChebNet variant of a GNN to identify and localize leakages for the BattLeDim 2020 dataset (Vrachimis et al., 2022).

Building on this progress, the water engineering community focused on state estimation of WDS based on sensor readings using graph learning techniques. Some of the first works approach the problem from a signal reconstruction perspective. Such an approach does not necessarily consider demands or flowrates in the system and relies solely on pressure readings. Hajgató et al. (2021) tested a ChebNet model on 3 different water networks separately. They achieved high accuracy of recovered pressure heads, equipping only 5 % of nodes with pressure meters. Zhou et al. (2022) introduced a non-parametric graph-based technique to reconstruct pressure signals. Their approach utilizes graph Fourier transform to recover heads on the nodes by assuming a smoothness of the signal on neighboring nodes. The term "smoothness" here refers to low spatial signal variability. These methods assume a uniform distribution of flow velocities and demands, which may not always be realistic. Nevertheless, Zanfei et al. (2023) applied this type of metamodel for the quantification of the uncertainty of nodal demands and pipe parameters, although the proposed method does not formally guarantee that the prediction intervals cover the true values.

State estimation based on sensor reading differs from response surface surrogate modeling. The latter which attempts to replicate the input-output relationship of the underlying physical simulator (Garzón et al., 2022). Xing and Sela (2022) showcased the first application of GNNs for response surface surrogate modeling of WDS. They considered varying demand patterns as an input feature (Xing and Sela, 2022). Additionally, they showed that the flowrates could be derived by using the Hazen-Williams equation and predicted nodal pressure. The loss function further takes the error in the flow rates into account. Such a construction of the loss function permits posing the problem as *semi-supervised* learning, where only a limited number of sensors are utilized during training. Similar work by Ashraf et al. (2023) introduced multi-hop filters in the graph convolutional network in order to process data from larger networks. However, the number of parameters in the model for those WDSs remained large.

1.3. Transferable metamodels

Aside from showcasing the robustness to pipe removal in (Xing and Sela, 2022), the evidence on the transferability of the trained model in the literature is fairly limited. A sufficiently transferable metamodel, if achievable, will obviate the need for re-training the surrogate model after modifying the network and could be applied to a network with an arbitrary layout and configuration. Ultimately, the model could replace the physical simulator in performing downstream tasks and tedious iterative optimization and training processes. One can draw analogies from the existing large language models pre-trained on large text corpora and capable of executing arbitrary tasks given a few examples (Brown et al., 2020; Devlin et al., 2019; Touvron et al., 2023).

The first exploratory work (Kerimov et al., 2023) studied the transferability of GNN-based metamodels by training multi-hop ChebNet models on multiple water networks simultaneously. The findings indicated that GNNs can learn shared representations across various WDS. However, their usefulness in unfamiliar domains is still limited.

We argue that this limitation is rooted in the lack of expressivity of GNNs when modeling physical processes in the pipes. In the physical simulators, fundamental variables such as flow rate, pipe properties, and headlosses are defined on the edge space. More to that, flowrates are present in both the equations of mass and energy conservation that define the well-established approach of global gradient (Todini and Pilati, 1988). The headloss near the reservoir is crucial as it affects nodal pressure downstream. GNNs, however, treat edges as elements of secondary importance and do not explicitly operate on the edge space. This also leads to the inability to consider input from the flow meters (Xing and Sela, 2022).

Recent advancements in graph-based deep learning might alleviate this issue. This includes novel models that operate on higher-order geometrical structures, e.g. edge-centered networks, and more general representations, i.e. simplicial and cellular complexes (Yang et al., 2022a,b; Bodnar et al., 2021). In these models, edges connect to each other via a common node, analogously to two nodes connected by an edge. The connectivity of the edge-centered network is represented with a laplacian matrix, which captures compelling properties of *conservation* of the signal between incident edges (Jia et al., 2019). This allows processing signals in the edge domain, e.g. performing signal reconstruction, smoothing, and trajectory embedding (Schaub and Segarra, 2018).

Based on the aforementioned arguments, this work introduces a novel approach for response surface surrogate modeling of WDS with an Edge-based Graph Neural Network (EGNN). By addressing the limitations of existing GNN-based surrogate models, our approach takes a step towards a first step flowrates in every pipe based on the given demands in consumption nodes and pipe parameters. Then the method derives pressures in every node from the predicted flowrates using one of two proposed techniques. To evaluate the performance we generate a synthetic dataset of hydraulic simulations based on multiple benchmark models available in the literature. We firstly compare the accuracy of predictions with the method proposed by Xing and Sela (2022) in the setting when the network topology is known in advance. Next, we evaluate the performance of EGNN in an unknown setting by jointly training it on a variety of topologies and evaluating it on several other WDS excluded from the training set. Lastly, the work examines the speedups of the edge-based approach.

2. Method

The following section presents our method. It starts with a mathematical formulation of the problem and introduces the reader to traditional GNN-based metamodels. The section contains a technical background on GNNs and highlights its core limitations. The method defines and Edge-based Graph Neural Network on the basis of GNN, highlights the necessary steps to switch from node-based to edge-based representation, and addresses the mentioned limitations.

2.1. Problem statement

We represent a water distribution network as a graph $G = G(\mathscr{V}, \mathscr{E})$. The graph consists of a node set $\mathscr{V} \subset V(G)$ with N_u junctions and N_r reservoirs, and the set of edges $\mathscr{E} \subset E(G)$. Input node variables, such as elevation, or demands and readings from the pressure sensors comprise node signals. Together they construct the node matrix $\mathbf{x}_n \in \mathbb{R}^{N\times F_n}$, where F_n represents the total number of input node features. Similarly, edge features can be concatenated into the matrix $\mathbf{x}_e \in \mathbb{R}^{E\times F_e}$, with a total of F_e features. The full description of the input parameters are included in Table 1. Tanks, pumps, and valves are not considered in this representation.

The output of the metamodel is a vector of flowrates $\mathbf{f}$ and vector of pressures $\mathbf{p}_u$ on the consumption nodes. Typically, metamodels based on machine learning techniques are *trained* on a set of simulations and evaluated on a separate set, i.e. test set. The performances on the test set should reflect how well these metamodels can be applied across different scenarios. Ideally, a metamodel should display strong *generalization* capabilities and allow for transferability (Garzón et al., 2022). Here, we define these properties as follows:

(i) *Generalization.* We define generalization as the ability of a trained metamodel to preserve good predictive accuracy when tested on the same WDS of the training set, but with unseen inputs, e.g. demands, pressures, or pipe parameters.

(ii) *Transferability.* We define transferability as the ability of a trained metamodel to preserve good predictive accuracy when tested on different WDS and inputs than those in the training set (Neuman and Bramburger, 2023). Strong transferability, by definition, implies strong generalization.

Table 1
List of input variables and parameters used for metamodelling with description.

Symbol	Description
$G \in \mathscr{G}$	Mathematical representation of a water distribution network as a graph, with $\mathscr{V}$ denoting the set of N nodes (including junctions and reservoirs) and $\mathscr{E}$ denoting the set of E edges (i.e. pipes).
$\mathbf{S}$	Graph shift operator matrix, defined as $\mathbf{S}_{ij} \neq 0$ if $(i, j) \in \mathscr{E}$, and $\mathbf{S}_{ij} = 0$ otherwise.
$\mathbf{x}_n \in \mathbb{R}^{N\times F_n}$	Node matrix representing input variables defined on the nodes. Here, N is the number of nodes and F_n is the number of node features.
$\mathbf{p}_r \in \mathbb{R}^{N_r}$	Vector of known heads *(m)*, where N_r is the number of nodes with known heads.
$\mathbf{q} \in \mathbb{R}^{N}$	Total nodal water flow *(m^3/s)*, with subsets of reservoir outflows $\mathbf{q}_r \in \mathbb{R}^{N_r}$ and consumer demand $\mathbf{q}_u \in \mathbb{R}^{N_u}$. In cases with a single source, $\mathbf{q}_r = \sum_i \mathbf{q}_u^{(i)}$
$\mathbf{x}_e \in \mathbb{R}^{E\times F_e}$	Edge matrix representing input variables defined on the edges. Here, E is the number of edges and F_e is the number of edge features.
$d, l, r \in \mathbb{R}^E$	Pipe diameters *(m)*, lengths *(m)*, and Hazen-Williams coefficients defined on the edges.

2.2. GNN-based metamodels

We pose the model in an Encoder-Processor-Decoder fashion, where each module in this composition is shared across all nodes (Sanchez-Gonzalez et al., 2020; Xing and Sela, 2022). The purpose of an encoder (ENC) is to transform $\mathbf{x}_n$ into a feature matrix $\mathbf{H}^{(0)} \in \mathbb{R}^{N\times C}$ in a hidden space of dimension C. The processor (PRC) then processes the feature matrix in this high-dimensional space with consideration of the topology. Finally, the decoder (DEC) projects the processed feature matrix $\mathbf{H}^{(L)}$ into single-dimensional output of pressures $\mathbf{p} \in \mathbb{R}^N$ on nodes. Thus, the final architecture will have the following structure:

$$\begin{aligned} \mathbf{H}^{(0)} &= \mathrm{ENC}(\mathbf{x}_n) \\ \mathbf{H}^{(L)} &= \mathrm{PRC}\left(\mathbf{H}^{(0)}\right) \\ \mathbf{p} &= \mathrm{DEC}\left(\mathbf{H}^{(L)}\right) \end{aligned} \tag{1}$$

2.2.1. Encoder

The encoding into a higher dimensional space can be performed with a multi-layer perceptron (MLP). An encoding MLP_ψ: $\mathbb{R}^{F^n} \rightarrow \mathbb{R}^C$ embeds the input features $\mathbf{x}_n$ and projects it into the initial hidden feature matrix $\mathbf{H}^{(0)}$

$$\mathbf{H}^{(0)} = \mathrm{MLP}_\psi(x_n) \tag{2}$$

Each layer within the MLP transforms the feature matrix with a linear transformation and a non-linearity function which enhances feature representation and improves the expressivity of the model. The encoder step pre-processes node features in parallel and does not include the topological information.

2.2.2. Processor

The feature matrix can then be propagated from one node to its neighbors by multiplying a *shift operator* $\mathbf{S}$ with a feature matrix. The shift operator is a matrix with its elements defined as $\mathbf{S}_{ij} \neq 0$ if $(i, j) \in \mathscr{E}$, and $\mathbf{S}_{ij} = 0$ otherwise (Isufi et al., 2022; Gama et al., 2020). In general, any connectivity matrix can act as a shift operator, be it an adjacency matrix $\mathbf{A}$ or a Laplacian matrix $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$. Here $\mathbf{D}$ is a diagonal matrix of node degrees. Applying the shift operator multiple times allows propagating signal from a larger hop neighborhood. The wider the receptive field (i.e. the number of layers) the wider the reach of information sharing throughout the graph, as depicted in Fig. 1.

The multiplication of the shift operator with a feature matrix can be performed as a sparse matrix multiplication (Yang et al., 2018). Such a property allows processing graphs of various sizes in parallel batches, which has been widely adopted in GNN frameworks (Paszke et al., 2019). Thus, the model needs not to be trained and evaluated on the same topology. More information can be found in Appendix A. We can use the shift operator to define the graph convolution layer for a node signal (e.g. $\mathbf{H}$) as

$$\mathbf{H}^{(l+1)} = \sigma\left(\mathbf{S}\mathbf{H}^{(l)}\boldsymbol{\Theta}^{(l)}\right) \tag{3}$$

where $\Theta \in \mathbb{R}^{CxC}$ is a trainable weight matrix, σ is a nonlinearity function, and l is the layer number. Multiplication with the weight matrix linearly transforms a node signal. In general, it is possible to apply an MLP before propagation at each graph convolutional layer to increase expressivity, e.g.

$$\mathbf{H}^{(l+1)} = \sigma\left(\mathbf{S}\cdot\mathrm{MLP}_\xi^{(l)}\left(\mathbf{H}^{(l)}\right)\right) \tag{4}$$

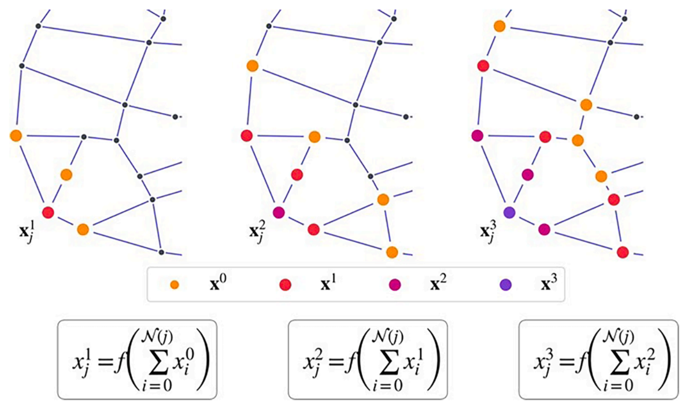

x_j^1
x_j^2
x_j^3
x^0
x^1
x^2
x^3
$x_j^1 = f\left(\sum_{i=0}^{\mathcal{N}(j)} x_i^0\right)$
$x_j^2 = f\left(\sum_{i=0}^{\mathcal{N}(j)} x_i^1\right)$
$x_j^3 = f\left(\sum_{i=0}^{\mathcal{N}(j)} x_i^2\right)$

Fig. 1. A 3-layered graph neural network. The figures from left to right depict the propagation of the information from the neighborhood $\mathcal{N}(j)=\{j\ (i, j) \in E\}$ of a node j to the node itself. Such computation is performed in parallel for every other node.

