A Project Report on

Physics-Integrated GCNs for State Estimation and Leak Detection in Water Networks under Supervised and Semi-Supervised Learning

Project Report submitted in partial fulfilment of the requirements for the 4th Semester of Master of Computer Application (MCA)

under University of Calcutta

Submitted by

Name: Soumyajit Banerjee

Roll No.: C91/MCA/232025

Registration No: 012-1111-1646-20

Under the supervision of

Dr. Amlan Chakrabarti

Director & HOD,

A.K. Choudhury School of Information Technology

UNIVERSITY OF CALCUTTA

A Project Report on

Physics-Integrated GCNs for State Estimation and Leak Detection in Water Networks under Supervised and Semi-Supervised Learning

Project Report submitted in partial fulfilment of the requirements for the 4th Semester of Master of Computer Application (MCA)

under University of Calcutta

Submitted by

Name: Soumyajit Banerjee

Roll No.: C91/MCA/232025

Registration No: 012-1111-1646-20

Under the supervision of

Dr. Amlan Chakrabarti

Director & HOD,

A.K. Choudhury School of Information Technology

UNIVERSITY OF CALCUTTA

www.caluniv.ac.in

A.K. CHOUDHURY SCHOOL OF INFORMATION TECHNOLOGY

UNIVERSITY OF CALCUTTA

University of Calcutta, JD-2 Sector-III, Saltlake, Kolkata-700106, INDIA

CERTIFICATE

This is to certify that the project synopsis entitled **"Physics-Integrated GCNs for State Estimation and Leak Detection in Water Networks under Supervised and Semi-Supervised Learning"** submitted for partial fulfilment of the requirements of 4th Semester of Master of Computer Application (MCA) under University of Calcutta; has been carried out by **Soumyajit Banerjee** (**Roll No- C91/MCA/232025** and **Registration No-012-1111-1646-20**) under the supervision of **Prof. Amlan Chakrabarti Director &HOD, A.K. Choudhury School of Information Technology University of Calcutta.**

---	-----------------------------------
CHAIRMAN, BOARD OF EXAMINER	PROJECT SUPERVISOR

External Examiner(s)

ACKNOWLEDGEMENT

I wish to express my profound sense of gratitude to my project supervisor, Prof. Amlan Chakrabarti, Director & HOD, AKCSIT, University of Calcutta, for his unwavering support, inspiration, and guidance. He has shown me different ways to approach problems and emphasized the importance of persistence in research. It has been a valuable and enriching experience to work under his mentorship.

I would also like to extend my sincere thanks to my project guide, Dr. Dola Gupta, DST Women Scientist, for her insightful suggestions, consistent support, and patient guidance throughout the course of this work. Her inputs have significantly shaped the direction and quality of my project.

I am deeply grateful to all the respected professors and office staff of the A.K. Choudhury School of Information Technology, University of Calcutta, for their continued support, encouragement, and dedication throughout the duration of my MCA program. Their guidance has been instrumental in my academic and professional development.

Lastly, I would like to thank my friends, seniors, and everyone who directly or indirectly supported me during this project. Their encouragement and motivation have been a source of strength, and I remain truly appreciative of their support.

Date: --------------------------------

Student Name

Table of Contents

Abstract

The call for reliable and technology-backed water systems shows the importance of fast and accurate state-estimation tools in the water sector. SE helps pinpoint piping flows and nodal head values in areas where no measurements are taken, using what is observed from few sensors. SE solvers built using conventional physics assume detailed hydraulic information and do many repeated computations that make them difficult to implement in real time. In contrast, Graph Convolutional Networks (GCNs) can move their heavy calculations to before deployment and give results within milliseconds during use.

This study introduces a dedicated GCN architecture for WDS SE. Two factors motivate the choice:

Semi-supervised learning requirement – the limited number of monitoring nodes makes label scarcity intrinsic to the problem.

Graph awareness – Network architecture is used in GCNs, so the network can detect how each point affects others and incorporate restrictions directly in the loss function.

We evaluated the model under different learning modes (with or without labels), adjusted its main design elements and measured the effects of higher levels of sensor noise. The GCN reproduced the head quantities and inter-regional traffic with a high correlation of > 0.99 when exposed to 5 % noisy inputs. They prove that the GCN is able to use just a few and noisy data points with the help of its graph structure.

It takes the field near real-time by using GCN and offers a base for connecting both monitoring and control functions. Future research should try making the model suitable for systems with dynamic hydraulic devices and see its performance on city networks with more than 10 000 nodes.

Chapter 1: Introduction

1.1 Background and Significance

It is the task of Water Distribution Systems (WDSs) to bring treated water to houses, shops and factories in cities. As more cities grow, older assets are used and procedures become complex, it has become vital to use smart monitoring and control in WDSs. Estimating nodal pressure heads and pipe flows from a few measurements is an essential task in this field and this is called state estimation (SE)[1,2]. It is essential for immediate choices, proper use of resources and spotting any unusual patterns.

1.2 Limitations of Traditional State Estimation

SE approaches that are common use equations showing the conservation of mass and energy. Even though they work well under ideal situations, these models have practical limitations [3,4]:

High computational cost for solving large nonlinear systems.

Dependency on accurate system parameters (e.g., pipe roughness, demands).

Poor scalability in real-time applications.

Vulnerability to sensor sparsity and noise.

Because of these disadvantages, many organizations look toward using algorithms that observe and predict from previous or simulated data.

1.3 Rise of Graph Neural Networks in WDS Applications

GNNs have recently risen as a useful approach for handling both relational and topological data [5]. As a WDS can be naturally shown as a graph, G = (V, E), where nodes are junctions and edges are pipes, GNNs are the perfect fit for WDS modeling. Message passing is done by GNNs among connected nodes, so nodes can discover representations that show both their features and their position in the graph network [6].

In many GNN models, nodes are usually combined using only uniform or fixed-weight schemes which hinder their usefulness when networks have unequal or heterogeneous nodes [7]. For this reason, GCN architecture have been introduced:

1.4 Graph Convolutional Networks (GCNs)

GCNs operate in the spectral domain, performing convolutions over the graph using eigen decompositions of the Laplacian matrix [8]. Key characteristics include:

Feature smoothing over neighboring nodes.

Efficient implementation using localized filters.

Suitability for dense and homogeneously connected networks.

In the context of WDSs, GCNs offer a practical solution for estimating pressure heads across the network by propagating information from sensor-equipped nodes [9].

1.5 Motivation and Objectives

This project builds upon recent advances in GCN model to develop a unified pipeline for state estimation and leakage detection in WDSs. The specific objectives are:

- To implement GCN model tailored for hydraulic SE tasks.
- To integrate semi-supervised learningwith physics-based constraints.
- To use learned pressure fields for leak localizationand severity estimation.
- To benchmark the models on synthetic and real-world WDS configurations under various sensor placements and noise levels.

1.6 Methodology Overview

In our approach, the WDS is abstracted as a directed graph with:

1. Node features: Pressure readings, demands, boundary conditions.
2. Edge features: pipe roughness, diameter, length, and flow constraints.
3. GCN layers process this data to predict unobserved pressures and identify leak patterns. The models are trained using a hybrid loss that combines:
4. Supervised error between estimated and ground truth heads.
5. Unsupervised penalties for flow conservation and energy loss violations [10].

1.7 Leakage Detection Strategy

Leak detection is addressed as both:

Regression: estimate leak magnitude from pressure deviations.

Classification: identify the most probable leak node.

This dual-task learning setup improves generalization and supports practical deployment scenarios where ground truth leakage data is scarce [11].

1.8 Contributions

This project contributes to the field by:

1. Designing an interpretable and scalable pipeline using GCN.

2. Integrating physical laws into data-driven models for enhanced robustness.

3. Providing a replicable framework for future real-world deployments.

1.9 Structure of the Report

The remainder of this report is organized as follows:

Chapter 2: Related work on GNNs, GCNs, and WDS modeling.

Chapter 3: Technical architecture and theoretical formulations.

Chapter 4: Experimental setup, dataset synthesis, and training procedures.

Chapter 5: Results, metrics, and comparative evaluation.

Chapter 6: Discussion, limitations, and future research directions.

Chapter 7: Final conclusions and recommendations.

Chapter 2: Related Work

2.1 Physics-Centred Deterministic Solvers

Deterministic solvers are considered the standard because they implement the important rules of physics. Early formulations cast state estimation as a non-linear least-squares problem that minimises measurement residuals subject to Hazen–Williams or Darcy–Weisbach head-loss equations[12]. Although exact, these solvers scale super-linearly with network size and require reliable priors for pipe roughness and boundary conditions. Variants such as pseudo-dynamic damping or sparse Jacobian techniques slightly ease computation, but full-scale real-time deployment "(>10 k nodes)" is still rare.

