Detecção de vazamentos em redes hidráulicas utilizando Redes Neurais de Grafos ⋆

Rodrigo P. Rolle * **Lucas N. Monteiro** * **Lucas R. Tomazini** *
Eduardo P. Godoy *

* *Universidade Estadual Paulista (UNESP), Sorocaba, SP. (e-mails: rodrigo.rolle, lucas.nunes, lucas.tomazini, eduardo.godoy@unesp.br).*

Abstract: This work approaches the problem of distributed monitoring of water distribution networks focusing on leakage detection, using computational simulation of water distribution networks and machine learning techniques. The implemented machine learning technique is based on Graph Neural Networks, which are structures with the capacity to take into account the spatial displacement of the measurement points in the network alongside the measurement data, thus providing insight regarding the leak location. A case study application was developed to evaluate the capacity of the algorithm. A hypothetical water distribution network was implemented in a software environment to enable data collection in diverse operation scenarios, especially leakages in different locations. The initial results demonstrate that the graph-based approach is a viable methodology for water leakage detection.

Resumo: Este trabalho aborda o problema do monitoramento distribuído das redes de distribuição de água com foco na detecção de vazamentos, utilizando simulação computacional de redes de distribuição de água e técnicas de aprendizado de máquina. A técnica de aprendizado de máquina implementada é baseada em Redes Neurais de Grafo, estruturas que consideram não apenas as variáveis medidas pelos sensores, mas também a disposição espacial dos pontos da rede e assim fornecendo indicativos da localização dos possíveis vazamentos. Um estudo de caso foi desenvolvido para a avaliação da capacidade do algoritmo. Um modelo hipotético de rede de distribuição de água foi estabelecido em ambiente de simulação computacional, de forma a viabilizar a coleta de dados em diversos cenários de operação, especialmente situações de vazamentos em diferentes localidades. Os resultados iniciais mostram que a abordagem baseada em grafos é uma metodologia viável de detecção de vazamentos.

Keywords: Water leak detection; water leak location; water distribution networks; deep learning; graph neural networks.

Palavras-chaves: Detecção de vazamentos de água; localização de vazamentos de água; redes de distribuição de água; aprendizado profundo; redes neurais de grafo.

1. INTRODUÇÃO

A gestão sustentável da água é um tópico que tem despertado a atenção de pesquisadores e empresas. O problema global de escassez de água é intensificado pelas perdas que ocorrem nas redes de distribuição e produz consequências ambientais e econômicas, particularmente nos países em desenvolvimento (Nkemeni et al., 2020). No contexto latino-americano, o Relatório Mundial sobre o Desenvolvimento de Recursos Hídricos da ONU (Connor, 2015) aponta que um passo prioritário é estabelecer e consolidar a governança hídrica. Dada a relativa abundância de água na região, a crise hídrica é de caráter institucional, não envolvendo necessariamente a falta de água para o uso. A falta de legislação rígida, o financiamento insuficiente e as limitações tecnológicas na rede de distribuição são apontadas como elos frágeis que precisam de atenção.

No Brasil, os Relatórios Anuais disponibilizados pelo Sistema Nacional de Informações sobre Saneamento (SNIS) desde 1995 retratam a evolução do setor hídrico. Segundo o Relatório Anual de 2021 (Brasil, 2021), a média nacional de perda de água tratada nos sistemas de distribuição é de 39,2%. Este cálculo engloba vazamentos, falhas nos sistemas de medição e ligações clandestinas.

A seleção de uma abordagem para detecção de vazamentos deve também levar em conta os aspectos operacionais das Redes de Distribuição de Água (RDAs): ao longo do tempo, as tubulações sofrem desgaste, rupturas, fissuras, alterações de diâmetro e fator de resistência devido à corrosão, falhas de solda nas juntas, entre outras ocorrências (Rojek and Studzinski, 2019). A demanda de água pelos consumidores é difícil de se prever e está sujeita a variações sazonais, especialmente no verão e épocas festivas. O desenho das redes é modificado à medida que as cidades se expandem. A combinação destes fatores faz com que as RDAs sejam ambientes heterogêneos e dinâmicos (Chan et al., 2018).

⋆ Este estudo foi financiado em parte pelo CNPq - contratos 142383/2019-8 e 303967/2021-8.

Os avanços da informática permitiram a implementação de técnicas de inteligência computacional, que são denominadas de forma genérica como Aprendizado de Máquina (ou *Machine Learning* - ML). Técnicas de ML tradicionais como as Redes Neurais Artificiais (RNA), Clusterização e Máquinas de Vetores de Suporte possuem limitações quanto à capacidade de extração de características e dependem de ajustes finos de diversos parâmetros, o que aumenta a dificuldade do projeto e prejudica a reusabilidade.

Grafos são estruturas que permitem a representação de relações entre componentes de sinais diversos. Por exemplo, num sistema de comércio virtual, os grafos permitem que se explorem as interações entre usuários para fornecer recomendações. Nas aplicações tradicionais de ML, normalmente são utilizadas representações mais simples (vetores e matrizes), onde perde-se informação relevante, como a dependência topológica da informação de cada nó (Scarselli et al., 2009). Por isso foram desenvolvidas técnicas de ML que trabalham com a informação na forma de grafos, chamadas Redes Neurais de Grafo (*Graph Neural Networks - GNN*) (Ruiz et al., 2021).

O presente trabalho tem como objetivo aplicar as Redes Neurais de Grafo ao problema de detecção de vazamentos em RDAs. Para isso, são elaborados modelos computacionais de RDAs para geração de dados e realização de testes. As contribuições deste trabalho são (1) implementar técnicas recentes de ML à gestão de redes hidráulicas; (2) realizar detecção de vazamentos com menor quantidade de dados e (3) estimar a localização de vazamentos com menor quantidade de sensores, explorando as relações e dependências contidas nos grafos.