We formulate the GNN by stacking L graph convolutional layers and employ it as a processor. Each

GNN layer updates the **H** based on the graph structure and captures the nodal interactions. However, theoretical research has identified a problem of "oversmoothing" that occurs in models with many layers (NT and Maehara, 2019). To combat oversmoothing, a residual connection, in general, adds the previously computed internal node matrix to the updated node matrix with a coefficient α, e.g.

$$\mathbf{H}^{(l+1)} = \mathbf{H}^{(l)} + \alpha\cdot\sigma\left(\mathbf{S}\cdot\mathrm{MLP}_{\xi}^{(l)}\left(\mathbf{H}^{(l)}\right)\right) \tag{5}$$

which reduces oversmoothing effect (Chen et al., 2020; Chamberlain et al., 2021b). This phenomenon occurs since propagation with **S** effectively averages out the signal, rendering it indistinguishable between nodes. In the context of WDS, similar pressures result in small headlosses.

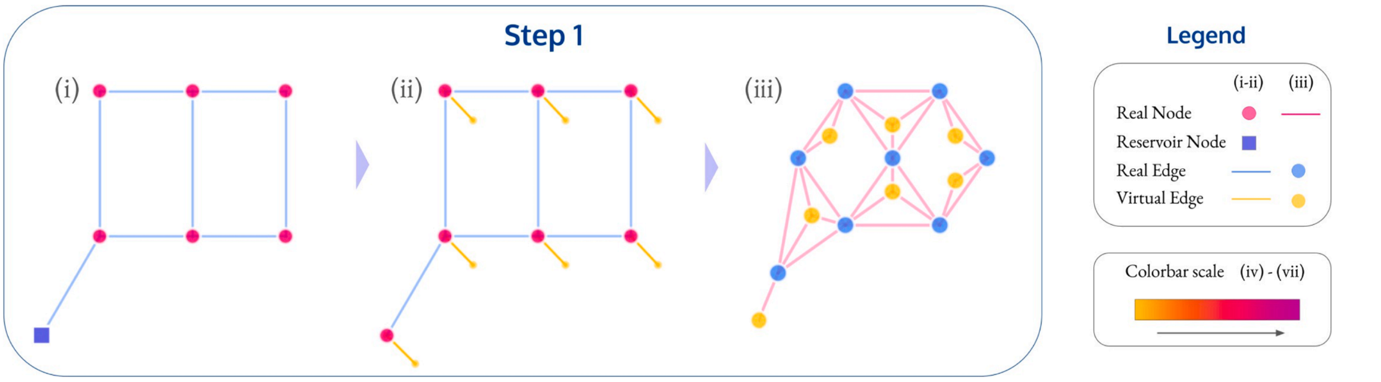
2.2.3. Decoder

Finally, the decoder $\mathrm{MLP}_{\phi} : \mathbb{R}^C \rightarrow \mathbb{R}$ obtains the desired output **p** from the hidden feature matrix in the last layer.

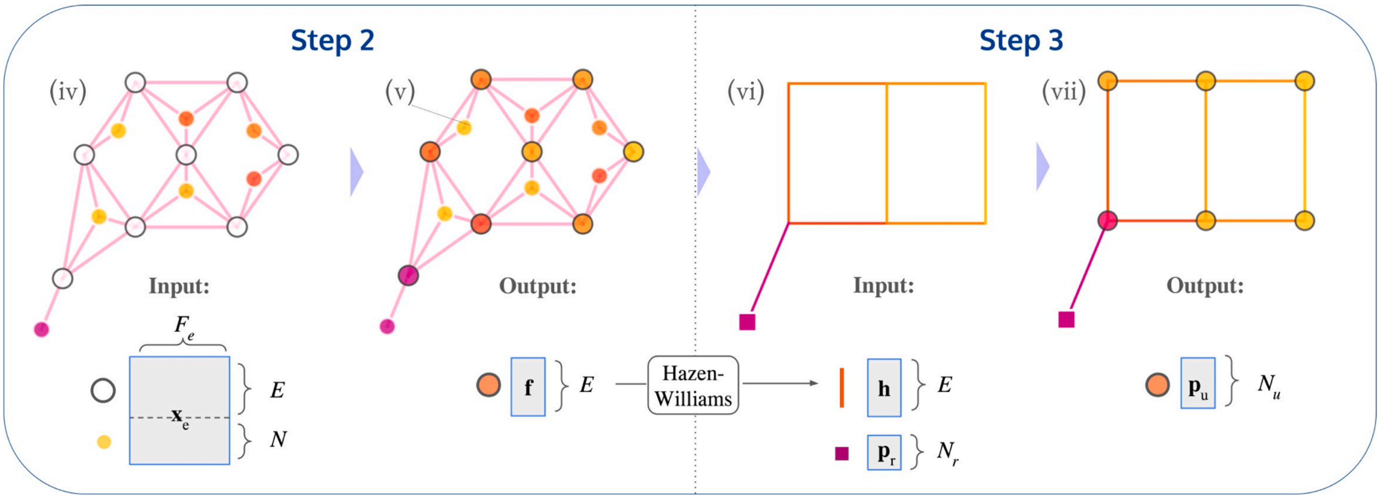
$$\mathbf{p} = \mathrm{MLP}_{\varphi}\left(\mathbf{H}^{(L)}\right) \tag{6}$$

2.3. Metamodeling with edge-based graph neural networks

To address the highlighted issue of oversmoothing in GNN-based metamodels, we propose an alternative approach by switching from node-based to edge-based representation. Instead of outputting pressures on nodes, EGNN predicts the flowrates **f** first by ingesting and processing the edge features $\mathbf{x}_e$ and. Using Hazen-Williams relationship, the flowrates are translated into headlosses. Finally, nodal pressures are derived from the headlosses. The procedure is performed in 3 steps and



(a) Switch from node-based to edge-based representation. (i) Original graph representation of WDS described in Section 2.2. (ii) Augmentation with virtual edges (Section 2.3.1). (iii) Final edge-based view (Section 2.3.2)



(b) Schematical depiction of the inputs and outputs of flowrate interpolation with EGNN (Step 2, Section 2.3.2) and pressure reconstruction (Step 3) described in Section 2.3.4. Pressure on nodes (vii) are calculated from **h** on edges and known pressure values **p** (vi)

Fig. 2. Overview of the method.

the overview is illustrated in Fig. 2.

2.3.1. Virtual sinks

As the first step of transformation into edge-based representation, the underlying graph G becomes directed with arbitrarily chosen edge orientations. Next, we couple every node in the directed graph with a virtual node $u_v \in \mathscr{V}_v$ where $\mathscr{V}_v$ is a set of virtual nodes. The set of edges is thus extended with a set of virtual edges $\mathscr{E}_v = \{(u, u_v) | u \in \mathscr{V}$ and and $u_v \in \mathscr{V}_v\}$. We then assign demand on the node u to the incidental virtual edge e_v, i.e.

$$\mathbf{f}(e_v) := \mathbf{q}(u) | u \in e_v \tag{7}$$

The sign of $\mathbf{f}(e)$ at edge e reflects the true orientation of the flow. If the sign is negative, the true orientation is the oppo site of the initially chosen one. The resulting matrix of edge input features is constructed based on three input parameters: demand $\mathbf{q}$, hydraulic resistance κ, and the indicator of the virtual pipes to help the model distinguish between two types of edges

$$\mathbf{x}_e = \begin{bmatrix} 0 & \kappa & 0 \\ \mathbf{q} & 0 & 1 \end{bmatrix} \tag{8}$$

The virtual sinks are assumed to be perfect and cause zero friction, while the flowrates on the real pipes are unknown and thus they are not in the input. Hydraulic resistance is derived from Hazen-Williams equations (Rossman, 2022) employed for the calculation of headloss along a pipe, (see Table 1).

$$\kappa = \frac{10.67 \cdot l}{r^{1.852} d^{4.870}} \tag{9}$$

The virtual connection to the reservoir would contain reservoir outflow. In a network with a single reservoir, we assume that a total reservoir outflow is equal to the sum of the demands.

2.3.2. Edge-based representation

We continue by defining the connectivity between an edge and a node $\mathbf{B} \in \mathbb{R}^{N \times E}$ as $b_{ij} = 1$ if the edge i is oriented towards the node j and $b_{ij} = -1$ otherwise, while $b_{ij} = 0$ when there otherwise no connectivity between the elements. Literature defines an Edge Laplacian as a result of the multiplication of a transposed incidence matrix with itself (see more in Schaub and Segarra (2018); Yang et al. (2022b).

$$\mathbf{L}_e = \mathbf{B}^T \mathbf{B} \tag{10}$$

The resulting matrix $\mathbf{L}_e \in \mathbb{R}^{E \times E}$ represents the connectivity between edges via common nodes and can act as a shift operator for the edge signal. The example of the connectivity is presented in Figure 2.2.3. For example, if 4 edges are connected through a single node, in the edge representation it will transform into 4 interconnected nodes.

According to Schaub et al. (2020) a normalized version of an Edge Laplacian is therefore calculated as follows, with $\mathbf{D}$ as a diagonal matrix of node degrees.

$$\widehat{\mathbf{L}}_e = \mathbf{B}^T \mathbf{D}^{-1} \mathbf{B} \tag{11}$$

An intriguing property of the incidence matrix $\mathbf{B}$ (and consequently, $\mathbf{L}_e$) is that it can function as an operator measuring the *divergence* of an edge signal which is related to mass conservation.

Inductive bias of mass conservation The divergence on a node i is calculated as the sum of signal on incidental edges with respect to their orientations

$$(\text{div } \mathbf{f})_i = \sum_{j,i \in \mathscr{E}} \mathbf{f}_{ji} - \sum_{i,j \in \mathscr{E}} \mathbf{f}_{ij} \tag{12}$$

The divergence of $\mathbf{f}$ is present in the equation of mass balance for water flow. In a WDS that operates without leakages, the divergence is equal to demands on nodes

$$\mathbf{B}\mathbf{f} = \mathbf{q} \tag{13}$$

However, $\mathbf{f}$ becomes divergence-free on real nodes when considering the flowrate on virtual edges. The divergence-free condition can be expressed with $\mathbf{L}_e$ and written as

$$\| \mathbf{B}\mathbf{f} \|_2^2 = \mathbf{f}^T \mathbf{B}^T \mathbf{B} \mathbf{f} = \mathbf{f}^T \mathbf{L}_e \mathbf{f} = 0 \tag{14}$$

This equation leads to an important aspect of adopting $\mathbf{L}_e$ as a shift operator. Instead of averaging out the features as in the case of shift operator for nodes, $\mathbf{L}_e$ drives the features to divergence-free values. Such a property has been successfully leveraged in determining synchronization and consensus of a dynamical system (Ziegler et al., 2022). By using $\mathbf{L}_e$ as a connectivity matrix we turn a limitation into a physics-based prior.

2.3.3. Flowrate and headloss interpolation

With the shift operator of the edge signal defined, we formulate an Edge-based Graph Neural Network with a residual connection as follows:

$$\mathbf{H}^{(l+1)} = \mathbf{H}^{(l)} + \alpha \cdot \sigma \left(\widehat{\mathbf{L}}_e \cdot \text{MLP}_{\xi}^{(l)} \left(\mathbf{H}^{(l)} \right) \right) \tag{15}$$

This time, $\mathbf{H} \in \mathbb{R}^{ExC}$ is an internal edge representation of the hidden size C. The update of the representation is based on Eq. (5). In a similar Encoder-Processor-Decoder fashion, EGNN here acts as a processor. The encoder MLP $_\psi$ and decoder MLP $_\varphi$ are separate MLPs:

$$\mathbf{H}^{(0)} = \text{MLP}_{\psi}(\mathbf{x}_e) \rightarrow \mathbf{f} = \text{MLP}_{\varphi}\left(\mathbf{H}^{(L)}\right) \tag{16}$$

Thus, the flowrate reconstruction module interpolates the flowrates on each real pipe. Assuming Hazen-Williams relationship, $\mathbf{f}$ determines the headlosses at the corresponding pipe e:

$$\mathbf{h}(e) = \kappa_e \cdot \mathbf{f}(e) \cdot |\mathbf{f}(e)|^{0.852} \tag{17}$$

Adding virtual pipes converts the nodal water demands $\mathbf{q}$ into edge features. By employing the demands as a known signal, we frame the problem as a signal reconstruction, similar to (Hajgató et al., 2021; Zhou et al., 2022) but on the edge space.