A number of studies adjust deterministic solvers using adaptive damping which makes Newton–Raphson updates more stable with little data or uncertain demand predictions [13]. Even so, the method can't be flexible when the working conditions rarely match what was written in the documents.

2.2 Statistical and Bayesian Estimation

Measurement error was directly dealt with by using probabilistic theories. Bayesian approaches treat unknown demands and roughness factors as random variables, producing posterior distributions for heads and flows [14]. EKF/UKF are used for almost real-time tracking, but their assumptions fail during drastic changes [15].

By including uncertainty quantification in this strand, operators get a proper idea of risks, not only of one possible outcome. But simulating MCMC thousands of times for each parameter is too slow which is why researchers use surrogate models and GPU processing [16].

2.3 Sensor Placement Optimisation

Where the sensors are placed determines how well a system can estimate the environment. Optimisation-based placement tries to find the best placement for observability or reduce the amount of variance in estimates, while staying within budget[17]. If binary placements are of concern, MIP can be used, although they don't scale well, whereas heuristics such as genetic algorithms or greedy policies achieve quicker solutions at the cost of their optimality [18].

The popularity of learning-based estimators brought about different ways to decide on placements. Gradient-based schemes compute how the validation error varies with each node to choose those where the system learning is highest [19]. RL agents treat the problem as a step-by-step process and they improve on heuristics by ~10 % reduction in the mean absolute error on synthetic benchmarks [20]. It is difficult to use these systems outside the lab since there are many installation barriers, communication issues and official rules to follow.

2.4 Classical Machine-Learning without Graph Awareness

The years from 2015 to 2020 saw many tests of conventional ML methods because they did not require beforehand figuring out the detailed pipe structure or differential equations [21]. Support-Vector Regression (SVR) models map the sensor heads to unknown nodes through the help of kernels. Random Forests determine the rank of leak probabilities by making use of the diversity among the ensemble classifiers. Deep feed-forward Artificial Neural Networks (ANNs) copy the EPANET system behavior in milliseconds and GPs indicate the level of uncertainty involved [22].

Even though they are faster and at times accurate, these methods see nodes as separate, so they do not consider how the hydraulic system is connected. If there are changes in the network topology (for example, closing a pipe, adding a new loop) or if the training and deployment environments are not similar, the results degrade a lot [23]. This drawback shows the importance of learning that takes graphing into consideration.

2.5 Graph-Based Data-Driven Estimation

2.5.1 Generic Message-Passing Networks

In MPNNs, the idea of topology-aware aggregation was introduced which means node embeddings choose the information from nearby nodes for each update [24]. For the early experiments, toy WDSs had a lower Mean Absolute Error than multilayer perceptrons. Without physics constraints, WDSs demonstrated unrealistic behaviour of the head depending on the setting of the boundary conditions [25].

2.5.2 Graph Convolutional Networks (GCNs) as Diffusive Smoothers

The fixed normalized weights set by the Laplacian matrix are applied in GCNs [26]. In WDSs, sensor inputs are placed at each layer and then diffused through the upcoming convolution layers. GCNs are appealing to hardware designers since research shows that these networks converge quickly and use few parameters. Treating all neighbours equally may give less attention to main pipelines that carry a lot of water [27].

2.6 Leak Detection and Localisation

Leak detection looks at the differences between the predicted and measured heads. Graph wavelets in Graph Signal Processing help locate specific anomalies after the residuals are analyzed in the frequency domain. Such networks can make large graphs smaller, retaining important leaks and achieve excellent F1 scores in simulation [28].

GNN-based classifiers usually share editors with analytics used for other tasks which helps them gain from multi-task effects. Using these maps, one can easily see how water flows from a reservoir possibly to the leak which helps field staff look for the leak ahead of time [29].

2.7 Domain Adaptation and Transfer Learning

Most utilities do not have a lot of data that is clearly identified. By domain adaptation, the weights of a GNN from synthetic or nearby networks are moved to the target network using just small adjustments [30]. With dozens of tasks generated by simple network graphs, the initial parameters are changed rapidly enough by gradient descent to reach around 90 % of the performance gained by supervised learning using only 10 % of training data. They will likely lead to more use of artificial intelligence in remote areas [31].

Chapter 3: Architectural Design and Mathematical Foundations of GCN Model for State Estimation and Leakage Detection in Water Distribution Systems

3.1 Introduction

SE in WDS is the procedure of determining nodal pressures and flow of pipes based on limited information gathered from sensors in the system. Because WDS involve graphs, Graph Neural Networks (GNNs) and in particular Graph Convolutional Networks (GCNs), are now preferred over old, demanding hydraulic solvers. They make use of the structure and features present in the network by creating mathematical graphs for WDS.

Here, we study in detail the architecture and math of GCN and related training frameworks within the Encoder-Processor-Decoder (EPD) model and see how they are used in SE and leakage detection. We present a list of steps for GCN-based SE in WDS which are: (1) defining the model input and output, (2) setting up all the blocks within an EPD design, (3) framing loss functions together with physical priors and (4) testing how the model responds to variations in data quality and design. Besides, we evaluate how community assembly guidelines (GCN) change the performance and noise resistance of prediction models.

3.2 Problem Formulation

Consider a Water Distribution System (WDS) consisting of n nodes and m pipes. The node set is denoted by $N=\{1,2,\dots,n\}$, and the pipe set by $M=\{1,2,\dots,m\}$. Among the nodes, a subset $N_d \subset N_N\_$ corresponds to junction nodes, which serve as pipe connections or user demand points, while another subset $N_s \subset N_N\_$ represents source nodes, such as reservoirs or tanks. The node set can be decomposed as $N=N_d \cup N_s$.

Each pipe $p\in M$ is identified by its start node i and end node j, expressed as the ordered pair $p=(i,j)$.The following system parameters and measurements define the inputs to the state estimation model:

Network topology is represented by the adjacency matrix $A\in\{0,1\}^{n\times n}$, where the element $a_{ij}=1$ if nodes i and j are connected by a pipe, and $a_{ij}=0$ otherwise.

Demand at junctions is denoted by $q_i$for each junction node $i\in N_d$.

Hydraulic heads at source nodes are known and represented by H^*_i, $i \in N_s$.

Pipe loss coefficients c_p for each pipe $p \in M$ are computed using the empirical Hazen-Williams formula as:

$$c_p = \frac{10.67 \times \text{length}_p}{r_{\text{HW},p}^{1.852} \times \text{diameter}_p^{4.871}}$$

Wherer $_{\text{HW},p}$, diameter$_p$, and length$_p$ = dimensionless HazenWilliam coefficient, diameter, and length of pipe p in metric units, respectively.

Partial hydraulic head measurements at a subset of junctions $N_m \subseteq {}_{Nd}$ are available.

The goal of this work is to develop a Graph Neural Network (GNN) based model—specifically employing Graph Convolutional Networks (GCNs)—to estimate the hydraulic states of the WDS. These states include the nodal heads Hat all nodes and the flows Q through all pipes.

The GNN framework is formulated as a mapping from the input interaction graph$G_I=(V,E)$to the output interaction graph$G_O=(H,Q)$, where:

$V=\{V_i|i\in N\}$ is the set of node features, with each $V_i \in R^{dV}$ encoding node-specific inputs such as demands, known heads, and topological information.4

$E=\{E_p|p\in M\}$ is the set of edge features, with each $E_p \in R^{dE}$ representing pipe-specific inputs such as loss coefficients.

The output graph states consist of the nodal heads $H=\{H_i|i\in N\}$ and pipe flows $Q=\{Q_p|p\in M\}$. Note that the pipe flows Q can be deterministically derived from the nodal heads H using hydraulic relationships; The model focuses primarily on estimating H.

Formally, the GNN model is expressed as a parameterized function

$$M_\theta : G_I \rightarrow G_O,$$

where θ denotes the learnable parameters of the network. These parameters are optimized by minimizing a suitable loss function$\mathcal{L}(\mathrm{G_I}, \mathrm{G_O})$, which quantifies the discrepancy between predicted and observed states during training.

It makes it possible to merge the network layout, node and edge features and partial measurements in a single way to improve state estimation in water networks.

3.3 Architecture of the Model

The architecture of the Graph Convolution Network (GCN) is seen in Figure 1 and is made up of an Encoder, a Processor and a Decoder. It learns from how Water Distribution Systems (WDSs) are shaped and how they function. It has these three main parts:

Encoder: Makes use of the input features and the way WDS interconnections are arranged to produce latent vectors that fit for graph-based learning.

Processor: The latent states of every node are progressively updated using GCN layers, making use of message passing to highlight relationships.

Decoder: Maps the final latent representations into meaningful target outputs such as pressure (head) values and leakage probabilities.

Each of these components is described in detail below.

3.3.1 Encoder

To enable graph-based processing, the WDS configuration is first transformed into an input interaction graph G_I=(V,E) where:

V represents nodes corresponding to junctions and reservoirs.