2. REDES NEURAIS DE GRAFO

Redes Neurais de Grafo (ou *Graph Neural Networks* - GNN) são uma técnica de ML que possibilita o processamento de informação na forma de grafos. Assim, as informações de topologia e dependência expressas no grafo passam a fazer parte do processamento dos dados. GNNs se baseiam em um mecanismo de difusão da informação, no qual os nós atualizam seus estados e trocam informação até atingir um equilíbrio estável (Scarselli et al., 2009). O restante do conteúdo desta seção é baseado no estudo realizado por Ruiz et al. (2021).

2.1 Modelagem Matemática

Um grafo é uma dupla $G = (V, E)$ composta por nós (ou vértices) $V = \{1, ..., n\}$, arestas E definidas como pares ordenados de nós (i, j) e pesos $w_{i,j}$ associados às arestas. Um grafo é dito simples quando não há *loops* (arestas que ligam um nó a ele mesmo) e há apenas um vértice entre cada par de nós (i, j), com $i \neq j$. A relação de adjacência entre os nós de G pode ser representada por uma matriz A não-negativa de dimensão $n \times n$. A posição $a_{i,j}$ da matriz de adjacência A é 1 se i e j são adjacentes (ou seja, existe uma aresta que interliga os nós i e j) e 0 em caso contrário. Ainda, $a_{i,j} = a_{j,i}$ quando o grafo G é não-direcionado (os vértices não possuem sentido definido) e as posições da diagonal principal $a_{i,i}$ são 0 se G é um grafo simples (Ruiz et al., 2021).

O objetivo das GNNs é implementar algoritmos de ML sobre os grafos. De forma geral, tem-se pares $(\mathbf{x}, \mathbf{y})$ compostos por um grafo de sinal de entrada $\mathbf{x} \in \mathbb{R}^n$ e um grafo de sinal de saída esperada $\mathbf{y} \in \mathbb{R}^n$. A terminologia "grafo de sinal" dada para $\mathbf{x}$ e $\mathbf{y}$ significa que os componentes x_i e y_i estão associados ao i-ésimo nó do grafo.

O par $(\mathbf{x}, \mathbf{y})$ é conjuntamente retirado de uma distribuição de probabilidade $p(\mathbf{x}, \mathbf{y})$ e o objetivo é encontrar uma função $\Phi : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ tal que $\Phi(\mathbf{x})$ seja uma aproximação de $\mathbf{y}$ sobre a distribuição de probabilidade $p(\mathbf{x}, \mathbf{y})$. Para isso, introduz-se a função de perda (*loss*) não-negativa $\mathcal{L}(\Phi(\mathbf{x}), \mathbf{y}) \geq 0$ tal que $\mathcal{L}(\Phi(\mathbf{x}), \mathbf{y}) = 0$ quando $\Phi(\mathbf{x}) = \mathbf{y}$, com o objetivo de mensurar a similaridade entre a saída da rede $\Phi(\mathbf{x})$ e a saída desejada $\mathbf{y}$. Assim, a função $\Phi^\dagger$ que melhor aproxima a saída desejada $\mathbf{y}$ pode ser definida como a função que minimiza a perda $\mathcal{L}(\Phi(\mathbf{x}), \mathbf{y})$ média sobre a distribuição de probabilidade $p(\mathbf{x}, \mathbf{y})$:

$$\Phi^\dagger = \underset{\Phi}{\operatorname{argmin}}\, \mathbb{E}[\mathcal{L}(\Phi(\mathbf{x}), \mathbf{y})] = \underset{\Phi}{\operatorname{argmin}} \int \mathcal{L}(\Phi(\mathbf{x}), \mathbf{y}) dp(\mathbf{x}, \mathbf{y}) \tag{1}$$

onde a esperança $\mathbb{E}[\mathcal{L}(\Phi(\mathbf{x}), \mathbf{y})]$ é definida como uma perda estatística. Logo, a (1) expressa um problema de minimização de perda estatística.

Uma condição importante para resolver (1) é a disponibilidade da distribuição de probabilidade $p(\mathbf{x}, \mathbf{y})$. Se esta é conhecida, a solução para (1) é computar uma distribuição posterior que dependa da forma da função de perda $\mathcal{L}(\Phi(\mathbf{x}), \mathbf{y})$. Entretanto, em problemas de ML, esta distribuição de probabilidade não é conhecida.

Normalmente, estão disponíveis coleções de Q amostras de informação $(\mathbf{x}_q, \mathbf{y}_q)$ retiradas da distribuição $p(\mathbf{x}, \mathbf{y})$ que são alocadas no conjunto de dados de treinamento $\mathcal{T} := (\mathbf{x}_q, \mathbf{y}_q)_{q=1}^Q$. Considerando que estas amostras são adquiridas de forma independente e o número de amostras de Q é grande, uma boa aproximação para a perda estatística em (1) é a média empírica $\bar{\mathcal{L}}(\Phi) := (1/Q) \sum_{q=1}^Q \mathcal{L}(\Phi(\mathbf{x}_q), \mathbf{y}_q)$. Portanto, para problemas de ML, deve-se buscar uma função Φ^* que minimize a média empírica $\bar{\mathcal{L}}(\Phi)$:

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}} \frac{1}{Q} \sum_{q=1}^{Q} \mathcal{L}(\Phi(\mathbf{x}_q), \mathbf{y}_q) \tag{2}$$

que é um problema de minimização de risco empírica (ERM - *Empirical Risk Minimization*). A função Φ^* é a função empírica ótima associada ao conjunto de dados de treinamento $\mathcal{T}$.

