KAPITEL 4 / *CHAPTER 4* [4]
FORECASTING WATER LEAKS IN PIPELINE NETWORKS WITH THE GRAPH CONVOLUTIONAL NETWORK APPROACH

DOI: 10.30890/2709-2313.2024-32-00-007

Introduction

With the advancement of industry and the process of urbanization, there has been an increasing need to transport water resources to cities via pipelines, both from dams and underground sources. Population growth has further escalated the demand for water resources, while global warming has led to a decrease in these vital resources. Given this scenario of rising demand and dwindling supply, it is imperative to transport water resources efficiently, minimizing any losses during transit. Water transportation and distribution are achieved through either open or closed pipeline networks, each with its own set of challenges.

Failures in these pipeline networks are a common and significant global issue. These failures can manifest in two primary forms: leaks and blockages. This study concentrates on detecting water leaks within pipeline networks. Leaks can result from a variety of causes, including pressure or temperature fluctuations, corrosion, wear and tear, and damage caused by third parties. Such leaks can lead to critical problems in pipeline transportation systems, making accurate leak detection essential to mitigate economic losses and protect the environment [1].

Therefore, it is crucial to rapidly detect and repair pipeline network failures. Prompt and precise intervention in these failures presents a critical decision-making challenge. There are two main approaches to detecting failures in pipeline networks: physical inspection and mathematical model simulation. Although physical methods are highly accurate in identifying the location and extent of failures, they are costly due to production stoppages [2]. Conversely, mathematical approaches can theoretically detect leaks at a much lower cost.

Various methods are employed to detect leaks in pipeline networks. Some of these methods include acoustic techniques, negative pressure wave measurement, input and

[4]**Authors:** *Mysak Pavlo Vasylovych, Mysak Ihor Vasylovych*

output value measurements of pressure and fluid velocity, vibration analysis, distributed fiber optic sensing, infrared cameras, and lidar systems. Machine learning algorithms, such as convolutional neural networks (CNNs) and artificial neural networks (ANNs), are increasingly used for leak detection and classification [3, 4]. Numerous studies have focused on detecting and localizing leaks in pipeline systems, contributing to a growing body of literature on machine learning applications for this purpose. However, research on machine learning methods for pipeline leak detection remains limited and under-explored.

A review of the literature reveals that studies on detecting leaks in pipelines using machine learning methods can be categorized into five main approaches. The machine learning models employed in these studies are as follows:

1. Negative Pressure Wave (NPW) Method: A wireless sensor network-based machine learning (WML) algorithm was utilized for this method.

2. Acoustic-Based Model: This method includes support vector machines (SVM), neural networks, artificial neural networks (ANNs), as well as a combination of SVM and Relevance Vector Machine (RVM) for leak detection.

3. Thermal Infrared (IR) Camera-Based Method: This approach utilized immune neural networks and convolutional neural network (CNN) techniques for leak detection, with some studies specifically focusing on petroleum pipelines.

4. Pressure-Monitoring Methods: These methods employed the sparrow search algorithm (SPSA) and CNN for leak detection in petroleum pipelines, along with a deep-learning method known as Deeppipe.

5. Fluid Transient Waves Method: This method utilized artificial neural networks (ANNs) and a machine learning (ML)-based framework for leak detection.

In summary, while significant strides have been made in detecting and addressing pipeline leaks through various methods, the integration of machine learning offers promising potential. Continued research and development in this area are essential to enhance the efficiency, accuracy, and cost-effectiveness of leak detection and repair in pipeline networks, ultimately contributing to better resource management and environmental protection [5, 6].

This academic literature review underscores the application of various machine learning algorithms for detecting leaks in pipelines. It demonstrates the effectiveness of different methods and encourages further research to enhance leak detection techniques. The reported accuracy rates for pipeline fault diagnosis using machine learning algorithms range from 78.51% to 99%. Theoretical representation of many complex systems encountered in real life can be challenging, but graph structures provide a viable solution.

Numerous structures, including airline networks, banking systems, social networks, medical systems, and supply chains, can be effectively represented using graph structures. For instance, intricate systems like distribution networks with multiple pipeline convergence or divergence points can be depicted through graphs. In the literature, pipeline networks are often represented using graphs, capturing the spatial correlations of the network with models such as the Graph Convolutional Network (GCN).

In this study, water-carrying pipelines are represented using graphs. The pressure values at specific points on the pipeline are visualized over time by modeling and observing pressure drops. The graph-based machine learning algorithm GCN is employed to detect pipeline leakage scenarios. Pipeline networks are rarely limited to a single pipeline; water distribution networks, for example, often feature multiple pipelines converging or diverging. Graph data structures offer advantages over traditional data structures in representing these complex systems.