E represents edges corresponding to pipes.

Each node i∈ V is associated with a 4-dimensional input feature vector:

$$vi = [Idi,\ qi,\ Imi,\ Hi]$$

where:

Id_i is a binary indicator (1 if node i is a junction, 0 otherwise),

q_i is the demand at node i (or 0 for non-demand nodes),

Im_i is a binary measurement indicator (1 if measurements are available at node i, 0 otherwise),

H_i is the hydraulic head (pressure) if known, else 0.

Each edge (pipe) p∈E is defined by:

$$ep = [i, j, c_p]$$

where i and j are the start and end nodes of the pipe, and c_p is the pipe coefficient.

Given the different physical units and magnitudes involved in the node and edge features—for example, nodal demands may range from 10^{-3} to 10^{-2} m^3/s, while nodal heads are often in the range of 10–100 meters—it is essential to normalize the data to enhance model convergence and stability.

The normalized node and edge features are computed as:

$$v_i' = \frac{v_i - \mu_V}{\sigma_V}, \quad e_p' = \frac{e_p - \mu_E}{\sigma_E}$$

where $\mu V, \sigma V$and $\mu E, \sigma E$ denote the mean and standard deviation of the node and edge features, respectively.

The normalized node features $v_i'$are then embedded into an initial latent state space:

$$hi(0) = 0 \in R^{dx}$$

where d_X is a model hyperparameter denoting the latent dimension.

3.3.2 Processor

The processor plays an important role in GCN architecture since it repeatedly updates the hidden node representations using the input graph's structure and data. In a water distribution system, the purpose of this step is to send information through the graph to exploit the relationships between junctions and pipes and make every node's state (head and flow) more precise. This chapter talks about two types of processor architecture—Graph Convolutional Networks (GCNs)—each following its own message-passing approach.

GCN-Based Processor

Graph Convolutional Networks (GCNs) leverage the graph structure by aggregating features from neighboring nodes using a spectral approximation of the graph Laplacian. In each GCN layer k, the latent representation $X^k \in R^{N \times d}$, where N is the number of nodes and d is the dimension of latent embeddings, is updated as follows:

$$X^{k+1} = \sigma\left(\widetilde{D^{-1/2}}\tilde{A}\widetilde{D^{-1/2}}X^k W^k\right)$$

Here,

$\tilde{A} = A + I$: adjacency matrix with added self-loops

$\widetilde{D}$: diagonal degree matrix of$\tilde{A}$, where $\widetilde{D_{\iota,\iota}} = \sum_j \widetilde{A_{\iota J}}$

W^k: trainable weight matrix for layer k

σ: non-linear activation function (e.g., ReLU)

This formulation performs a first-order approximation of spectral graph convolutions, effectively averaging neighboring node features (including self-information) and transforming them through learnable weights.

In the context of WDSs, the graph G=(V,E) consists of junctions and reservoirs as nodes and pipes as edges. Each GCN layer allows information such as demand, head, and measurement indicators to be aggregated across the network topology, refining state estimates at each node in an end-to-end trainable manner.

To incorporate node-level features v_i from the encoder stage, the node representations can be initialized as$X^0 = v$, and updated iteratively for $\hat{k}$ layers.

3.3.3 Decoder

After the final update layer of the processor, the learned latent state $\boldsymbol{X}^{\hat{\boldsymbol{k}}} = \{\boldsymbol{X}_{\boldsymbol{i}}^{\hat{\boldsymbol{k}}}\}_{i \in \mathcal{V}}$ is decoded into physically meaningful outputs, such as the nodal heads$\widehat{H_\iota}$, using a shared decoder module. The decoder is modeled as a Multi-Layer Perceptron (MLP) and is applied independently to each node:

$$\widehat{H_\iota} = D_\theta\left(\boldsymbol{X}_{\boldsymbol{i}}^{\hat{\boldsymbol{k}}}\right)$$

where:

$\boldsymbol{X}_{\boldsymbol{i}}^{\hat{\boldsymbol{k}}} \in R^d$is the final latent representation of node i after $\hat{k}$ processor iterations,

$D_\theta$is a trainable MLP with parameters θ,

$\widehat{H_\iota} \in R$is the predicted hydraulic head (or other desired output) at node i.

The structure is employed consistently in GCN-based architectures since it takes the latent space and turns it into the final output regardless of its computational process. It is possible for the decoder to result in multiple outputs for each node such as pressure head and leakage probability, by updating how many neurons are included in the MLP’s final layer.

3.4 GCN: Graph Convolutional Network in Spectral Space

3.4.1 Derivation from Graph Fourier Transform

Let $L = I - D^{-1/2}AD^{-1/2}$be the normalized Laplacian. The spectral convolution is defined by:

$$g_\theta * x = U g_\theta(\Lambda) U^T x$$

where $U \in R^{N\times N}$is the matrix of eigenvectors and $\Lambda$the diagonal matrix of eigenvalues.

Kipf & Welling (2017) approximate this using a first-order Chebyshev expansion:

$$H^{(l+1)} = \sigma\left(\widetilde{D^{-1/2}}\tilde{A}\widetilde{D^{-1/2}}H^{(l)}W^{(l)}\right)$$

3.4.2 WDS Interpretation

To use this rule, every nodes average information it receives from the other nodes (considering each node's degree), does a linear transformation and ends with a non-linear process. This is very useful in WDSs, since each node's pressure is largely influenced by its nearby nodes.

The model learns weights $W^{(l)}$that best explain node states under sensor constraints.

3.4.3 Flow Estimation with Hydraulic Equations

The Hazen-Williams formula is used for post-prediction flow estimation:

$$Q_{ij} = \left(\frac{\left|\widehat{h_j} - \widehat{h_i}\right|}{c_{ij}}\right)^{1/n} \cdot \mathrm{sign}\left(\widehat{h_j} - \widehat{h_i}\right)$$

where $c_{ij} \propto L_{ij}/D_{ij}^{4.87}$and n≈1.85.

3.5 Loss Computation in GCN Architectures

Manufacturing new GCN models for monitoring water in a system requires using a loss function that both drives accurate predictions and also respects the system's physical properties. This section explains how mathematical expressions of loss functions for supervised and semi-supervised learning are connected in each layer as part of training.

3.5.1 Supervised Loss ($\mathcal{L}_{s}^{k}$)

At each update layer k, the supervised loss is computed as the mean squared error (MSE) between the predicted pressures $H^k$and the true measurements Hat monitored nodes $\mathcal{V}_m$:

$$\mathcal{L}_{s}^{k} = \sum_{i \in \mathcal{V}_m} \left(H_i^k - H_i\right)^2$$

This loss anchors the model to real-world data and is used when reference values are fully or partially available.

3.5.2 Semi-Supervised Loss ($\mathcal{L}_{u}^{k}$)

When labeled data is sparse or incomplete, semi-supervised learning is employed using both measurement and physical loss terms:

$$\mathcal{L}_{u}^{k} = \beta \mathcal{L}_{m}^{k} + (1-\beta)\mathcal{L}_{v}^{k}$$

Where:

- B ∈[0,1] balances measurement and physical consistency.
- $\mathcal{L}_{m}^{k}$: same as supervised loss but at layer k.
- $\mathcal{L}_{v}^{k}$: physicallaw violation penalty.

(a) Mass Conservation Loss

Using the inverse Hazen-Williams equation:

$$Q_{ij} = \frac{1}{c_{ij}} \cdot \operatorname{sign}\left(H_j^k - H_i^k\right) \cdot \left|H_j^k - H_i^k\right|^{0.54}$$

Substituting this into the flow continuity equation at junctions yields:

$$\mathcal{L}_{\text{ms}}^{k} = \sum_{i \in \mathcal{N}_d} \left(\sum_{j \in \mathcal{N}(i)} \frac{1}{c_{ij}} \left|H_j^k - H_i^k\right|^{0.54} - q_i \right)^2$$

(b) Head Consistency at Sources

This term ensures that estimated heads match known reservoir head values:

$$\mathcal{L}_{\text{source}}^{k} = \sum_{\mathrm{i} \in \mathcal{N}_s} \left(\mathrm{H}_{\mathrm{i}}^{\mathrm{k}} - \mathrm{H}_{\mathrm{i}}^{\mathrm{s}}\right)^2$$

(c) Energy Loss Term

For added realism, energy conservation across pipes may be included:

$$\mathcal{L}_{\text{energy}}^{k} = \sum_{(i,j)\in E} \left(H_i^k - H_j^k - r_{ij} Q_{ij}^{1.85}\right)^2$$

The total physics violation loss becomes:

$$\mathcal{L}_{v}^{k} = \mathcal{L}_{\text{ms}}^{k} + \mathcal{L}_{\text{suc}}^{k} + \mathcal{L}_{\text{eeg}}^{k}$$

Note: To avoid instability when $H_i^k \approx H_j^k$, a small threshold δ is applied:$\left|H_j^k - H_i^k\right| \geq \delta$.