Nota-se que a solução para (2) é elementar: uma vez que $\mathcal{L}(\Phi(\mathbf{x}), \mathbf{y}) = 0$ quando $\Phi(\mathbf{x}) = \mathbf{y}$ e não-negativa em caso contrário, basta que $\Phi(\mathbf{x}_q) = \mathbf{y}_q$ para todas as amostras $\mathbf{x}_q$ observadas (ou alguma espécie de média, caso a mesma entrada $\mathbf{x}_q$ seja observada diversas vezes). Entretanto, (2) só faz sentido quando se tem acesso a todas as possíveis amostras $\mathbf{x}_q$, o que não ocorre na prática. De fato, nas aplicações de ML busca-se inferir (ou aprender) os valores de $\mathbf{y}$ para amostras $\mathbf{x}$ que nunca foram observadas.

Desta forma, convém estabelecer um parâmetro de aprendizado $\mathcal{H}$ que restringe a família de funções Φ que são

admissíveis em (2). Assim, em vez de realizar a busca sobre $\Phi(\mathbf{x})$, a busca é realizada sobre funções $\Phi(\mathbf{x};\mathcal{H})$ de forma que o problema de ERM expresso em (2) pode ser substituído pela formulação:

$$\mathcal{H}^* = \underset{\mathcal{H}}{\operatorname{argmin}} \frac{1}{Q}\sum_{q=1}^{Q}\mathcal{L}(\Phi(\mathbf{x}_q;\mathcal{H}),\mathbf{y}_q) \tag{3}$$

Uma escolha de parâmetros específica é o conjunto de funções lineares da forma $\Phi(\mathbf{x};\mathbf{H}) = \mathbf{H}\mathbf{x}$, tal que (2) pode ser reescrita como:

$$\mathbf{H}^* = \underset{\mathbf{H}}{\operatorname{argmin}} \frac{1}{Q}\sum_{q=1}^{Q}\mathcal{L}(\mathbf{H}\mathbf{x}_q),\mathbf{y}_q) \tag{4}$$

ou ainda, a função $\Phi(\mathbf{x};\mathcal{H})$ pode ser uma RNA ou uma GNN.

É importante ter em vista que o projeto de um sistema de ML é equivalente à seleção de parâmetros de aprendizado apropriados, como mostrado em (3). Nela, a única escolha do projetista é a classe de funções $\Phi(\mathbf{x};\mathcal{H})$, caracterizada pelas diferentes escolhas de $\mathcal{H}$. Ainda, esta escolha de parametrização determina como a função $\Phi(\mathbf{x};\mathcal{H})$ generaliza a partir de amostras do conjunto de treinamento $(\mathbf{x}_q,\mathbf{y}_q) \in \mathcal{T}$ para sinais não-observados $\mathbf{x}$.

Um método conveniente para o processamento de grafos de sinais é o filtro convolucional de grafo. Para definir esta operação, tem-se que $\mathbf{S} \in \mathbb{R}^{n\times n}$ denota uma representação matricial do grafo. O filtro de ordem K possui coeficientes h_k que são agrupados no vetor $\mathbf{h} = [h_0;...;h_K]$. O filtro convolucional de grafo aplicado ao grafo de sinal $\mathbf{x}$ é uma expressão polinomial na representação matricial:

$$\mathbf{u} = \sum_{k=0}^{K} h_k\mathbf{S}^k\mathbf{x} = \Phi(\mathbf{x};\mathbf{h};\mathbf{S}) \tag{5}$$

definida como $\Phi(\mathbf{x};\mathbf{h};\mathbf{S})$ no lado direito da igualdade para representar a saída de um filtro de grafo com coeficientes $\mathbf{h}$ implementado sobre a representação matricial $\mathbf{S}$ e aplicada ao grafo de sinal $\mathbf{x}$. A saída $\mathbf{u} = \Phi(\mathbf{x};\mathbf{h};\mathbf{S})$ também é um grafo de sinal. Na Equação (5), $\mathbf{S}$ é o grafo de operador de deslocamento, que pode ser interpretado como a matriz de adjacência do grafo com entradas $S_{ij} = w_{ij}$. Da mesma forma, pode-se trabalhar com matrizes Laplacianas ou versões normalizadas tanto da matriz de adjacência quanto da Laplaciana.

Uma característica do filtro de grafo é a sua localidade. A sequência de difusão pode ser definida como a coleção de grafos de sinais $\mathbf{z}_k = \mathbf{S}^k\mathbf{x}$, assim o filtro descrito em (5) pode ser reescrito como $\mathbf{u} = \sum_{k=0}^{K} h_k\mathbf{z}_k$. A sequência de difusão é dada pela recursão $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$, sendo $\mathbf{z}_0 = \mathbf{x}$. Observando que $S_{ij} \neq 0$ apenas quando o par (i,j) é uma aresta do grafo, percebe-se que a sequência de difusão satisfaz:

$$z_{k,i} = \sum_{j:(i,j)\in\varepsilon} S_{ij}z_{k-1,j} \tag{6}$$

logo, pode-se interpretar o filtro de grafo em (5) como uma operação que propaga informação através de nós adjacentes (Fig. 1).

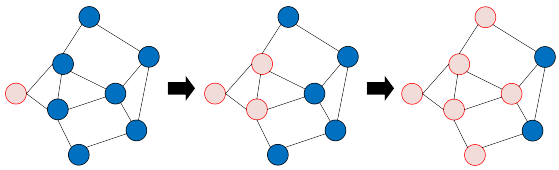

Figura 1. Esquemático de propagação de informação no filtro convolucional de grafo.