2.3.4. Pressure reconstruction module

The last step in the pipeline is the derivation of pressures from the output of the neural network. Below we propose two possible approaches and the necessary steps before the reconstruction. Note that the virtual connections are not considered in this step.

The derivation of heads on the unknown nodes $\mathbf{p}_u$ is based n the law of energy conservation (Todini and Rossman, 2013)

$$\mathbf{B}_u^T \cdot \mathbf{p}_u + \mathbf{h} = -\mathbf{B}_r^T \cdot \widehat{\mathbf{p}}_r \tag{18}$$

where $\widehat{\mathbf{p}}_r$ is the known nodal head (e.g. pressure sensor or a reservoir head), $\mathbf{B}_r \in \mathbb{R}^{N_r \times E}$ and $\mathbf{B}_u \in \mathbb{R}^{N_u \times E}$ are the incidence matrices of the nodes with known (total N_r) and unknown (total N_u) head values correspondingly. Examples of the former are reservoir nodes or pressure meters. Both proposed manners derive $\mathbf{p}_u$ based on matrix inversion:

(i) *Least squares method* This approach is based on the solution of Eq. (18) via solving the least squares problem. This method still acts as a bottleneck as it requires solving problems for each network one by one. This work employs the implementation of a least squares solver in PyTorch (Paszke et al., 2019).

(ii) *Power approximation method* To address this limitation we provide an alternative approach. Multiplying both sides of Eq. (18) with $\mathbf{B}_u$ on the left we obtain the following equation

$$\mathbf{L}^{(u)} \cdot \mathbf{p}_u = -\mathbf{B}_u \left(\mathbf{h} + \mathbf{B}_r^T \cdot \mathbf{p}_r \right). \tag{19}$$

Here, $\mathbf{L}^{(u)} \in \mathbb{R}^{N_u x N_u}$ is the graph laplacian of the junction nodes.

To circumvent the need for a linear solver we engage the properties of the geometric series of a matrix (Stewart(1998)). In fact, from the properties of the Neumann series, it follows that if an arbitrary matrix $\mathbf{I}-\mathbf{P}$ is non-singular and the powers of $\mathbf{P}^k$ converge zero, then it is possible to approximate a matrix inverse with an infinite sum as follows

$$\mathbf{P}^{-1} = \sum_{i=0}^{\infty} (\mathbf{I}-\mathbf{P})^i \tag{20}$$

It is sufficient to show that the largest eigenvalue a real symmetric matrix $\mathbf{P}$ is $\lambda_{max} < 2$ to satisfy the convergence requirement. The largest eigenvalues of the normalized graph laplacian $\widehat{\mathbf{L}} = \mathbf{D}^{-1}\mathbf{L}$ is bounded by 2 (Kipf and Welling, 2016). Moreover, the $\lambda_{max} = 2$ if and only if the graph G is bipartite (Beers and Mulas, 2024). The infrastructure network needs to have a grid-like structure (Bandelt et al., 2010).

Normally, calculation of the power of the laplacian is a slower method than obtaining the inverse. It is possible, however, to leverage batching and GPU parallelization, which substantially increases speed.

2.3.5. *Loss function*

The edge-based model is trained in a supervised fashion by minimizing the loss function that consists of two components

$$\mathscr{L} = \alpha_1 \| \boldsymbol{\kappa}(\mathbf{f}-\widehat{\mathbf{f}})\|_2^2 + \alpha_2 \| \mathbf{h}-\widehat{\mathbf{h}}\|_1 \tag{21}$$

where $\mathbf{f}$ and $\widehat{\mathbf{f}}$ are predicted and true flowrates on each real edge, while $\mathbf{h}$ and $\widehat{\mathbf{h}}$ are derived and true headlosses on each real edge. Each term in (21) is calculated as a deviation from the true values derived by a hydraulic simulator α_1 and α_2 are weighting coefficients.

The first term is weighted by hydraulic resistance κ. This term overrepresents the pipes with a potential high headloss and avoids erroneous predictions of such conduits to propagate to the remaining elements. Previous studies have shown the efficiency of this approach, e.g., (Kerimov et al., 2023)

The second term in the loss function is obtained with a familiar equation employed in the physical simulator. Based on the predicted flowrates on real edges $\mathbf{f}$ we derive respective headlosses via the Hazen-Williams equation.

$$\widehat{\mathbf{h}} = \boldsymbol{\kappa}\cdot\widehat{\mathbf{f}}^{1.852} = \mathbf{B}^T\cdot\mathbf{p} \tag{22}$$

This term is calculated with L_1-norm to keep the values in the same order of magnitude as the values in the first term.

3. Experiments

This section describes the experiments in this study. Based on definitions of generalization and transferability in Section 2.1 we compose two sets of experiments. We describe case studies and the data generation procedure for each of them. The first experiment (Section 3.2) compares the generalization and performances of node-based and edge-based models. The experiment contains two subsets (A and B). The second experiment measures the transferability of an EGNN in an unknown setting (Section 3.3). The last experiment evaluates the runtime and speed-up of our method (Section 3.4).

3.1. *Case studies*

Selected case studies are shown on Table 2, along with their number of nodes, pipes, reservoirs in the system, and the network diameter. The network (or graph) *diameter* here characterizes the shortest distance between the two farthest nodes in a network. The same table also shows which case studies were used for each experiment.

The chosen WDSs do not feature hydraulic elements, such as pumps and valves, while existing tanks are transformed into reservoirs. The modeling of physical components can be studied in future work, for example by including them as a binary flag per each node or edge in node or edge features. Some of the networks contain pairs of junctions which are connected with parallel pipes, that were originally used within the optimization task (Lee and Lee, 2001). In the graph representation that results in a multi-edge setting, complicates the calculation of the Edge-Laplacian. To avoid that, we merged those pipes into one. The final diameter of the merged pipe is derived to represent equivalent losses in energy as the initial setting. For each of these networks, we employed the WNTR Python package with EPANET simulator as the backbone to generate steady-state simulations for every experiment (Klise et al., 2018).

3.1.1. *Data generation*

Based on the selected networks, we generate individual datasets of steady state simulations for each experiment. Within each experiment, the simulations are produced by sampling one or more of the following parameters:

- *Nodal demands* $\mathbf{q}$ were sampled uniformly around a base value shared across nodes. For some WDSs, this sampling strategy results in hydraulically unfeasible simulations (i.e. negative pressure on the nodes). Instead, we uniformly sampled demands around the original base demand on each node provided in the *.inp* file.
- *Pipe parameters* κ include length, diameter, and Hazen-Williams coefficients. The values are sampled uniformly within a range centered on each original pipe characteristic (± 20 % of the original values). To avoid unrealistic configurations, leading to very low or very high pressures (Zischg et al., 2015), the bounds may be lowered. The altered values of the Hazen-Williams coefficients are selected from distributions that reflect commercial ranges (50 to 150).
- *Topology* G modifications diversify the variety of networks in the training data and benefits the transferability (Xie et al., 2022). For these modifications, each pipe is removed with a probability of 10 %. After that, with a probability of 25 % the network undergoes one of

Table 2
Description of water networks used in the experiments ordered by the number of pipes.

	APULIA	JILIN	BAK	ASnet2	PES	LT	NT3	ZJ	MOD	KL
Parameters										
Network diameter	10	10	14	14	22	22	30	26	38	53
Nodes	24	28	36	51	71	93	97	114	272	936
Pipes	34	34	58	65	99	109	119	164	317	1274
Reservoirs	1	1	1	1	3	1	3	1	4	1
Experiments										
Generalization of GNNs and EGNNs			✓	✓		✓		✓		✓
Transferability of EGNNs: Training*			✓		✓	✓	✓		✓	✓
Transferability of EGNNs: Testing	✓	✓		✓				✓		
Speed-up	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

* The networks are combined into a single set.

the skeletonization procedures (i.e. parallel pipe removal, trim branching, etc.), as described in Table 3 (Cesario and Association, 1995). The probability values are taken arbitrarily to introduce variations in the topology. This procedure can be repeated for multiple iterations.

Table 4 summarizes the modifications applied in each experiment. When a parameter is not sampled, it is the same in all simulations within both the training and testing sets.

3.2. *Generalization of GNNs and EGNNs*

This experiment provides the comparison of the performance by training and evaluating within the same WDSs.

With this experiment, we aim to assess how accurately each of the models predicts $\mathbf{p}_u$ and $\mathbf{f}$ in different cases when the network layout is known in advance and used during training.

The models are evaluated on two subsets: a dataset with varying demands and fixed κ (Subset A), and a dataset with varying demands and pipe parameters (Subset B). In both subsets, the topology is the same in both training and testing set. We are interested in studying the change in the performance with the introduction of additional complexity. The models are trained and evaluated separately on the basis of 5 networks with single reservoirs: BAK (Fig. 3c), ASnet2 (Fig. 3d), area C of l-Town (Fig. 3f), ZJ (Fig. 3h), and KL (Fig. 3j). The number of trained models within this experiment totals to 10. For each network, except for KL, we generated 10,000 steady-state simulations by altering demands, and another 10,000 simulations by varying both demands and pipe characteristics. For KL, specifically, the number of simulations is 2000 due to the large size of the network. Each of the subsets is split into training, validation, and test subsets with a ratio of 80:10:10.

3.2.1. *Description of models*

As a benchmark GNN candidate we selected the work provided by Xing and Sela (2022). The model is similar to the GNN described in 2.2 but extends a classical GNN by including extra steps in the processor module, such as separate multi-layer perceptrons for in- and out-going edges and self-loops, an intermediate supervised loss function, and inclusion of the pipe parameters in the processor module. The original study was originally only applied on a single network (i.e. ASnet2). In this work, we were able to adapt the same architecture to create a metamodel and apply it to different water networks using the original code of the authors. We used the same set of hyperparameters as provided in the original work. Both EGNN and GNN have 20 hidden units in each linear layer. The number of layers L corresponds to the diameter of each WDS. Given that a WDS is a complex and interconnected system, it is crucial for the receptive field to encompass the entire network. The total number of parameters in the GNN is larger due to the more complex architecture. A detailed description of the chosen hyperparameters is provided in Appendix B. https://github.com/erytheis/egnn

The performance is measured with the coefficient of determination

$$R^2 = 1 - \frac{\| \mathbf{y} - \widehat{\mathbf{y}}\|_2^2}{\| \mathbf{y} - \overline{\mathbf{y}}\|_2^2} \tag{23}$$

Here $\mathbf{y}$, $\widehat{y}i$ are vectors of values produced correspondingly by the neural network and EPANET, and $\widehat{\mathrm{y}}$ is a mean nodal or edge values obtained with EPANET.

Table 3
Summary of applied topological modifications.

Skeletonization technique	p	Number of iterations
Pipe removal	0.10	
Parallel pipe merge	0.25	
Series pipe merge	0.25	1–10
Branch trim	0.25	

Table 4
Summary of variabilities within each experiment.

		Modifications					
		Training set			Testing set		
		q	κ	G	q	κ	G
Generalization of GNNs and EGNNs	A	✓			✓		
	B	✓	✓		✓	✓	
Transferability		✓	✓	✓	✓	✓	✓

3.3. *Transferability of EGNNs*

In the second experiment, we assess the performance of EGNNs trained on a subset of networks, when tested on a subset of unseen networks. We do not perform these tasks for the GNN since the available implementation does not allow for training across different water network sizes and layouts. We jointly train the EGNNs on multiple WDSs to increase generalization capabilities. Due to the limited number of publicly available water distribution systems, we apply a set of topological modifications to the training set described in Section 3.1.1. With this augmentation technique, we generated 13,600 samples, divided into training and validation subsets with a ratio of 80:20.

We measure transferability by computing the discrepancy between predicted flowrates and heads in terms of $\boldsymbol{R}^2$, MAE, and RMSE for WDSs not featured in the training set. The systems selected for training and test sets are marked in Table 2. These cases vary in network size, pipe parameters, and demand distribution. For test networks, we used the same data generation strategy as in the experiment described in Section 3.2 with 100 generated steady states per case. The demands for every node in the test set were sampled uniformly around 5 L/s.