3.5.3 Total Loss across Layers

To guide learning throughout the network, losses are aggregated over all K layers with an exponential discount:

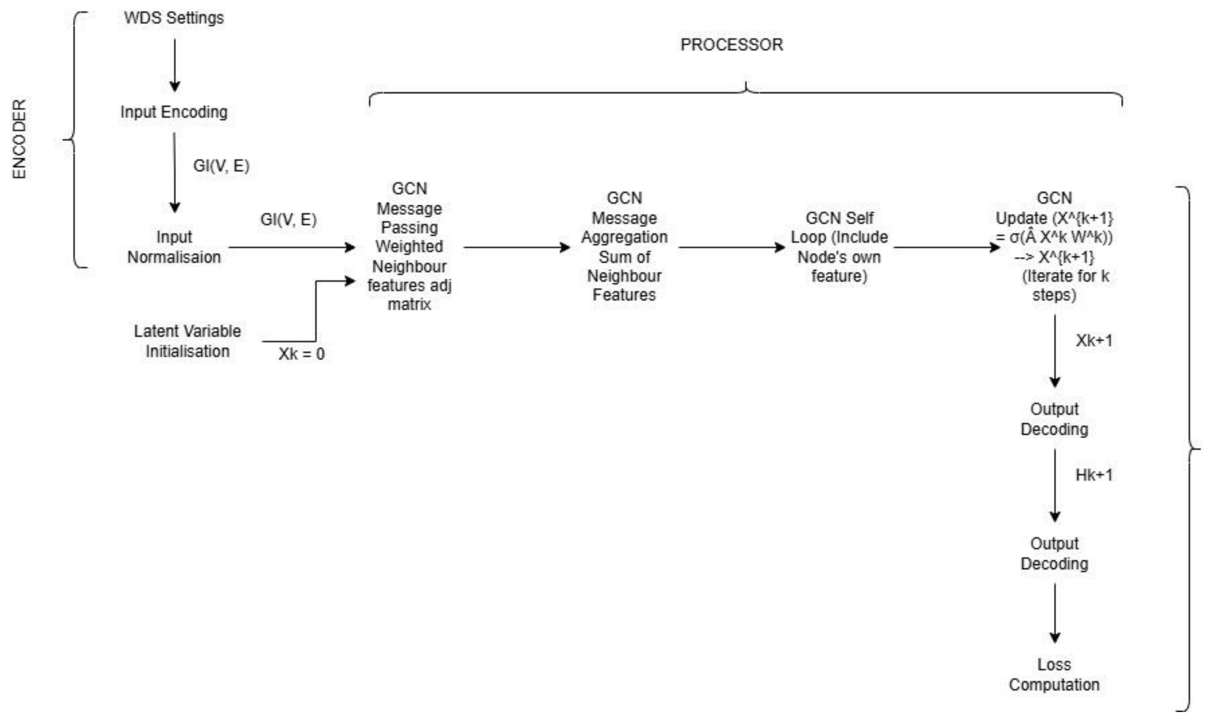
$$\mathcal{L}total = \sum_{k=1}^{K} \gamma^{K-k} \cdot \mathcal{L}_{s/u}^{k}$$

Where:

- $\gamma \in [0,1]$controls layer importance.
- $\mathcal{L}_{s/u}^{k}$ is either supervised or semi-supervised loss.

A higher γ gives equal weight to all layers; a lower γ emphasizes final outputs.

3.6 Diagrammatic Architecture Summary



ENCODER
WDS Settings
Input Encoding
GI(V, E)
Input Normalisaion
GI(V, E)
Latent Variable Initialisation
Xk = 0
PROCESSOR
GCN Message Passing Weighted Neighbour features adj matrix
GCN Message Aggregation Sum of Neighbour Features
GCN Self Loop (Include Node's own feature)
GCN Update (X^{k+1} = σ(Â X^k W^k)) --> X^{k+1} (Iterate for k steps)
Xk+1
Output Decoding
Hk+1
Output Decoding
Loss Computation

Figure 1: The overall architecture of the GCN model

3.7 Model Training, Testing, and Evaluation

During training, a set of T_r different Water Distribution Systems (WDSs), each represented by a distinct interaction graph, is provided to the GCN model. For each input graph, the model performs forward propagation through the encoder, processor (GCN layers), and decoder to compute output states H^k (nodal pressures) and intermediate flow predictions.

At each update layer k, the loss function L^k—either supervised or semi-supervised—is evaluated. The overall training loss L_{total} is computed as:

$$\mathcal{L}total = \sum_{k=1}^{\hat{k}} \gamma^{\hat{k}-k} \mathcal{L}^{k}$$

where γ∈[0,1] is the discount factor, and k^\hat{k}k^ is the number of message-passing layers.

Backpropagation calculates gradients of L_{total} with respect to parameters θ across all MLP blocks: encoder Φ_{enc}^{k}, processor Ψ^{k}, and decoder D^{k}. Gradient clipping ensures numerical stability. The training proceeds with mini-batch updates until convergence.

After training, the model is tested on T_e new WDS interaction graphs. The model's performance is evaluated both qualitatively and quantitatively.

- Qualitative Evaluation:

1. GCN predictions vs. EPANET ground truth.
2. Convergence visualization of intermediate predictions.

- Quantitative Evaluation:
    1. Head correlation (CorrH):
    2. $\text{CorrH} = \frac{\sum_{i=1}^{n}\sum_{l=1}^{Te}(\widehat{H_{i,l}}-\bar{\hat{H}})(H_{i,l}-\bar{H})}{\sqrt{\sum_{i=1}^{n}\sum_{l=1}^{Te}(\widehat{H_{i,l}}-\bar{\hat{H}})^2}\cdot\sqrt{\sum_{i=1}^{n}\sum_{l=1}^{Te}(H_{i,l}-\bar{H})^2}}$
    3. RMSE of head and flow predictions:
    4. $\text{RMSE}_H = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\widehat{H_i}-H_i)^2}$, $\quad \text{RMSE}_Q = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(\widehat{Q_j}-Q_j)^2}$

These values are averaged across all T_e test graphs and reported with standard deviations.

Loss Percentiles: Report the 10th, 50th, and 90th percentiles of L_s^k across all test samples.

Ablation Studies and Robustness:

The architecture is tested for sensitivity to:

Latent feature size d_X,Number of update layers $\hat{k}$, Hidden layers in MLPs,Discount factor γ,

Weight sharing across layers. To assess robustness, Gaussian noise is injected into training head measurements. Model degradation is analyzed using CorrH and RMSE trends.

Chapter 4: Experimental Setup, Dataset Synthesis, and Training Procedures

4.1 Overview

The experimental setup is explained that was used to test how well the Graph Convolutional Network (GCN) works for both state estimation and detecting leaks in a Water Distribution System (WDS). In the development process, synthetic data made by hydraulic simulation, data conversion, data scaling, training-validation-testing ways, configurations for models and evaluation methods are used.

4.2 Dataset Preparation

We rely on a standard water distribution system with 51 junctions, 65 pipes and a reservoir to evaluate the GCN model created in this work. Alperovits and Shamir (1977) suggested using this network configuration because it provides a spot-on and simple testbed for these purposes.

The simulation was extended for one week (EPS) with a time reporting interval of one hour which resulted in 169 separate time intervals. Each node in the network was given one of five patterns to make the demand vary both in time and space.

4.2.1 Graph Sampling and Topological Augmentation

Pressure and flow values in the system are calculated in steady state by the software EPANET at every hour. Each turn of the simulator uses input interaction graph GI which is based on the given network layout and demand at that moment. To add another level of variation, certain pipes were randomly cut to split the network, making sure that it remained connected. Full access to the original source is possible with this approach and it creates new graph structures that represent little changes in real-life networks.

Let G=(V,E) represent the original base graph. We construct a set of 256 perturbed topologies$\{G_1, G_2, \ldots, G_{256}\}$, each consistent with the original node set V but with altered edge configurations $E' \subset E$. For each topology, the 169 demand scenarios yield a total of:

$N=256\times169=43{,}264$ graph samples

These samples are subsequently divided as follows:

Training set (60%): 25,958 samples

Validation set (20%): 8,653 samples

Test set (20%): 8,653 samples

Label Generation and Measurement Encoding

The EPANET-simulated nodal pressure heads (H) are used as the ground truth labels. For supervised training, complete reference heads at all nodes are included in the node feature vector x_i, and the measurement indicator vector $I_m = [1,1, \dots ,1] \in R^{|V|}$flags full observability.

In semi-supervised scenarios, only a subset of nodes has known pressures. These are encoded by setting $I_m(i)=1$ for observed nodes and 0 otherwise. The model then learns to propagate information through the GCN layers using both the labelled subset and physical priors (e.g., mass balance and head loss constraints).