As GNNs são uma expansão dos filtros de grafo que utilizam não-linearidades ponto-a-ponto, que são funções não-lineares aplicadas independentemente a cada componente de um vetor. Seja $\sigma : \mathbb{R} \longrightarrow \mathbb{R}$ uma função de variável única (escalar), que pode-se estender para a função vetorial $\sigma : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ por aplicação independente a cada elemento. Sendo $\mathbf{u} = [u_1;...;u_n] \in \mathbb{R}^n$, o vetor de saída $\sigma(\mathbf{u})$ será dado por:

$$\sigma(\mathbf{u}) : [\sigma(\mathbf{u})]_i = \sigma(u_i) \tag{7}$$

assim, o vetor de saída pode ser representado como $\sigma(\mathbf{u}) = [\sigma(u_1);...;\sigma(u_n)]$. Por simplicidade, utilizou-se σ para denotar tanto a função escalar quanto a função vetorial ponto-a-ponto.

Numa GNN de camada única, o grafo de sinal $\mathbf{u}$ passa por uma função não-linear ponto-a-ponto que satisfaça (7) para obter:

$$\mathbf{z} = \sigma(\mathbf{u}) = \sigma\left(\sum_{k=0}^{K} h_k\mathbf{S}^k\mathbf{x}\right). \tag{8}$$

A transformação mostrada em (8) é chamada de Perceptron de grafo e ilustrada na Fig. 2. Diferentemente do filtro de grafo, o Perceptron de grafo é uma função não-linear da entrada. Entretanto, implementa uma forma muito simples de processamento não-linear, uma vez que a não-linearidade não mistura componentes do sinal. Os componentes do sinal são misturados pelo filtro de grafo, mas são processados elemento a elemento através de σ.

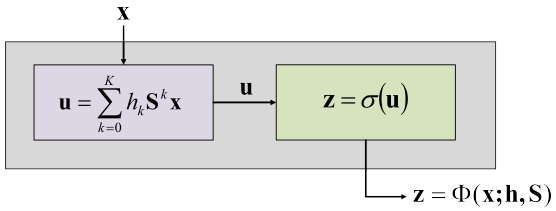

Figura 2. Esquemático do Perceptron de grafo, que é composto por um filtro de grafo associado a uma não-linearidade ponto-a-ponto.

Os Perceptrons de grafo podem ser empilhados em camadas para criar GNNs de múltiplas camadas. Este procedimento pode ser matematicamente expresso como uma composição de funções, onde as saídas de uma camada se tornam as entradas da próxima camada. Seja $l = 1,...,L$ o índice de camadas e $\mathbf{h}_l = h_{lk}{}_{k=0}^{K}$ coleções de $K+1$ coeficientes de filtros de grafos associados a cada camada. Cada

um destes conjuntos de coeficientes define um filtro de grafo $\Phi(\mathbf{x}; \mathbf{h}_l, \mathbf{S}) = \sum_{k=0}^{K} h_{lk}\mathbf{S}^k\mathbf{x}$. A camada l recebe como entrada a saída $\mathbf{x}_{l-1}$ da camada $l-1$, que é processada com o filtro $\Phi(\mathbf{x}; \mathbf{h}_l, \mathbf{S})$ para obter o resultado intermediário:

$$\mathbf{u}_l = \mathbf{H}_l(\mathbf{S})\mathbf{x}_{l-1} = \sum_{k=0}^{K} h_{lk}\mathbf{S}^k\mathbf{x}_{l-1} \tag{9}$$

onde, por convenção, define-se $\mathbf{x}_0 = \mathbf{x}$, de forma que o grafo de sinal $\mathbf{x}$ é a entrada da GNN. Assim como acontece no Perceptron de grafo, este resultado intermediário é processado por uma função não-linear ponto-a-ponto (que é a mesma em todas as camadas) para produzir a saída da l-ésima camada:

$$\mathbf{x}_l = \sigma(\mathbf{u}_l) = \sigma\left(\sum_{k=0}^{K} h_{lk}\mathbf{S}^k\mathbf{x}_{l-1}\right). \tag{10}$$

Após a repetição recursiva de (9) e (10) para $l = 1, ..., L$ obtém-se $\mathbf{x}_L$, que não recebe pós-processamento e é considerada a saída $\mathbf{z} = \mathbf{x}_L$. Para representar a saída da GNN, é definida a matriz de filtro $\mathbf{H} := \mathbf{h}_{l\,l=1}^{L}$ que agrupa os L conjuntos de coeficientes dos filtros em cada camada. Assim, pode-se definir também o operador $\Phi(.; \mathbf{H}, \mathbf{S})$ como:

$$\Phi(\mathbf{x}; \mathbf{H}, \mathbf{S}) = \mathbf{x}_L \tag{11}$$

É importante ressaltar que em (11), a saída da GNN $\Phi(\mathbf{x}; \mathbf{H}, \mathbf{S}) = \mathbf{x}_L$ é derivada da aplicação recursiva das Equações (9) e (10) para $l = 1, ..., L$ com $\mathbf{x}_0 = \mathbf{x}$. Esta notação enfatiza que a saída de uma GNN é dependente do filtro $\mathbf{H}$ e do grafo de operador de deslocamento $\mathbf{S}$.

Os conjuntos de coeficientes de filtros $\mathbf{H}$ que definem o operador da GNN em (11) são escolhidos de forma a minimizar a perda (*loss*) de treinamento, como descrito na Equação 3:

$$\mathbf{H}^* = \underset{\mathbf{H}}{\operatorname{argmin}} \frac{1}{Q}\sum_{q=1}^{Q} \mathcal{L}(\Phi(\mathbf{x}_q; \mathbf{H}, \mathbf{S}), \mathbf{y}_q). \tag{12}$$

De modo análogo ao caso dos filtros de grafo, a otimização é realizada sobre a matriz do filtro $\mathbf{H}$ com o operador de deslocamento $\mathbf{S}$ dado.