Conventional machine learning models like Support Vector Machines (SVM) and Artificial Neural Networks (ANNs) often fail to account for the inherent dependencies among nodes and edges in pipeline networks. However, recognizing that a leakage in one pipeline can affect interconnected pipelines is crucial. This study proposes an approach leveraging graph data structures to represent pipeline networks effectively, utilizing GCN as a graph-based machine learning model [7]. GCN can capture and account for the interdependencies between nodes and edges, enhancing accuracy and predictive capabilities tailored to pipeline networks.

In the literature, GCN studies generally focus on object recognition and

classification, achieving accuracy rates of 75% in urban pipeline networks, 94% in image processing, 88% in customer product recommendations, and 86% in fault detection on steam turbines. This study aims to predict leaks by monitoring pressure drops in water-carrying pipelines, focusing on accurately detecting pipeline leaks.

In this research, the network structure of pipelines is represented using graph representations, and the GCN model is utilized. GCNs offer an advantage in capturing the intricate relationships among pipeline connections. Data for the GCN algorithm were collected from a test set using experimental methods, where a leakage scenario was deliberately created. Two datasets, edge and node, were specifically constructed for this purpose. The performance of the GCN algorithm in leak detection was compared with existing studies in the literature, with a particular focus on a comparative analysis with traditional machine learning methods, notably using the SVM algorithm as a reference.

Given the escalating demand for water resources and the concurrent decline in availability, preventing leaks in pipeline transportation systems has become paramount. Detecting leaks facilitates prompt intervention, maintenance, and repair, minimizing economic losses and mitigating environmental impact. The findings of this study highlight the efficacy of the GCN algorithm in water leakage detection and underscore the potential of graph-based machine learning in tackling complex problems.

4.1. Materials and Methods

This study utilized graph-based machine learning algorithms, specifically Graph Convolutional Networks (GCN) and Support Vector Machines (SVM). These algorithms were implemented using various Python libraries, including StellarGraph, Pandas, NumPy, Scikit-learn (Sklearn), TensorFlow, IPython, Matplotlib, and Pyvis.

The performance evaluations of the GCN and SVM algorithms were conducted using several metrics: confusion matrix, accuracy, precision, recall, and F1 score. These metrics provide a comprehensive assessment of the models' performance,

allowing for a detailed analysis of their strengths and weaknesses in detecting pipeline leaks.

To prevent overfitting, the GCN algorithms were implemented with early-stopping methods. Early stopping is a regularization technique that terminates training when the performance on a validation set begins to deteriorate, ensuring that the model maintains its ability to generalize to unseen data [8].

In our lives, non-Euclidean structured data exist in various forms, such as molecular structures in chemical analysis, sensor networks in communication networks, and social networks in social sciences. Traditional convolutional neural networks (CNNs) struggle to perform local feature extraction on graph data due to the unstructured nature of these data and the unique surrounding structure of each node. Despite these challenges, researchers are eager to extend deep learning models to non-Euclidean domains, given deep learning's proven efficacy in addressing a wide range of problems in acoustics, computer vision, and natural language processing.

Recently, there has been an increasing focus on using deep learning methods to analyze graph data. These methods can be broadly categorized into two main approaches. The first approach involves embedding nodes in a low-dimensional vector space to learn a uniform length representation for each node, with network embedding being a representative example. The second approach applies various deep learning models directly to graph data, which can be further divided into five categories based on model structure and training strategy: graph recurrent neural networks, graph convolutional networks (GCNs), graph autoencoders, graph reinforcement learning, and graph adversarial methods. Among these, GCNs have emerged as a particularly prominent branch of graph deep learning models [9].

GCNs function similarly to CNNs in that they act as feature extractors. Unlike traditional deep learning models such as CNNs and recurrent neural networks, GCNs can process non-Euclidean structured data. The concept of graph convolutional operations originates from the field of graph signal processing. Bruna et al. first proposed a CNN formula for graphs, defining convolution in the spectral domain based on the graph Laplacian. Subsequent contributions by Duvenaud et al., Kipf and

Welling, and Atwood and Towsley introduced various GCN models that replaced traditional graph filters with adjacency matrices featuring self-loops. These models used propagation rules to calculate the output of each neural network layer during weight updates. Defferrard et al. extended this framework by incorporating fast local spectral filters and efficient merging operations.