3.4. *Speed-up*

The ultimate goal of a response surface metamodel is to provide computational gains with a minimal loss of accuracy. This experiment provides the evaluation of the execution time for each step in our method. According to the discussion in the method section, our approach presumes execution in two steps, with one of the 2 variants of the pressure reconstruction stage. Based on every network we measured the required time to obtain the end result and visualized the results. For each case study, we constructed a model with the number of layers that is equal to the diameter of the network and with a hidden size $\mathscr{C}$ of 20 units. We compare those speeds with the time required to run a simulation with EPANET as a baseline. Note that we do not compare the speed-ups of EGNN with the ones of GNN. The latter utilizes TensorFlow (Abadi et al., 2015) and, in general, results in faster computations than PyTorch. However, it requires additional time for compilation before running the model.

4. Results & discussion

4.1. *Generalization of GNNs and EGNNs*

We begin this section by presenting the comparison of the performances of metamodels based on EGNN and GNN. Following, we visualize the robustness of their prediction to changing pipe parameters and demands.

Fig. 4 summarizes the results of the evaluation of the models. As expected, the performance of the models trained on Subset A (tops of the bars) is higher than the one of Subset B (bottoms of the bars). We see that the edge-based approach consistently outperforms the node-based GNN in terms of flowrates. In terms of junction heads and pressures, EGNNs are more likely to provide reliable results when the parameters are varied (Subset B). When the pipe parameters are sampled in the training and testing sets, the influence of energy losses due to the friction in pipes

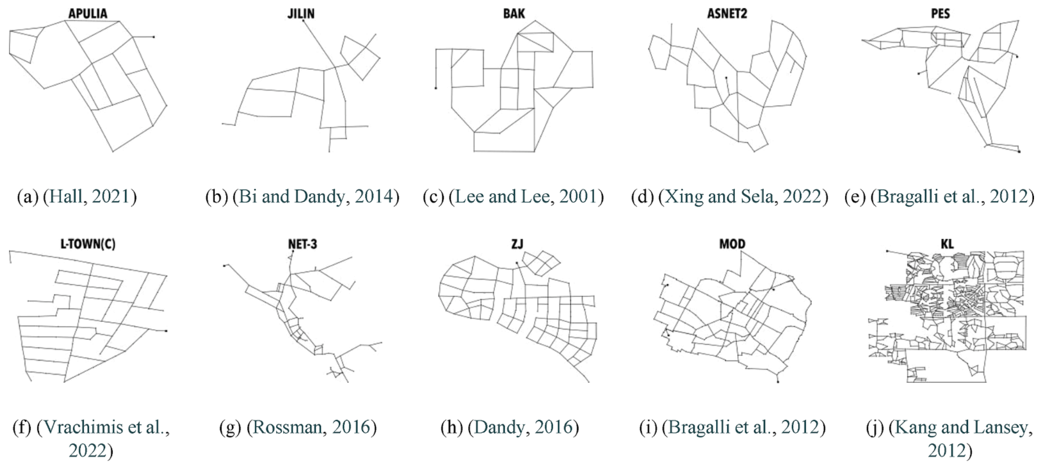


Fig. 3. Water distribution networks used in the experiments (Bragalli et al., 2012; Kang and Lansey, 2012; Dandy, 2016; Hall, 2021; Rossman, 2016).

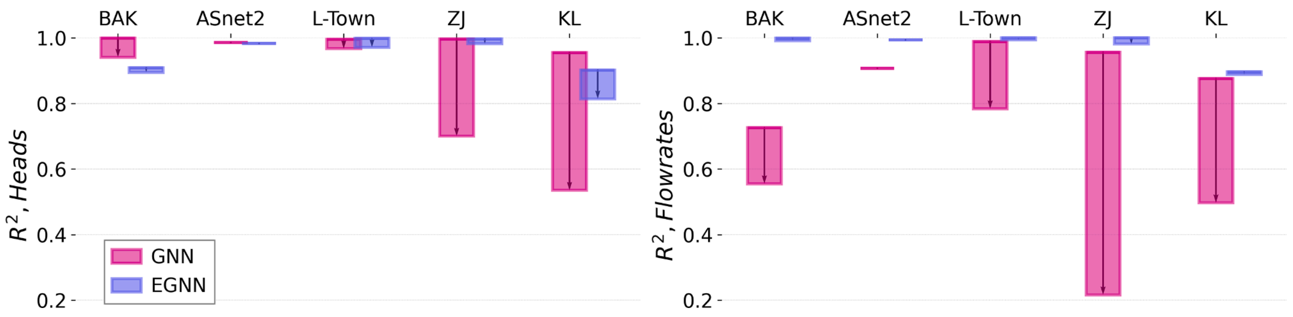


Fig. 4. Degradation of performance in terms of R^2 with the introduction of variability of the parameters in terms of predicted heads (left) and flowrates (right) between GNN (pink) and EGNN (purple). The top of the bar corresponds to the accuracy of the model trained on the dataset with fixed pipe parameters (Subset A in Section 3.2), while the bottom of the bar corresponds to the accuracy of the model trained on the dataset with variabilty in κ and **q** (Subset B in Section 3.2).

becomes more prominent. Thus, adding this variability increases the complexity of the problem as the model must adapt to a wider range of data (Sato et al., 2021). The improved generalization becomes especially visible for larger WDSs, as can be seen in Fig. 4. There, the performance of GNN degrades significantly more than the one of EGNN. We included further comparison in terms of MAE to Appendix The primary difference in generalizability arises from the challenges of GNNs to effectively model pipe headlosses, as energy loss within the pipes are intricately tied to the flowrates. In contrast, EGNNs gain an advantage in this problem due to their edge centering and iterative updates of edge representations. It initially predicts flowrates and subsequently deduces pressure values, thus decomposes a complex problem into smaller steps.

The second distinction arises from the effect of stacking multiple layers in each method, which is essential to cover the whole WDS. In the case of nodes, this setting may lead to oversmoothing of the nodal signal (NT and Maehara, 2019) such as node heads, which leads to small variations in neighboring nodes, i.e. small headlosses. Such an inductive bias can be detrimental as it disregards the physical relation between the headloss and the flowrates. In the case of $\mathbf{L}_e$ smooth signal is *divergent-free* or mass-conservative on the junctions. Thus, an EGNN with multiple layers drives the prediction of flowrates towards mass-conservative values. This results in a more stable model for larger WDSs.

4.2. Transferability of EGNNs

Table 5 summarizes the results of the evaluation of the model trained on multiple networks simultaneously. It shows that the R^2 of the predictions of flowrates and pressures on the unseen settings are consistently above 0.8. Generally, the flowrate accuracy is higher for smaller networks. In addition, the model accurately approximates the pressure values for all networks. The results present good evidence of the transferability of the model in unseen domains when the distribution of nodal demands is similar to the one in the training set. Further analysis of the model's sensitivity to input nodal demands is provided in the Appendix C.2.

Moving to specific network cases, Figs. 5 and 6 showcase the transferability of EGNN on the example ASnet2, ZJ, Apulia and Jilin. Although these networks were not present in the training dataset, EGNN performs well. The error is manifested in the underestimation of

Table 5
Results of evaluation of transferability of EGNN (Section 4.2).

Case study	Maximum κ	Pressures, (m)			Flowrates, (L/s)		
		R^2	RMSE	MAE	R^2	RMSE	MAE
ASnet2	$4.5 \cdot 10^1$	0.832	0.069 ± 0.017	0.040 ± 0.011	0.793	10.785 ± 6.716	9.416 ± 6.192
ZJ	$3.5 \cdot 10^2$	0.858	0.233 ± 0.108	0.135 ± 0.050	0.848	19.689 ± 13.190	17.122 ± 11.598
Jilin	$1.5 \cdot 10^4$	0.950	0.552 ± 0.324	0.379 ± 0.218	0.983	2.419 ± 2.346	2.015 ± 2.179
Apulia	$7.4 \cdot 105$	0.883	0.907 ± 0.862	0.402 ± 0.210	0.982	2.820 ± 2.055	2.218 ± 1.687

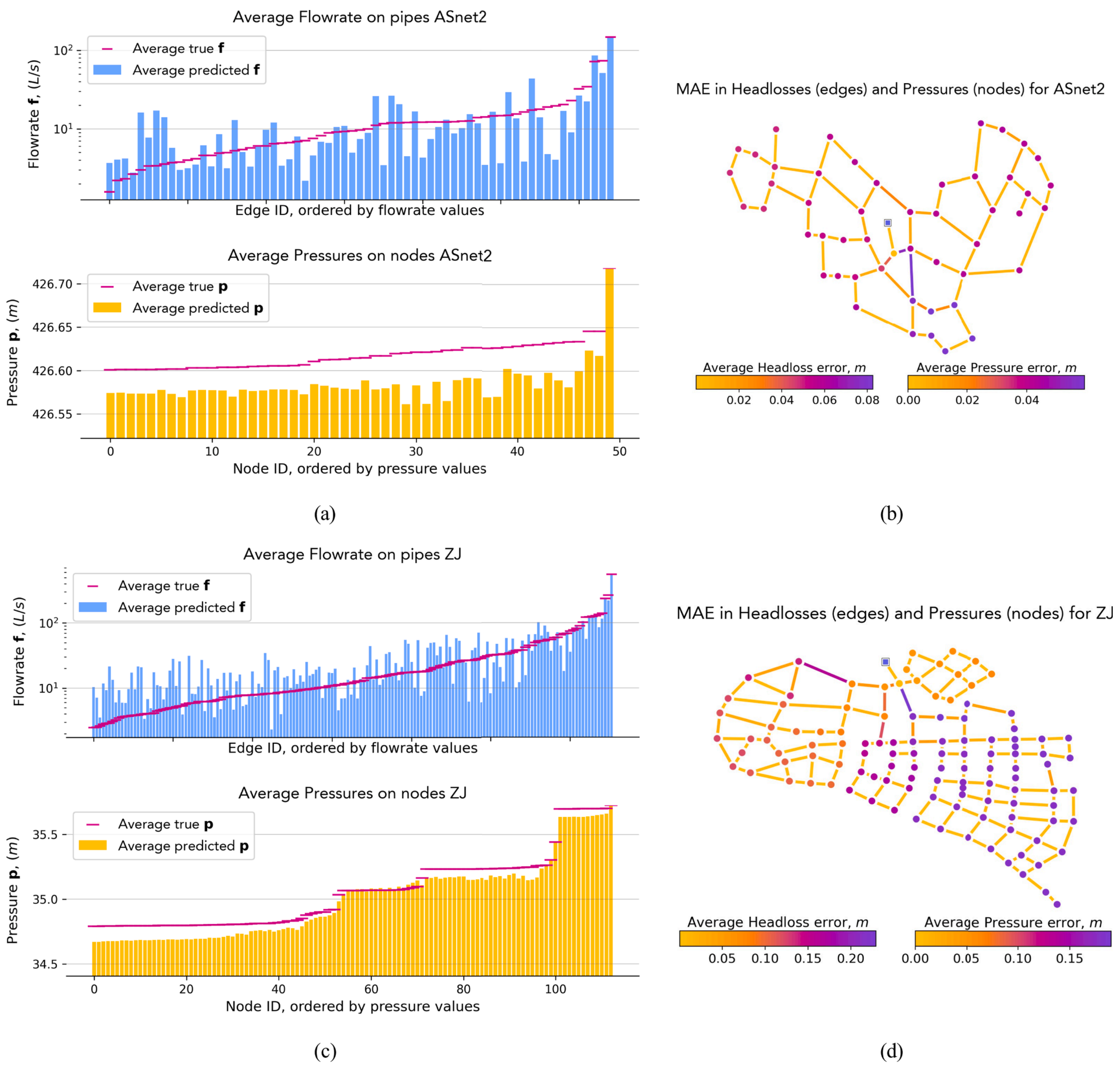


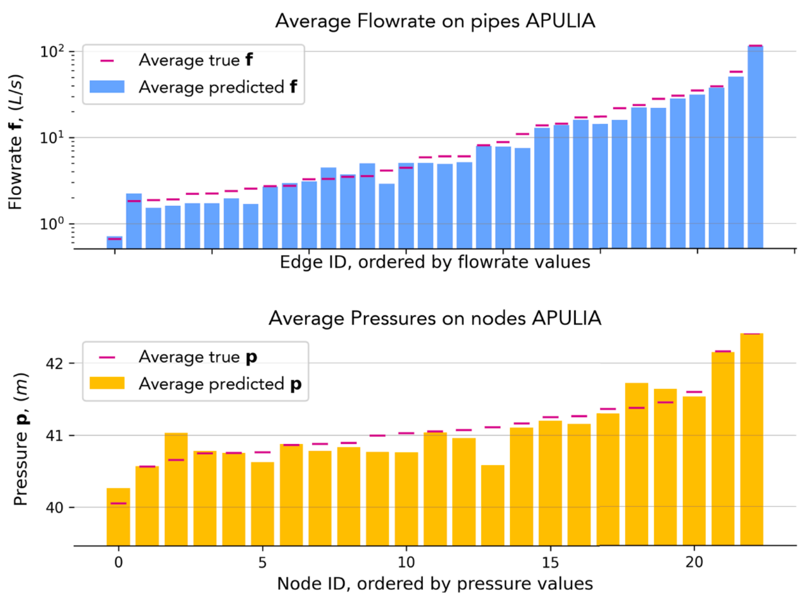
Fig. 5. Detailed evaluation of transferability on the example of ASnet2 (a, b) and ZJ (c, d) averaged over ($n = 100$) simulations for each case study. The left side shows the comparison of $\mathbf{f}$ with $\mathbf{f}$ and $\mathbf{p}$ with $\widehat{\mathrm{p}}$. The right side depicts the layouts of the networks with the color on nodes and edges indicating an average absolute error of derived pressures and headlosses correspondingly (b,d).

pressures for certain areas of the networks. Simultaneously, the highest errors in predicted headlosses $\mathbf{h}$ appear near the reservoirs, particularly in pipe 62 in the case of ASnet2 in Fig. 5b and pipe 29 in the case of Jilin in Fig. 6d. These pipes connect districted zones to the rest of the network. Consequently, the errors in headloss propagate downstream to the rest of the sectors.