2.2 Propriedades

De acordo com Ruiz et al. (2021), GNNs possuem três características importantes que ratificam o potencial da técnica. Em primeiro lugar, GNNs são equivariantes a permutações no grafo. Isto significa que nós com vizinhos análogos, realizando observações análogas, realizam as mesmas operações. Isto permite que a rede "preencha" informações desconhecidas de um nó se as estruturas locais do grafo forem as mesmas. Em segundo lugar, GNNs são estáveis a deformações no grafo, ou seja, o princípio da equivariância se aplica mesmo se as estruturas de vizinhança locais forem apenas similares, e não necessariamente idênticas. Finalmente, GNNs são transferíveis, de forma que podem ser treinadas e executadas em grafos de diferentes tamanhos.

Estas três características conferem às GNNs um grande potencial de generalização. Para as aplicações de detecção de anomalias, isto significa que os modelos não aprendem as características do sistema em si, mas heurísticas que determinam a ocorrência e a localização da anomalia com base nos dados disponíveis.

2.3 Trabalhos relacionados

Os grafos têm sido utilizados para aplicações de detecção de anomalias em diversos contextos. Em Candelieri et al. (2014) a estrutura de grafo é utilizada para realizar clusterização de redes de distribuição de água e assim localizar os pontos de vazamento. A metodologia é aplicada a dados de simulação hidráulica gerados pelo aplicativo EPANET e mostra como a clusterização baseada em grafo é mais eficiente que as técnicas convencionais de ML.

No trabalho desenvolvido por Ponti et al. (2021), os autores apresentam uma proposta de métrica de vulnerabilidade baseada em grafo para RDA urbanas. Representando a RDA como um grafo, os autores analisam cada nó como um espaço de informação, cujos elementos (nós) possuem distribuições de probabilidade que caracterizam a distância entre um nó e outro. O grafo, então, torna-se uma agregação das distribuições de probabilidade dos nós. A métrica de vulnerabilidade é parte de um *framework* de análise das redes para identificação de pontos críticos e suporte à tomada de decisão para manutenção preventiva. Não há coleta de dados como pressão ou vazão.

Em Romero et al. (2020) os autores apresentam uma técnica de localização de vazamentos em RDA utilizando a combinação de *Deep Learning* para análise dos dados de medição e clusterização baseada em grafos para localização do ponto de vazamento. A série temporal de dados de pressão nos nós é convertida em imagem através de uma técnica chamada *Gramian Angular Field*. Em seguida, um processo de clusterização é realizado para dividir a rede em sub-redes menores de acordo com as informações topológicas do grafo associado. Finalmente, os *clusters* menores são analisados por uma rede neural profunda para identificação de vazamentos. A metodologia foi testada com dados de simulação computacional.

No ramo de sistemas elétricos, de Freitas and Coelho (2021) apresentam um método de localização de faltas baseado numa variante de GNN, chamada *Gated Graph Neural Networks* (GGNN). O objetivo do trabalho foi criar um *framework* mais genérico para a detecção de falhas, utilizando as propriedades de GNN para obter uma rede flexível, que se baseia em heurísticas para realizar detecção, em contraste aos modelos tradicionais de RNA que aprendem características específicas de cada sistema para identificar as faltas. A coleta de dados foi realizada em ambiente de simulação computacional, utilizando como base modelos de redes de distribuição brasileiras. Na ocorrência de uma falta, a GGNN indica qual a probabilidade de cada nó ser o local da falta, sendo que a indicação pode ser de um único nó ou de possíveis nós (normalmente na mesma região), quando não é possível identificar o nó exato.

Nota-se que as aplicações baseadas na teoria dos grafos, em particular GNNs, ganharam relevância no meio acadêmico, trazendo características muito úteis para sistemas de

monitoramento, tais como a capacidade de generalização e uso eficiente de informações de campo (especialmente a disposição espacial dos nós). Este tipo de abordagem tem sido utilizada não apenas no contexto hidráulico, mas também em áreas como Eletrônica de Potência, Cibersegurança e outras.

3. MATERIAIS E MÉTODOS

3.1 Modelagem de RDAs

O uso de algoritmos de aprendizado de máquina para detecção de vazamentos requer grandes conjuntos de dados para o treinamento. Por isso, convém estabelecer modelos computacionais que permitam a simulação de diversos cenários, incluindo vazamentos em locais distintos.

Neste trabalho, foi utilizado o *software* EPANET para a criação de modelos de redes hidráulicas. No EPANET, cada nó é uma entidade genérica que permite diversas interpretações. Nas simulações de RDAs no contexto deste trabalho, temos os nós (i) de junção (ramificações da rede, ligações em "T" ou outros formatos), nos quais não há consumo nem medição de variáveis; (ii) de medição, onde há a medição de variáveis (normalmente posicionados na entrada da rede); (iii) de consumo, que são os pontos de consumo normal de água, onde há também a medição individual de consumo e (iv) de vazamento, que são pontos colocados especificamente para a simulação de vazamento.

Dos diversos tipos de RDAs existentes, o escopo deste trabalho são RDAs de "pequeno porte" (tais como conjuntos residenciais ou instalações industriais), em contrapartida às RDAs de "grande porte", que envolvem distribuição de água ao longo de cidades inteiras ou múltiplas cidades. Desta forma, os estudos serão voltados para tubulações com dimensão de até duas polegadas e distâncias de dezenas ou centenas de metros. Cada nó de consumo representa uma unidade de consumo individual (residência, ponto comercial, setor de uma instalação industrial, entre outros).

O sistema de abastecimento é modelado como um reservatório de nível fixo em conjunto com uma bomba, replicando a distribuição de água em ambiente urbano. Diversos pontos de consumo externos são posicionados para representar o consumo de água no restante da rede, de forma a produzir as variações de demanda que normalmente ocorrem nas RDAs.