In recent years, the application of convolutional techniques to graphs has surged, leading to the development of numerous GCN-related models. Graph convolution can be categorized into two main branches: spectral convolution and spatial convolution. When a polynomial spectral kernel is used, these two approaches can overlap. Spectral domain GCNs are powerful but have several limitations:

1. Scalability: The training of GCNs requires knowledge of the adjacency matrix for all training and test nodes, making the process transductive and unsuitable for large-scale graphs.

2. Graph Type Restrictions: Spectral domain GCNs are limited to undirected graphs because the Laplacian matrix used requires matrix symmetry for eigendecomposition, a prerequisite for spectral convolution.

3. Layer Depth: GCNs are typically confined to shallow layers. Each GCN layer performs Laplace smoothing, and adding multiple layers can result in overly similar node attributes within the same connected component, leading to excessively smooth output features.

Many recent studies aim to address these limitations. The spatial domain graph convolution method, for instance, compensates for some shortcomings of spectral domain GCNs by saving memory at the cost of time efficiency, resulting in higher complexity. Recent research has made significant progress in handling large-scale graph data, directed graphs, multiscale graph tasks, dynamic graphs, and relational graphs.

Support Vector Machines (SVM). Support Vector Machine (SVM) is one of the most popular supervised learning algorithms, used for both classification and regression problems. However, it is primarily utilized for classification tasks in machine learning.

The primary objective of the SVM algorithm is to find the optimal hyperplane that can effectively segregate an n-dimensional space into distinct classes. This allows for the accurate categorization of new data points in the future. The optimal decision boundary created by SVM is known as the hyperplane (fig. 1).

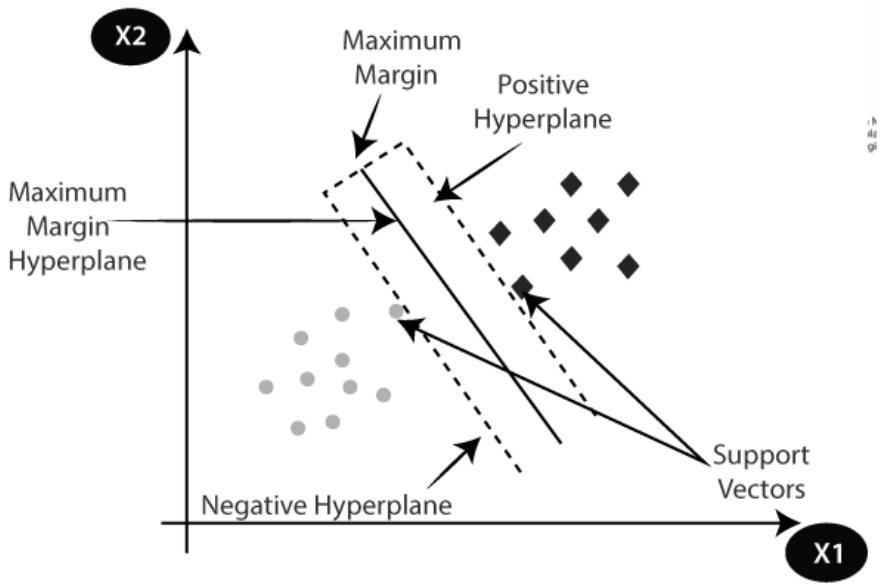

Figure 1 - Hyperplane and Support Vectors in the SVM Algorithm

SVM identifies the extreme points or vectors that assist in forming this hyperplane. These extreme cases are termed support vectors, giving the algorithm its name, Support Vector Machine. For instance, consider a scenario where there are two distinct categories classified using a decision boundary or hyperplane.

To illustrate SVM, imagine we encounter a peculiar animal that shares characteristics of both cats and dogs. We want to create a model that can accurately identify whether this creature is a cat or a dog. By using the SVM algorithm, we train our model with numerous images of cats and dogs, allowing it to learn their distinguishing features. When we test the model with this unusual creature, the support vectors create a decision boundary between the data (cats and dogs) and identify the extreme cases. Based on these support vectors, the model classifies the creature correctly as a cat.

SVM algorithms have diverse applications, including face detection, image classification, and text categorization.

Types of SVM

1. Linear SVM: Linear SVM is used for linearly separable data. If a dataset can be classified into two classes using a single straight line, it is termed linearly separable data, and the classifier used is called a Linear SVM classifier.

2. Non-linear SVM: Non-linear SVM is used for non-linearly separable data. If a dataset cannot be classified using a straight line, it is termed non-linear data, and the classifier used is called a Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM Algorithm

Hyperplane: In n-dimensional space, there can be multiple decision boundaries or lines that segregate the classes. The goal is to identify the best decision boundary that classifies the data points effectively. This optimal boundary is known as the hyperplane of SVM. The dimensions of the hyperplane depend on the number of features present in the dataset. For instance, if there are two features, the hyperplane will be a straight line. If there are three features, the hyperplane will be a two-dimensional plane. The hyperplane is always created to have the maximum margin, which is the maximum distance between the data points of different classes.