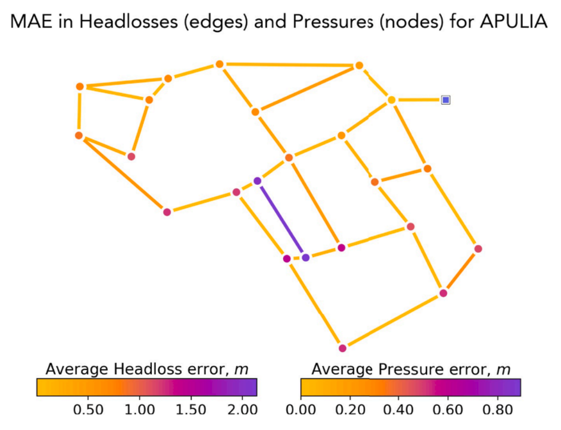
One can additionally observe in Figs. 5a and 5c, that the relative error in $\mathbf{f}$ is generally higher for the smaller flows. Since the optimization is based on minimizing mean squared error in flowrates, these errors are more visible on pipes with low water flow. Nonetheless, large relative errors in predicted flowrates can lead to substantial disparities in headloss, as defined by Eq. (17). This is specifically evident when dealing with smaller pipe diameters, for example in the case of Apulia in Fig. 6b. The pipe with the largest headloss error features the highest κ and the lowest flowrate $\mathbf{f}$. We introduced a weighting term with κ in the loss term in Eq. (21) to alleviate the issue. Because the loss is calculated per pipe, the weighting oversamples pipes with small diameters and prioritizes the important pipes during optimization.

4.3. Speed-up

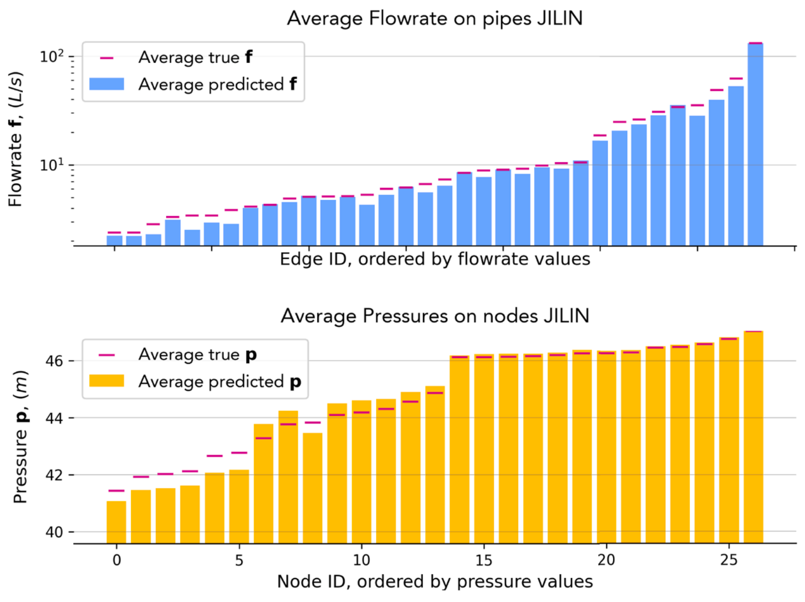
Fig. 7a depicts the results of speed up experiments as a comparison of computational gains of the total model with each pressure reconstruction method. As we see iterative smoothing in general provides more speed improvements than the least squares, as it can be executed in parallel on a GPU. Thus, with the same graphic memory limit, iterative smoothing is significantly faster for smaller networks (up to 350 times faster). For larger networks, the methods seem to be providing relatively



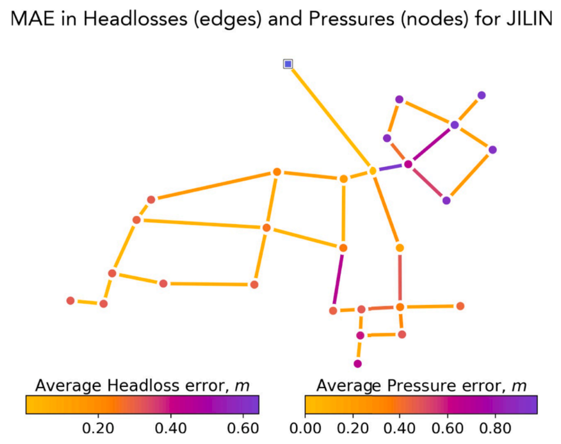
(a)


MAE in Headlosses (edges) and Pressures (nodes) for APULIA
Average Headloss error, m
0.50
1.00
1.50
2.00
Average Pressure error, m
0.00
0.20
0.40
0.60
0.80

(b)



(c)


MAE in Headlosses (edges) and Pressures (nodes) for JILIN
Average Headloss error, m
0.20
0.40
0.60
Average Pressure error, m
0.00
0.20
0.40
0.60
0.80

(d)

Fig. 6. Detailed evaluation of transferability on the example of(a, b) and Jilin (c, d) averaged over ($n = 100$) simulations for each case study. The left side shows the comparison of $\mathbf{f}$ with $\hat{\mathbf{f}}$ and $\mathbf{p}$ with $\hat{\mathrm{p}}$. The right side depicts the layouts of the networks with the color on nodes and edges indicating an average absolute error of derived pressures and headlosses correspondingly (b,d).

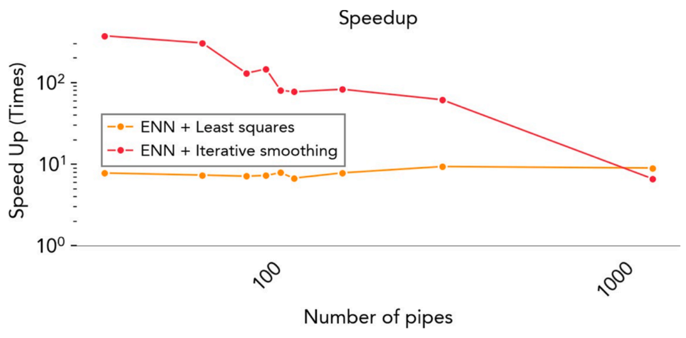
similar speedups to the least squares method (up to 8 times faster). This is due to the fact that each iteration described in Eq. (20) propagates into the nearest neighborhood. Thus, the number of iterations required for convergence is scaled with the network diameter. Still, both of the methods act as bottlenecks in the system, as seen in Fig. 7b.

4.4. Limitations & future work

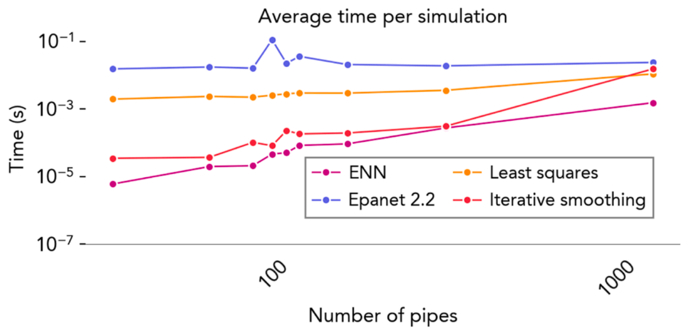
Although EGNNs show better performance for most of the networks and display good evidence for transferability, there are several possible paths for improvement.

Firstly, we identified that the main difficulties arise in modeling the pipes near the reservoir and pipes with high hydraulic resistance κ. Future avenues could employ an architecture that simultaneously predicts on a nodal and pipe level. The main purpose of such architecture is to achieve differentiable predictions of both heads and pipe flows. This can be done, for example, by updating nodal representation simultaneously with the one of edges and predicting the pressures in the output. Such a setting will optimize for the accuracy of pressures during training, and provide further speed-ups. Further analysis is required to ensure that this setting will result in the same transferability as for EGNN. Another option could leverage a richer training corpus composed of a multitude of synthetically generated networks (Sitzenfrei et al., 2013) and various demand sampling techniques.

Secondly, both EGNNs and GNNs perform computation locally, i.e. for every node or edge and its neighborhood in parallel. Since physical



(a) Evaluation of speed-up with respect to EPANET



(b) Evaluation of average execution time required by EGNN to compute flows, pressure reconstruction methods, and EPANET

Fig. 7. Comparison of execution time and speed-up.

processes within WDS are not strictly local and are dependent on the global topology altogether (Todini and Pilati, 1988), the perceptive field of the deep learning model must be relative to the network diameter to convey the global topology. This results in scaling difficulties and challenges with propagating the signal from distant nodes (NT and Maehara, 2019). Conversely, hydraulic simulators effectively approach this challenge by approximating the state with the Global Newton-Raphson algorithm (Todini and Rossman, 2013). Possible directions to tackle the issue include decoupling the WDS graph from the computational graph, for example with a coarser network (Shamir and Salomons, 2008; Paluszczyszyn et al., 2013) or graph rewiring techniques (Gasteiger et al., 2019; Chamberlain et al., 2021a).

Lastly, representing other elements of WDS, such as multiple pumps, valves, and tanks remains an open challenge. At this point, the EGNN can include readings from flow meters as an input, while further research could investigate incorporating readings from pressure meters and reservoir heads in the input of the model as a part of $\mathbf{x_e}$. This will extend the analysis to networks with multiple reservoirs. In the extended period simulation, tanks can be included similarly to Rossman (2022). However, that imposes additional complications, as the prediction errors at each steady state can propagate to the following time steps. Posing the problem as a time series rather than a steady state prediction might be beneficial. In that case the metamodel inputs and outputs several prediction for several timesteps, similar to metamodels that accelerate simulations for urban drainage systems (Garzón et al., 2022).

5. Conclusion

In this study, we proposed a new approach to metamodelling of WDS using edge-based graph neural networks. The metamodelling is performed in 2 consecutive steps: flowrate prediction and pressures derivation. We additionally presented a novel approach to the derivation of pressures from flowrates using power approximation.

The study evaluated 2 main properties of the model: generalization to varying demands and pipe parameters in the dataset with fixed topology and the transferability to unknown topology. The model shows better generalization capabilities than the traditional GNN-based metamodels, consistently providing higher prediction accuracy in terms of flowrates. For most of the cases, EGNN shows similar or better results in terms of predicted pressures. We additionally displayed that EGNN can generalize the predictions out-of-sample. By leveraging inductive biases rooted in the physical laws of mass conservation, the model presents the best transferability so far. This method showed higher speedups for smaller WDSs (up to 350 times for smaller networks and 8 times for larger networks) and is parallelizable on GPUs.

Improved generalization of a metamodel can be leveraged in downstream tasks that require evaluation of alternative layouts and pipe parameterizations. We believe the proposed methodology is a first step towards the development of a fully transferable EPANET metamodel, capable of accurate zero-shot predictions for unseen case studies. Future research could address the mentioned limitations and investigate the performance of our method with available sensor readings and the applicability of EGNNs in surrogate modeling over an extended period.

CRediT authorship contribution statement

Bulat Kerimov: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Conceptualization. **Riccardo Taormina:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization. **Franz Tscheikner-Gratl:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The code and data are available from an online repository at https://github.com/erytheis/egnn.