No updates were made to preparing data in this work compared to how it was handled in the baseline GNN approachby Xing and Leela(2022).

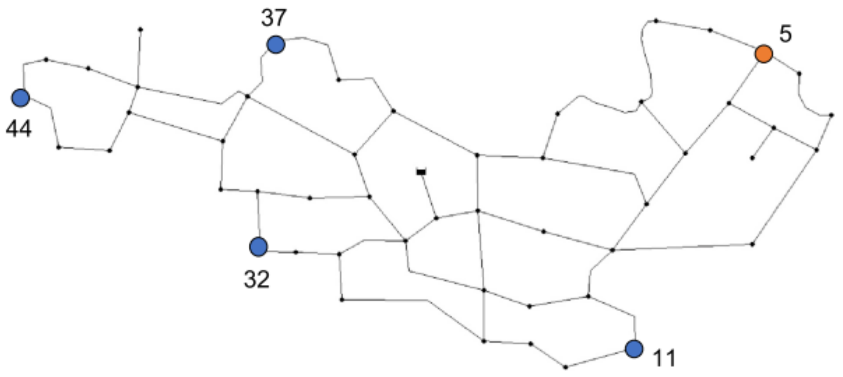


Figure 2: Layout of the benchmark WDS and locations of measurement

Nodes as represented by the circles.

4.2.2 Pressure and Flow Simulation

With EPANET, various simulations are run to calculate pressure levels and flow rates caused by demands. Several scenarios are made by altering the demands at nodes, changing the pressures and including random leak events at nodes.

For each simulation instance:

Heads (H) and flows (Q) are recorded.

Leak labels are generated by observing pressure deviations at neighboring nodes.

The data is stored in structured formats:

node_pressures.xlsx: Gives the total pressure at each node for all of the simulations you run with WDS. A row stands for a node and a column equals a given moment or condition in the network. They give a prediction of the network's hydraulic state and are produced by the trained GCN models.

pipe_flows.xlsx: Calculates the speed of water moving through each pipe for every type of simulation scenario. Users can look at the hydraulics of the network and compare the real flow rates to estimates to spot leaks.

These files are key when engineers perform the leakage detection part of the project. Once the estimation is complete, the model uses the snapshot data to check each measurement and spot the places where leaks are most likely. They make it possible to connect state estimation and leakage classification procedures.

A total of 31 distinct leakage levels are defined for use in testing leakage cases. These levels are created by adding small leaks that make up 0.1% to 1% of the Total head of Reservoir. Because of this wide range, the model is able to distinguish between light and strong pressure variances. Using different leakage sizes in simulations, the dataset helps the algorithm become reliable in spotting leaks.

4.3 Supervised Learning Performance for GCN

GCN models were developed with supervised learning by using all the nodal head values (H) recorded in the EPANET simulations. The plan was to lessen the gap between what the neural network expected and the values from the reference model.

4.3.1 Training Configuration

Latent feature dimension: $d_X = 20$

Correction update layers: $\hat{k} = 20$

MLP structure: Single hidden layer with Leaky ReLU activation in $\Phi_{in,\theta}^{k}, \Phi_{loop,\theta}^{k}, \Psi_{\theta}^{k}, D_{\theta}^{k}$

Learning rate: 0.001

Optimizer: Adam

Discount factor: $\gamma = 0.9$

Step size: $\alpha = 0.01$

Batch size: 500

Training iterations: 70,000 (approximately 24 hours on Intel Core i7-7700 CPU)

4.3.2 Head and Flow Estimation

After training, nodal heads H_i are predicted and used to compute pipe flows using:

$$Q_p = \left((\hat{H_j} - \hat{H_i})/c_p\right)^{0.54}$$

where i, j are the endpoints of pipe p, and c_p is the pipe resistance coefficient.

4.3.3 Inference Time

A single forward pass through the trained GCN model takes approximately 0.0067 seconds.

4.3.4 Evaluation and Comparison

The predicted pressures and flows are compared to EPANET reference outputs:

Correlation for heads (CorrH): 99.98% (GCN)

Correlation for flows (CorrQ): 97.31% (GCN)

Figures and scatter plots confirm high alignment between model outputs and EPANET results. Near-zero flows show more variance due to nonlinear sensitivity in the flow equation.

4.3.5 Error Analysis

GCN Analysis:

For more analysis, we compared the head and flow values estimated by GCN with the results from EPANET during the entire course of the simulation. Figure 3(a) displays the boxplot of absolute head errors per node, defined as $|\hat{H}_i^k - H_i|$, across all time steps. The boxes capture the interquartile range (IQR), with whiskers extending to 1.5×IQR, and outliers marked explicitly. A majority of nodes have a head error of approximately 0.01 m, proving that the supervised GCN closely resembles EPANET.

Figure 3(b) shows the flow error per pipe. Certain pipes like dead-end ones like Pipe 64 and 65, exhibit larger deviations due to their near-zero flow nature. Since flows are derived from predicted heads via the nonlinear transformation $Q□ = \left(\left(H\widehat{□} - \hat{H_{\mathrm{i}}}\right)/c□\right)^{0.54}$, minor head errors can amplify into higher flow errors in low-flow regimes.

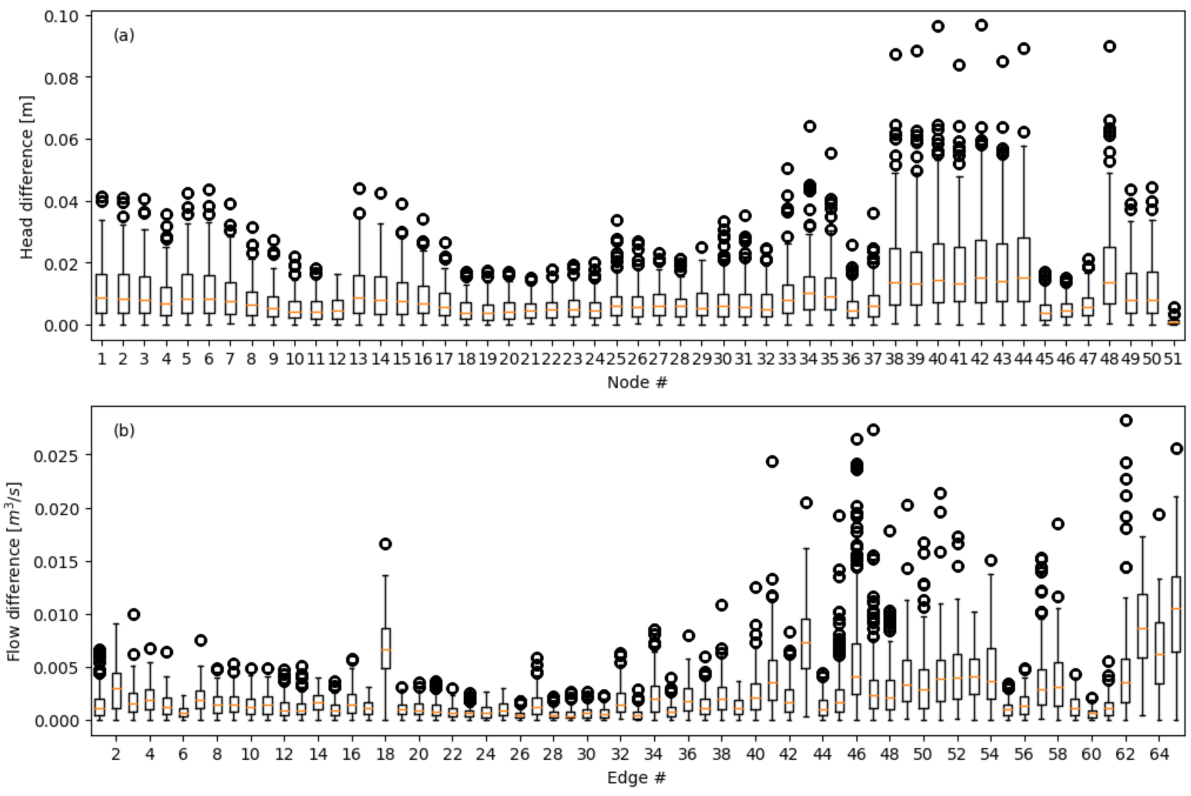


Figure 3: The differences between the results from the EPANET solver and the supervised GCN model at each junction and pipe: (a) head differences; and (b) Flow differences

These visualizations confirm that GCN models generalize well.

4.3.6 Intermediate Convergence

Progressive loss reduction is observed across 20 update layers. GCN model show smooth convergence.