Os modelos devem considerar também a natureza estocástica das demandas de usuários, que neste trabalho é representada por diferentes padrões de consumo que são atribuídos por meio de sorteio a cada dia simulado (período de 24 horas). Além da variação no padrão de consumo, o consumo-base (valor de referência que multiplica os fatores do padrão de consumo) também é sorteado a cada dia de simulação, dentro de um intervalo de valores predefinido. A coleta dos dados é realizada em intervalos de tempo predefinidos. No contexto deste trabalho, convém manter este intervalo na ordem de minutos a dezenas de minutos, para que seja possível integrar a solução a aplicações práticas do contexto de Internet das Coisas, sobretudo redes de comunicação de baixo *throughput* e longo alcance (tecnologias LPWAN - *Low-Power Wide Area Networks*).

Após a criação e parametrização da rede no *software* EPANET, o modelo pode ser exportado. Neste trabalho, utilizou-se o EPANET-MATLAB *toolkit*, uma ferramenta que carrega os modelos do EPANET no contexto do *software* de cálculos MATLAB, desta forma podem ser elaborados *scripts* para automatizar as simulações e criar os conjuntos de dados de simulação (*data sets*).

3.2 Algoritmo de ML

A estrutura do algoritmo de ML utilizado é apresentada na Fig. 3.

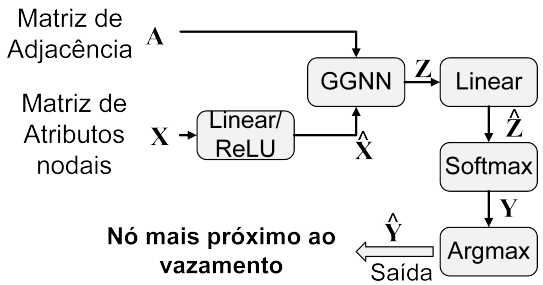

Figura 3. Esquemático do algoritmo de ML baseado em GGNN para detecção de vazamentos.

As entradas são a matriz de adjacências e a matriz que contém os atributos dos nós (no caso, as medições de pressão em cada nó). A primeira camada é uma combinação linear padrão combinada com uma função de ativação ReLU (*Rectified Linear Unit*):

$$\hat{\mathbf{X}} = \mathrm{ReLU}(\mathbf{W}_{in}\mathbf{X} + \mathbf{b}_{in}) \tag{13}$$

onde $\mathbf{W}_{in} \in \mathbb{R}^{mxr}$ é a matriz de pesos de entrada, $\mathbf{b}_{in}$ são os *biases* de entrada [1] e ReLU(.) é uma função elemento a elemento tal que $\mathrm{ReLU}(x) = \max(0, x)$. A variável r representa o número de atributos de entrada, enquanto a variável m define o tamanho dos estados escondidos. Quanto mais estados escondidos, maior é a capacidade de aprendizagem do modelo, porém um número de estados muito alto aumenta o custo computacional e o risco de *overfitting* [2].

Na próxima camada, uma nova representação $\mathbf{Z}$ do grafo é obtida de $\hat{\mathbf{X}}$ e $\mathbf{A}$ utilizando uma GGNN. GGNNs (Li et al., 2015) são expansões das GNNs tradicionais com treinamento diferenciado, no qual o vetor de estados de cada nó é atualizado num número fixo de etapas. Assim, os gradientes podem ser computados através de retropropagação ao longo do tempo (de Freitas and Coelho, 2021):

$$\mathbf{Z} = \mathrm{GGNN}(\hat{\mathbf{X}}, \mathbf{A}) \tag{14}$$

O vetor de estados final $\mathbf{z}_v$ de cada nó contém informação de um certo número de nós vizinhos. O tamanho desta

[1] *Bias* é um valor que se soma ao final do cálculo realizado por um determinado neurônio. Ele serve para calibração do resultado final do neurônio.

[2] *Overfitting* é o fenômeno resultante do excesso de treinamento da rede, que faz com que a rede "decore" os dados do conjunto de treinamento em vez de "aprender" as relações entre os dados. A rede perde a capacidade de generalização.

vizinhança é um dos hiperparâmetros (chamado "passes de propagação" ou *propagation steps*) que podem ser configurados no modelo.

A seguir, uma gradação (*score*) $\hat{\mathbf{z}}_v$ é atribuída a cada vetor de estados $\mathbf{z}_v$ através de uma combinação linear:

$$\hat{\mathbf{Z}} = \mathbf{W}_{out}\mathbf{Z} + \mathbf{b}_{out} \tag{15}$$

onde $\mathbf{W}_{out} \in \mathbb{R}^m$ são os pesos da saída e $\mathbf{b}_{out}$ são os *biases* da saída. Após, a função *softmax* normaliza as saídas em uma distribuição de probabilidades, tal que:

$$y_v = \frac{e^{\hat{z}_v}}{\sum_{u\in\gamma} e^{\hat{z}_u}} \tag{16}$$

é a probabilidade de o nó v ser o nó mais próximo ao vazamento. Para o cômputo da acurácia do algoritmo, a distribuição de probabilidades é "arredondada" pela função *Argmax*, sendo que o nó com maior probabilidade recebe valor 1 e os demais recebem valor 0. Esta operação possibilita a composição das matrizes de confusão.

4. ESTUDO DE CASO

Foi elaborada uma RDA hipotética para o estudo de caso (Fig. 4). Ela representa um conjunto residencial com 4 casas (pontos amarelos), cada uma delas equipada com um medidor na sua entrada. Há um medidor instalado na entrada do conjunto (ponto verde). Foram inseridos 30 pontos de consumo externos (pontos roxos) que representam as demandas de consumo externas à RDA monitorada. Tais demandas influenciam na dinâmica de abastecimento urbano, por isso são inseridas no modelo. Os pontos pretos representam pontos de junção (ligações em T ou outros formatos). Os pontos vermelhos são utilizados para simular vazamentos. As distâncias mais relevantes são destacadas por cotas (fora de escala).