Acknowledgements

The authors acknowledge the financial support through the NTNU Green2050 Centre for Green Shift in the Built Environment and the TU Delft AI Labs programme. The authors would also like to thank Maosheng Yang, Dr Elvin Isufi, and other fellow colleagues in AIdroLab and NTNU for their insightful comments.

Appendix A. Vector representation of batching

Multiple networks can be processed in a batch. In that case, Laplacian matrices are stacked as a block diagonal matrix that represents the connectivity of a large graph network with n connected components, where n is the batch size. Edge features and flowrates are concatenated in the first dimension, i.e.

$$\mathbf{L}_e = \begin{bmatrix} \mathbf{L}_e^{(1)} & & \\ & \ddots & \\ & & \mathbf{L}_e^{(n)} \end{bmatrix} \quad \mathbf{x}_e = \begin{bmatrix} \mathbf{x}_e^{(1)} \\ \vdots \\ \mathbf{x}_e^{(n)} \end{bmatrix} \quad \mathbf{f} = \begin{bmatrix} \mathbf{f}^{(1)} \\ \vdots \\ \mathbf{f}^{(n)} \end{bmatrix}$$

As a block diagonal matrix, combined $\mathbf{L}_e$ is sparse. Sparse aggregation techniques allow for parallel multiplication of thousands of WDSs (Yang et al., 2018).

Appendix B. Training details

Below we provide hyperparameters selected for the experiment. All of the experiments were executed on the cluster node with Intel Xeon(R) Gold 6132 CPU and NVIDIA V100 with 16GB graphic memory. The batch sizes for each experiment were selected to fully fill the GPU.

B.1. Experiment 1

For each of the case studies, we trained ENNs with Adam optimizer for 70,000 epochs with a learning rate of 0.003. The number of layers L corresponds to the diameter of each WDS. For every case except for BAK, we chose a 2- layer MLPs as an encoder MLP$_\psi$ and a single-layer MLP$_\xi$. A nonlinearity σ in the encoder ψ_e is LeakyReLU with a coefficient of 0.2. Only the first term in loss function 21 is used, as it was sufficient.

For BAK the training procedure required additional fine-tuning. After 10,000 epochs, we decreased the learning rate to 0.0003. The coefficients α_1 and α_2 in the loss function are 1000 and 1 correspondingly. Additionally, ξ is a 2-layer MLP with Tanh activation function.

For the baseline GNN, we used the same set of hyper-parameters as provided in the original work. The provided code uses TensorFlow framework.

B.2. Experiment 3

In the case of transferable training we additionally normalized the input features κ between and scaled the flowrates by multiplying by 10. The selected learning rate is equal to 0.001 and a batch size is selected as 1000. We additionally applied a gradient clipping with a maximum value of 0.01. We selected LeakyReLU with the coefficient of 0.2 as a nonlinearity σ in the encoder ψ_e. For the rest of the network, the σ is a Tanh. After each layer of MLP we applied a dropout with the probability of 0.1. In addition, we randomly change the orientation in the representation every 25 epochs.

Appendix C. Additional experiment results

C.1. Generalization

This section includes a visualization of the evaluation of trained EGNN models in Experiment 3.2. Particularly, Fig. C.1 presents the difference in performances of trained models measured by MAE. The plot shows the same trend as in Fig. 4.

Since the network topology is the same in the training and testing set, the prediction accuracy is visibly higher. However, the limitations highlighted in Section 4.2 remain. Particularly in Fig. 5 we see errors in pressures, propagated from the errors in flowrates. Additionally, relative errors in predicted flowrates are higher for smaller flows.

C.2. Model sensitivity to demand values

In a separate experiment, we explore the sensitivity of the model trained in Section 3.3 to nodal demands and energy losses within the system. This evaluation setting differs from the one presented in Table 5. Instead of sampling $\mathbf{q}$ uniformly, we employed the original values from *.inp* files and scaled them with a fixed multiplier m. The original demand values are also visualized in Fig. C.3. We obtained 200 simulations per network by varying m between 0 and 100 % (400 % for ASnet2) with a step of 0.5 %. We use the head difference between the reservoir and a node with the lowest pressure

$$\Delta\mathbf{p}_{max} = \mathbf{p}_r - \min_{i\in\mathscr{V}}\mathbf{p}_i$$

as a proxy for the total energy losses in the system.

Fig. C.4 depicts the sensitivity of the model to the input demands and energy losses. For every network except the largest, i.e. ZJ, the model produces predictions with positive R^2 when $\Delta\mathbf{p}_{max}$ is contained within 10 m. Depending on the network, $\Delta\mathbf{p}_{max}$ corresponds to specific demand ranges, mostly under 10 L/S

The root cause of the limitation arises from the distribution of the nodal demands in the training set. As we see in Fig. C.5, the demands are contained within the range of 0 and 5 L/S and are sampled uniformly. Conversely, the original $\mathbf{q}$ from *.inp* files does not follow the same distribution (Fig. C.3). This is especially visible in Apulia, where the lower bound of the demands is located on 8 L/S, contrary to 0 L/S in all other networks. This could explain a discrepancy between accuracies depicted in Fig. C.4 and in Table 5.

One can also observe that the accuracy decreases when $\Delta\mathbf{p}_{\max}$ is small. This could be the result of the higher relative errors for smaller flowrates mentioned above and a general sensitivity of R^2 to errors when the target values are small. A broader range of demand distributions and alternative sampling techniques, such as latin hypercube (Babayan et al., 2005) and sparse sampling, may hold the potential for improvement.

Fig. C.1
Fig. C.2
Fig. C.3
Fig. C.4
Fig. C.5

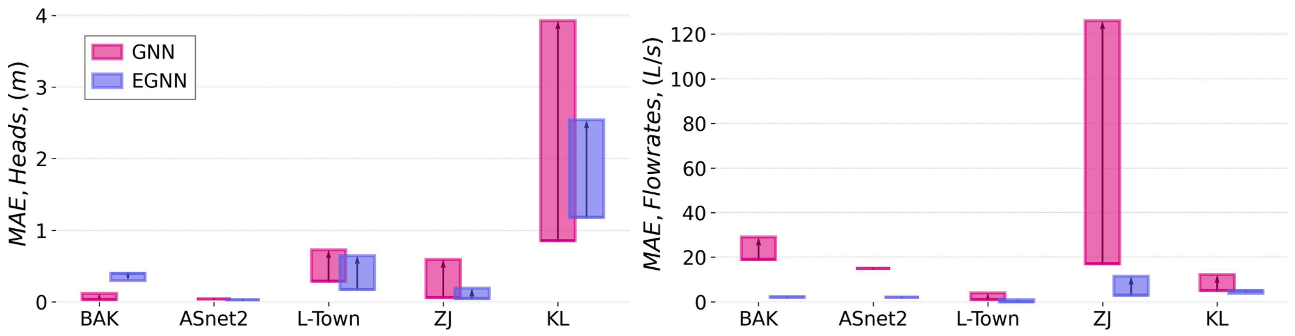
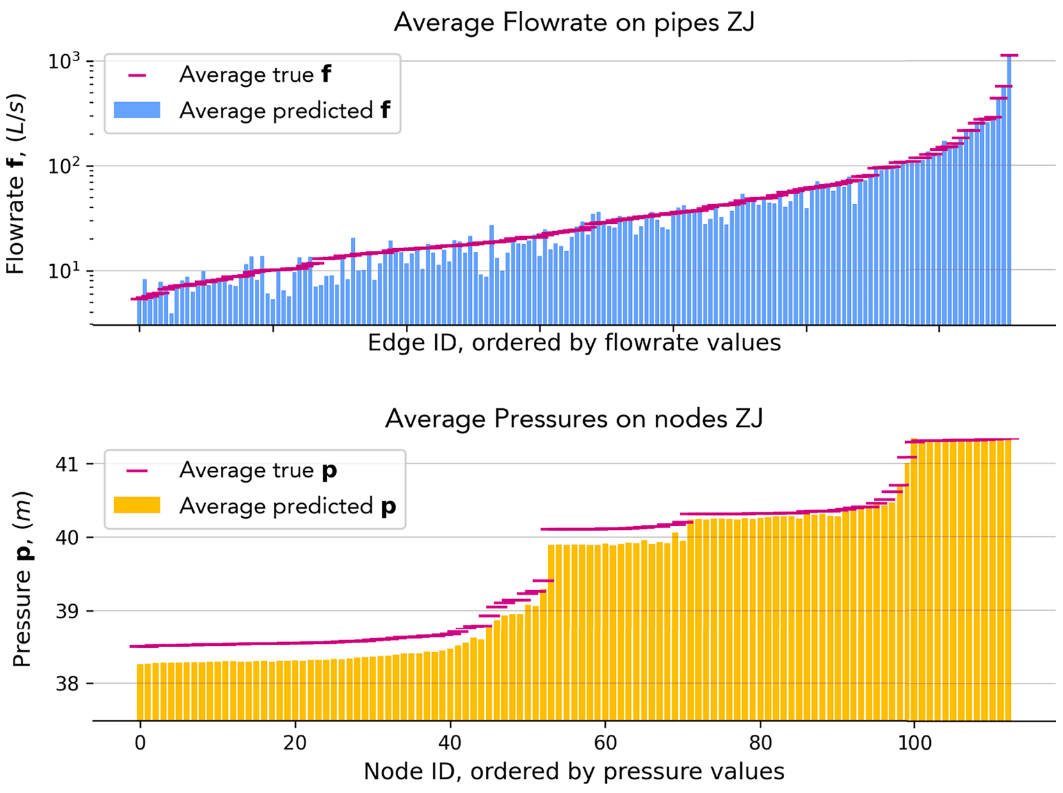
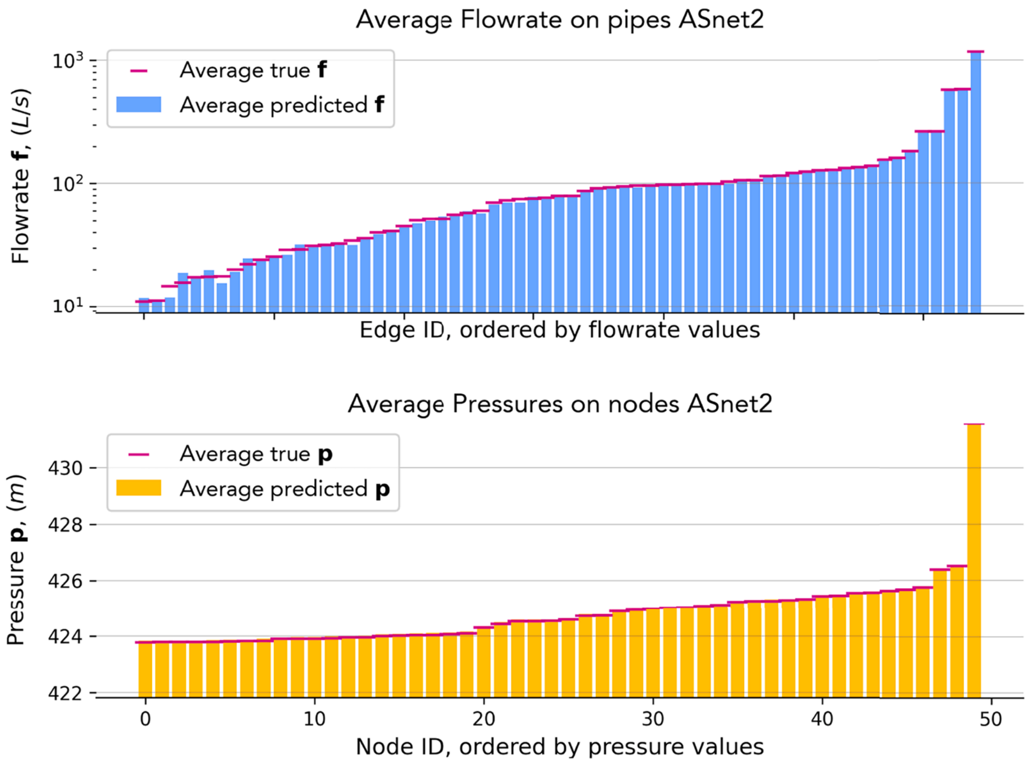


Fig. C.1. Degradation of performance (MAE) with the introduction of variability of the parameters in terms of predicted heads (left) and flowrates (right) between GNN (pink) and EGNN (purple). The top of the bar corresponds to the accuracy with fixed pipe parameters (Subset A in Section 3.2), while the bottom of the bar corresponds to the accuracy with introduced variability in κ and $\mathbf{q}$ (Subsest B in Section 3.2).