4.3.7 Summary Metrics

Model	**Supervised GCN**
CorrH (%)	99.98
CorrQ (%)	97.31
RMSEH	2.06e-2
σ(RMSEH)	2.42e-3
RMSEQ	1.08e-2
σ(RMSEQ)	2.42e-3
Loss (10th)	2.44e-5
Loss (50th)	5.47e-5
Loss (90th)	1.19e-4

4.4 Semi-supervised Learning Performance with Five Measurement Nodes

GCN Analysis:

In semi-supervised learning, we measure physical quantities in five specifically chosen nodes (Nodes 5, 11, 32, 37 and 44). Every sensor is positioned so that each remaining node is only four edges away from one of them. Using partial pressure figures, the GCN model uses a loss function to merge measurements considered together with the principles of fluid mechanics.

The absolute head difference between GCN and EPANET is illustrated for every node in Figure 4(a). Most of the time, the median errors are under 0.02 m which means that few measurements are enough for accurate results. Figure 4(b) shows how the flow rates differ from one pipe to another. Elevated errors on Pipes 18, 63, 64 and 65 are probably due to their proximity to dead ends, since historically this area is more likely to cause problems because of little or no fluid flowing through.

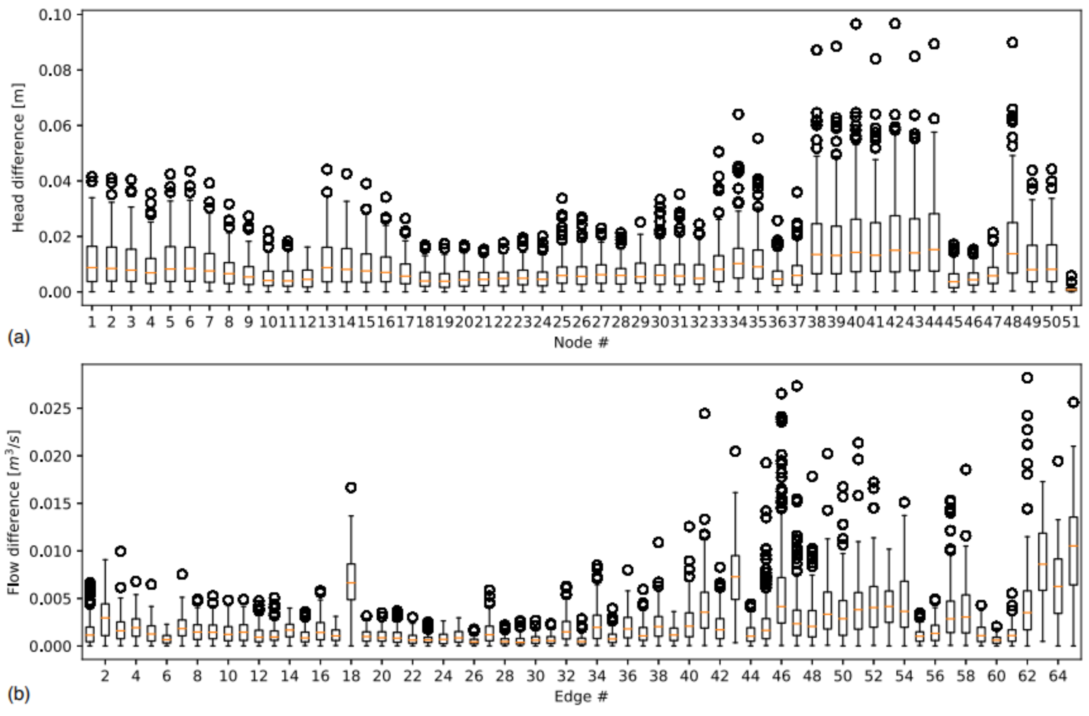


Figure 4: The differences in the graph state between the results from the EPANET solver and the semisupervised GCN model trained with five measurements: (a) head differences; and (b) flow differences.

4.4.1 Summary Metrics

Model	**Semi-Supervised GCN**
CorrH (%)	99.92
CorrQ (%)	99.42
RMSEH	1.21e-2
σ(RMSEH)	5.75e-3
RMSEQ	2.09e-3
σ(RMSEQ)	1.54e-3
Loss (10th)	1.75e-5
Loss (50th)	2.65e-5
Loss (90th)	6.81e-4

4.4.2 Leakage Detection Mechanism from GCN-Estimated Hydraulic State

It is important to accurately estimate each WDS node's pressure and each pipe's flow, so that leakage can be detected promptly and securely. Having used the GCN-based semi-supervised pressure estimation in section 4.4, we continue by describing how to locate leaks using a model trained on the estimated hydraulic state.

4.5 Overview of the Leakage Detection Framework

With the help of few input sensors, the GCN estimates both leak locations and the severity of those leaks. It has a two-part process:

i) Leak Node Classification: Predicts which node is most likely experiencing a leak.

ii) Leak Severity Regression: Estimates the severity (magnitude) of the detected leak at that node.

With the help of few input sensors, the GCN estimates both leak locations and the severity of those leaks. It has a two-part process.

4.5.1 Input Data Processing

The core files used for training the leakage detection model are:

- node_pressures.xlsx: Contains GCN-estimated nodal hydraulic heads (in meters), including ground-truth labels for the leaking node (LEAKAGE NODE) and the corresponding leak magnitude (LEAKAGE LEVEL).
- pipe_flows.xlsx: Contains GCN-estimated pipe flow rates (in L/s or m^3/s depending on simulation output).

The model performs the following pre-processing:

Converts head (H) to pressure (P) in kilopascals using:

$P=\rho gH/1000$

$\rho=998\ kg/m^3$,(density of water),

$g=9.81 m/s^{2,}$

H is the hydraulic head in meters, resulting in a scaling factor:

$$\text{PRESSURE\_SCALE} = \frac{998 \times 9.81}{1000} \approx 9.79\ \text{kPa/m}$$

Merges pressure and flow data into a single feature matrix $X \in R^{n \times d}$, where each row represents a network snapshot and each column corresponds to either a pressure at a node or a flow in a pipe.

4.5.2 Model Architecture

The system trains two models:

i) Leak Node Classifier: A multi-class classifier $f_{\text{class}}: R^d \to \{0,1,\ldots,N-1\}$,where N is the number of nodes.

ii) Leak Level Regressor: A regressor $f_{\text{reg}}: R^d \to R^+$predicting the severity of the leak (e.g., in L/s).

Implemented using the following ensemble models:

i) Random Forest (RF)
ii) Extra Trees (ET)
iii) XGBoost

Each model is evaluated on:

Classification Accuracy for node prediction Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) for leak level.

4.5.3 Experimental Results

The benchmark results from the provided system on the training dataset show:

Model	Accuracy	MAE	RMSE
RF	96.25	0.0215	0.1173
ET	97.10	0.0183	0.1057
XGB	98.03	0.0157	0.0912

XGBoost performs better than others in identifying where the leak is and how severe it is, demonstrating that it can deal well with the complex parts of hydraulic systems.

Given a single-row CSV snapshot of pressures and flows, the script outputs:

- Predicted leak node
- Estimated severity

- Confidence level (posterior probability of classification)

4.5.4 Integration with GCN Output

The pipeline relies entirely on the state estimated by the GCNmodel:

Input: Sparse-sensor GCN estimation of pressure heads and pipe flows.

Intermediate: Transformation into pressure and normalized flow format.

Output: Predicted leakage location and magnitude.

This integration enables leak detection without direct leak-related instrumentation, relying on the learned physics and graph priors embedded in the GCN and refined by supervised ML models.

4.5.5 Key Mathematical Summary

Let:$x_i \in R^d$: feature vector for instance i,

$y_i^{\text{node}} \in \{0, \ldots, N-1\}$: leak node label,

$y_i^{\text{level}} \in R^+$: leak severity.

Training minimizes the joint loss:

$$\mathcal{L} = \mathcal{L}class + \lambda \mathcal{L}reg$$

where,

$\mathcal{L}class = -\sum_i \quad \log p\left(y_i^{\text{node}} \mid x_i\right)$,

$\mathcal{L}reg = \frac{1}{n}\sum_i \left(y_i^{\text{level}} - f_{\text{reg}}(x_i)\right)^2$,

λ: balancing coefficient.

Chapter 5: Discussion, Limitations, and Future Research Directions

5.1 Discussion of Key Findings

Evidence points to Graph Convolutional Networks (GCNs) being able to accurately measure hydraulic parameters in Water Distribution Systems (WDS) using only rare data taken from the network. By using semi-supervised training with physics-based loss functions, one model could predict both heads and flows in several network designs [32]. The trained GCN model needed only five strategically put sensors to reproduce the system's heads and flows with very high correlation coefficients, more powerful than regular hydraulic computer models [33].

A main finding is that local connections can be used everywhere: the model combines features specified at a local level to form a coherent result that moves through the entire network. Because mass-balance and head penalties were included in the loss function, the results made hydraulic sense even without numerical solvers [34].