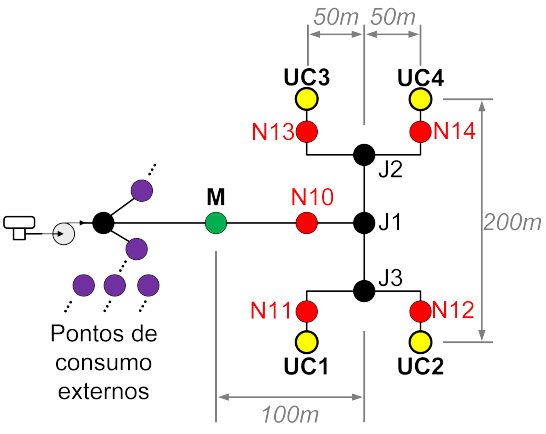

Figura 4. Mapa da RDA do Estudo de Caso.

Este modelo dispõe de 16 padrões de consumo diários, que são atribuídos aleatoriamente às unidades consumidoras para as simulações diárias. Há 11 padrões de consumo adicionais que modelam o consumo dos pontos externos à rede monitorada, também atribuídos por meio de sorteio, que representam as variações de demanda na rede. Os vazamentos também são representados por padrões de consumo, com valores fixos. Os vazamentos podem perdurar o dia todo (24 horas) ou por metade do dia (início às 12h), com volume de 2 litros por minuto.

A sequência de simulações é dada por: 100 simulações sem vazamento; 50 simulações com vazamento de dia inteiro no nó N10; 50 simulações com vazamento de meio dia no nó N10 e assim sucessivamente até o nó N14, totalizando 600 dias.

Nos experimentos, é coletada a pressão em cada nó de medição da rede. São coletadas, também, as medidas de pressão nos nós de junção (bifurcações, junções em "T" e outras). Estes dados são necessários para povoar de dados todos os nós do grafo. Embora na prática esta coleta de dados seja inviável, ela foi utilizada para verificar a aplicabilidade do algoritmo para os fins desta pesquisa. Futuramente, estes dados serão estimados a partir de regressões lineares ou funções de aproximação.

Durante o processo de conversão da rede hidráulica para grafo, há uma mudança de nomenclatura, uma vez que os nós do grafo são contados a partir de zero. Por isso, nos conjuntos de treinamento e teste, os rótulos (*labels*) de cada classe são atribuídos de acordo com o número de cada nó do grafo. O grafo equivalente à RDA [3] é apresentado na Fig. 5.

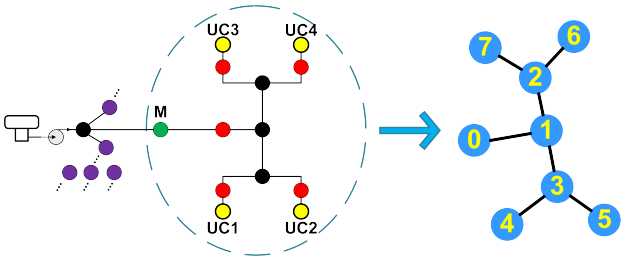

Figura 5. Grafo equivalente da RDA do Estudo de Caso.

Uma vez que a saída do algoritmo é a indicação de um nó (no caso, o nó mais próximo do vazamento), cada ponto de vazamento é associado ao nó imediatamente à sua frente. Quando não há vazamento, a saída da rede é o nó de medição da entrada (M). O vazamento em N10 é atribuído ao nó J1 e assim sucessivamente. Nota-se que os nós 2 e 3 do grafo não possuem nenhum vazamento associado a eles. A definição dos rótulos é dada na Tabela 1.

Tabela 1. Atribuições de rótulos para as classes de vazamentos da Rede 1.

Classe	Nó no mapa	Atribuição equivalente no grafo
Sem vazamento	M	0
Vazamento em N10	J1	1
Vazamento em N11	UC1	4
Vazamento em N12	UC2	5
Vazamento em N13	UC3	6
Vazamento em N14	UC4	7

Após o carregamento e normalização dos dados, pode-se executar o algoritmo da GNN. O modelo é implementado com base na matriz de adjacência da RDA proposta e nos

[3] Nota-se que os pontos de vazamento não compõem o grafo, porque não é possível saber de antemão onde acontecerá o vazamento.

dados de pressão calculados por simulação computacional. Os seguintes parâmetros foram utilizados no modelo (GNN):

- Tamanho da vizinhança: 5
- Tamanho da camada escondida: 256
- Tamanho do lote: 300
- Função de custo (*loss*): entropia cruzada
- Função de otimização: Adam
- Subdivisões do algoritmo de validação cruzada (*K-folds*): 10
- Número de épocas: 300

Este algoritmo atua como um classificador nodal, que aponta o nó mais próximo ao vazamento. Quando não há vazamento, o algoritmo foi configurado para apontar o nó do medidor de entrada.

Ao longo do treinamento, as saídas produzidas pelo algoritmo são comparadas com os resultados esperados, utilizando a função de custo (*loss*). No final, são gerados gráficos apresentando a evolução do *loss* e da acurácia ao longo do processo de treinamento e teste. Além destes, foram utilizadas matrizes de confusão. Elas permitem uma análise mais específica do desempenho do algoritmo, pois a acurácia é avaliada classe a classe. Sendo assim, pode-se observar em quais classes houve maior ou menor acurácia.

5. RESULTADOS E DISCUSSÕES

Ao longo dos estudos, observou-se que a distância entre os pontos de medição é um fator importante para a acurácia do processamento. Isto ocorre porque pontos de medição muito próximos apresentam pressão similar, de forma que os algoritmos não conseguem diferenciar os eventos de vazamento e operação normal. À medida que se aumenta a distância entre os pontos de medição, as perdas de carga decorrentes de atrito nos canos e formato da tubulação (derivações, curvas e outros obstáculos) se tornam mais evidentes. Sendo assim, há uma diferença mais notória entre as medições nos pontos, o que facilita a distinção dos eventos anormais.