(a)



(b)

Fig. C.2. Visualization of generalization to **q** and κ (Subset B of Experiment 3.2) on the example of ZJ (top) and ASnet2 (bottom). The values are averaged over ($n =$ 1000) samples.

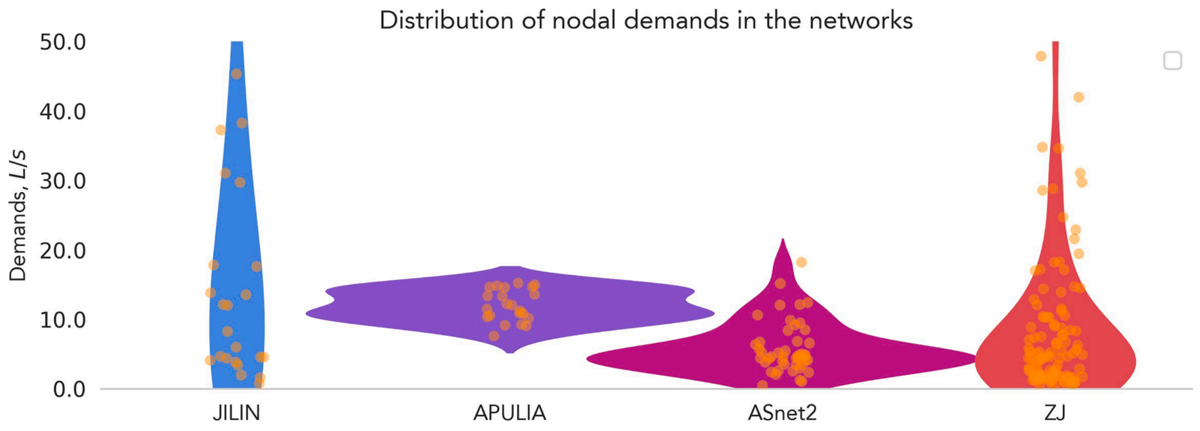


Fig. C.3. Demand distribution in the original *.inp* files.

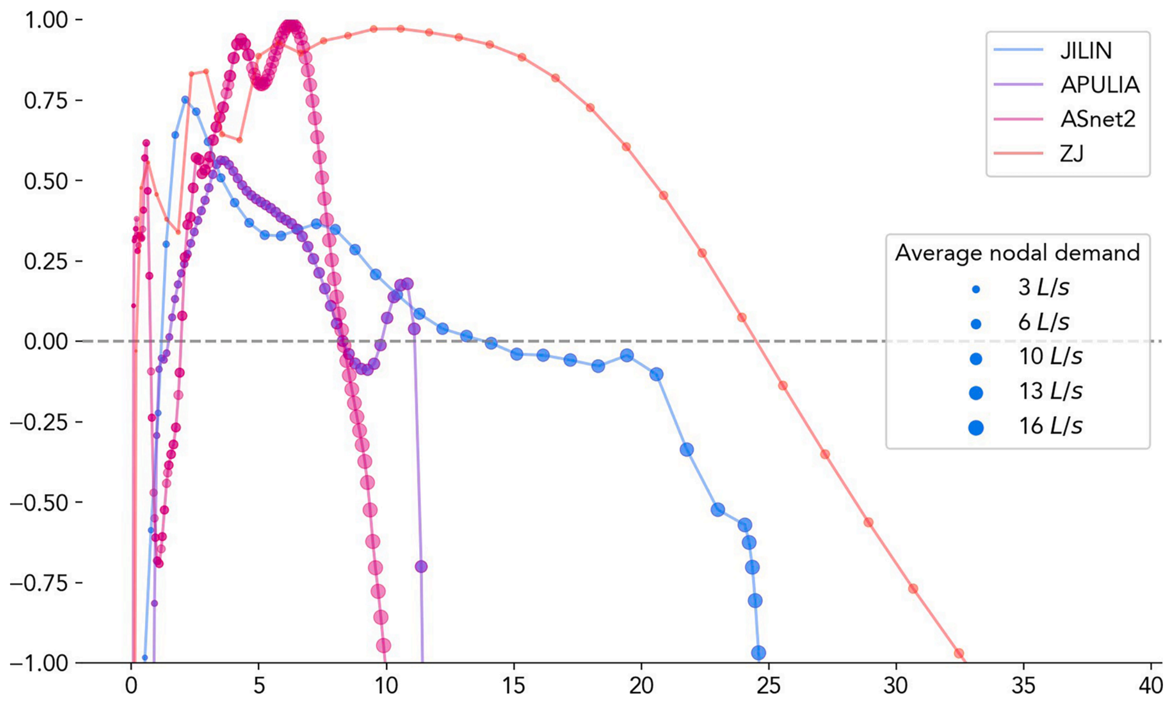


Fig. C.4. Sensitivity of the prediction accuracy to the maximum head difference and average nodal demand. Each point corresponds to a simulation and is scaled by the average nodal demand.

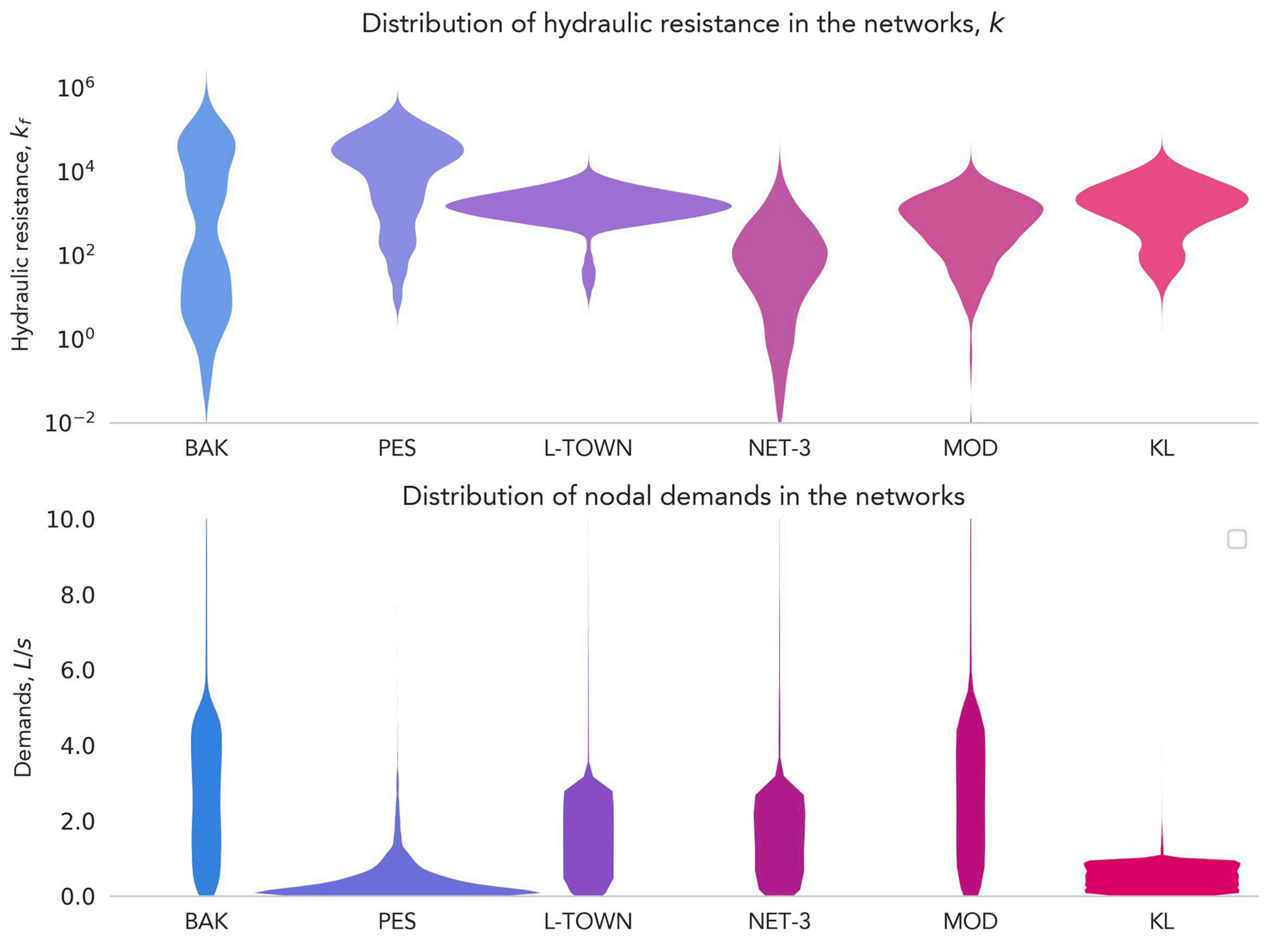


Fig. C.5. Demand and pipe parameter distributions for generated networks in the training set of the experiment 3.3.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: large-scale machine learning on heterogeneous systems. URL: https://www.tensorflow.org/.software available from tensorflow.org.

Ashraf, I., Hermes, L., Artelt, A., Hammer, B., 2023. Spatial graph convolution neural networks for water distribution systems. Advances in intelligent data analysis XXI. Springer, Nature Switzerland, Cham, pp. 29–41.

Babayan, A., Kapelan, Z., Savic, D., Walters, G., 2005. Least-cost design of water distribution networks under demand uncertainty. J. Water. Resour. Plan. Manage 131 (5), 375–382. https://doi.org/10.1061/(ASCE)0733-9496(2005)131:5(375).

Bandelt, H.J., Chepoi, V., Eppstein, D., 2010. Combinatorics and geometry of finite and infinite squaregraphs. SIAM. J. Discret. Math. 24 (4), 1399–1440.

Beers, L., Mulas, R., 2024. At the end of the spectrum: chromatic bounds for the largest eigenvalue of the normalized Laplacian. arXiv preprint arXiv:2402.09160.

Belghaddar, Y., Chahinian, N., Seriai, A., Begdouri, A., Abdou, R., Delenne, C., 2021. Graph convolutional networks: application to database completion of wastewater networks. Water 13 (12). https://doi.org/10.3390/w13121681. URL: https://www.mdpi.com/2073-4441/13/12/1681.

Bi, W., Dandy, G., 2014. Optimization of water distribution systems using online retrained metamodels. J. Water. Resour. Plan. Manage 140, 04014032. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000419.

Bodnar, C., Frasca, F., Wang, Y.G., Otter, N., Montúfar, G., Liò, P., Bronstein, M., 2021. Weisfeiler and Lehman go topological: message passing simplicial networks. URL: https://arxiv.org/abs/2103.03212, doi:10.48550/ARXIV.2103.03212.

Bragalli, C., D'Ambrosio, C., Lee, J., Lodi, A., Toth, P., 2012. On the optimal design of water distribution networks: a practical MINLP approach. Optim. Eng. 13 (2), 219–246. https://doi.org/10.1007/s11081-011-9141-7.

Broad, D.R., Dandy, G.C., Maier, H.R., 2004. A metamodeling approach to water distribution system optimization. Critical transitions in water and environmental resources management. https://doi.org/10.1061/40737(2004)453.

Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D., 2020. Language models are few-shot learners URL: https://arxiv.org/abs/2005.14165, doi:10.48550/ARXIV.2005.14165.

Cesario, L., Association, A.W.W., 1995. Modeling, analysis, and design of water distribution systems. American Water Works Association. URL: https://books.google.no/books?id=7SBSAAAAMAAJ.

Chamberlain, B., Rowbottom, J., Eynard, D., Di Giovanni, F., Dong, X., Bronstein, M., 2021a. Beltrami flow and neural diffusion on graphs. Advances in neural information processing systems. Curran Associates, Inc., pp. 1594–1609. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/0cbed40c0d920b94126eaf5e707be1f5-Paper.pdf

Chamberlain, B., Rowbottom, J., Gorinova, M.I., Bronstein, M., Webb, S., Rossi, E., 2021b. GRAND: graph neural diffusion. In: Proceedings of the 38th international conference on machine learning, PMLR, pp. 1407–1418. URL: https://proceedings.mlr.press/v139/chamberlain21a.html.

Chen, M., Wei, Z., Huang, Z., Ding, B., Li, Y., 2020. Simple and deep graph convolutional networks. In: Proceedings of the 37th International conference on machine learning, PMLR, pp. 1725–1735. URL: https://proceedings.mlr.press/v119/chen20v.html.

Dandy, G., 2016. 06 Zhi Jiang. International systems 6. URL: https://uknowledge.uky.edu/wdst_international/.

Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. BERT: pre- training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (Long and Short Papers), Association for com- putational linguistics, minneapolis, minnesota, pp. 4171–4186. https://doi.org/10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

Donon, B., Clement, R., Donnot, B., Marot, A., Guyon, I., Schoenauer, M., 2020. Neural networks for power flow: graph neural solver. Electric Power Syst. Res. 189, 106547 https://doi.org/10.1016/j.epsr.2020.106547.