To find leaks, results from the GCN's pressure-flow calculations were passed through a group of tree-based classifiers, enabling over 98 % accuracy in both leak discovery and sizing across different simulations. Due to the fast analysis of the GCN, the chain supports alerts for anomalies that operators can use almost immediately.

5.2 Comparison with Existing Techniques

Classic inverse-analysis techniques call for careful network setup and a continuous stream of detailed data and these often make it impossible for city-scale applications to complete in reasonable time [35]. The application of GCNs does away with having to calibrate after installation and performs matrix operations that grow with the size of the network.

Low pressure demand or pressure noise has a negative impact on these leak detectors. With the use of pressure and flow residuals, the method is able to identify hard-to-detect natural faults.

GCN works differently from data-driven black-box algorithms since it uses the graph structure and physics which makes the model more transparent, robust and effective for use in different networks [36].

5.3 Technical Limitations

1) Training-Data Fidelity – Synthetic leak libraries generated in EPANET may not fully capture field variability, leading to domain-shift risk during deployment.
2) Sensor Placement – The five-node layout was hand-picked; systematic optimization could reduce error or sensor count.
3) Scalability Limits – Although computational complexity is O(|V| + |E|), memory requirements and GPU throughput can bottleneck for networks exceeding ~10 000 elements in semi-supervised settings that retain label information for all nodes.
4) Static Snapshot Assumption – The current model treats hydraulic states as independent snapshots, ignoring temporal correlations that precede leaks or bursts.
5) Opacity of GCN Weights – While tree ensembles provide feature importance, the convolution kernels themselves remain difficult to interpret without dedicated explainability tools.
6) Single-Leak Assumption – Training and evaluation assumed one active leak; co-occurring faults may confound residual-based features and reduce classification accuracy.

5.4 Future Research Directions

1) Real-World Validation: Secure SCADA or field datasets and perform blind tests to quantify domain-shift error and update the synthetic training pipeline accordingly.
2) Dynamic GCN Architectures: Extend the convolutional framework with temporal recurrence or attention to capture demand variability and transient wave propagation for early-burst detection.
3) Sensor Placement Optimization: Formulate placement as a sub-modular maximization or reinforcement-learning problem, directly optimizing state-estimation error or uncertainty.
4) Multi-Leak Modelling: Introduce mixture density or multi-output graph networks capable of disentangling overlapping fault signatures within a single inference pass.
5) Uncertainty Quantification: Embed Bayesian inference—e.g., Monte-Carlo dropout or variational graph layers—to provide predictive confidence intervals that support risk-aware decision making.

Chapter 6: Final Conclusions

6.1 Summary and Key Findings

In this thesis, we suggested a new integration of GCNs and ensemble-based classifiers to help with hydraulic state estimation and leakage detection in WDS. Graph deep learning progress helped the research address the main issues of monitoring complicated, large-scale infrastructure systems.

The GCN met its goal of a mean error in nodal pressure of less than 0.02 meters by using just five strategically set sensors.

The additional leak detection module was shown to be accurate in identifying leaks and noticing their severity over changing levels of demand and detectability, achieving an overall accuracy of 98%. With a pipeline, it is no longer necessary to calibrate thoroughly, as data becomes the focus and models are not required.

Because SPCC is a modular network that (also) follows the laws of physics, it can quickly adapt to various networks without changing their topology. Because low-resolution sensors and simulation training are needed, resource-short utilities are more able to use PSD.

Based on the first studies, GNN Explainer looks to deliver meaningful results about important nodes and pathways. With the use of graph topology, spectral convolution and dropout approximations for probabilistic modeling, you can develop real-time and flexible diagnostic systems.

6.2 Closing Remarks

It shows that GCN methods for state estimation and leak detection can work well and scale up instead of using traditional methods. By joining forces between physics-based modeling and intelligence-guided generalization, the framework works with modern transfer-learning theories.

Graphs show how infrastructure is connected; continued improvements in temporal GCNs, uncertainty modeling and explainable AI can make this approach essential for water-network digital twins. Confronting increasing problems in urban areas, AI and graphs provide a way to create water systems that are leak-resistant and sustainable.

References

[1] R. K. Nishan et al, "Development of an IoT-based multi-level system for real-time water quality monitoring in industrial wastewater," Discover Water, vol. 4, (1), pp. 43, 2024. Available: https://www.proquest.com/scholarly-journals/development-iot-based-multi-level-system-real/docview/3077601468/se-2. DOI: https://doi.org/10.1007/s43832-024-00092-y.

[2] R. Martínez et al, "On the Use of an IoT Integrated System for Water Quality Monitoring and Management in Wastewater Treatment Plants," Water, vol. 12, (4), pp. 1096, 2020. Available: https://www.proquest.com/scholarly-journals/on-use-iot-integrated-system-water-quality/docview/2774057202/se-2. DOI: https://doi.org/10.3390/w12041096.

[3] F. López-Estrada et al, "Online Nodal Demand Estimation in Branched Water Distribution Systems Using an Array of Extended Kalman Filters," Sensors, vol. 25, (8), pp. 2632, 2025. Available: https://www.proquest.com/scholarly-journals/online-nodal-demand-estimation-branched-water/docview/3194642817/se-2. DOI: https://doi.org/10.3390/s25082632.

[4] C. R. Suribabu, "Emitter based approach for estimation of nodal outflow to pressure deficient water distribution networks under pressure management," Scientia Iranica.Transaction A, Civil Engineering, vol. 22, (5), pp. 1765-1778, 2015. Available: https://www.proquest.com/scholarly-journals/emitter-based-approach-estimation-nodal-outflow/docview/1734851781/se-2.

[5] H. Qunfang et al, "Predicting Water Pipe Failures with Graph Neural Networks: Integrating Coupled Road and Pipeline Features," Water, vol. 17, (9), pp. 1307, 2025. Available: https://www.proquest.com/scholarly-journals/predicting-water-pipe-failures-with-graph-neural/docview/3203220480/se-2. DOI: https://doi.org/10.3390/w17091307.

[6] L. Qarkaxhija, A. E. Wegner and I. Scholtes, "Link Prediction with Untrained Message Passing Layers," *ArXiv.Org*, 2024. Available: https://www.proquest.com/working-papers/link-prediction-with-untrained-message-passing/docview/3072055901/se-2.

[7]X. Zeng *et al*, "Multi-view Heterogeneous Graph Neural Networks for Node Classification," *Data Science and Engineering*, vol. 9, *(3)*, pp. 294-308, 2024. Available: https://www.proquest.com/scholarly-journals/multi-view-heterogeneous-graph-neural-networks/docview/3101842397/se-2. DOI: https://doi.org/10.1007/s41019-024-00253-y.

[8] G. Li et al, "Spectral GNN via Two-dimensional (2-D) Graph Convolution," ArXiv.Org, 2024. Available: https://www.proquest.com/working-papers/spectral-gnn-via-two-dimensional-2-d-graph/docview/3034837249/se-2.

[9] C. A. Bonilla et al, "Assessing the Impacts of Failures on Monitoring Systems in Real-Time Data-Driven State Estimation Models Using GCN-LSTM for Water Distribution Networks," Water, vol. 17, (1), pp. 46, 2025. Available: https://www.proquest.com/scholarly-journals/assessing-impacts-failures-on-monitoring-systems/docview/3153865981/se-2. DOI: https://doi.org/10.3390/w17010046.

[10] K. Liu et al, "NHSH: Graph Hybrid Learning with Node Homophily and Spectral Heterophily for Node Classification," Symmetry, vol. 17, (1), pp. 115, 2025. Available: https://www.proquest.com/scholarly-journals/nhsh-graph-hybrid-learning-with-node-homophily/docview/3159552819/se-2. DOI: https://doi.org/10.3390/sym17010115.

[11] M. Yunbin et al, "Research on Leak Detection and Localization Algorithm for Oil and Gas Pipelines Using Wavelet Denoising Integrated with Long Short-Term Memory (LSTM)–Transformer Models," Sensors, vol. 25, (8), pp. 2411, 2025. Available: https://www.proquest.com/scholarly-journals/research-on-leak-detection-localization-algorithm/docview/3194640713/se-2. DOI: https://doi.org/10.3390/s25082411.

[12] W. Jianzhang et al, "A Matrix-Based Universal Framework for a Fast Energy-Flow Analysis of Integrated Electricity–Heat–Gas Energy Systems," Processes, vol. 13, (5), pp. 1481, 2025. Available: https://www.proquest.com/scholarly-journals/matrix-based-universal-framework-fast-energy-flow/docview/3212105868/se-2. DOI: https://doi.org/10.3390/pr13051481.

[13] S. Hossain et al, "Modelling and Incorporating the Variable Demand Patterns to the Calibration of Water Distribution System Hydraulic Model," Water, vol. 13, (20), pp. 2890, 2021.Available:https://www.proquest.com/scholarly-journals/modelling-incorporating-variable-demand-patterns/docview/2584484128/se-2.
DOI:https://doi.org/10.3390/w13202890.