No estudo de caso implementado, verifica-se pela curva de acurácia (Fig. 6) que a acurácia média obtida no conjunto de teste foi de 78,5%.

A matriz de confusão resultante do conjunto de teste é apresentada na Fig. 7. O pior desempenho foi obtido na classe 1 (vazamento no nó N10), onde 55,3% das classificações foram corretas e 35,8% foram erroneamente classificadas como não-vazamento (classe 0). Ainda, na coluna referente à classe 0 (não-vazamento), observa-se que o acerto foi de 78,4%, sendo que 20,4% das amostras forram erroneamente classificadas na classe 1. Nas demais classes, a acurácia foi superior a 90%.

6. CONCLUSÃO

Este trabalho apresenta os primeiros resultados da implementação de Redes Neurais de Grafo à detecção de vazamentos em RDAs. O estudo de caso apresentado foi baseado numa RDA hipotética criada em ambiente computacional. Assim, foi possível obter dados suficientes para o treinamento do algoritmo e análise dos resultados.

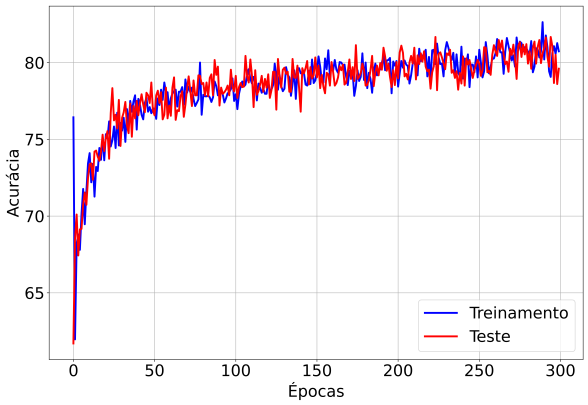

Figura 6. Gráfico de acurácia resultante dos conjuntos de dados de treinamento e teste do algoritmo de GNN.

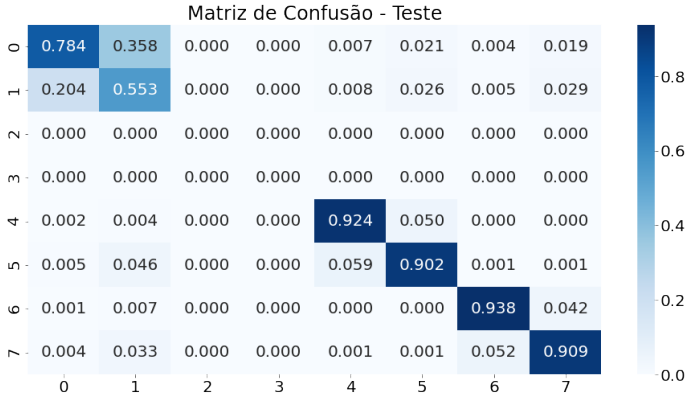

Figura 7. Matriz de confusão resultante do conjunto de dados de teste do algoritmo de GNN.

No estudo de caso proposto, o algoritmo baseado em GNN foi capaz de obter acurácia de 78%. Este algoritmo é um classificador nodal, que aponta o nó conhecido da rede (medidor, junta ou unidade consumidora) mais próximo ao vazamento ativo. Estes resultados mostram que a abordagem baseada em GNN é capaz de performar mesmo em redes de sensores com taxas limitadas de aquisição de dados.

Nesta aplicação, foram utilizados apenas dados de pressão, com aquisição de dados a cada 10 minutos. Futuramente, serão realizados ensaios envolvendo outras variáveis, como vazão e volume. Também serão necessários ensaios com RDAs de diferentes tamanhos e quantidades de pontos, a fim de se ter uma ideia mais precisa da sensibilidade (volume mínimo de vazamento identificável) e da acurácia em diferentes cenários.

REFERÊNCIAS

Brasil (2021). Diagnóstico temático dos serviços de água e esgotos - ano 2020. *Ministério do Desenvolvimento Regional*.

Candelieri, A., Conti, D., and Archetti, F. (2014). A graph based analysis of leak localization in urban water networks. *Procedia Engineering*, 70, 228–237.

Chan, T.K., Chin, C.S., and Zhong, X. (2018). Review of current technologies and proposed intelligent methodo-

logies for water distributed network leakage detection. *IEEE Access*, 6, 78846–78867.

Connor, R. (2015). *The United Nations world water development report 2015: water for a sustainable world*, volume 1. UNESCO publishing.

de Freitas, J.T. and Coelho, F.G.F. (2021). Fault localization method for power distribution systems based on gated graph neural networks. *Electrical Engineering*, 1–8.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. (2015). Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*.

Nkemeni, V., Mieyeville, F., and Tsafack, P. (2020). A distributed computing solution based on distributed kalman filter for leak detection in WSN-based water pipeline monitoring. *Sensors*, 20(18), 5204.

Ponti, A., Candelieri, A., Giordani, I., and Archetti, F. (2021). A novel graph-based vulnerability metric in urban network infrastructures: The case of water distribution networks. *Water*, 13(11), 1502.

Rojek, I. and Studzinski, J. (2019). Detection and localization of water leaks in water nets supported by an ICT system with artificial intelligence methods as a way forward for smart cities. *Sustainability*, 11(2), 518.

Romero, L., Blesa, J., Puig, V., Cembrano, G., and Trapiello, C. (2020). First results in leak localization in water distribution networks using graph-based clustering and deep learning. *IFAC-PapersOnLine*, 53(2), 16691–16696.

Ruiz, L., Gama, F., and Ribeiro, A. (2021). Graph neural networks: Architectures, stability, and transferability. *Proceedings of the IEEE*, 109(5), 660–682.

Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *IEEE transactions on neural networks*, 20(1), 61–80.