Fuertes, P.C., Alzamora, F.M., Carot, M.H., Campos, J.A., 2020. Building and exploiting a digital twin for the management of drinking water distri- bution networks. Urban. Water. J. 17 (8), 704–713. https://doi.org/10.1080/1573062X.2020.1771382.

Gama, F., Isufi, E., Leus, G., Ribeiro, A., 2020. Graphs, convolutions, and neural networks: from graph filters to graph neural networks. IEEe Signal. Process. Mag. 37, 128–138. https://doi.org/10.1109/MSP.2020.3016143.

Garzón, A., Kapelan, Z., Langeveld, J., Taormina, R., 2022. Ma- chine learning-based surrogate modeling for urban water networks: review and future research directions. Water. Resour. Res. 58 (5) https://doi.org/10.1029/2021WR031808. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2021WR031808.

Gasteiger, J., Weißenberger, S., Günnemann, S., 2019. Diffusion improves graph learning. In: Proceedings of the 33rd international conference on neural information processing systems.

Hajgató, G., Gyires-Tóth, B., Paál, G., 2021. Reconstructing nodal pres- sures in water distribution systems with graph neural networks URL: https://arxiv.org/abs/2104.13619, arXiv:2104.13619.

Hall, A., 2021. 01 Apulia. URL: https://uknowledge.uky.edu/wdst_international/1. international Systems, 1.

Isufi, E., Gama, F., Shuman, D.I., Segarra, S., 2022. Graph filters for signal processing and machine learning on graphs. arXiv:2211.08854.

Jia, J., Schaub, M.T., Segarra, S., Benson, A.R., 2019. Graph-based semi- supervised &amp active learning for edge flows. In: Proceedings of the 25th ACM SIGKDD International conference on knowledge Discov- ery Data Mining. https://doi.org/10.1145/3292500.3330872.

Kang, D., Lansey, K., 2012. Revisiting optimal water-distribution system design: issues and a heuristic hierarchical approach. J. Water. Resour. Plan. Manage 138, 208–217. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000165.

Kerimov, B., Bentivoglio, R., Garzón, A., Isufi, E., Tscheikner-Gratl, F., Steffelbauer, D.B., Taormina, R., 2023. Assessing the performances and transferability of graph neural network metamodels for water distribution systems. J. Hydroinformatics. https://doi.org/10.2166/hydro.2023.031.

Kipf, T.N., Welling, M., 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 .

Klise, K.A., Murray, R., Haxton, T., 2018. An overview of the water net- work tool for resilience (WNTR). In: 1st International Water Distribution Systems Analysis (WDSA) /3rd International Conference on Computing and Control for the Water Industry (CCWI) Joint Conference.

Lee, S.C., Lee, S.I., 2001. Genetic algorithms for optimal augmentation of water distribution networks. J. Korea Water Res. Ass. 34 (5), 567–575.

Levie, R., Isufi, E., Kutyniok, G., 2019. On the transferability of spectral graph filters. In: 2019 13th International conference on sampling theory and applications (SampTA), pp. 1–5. URL: https://api.semanticscholar.org/CorpusID:59413913.

Lima, G.M., Brentan, B.M., Manzi, D., Luvizotto, Edevar, J., 2017. Metamodel for nodal pressure estimation at near real-time in water distribution systems using artificial neural networks. J. Hydroinform. 20 (2), 486–496. https://doi.org/10.2166/hydro.2017.036 arXiv. https://iwaponline.com/jh/article-pdf/20/2/486/657814/jh0200486.pdf.

Makropoulos, C., Savić, D.A., 2019. Urban hydroinformatics: past, present and future. Water 11 (10). https://doi.org/10.3390/w11101959. URL: https://www.mdpi.com/2073-4441/11/10/1959.

Mala-Jetmarova, H., Sultanova, N., Savic, D., 2018. Lost in optimisation of water distribution systems? A literature review of system design. Water 10 (3). https://doi.org/10.3390/w10030307. URL. https://www.mdpi.com/2073-4441/10/3/307.

McClymont, K., Keedwell, E., Savic, D., 2015. An analysis of the interface between evolutionary algorithm operators and problem features for water resources problems. a case study in water distribution network design. Environm. Modell. Software 69, 414–424. https://doi.org/10.1016/j.envsoft.2014.12.023. URL. https://www.sciencedirect.com/science/article/pii/S1364815215000225.

Meijer, D., Post, J., van der Hoek, J.P., Korving, H., Langeveld, J., Clemens, F., 2021. Identifying critical elements in drinking water distribution networks using graph theory. Struct. Infrastruct. Eng. 17 (3), 347–360. https://doi.org/10.1080/15732479.2020.1751664 arXiv.

Neuman, A.M., Bramburger, J.J., 2023. Transferability of graph neu- ral networks using graphon and sampling theories. arXiv preprint arXiv:2307.13206 .

NT, H., Maehara, T., 2019. Revisiting graph neural networks: all we have is low-pass filters. ArXiv. abs/1905.09550. URL https://api.semanticscholar.org/CorpusID:162183860.

Paluszczyszyn, D., Skworcow, P., Ulanicki, B., 2013. Online simplificaton of water distribution network models for optimal scheduling. J. Hydroinformatics 15 (3), 652–665. https://doi.org/10.2166/hydro.2013.029. URLarXiv. https://iwaponline.com/jh/article-pdf/15/3/652/387014/652.pdf.

Pasha, M.F.K., Lansey, K., 2014. Strategies to develop warm solutions for real-time pump scheduling for water distribution systems. Water Resour. Manage. 28, 3975–3987.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. Pytorch: an imperative style, high-performance deep learning library. In: Advances in neural information processing systems 32. Curran Associates, Inc., pp. 8024–8035. https://doi.org/10.1016/j.envsoft.2013.05.006. URL. http://papers.neurips.cc/paper/

Stewart, G., 1998. Matrix algorithms: volume 1, basic decompositions. Society for industrial and applied mathematics, pp. 54–55. URL: https://books.google.no/books?id=XHOQ_HU-85IC.

Todini, E., Pilati, S., 1988. A gradient algorithm for the analysis of pipe networks. Comput. Applicat. Water Supp. 1, 1–20.

Todini, E., Rossman, L.A., 2013. Unified framework for deriv- ing simultaneous equation algorithms for water distribution net- works. J. Hydraulic Eng. 139 (5), 511–526. https://doi.org/10.1061/(ASCE)HY.1943-7900.0000703. URL. https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29HY.1943-7900.0000703.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Canton Ferrer, C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Singh Koura, P., Lachaux, M.A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E.M., Subramanian, R., Tan, X.E., Tang, B., Taylor, R., Williams, A., Kuan, J.X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., Scialom, T., 2023. Llama 2: open foundation and fine-tuned chat models. arXiv e-prints p. arXiv:2307.09288, arXiv:2307.09288. doi:10.48550/arXiv.2307.09288, arXiv:2307.09288.

Tsiami, L., Makropoulos, C., 2021. Cyber—Physical attack detection in water distribution systems with temporal graph convolutional neural networks. Water. (Basel) 13 (9). URL. https://www.mdpi.com/2073-4441/13/9/1247.

Vrachimis, S.G., Eliades, D.G., Taormina, R., Kapelan, Z., Ostfeld, A., Liu, S., Kyriakou, M., Pavlou, P., Qiu, M., Polycarpou, M.M., 2022. Battle of the leakage detection and isolation methods. J. Water. Resour. Plan. Manage 148 (12). https://doi.org/10.1061/(ASCE)WR.1943-5452.0001601. 9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Xie, Y., Xu, Z., Zhang, J., Wang, Z., Ji, S., 2022. Self-supervised learning of graph neural networks: a unified review. arXiv:2102.10757.

Rong, K., Fu, M., Chen, J., Zheng, L., Zheng, J., Yaseen, Z., 2021. Graph neural network for integrated water network partitioning and dynamic district metered areas. Sci. Rep. 12 https://doi.org/10.21203/rs.3.rs-772506/v1.

Rossman, L.A., 2016. 06 EPANET Net 3. software manual examples. https://uknowledge.uky.edu/wdst_manuals/2.

Rossman, L.A., 2022. Epanet 2.

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., Battaglia, P., 2020. Learning to simulate complex physics with graph networks. In: III, H.D., Singh, A. (Eds.), Proceedings of the 37th International conference on machine learning, PMLR, pp. 8459–8468.

Sato, R., Yamada, M., Kashima, H., 2021. Random features strengthen graph neural networks. In: Proceedings of the 2021 SIAM international conference on data mining (SDM), SIAM, pp. 333–341.

Schaub, M.T., Benson, A.R., Horn, P., Lippner, G., Jadbabaie, A., 2020. Random walks on simplicial complexes and the normalized hodge 1- laplacian. SIAM Rev. 62 (2), 353–391. https://doi.org/10.1137/18m1201019.

Schaub, M.T., Segarra, S., 2018. Flow smoothing and denoising: graph signal processing in the edge-space. In: 2018 IEEE Global conference on signal and information processing (GlobalSIP). IEEE. https://doi.org/10.1109/globalsip.2018.8646701.

Shamir, U., Salomons, E., 2008. Optimal real-time operation of urban water distribution systems using reduced models. J. Water. Resour. Plan. Manage 134 (2), 181–185. https://doi.org/10.1061/(ASCE)0733-9496(2008)134:2(181). URL. https://ascelibrary.org/doi/abs/10.1061/%28ASCE%290733-9496%282008%29134%3A2%28181%29.

Sitzenfrei, R., Möderl, M., Rauch, W., 2013. Automatic generation of water distribution systems based on gis data. Environ. Modell. Software 47, 138–147. URL. https://www.sciencedirect.com/science/article/pii/S1364815213001163.

Xing, L., Sela, L., 2022. Graph neural networks for state estimation in water distribution systems: application of supervised and semisupervised learning. J. Water. Resour. Plan. Manage 148. https://doi.org/10.1061/(ASCE)WR.1943-5452.0001550.

Yang, C., Buluç, A., Owens, J.D., 2018. Design principles for sparse matrix multiplication on the GPU. In: Euro-Par 2018: Parallel Pro- cessing: 24th international conference on parallel and distributed computing, Turin, Italy. Springer- Verlag, Berlin, Heidelberg, pp. 672–687. https://doi.org/10.1007/978-3-319-96983-1_48. August 27 - 31, 2018, Proceedings.

Yang, M., Isufi, E., Leus, G., 2022a. Simplicial convolutional neural net- works. In: Proceedings of the ICASSP 2022 - 2022 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, United States, pp. 8847–8851. https://doi.org/10.1109/ICASSP43922.2022.9746017.

Yang, M., Isufi, E., Schaub, M.T., Leus, G., 2022b. Simplicial convolutional filters. IEEE Transact. Signal Process. 70, 4633–4648. https://doi.org/10.1109/tsp.2022.3207045.

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. Sustain. Cities. Soc. 86, 104090 https://doi.org/10.1016/j.scs.2022.104090. URL. https://www.sciencedirect.com/science/article/pii/S2210670722004073.

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 Res., 120264 https://doi.org/10.1016/j.watres.2023.120264. URL. https://www.sciencedirect.com/science/article/pii/S0043135423007005.

Zhou, X., Liu, S., Xu, W., Xin, K., Wu, Y., Meng, F., 2022. Bridginghydraulics and graph signal processing: a new perspective to estimate water distribution network pressures. Water Res. 217, 118416 https://doi.org/10.1016/j.watres.2022.118416. URL. https://www.sciencedirect.com/science/article/pii/S0043135422003724.

Ziegler, C., Skardal, P.S., Dutta, H., Taylor, D., 2022. Balanced Hodge Laplacians optimize consensus dynamics over simplicial complexes. Chaos: An Interdisciplin. J. Nonlinear Sci. 32 (2).

Zischg, J., Mair, M., Rauch, W., Sitzenfrei, R., 2015. Stochastic performance assessment and optimization strategies of the water supply network transition of kiruna during city relocation, pp. 853–862. URL: https://ascelibrary.org/doi/abs/10.1061/9780784479162.080, doi:10.1061/9780784479162.080, arXiv:https://ascelibrary.org/doi/pdf/10.1061/9780784479162.080.

Örn Garðarsson, G., Boem, F., Toni, L., 2022. Graph-based learning for leak detection and localisation in water distribution networks. In: 11th IFAC symposium on fault detection, supervision and safety for technical processes SAFEPROCESS 2022, pp. 661–666. https://doi.org/10.1016/j.ifacol.2022.07.203. URL. https://www.sciencedirect.com/science/article/pii/S2405896322005882.