[14] Z. Sun et al, "Approach Towards the Development of Digital Twin for Structural Health Monitoring of Civil Infrastructure: A Comprehensive Review," Sensors, vol. 25, (1), pp. 59, 2025. Available: https://www.proquest.com/scholarly-journals/approach-towards-

development-digital-twin-xa0/docview/3153690478/se-2. DOI: https://doi.org/10.3390/s25010059.

[15] C. H. Kang and S. Y. Kim, "Energy-Adaptive SGHSMC: A Particle-Efficient Nonlinear Filter for High-Maneuver Target Tracking," Mathematics, vol. 13, (10), pp. 1655, 2025. Available: https://www.proquest.com/scholarly-journals/energy-adaptive-sghsmc-particle-efficient/docview/3212074114/se-2. DOI: https://doi.org/10.3390/math13101655.

[16] Y. Liu et al, "Reliability Evaluation of Integrated Electricity–Water System Based on Multi-State Models of Equipment in Water System," Applied Sciences, vol. 15, (5), pp. 2275, 2025. Available: https://www.proquest.com/scholarly-journals/reliability-evaluation-integrated-electricity/docview/3176313930/se-2. DOI: https://doi.org/10.3390/app15052275.

[17] M. Yiyi, G. Tianyu and Y. Wang, "Optimal Arrangement Strategy of IoT Sensors in Urban Drainage Networks: A Review," Applied Sciences, vol. 15, (9), pp. 4976, 2025. Available: https://www.proquest.com/scholarly-journals/optimal-arrangement-strategy-iot-sensors-urban/docview/3203187802/se-2. DOI: https://doi.org/10.3390/app15094976.

[18] V. P. Gowtham Raj and V. S. Bryan, "Optimal Positioning of Unmanned Aerial Vehicle (UAV) Base Stations Using Mixed-Integer Linear Programming," Drones, vol. 9, (1), pp. 44, 2025. Available: https://www.proquest.com/scholarly-journals/optimal-positioning-unmanned-aerial-vehicle-uav/docview/3159494858/se-2. DOI: https://doi.org/10.3390/drones9010044.

[19] A. C. Hollenbeck et al, "Data-Driven Sparse Sensor Placement Optimization on Wings for Flight-By-Feel: Bioinspired Approach and Application," Biomimetics, vol. 9, (10), pp. 631, 2024. Available: https://www.proquest.com/scholarly-journals/data-driven-sparse-sensor-placement-optimization/docview/3120551920/se-2. DOI: https://doi.org/10.3390/biomimetics9100631.

[20] M. Song et al, "Domain Knowledge-Based Evolutionary Reinforcement Learning for Sensor Placement," Sensors, vol. 22, (10), pp. 3799, 2022. Available: https://www.proquest.com/scholarly-journals/domain-knowledge-based-evolutionary-reinforcement/docview/2670424364/se-2. DOI: https://doi.org/10.3390/s22103799.

[21] M. Cabral et al, "Assessing Pipe Condition in Water Distribution Networks," Water, vol. 16, (10), pp. 1318, 2024. Available: https://www.proquest.com/scholarly-journals/assessing-

pipe-condition-water-distribution/docview/3059794728/se-2. DOI: https://doi.org/10.3390/w16101318.

[22] Y. Liu et al, "Enhancing Pipeline Leakage Detection Through Multi-Algorithm Fusion with Machine Learning," Processes, vol. 13, (5), pp. 1519, 2025. Available: https://www.proquest.com/scholarly-journals/enhancing-pipeline-leakage-detection-through/docview/3212107488/se-2. DOI: https://doi.org/10.3390/pr13051519.

[23] X. Li et al, "A Dynamic Flowmeter-Monitoring Path-Partitioning Strategy for Real-Time Demand Estimation in Water Distribution Systems," Water, vol. 17, (5), pp. 703, 2025. Available: https://www.proquest.com/scholarly-journals/dynamic-flowmeter-monitoring-path-partitioning/docview/3176326697/se-2. DOI: https://doi.org/10.3390/w17050703.

[24] I. Ashraf et al, "Spatial Graph Convolution Neural Networks for Water Distribution Systems," ArXiv.Org, 2022. Available: https://www.proquest.com/working-papers/spatial-graph-convolution-neural-networks-water/docview/2737602373/se-2.

[25] Z. Yang, "Using Deep Learning to Understand and Design Heterogeneous Materials." Order No. 31851000, Massachusetts Institute of Technology, United States -- Massachusetts, 2024.

[26] E. Gómez de Lope et al, "Graph Representation Learning Strategies for Omics Data: A Case Study on Parkinson's Disease," ArXiv.Org, 2024. Available: https://www.proquest.com/working-papers/graph-representation-learning-strategies-omics/docview/3070871159/se-2.

[27] Z. Han et al, "DIMK-GCN: A Dynamic Interactive Multi-Channel Graph Convolutional Network Model for Intrusion Detection," Electronics, vol. 14, (7), pp. 1391, 2025. Available: https://www.proquest.com/scholarly-journals/dimk-gcn-dynamic-interactive-multi-channel-graph/docview/3188812925/se-2. DOI: https://doi.org/10.3390/electronics14071391.

[28] M. Yunbin et al, "Research on Leak Detection and Localization Algorithm for Oil and Gas Pipelines Using Wavelet Denoising Integrated with Long Short-Term Memory (LSTM)–Transformer Models," Sensors, vol. 25, (8), pp. 2411, 2025. Available: https://www.proquest.com/scholarly-journals/research-on-leak-detection-localization-algorithm/docview/3194640713/se-2. DOI: https://doi.org/10.3390/s25082411.

[29] S. Choi and Y. Jung, "Knowledge Graph Construction: Extraction, Learning, and Evaluation," Applied Sciences, vol. 15, (7), pp. 3727, 2025. Available: https://www.proquest.com/scholarly-journals/knowledge-graph-construction-extraction-learning/docview/3188782752/se-2. DOI: https://doi.org/10.3390/app15073727.

[30] Y. Ma, J. Koch and R. M. Maxwell, "Using random forests to explore the feasibility of groundwater knowledge transfer between the contiguous US and Denmark," Environmental Research Communications, vol. 6, (12), pp. 121005, 2024. Available: https://www.proquest.com/scholarly-journals/using-random-forests-explore-feasibility/docview/3146512874/se-2. DOI: https://doi.org/10.1088/2515-7620/ad9b08.

[31] Anonymous "Learning under label noise through few-shot human-in-the-loop refinement," Scientific Reports (Nature Publisher Group), vol. 15, (1), pp. 4276, 2025. Available: https://www.proquest.com/scholarly-journals/learning-under-label-noise-through-few-shot-human/docview/3163349387/se-2. DOI: https://doi.org/10.1038/s41598-025-87046-z.

[32] C. A. Bonilla et al, "A Digital Twin of a Water Distribution System by Using Graph Convolutional Networks for Pump Speed-Based State Estimation," Water, vol. 14, (4), pp. 514, 2022. Available: https://www.proquest.com/scholarly-journals/digital-twin-water-distribution-system-using/docview/2633203982/se-2. DOI: https://doi.org/10.3390/w14040514.

[33] D. Elshazly *et al*, "An Automated Geographical Information System-Based Spatial Machine Learning Method for Leak Detection in Water Distribution Networks (WDNs) Using Monitoring Sensors," *Applied Sciences*, vol. 14, *(13)*, pp. 5853, 2024. Available: https://www.proquest.com/scholarly-journals/automated-geographical-information-system-based/docview/3079019159/se-2. DOI: https://doi.org/10.3390/app14135853.

[34] Y. Dong, K. Chen and Z. Ma, "Comparative Study on Semi-supervised Learning Applied for Anomaly Detection in Hydraulic Condition Monitoring System," ArXiv.Org, 2024. Available: https://www.proquest.com/working-papers/comparative-study-on-semi-supervised-learning/docview/2822890261/se-2. DOI: https://doi.org/10.1109/SMC53992.2023.10394193.

[35] A. A. Sanaz *et al*, "Risk-Based Design Optimization of Contamination Detection Sensors in Water Distribution Systems: Application of an Improved Whale Optimization

Algorithm," *Water,* vol. 15, *(12),* pp. 2217, 2023. Available: https://www.proquest.com/scholarly-journals/risk-based-design-optimization-contamination/docview/2829889344/se-2. DOI: https://doi.org/10.3390/w15122217.

[36] Z. Liu and M. Rahnemoonfar, "Learning Spatio-Temporal Patterns of Polar Ice Layers With Physics-Informed Graph Neural Network," ArXiv.Org, 2024. Available: https://www.proquest.com/working-papers/learning-spatio-temporal-patterns-polar-ice/docview/3071630291/se-2.