Support Vectors: Support vectors are the data points or vectors closest to the hyperplane and influence its position. These vectors support the hyperplane, hence the name support vectors.

SVM algorithms have diverse applications, including face detection, image classification, and text categorization.

4.2. Empirical Study

This study conducted an experimental investigation to obtain the necessary datasets for the proposed GCN and SVM algorithms, aiming to predict pipeline leaks. The experimental design and implementation phase consisted of four main steps:

1. Research Design

2. Selection of Data Collection Methods

3. Data Collection Process

4. Data Recording

During the research design phase, the data to be collected (such as pressure and flow rate) were identified. The experimental dataset was prepared based on a created leak scenario, and the sensors were calibrated accordingly. The limitations and constraints of this research were also identified.

Research Limitations

1. The system is assumed to be represented by eight pressure sensors and two flow meters.

2. The experimental setup and measurement instruments are assumed to be unaffected by external factors such as heat, vibration, noise, and light.

3. The region's characteristics where the experimental setup is located, such as altitude, are assumed not to affect measurement precision.

4. The PVC pipe used in the experimental setup is assumed to represent other pipe types used in transmission lines.

5. The findings are limited to the data obtained from the experimental setup.

6. The representation of liquid fluids is limited to water, used in the experimental setup.

7. The pressure measurements are limited to the eight pressure sensors, two flow meters, and PVC pipes used in the experimental setup.

An experimental setup was designed to create datasets required for operating graph-based machine learning algorithms. A leakage scenario was created to represent potential leakage situations. The data collection method and recording processes were meticulously planned. During the selection of data collection methods, the nature and volume of the data were considered to ensure accurate and consistent data collection [10]. A 12-channel data acquisition card, recording ten data points per second, was used for data recording. The recorded data were transferred to electronic spreadsheets, and any incorrect or missing data were identified and corrected. For graph-based machine learning algorithms, the obtained data must be represented by nodes and edges [11]. In the experimental dataset, the pressure sensors represent the nodes, and the

connections between these sensors represent the edges [12]. A total of eight pressure sensors were used in the dataset. The dataset was standardized using the Python programming language and the Sklearn and StandardScaler libraries, as the measurement intervals for variables such as flow rate and pressure differed. Additionally, the dataset was normalized using the MinMaxScaler library, as artificial intelligence models perform more efficiently with values ranging between 0 and 1.

Conclusion

When reviewing the literature, it was found that machine learning algorithms have been utilized for fault diagnosis in pipeline systems, yielding accuracy rates ranging from 78.51% to 99%. However, it is notable that non-graph-based machine learning models, such as SVM, ANNs, RVM, CNN, SPSA, and WML, have been predominantly employed for predicting and detecting leaks in pipelines.

In this study, the GCN model achieved a detection accuracy of 95% for leak conditions, 94% for risky situations, and 100% for normal situations [13]. The SVM (Support Vector Machine) model was used as a reference to determine the GCN model's effectiveness. The SVM model achieved a detection accuracy of 81% for leaks, 82% for risky conditions, and 100% for normal conditions. Overall, when comparing the performance of the two models (considering leaks, risky conditions, and normal conditions), SVM achieved an accuracy of 87%, while GCN achieved 94% accuracy [14].

Conclusions Based on the Findings

1. Water Resource Management and Sustainability: The results have significant implications for water resource management, reducing water losses, and promoting sustainable living.

2. GCN Model's High Accuracy: The GCN model demonstrated high accuracy and performance in predicting pipeline water leaks, including leaks and risky conditions. The results highlight the effectiveness of the graph-based machine learning

model. The high accuracy of the GCN model in detecting risky conditions is crucial for early leak detection and preventive maintenance planning. Situations labeled as risky indicate the need for action to prevent leaks, allowing for proactive intervention and maintenance to prevent major leaks.

3. Performance in Normal Conditions: Both GCN and SVM models exhibited equally good performance in detecting normal conditions.

4. GCN Model Superiority: The GCN model outperformed the SVM model when comparing the prediction results for leaks and risky conditions. The GCN algorithm's ability to analyze edge and spatial relationships significantly enhanced leak detection.

The results indicate the potential of graph-based machine learning methods in solving complex problems such as detecting water leaks in pipelines. Based on the experience gained from this study, future research aims to replicate this study with other graph-based machine learning algorithms, such as GraphSAGE, HinSAGE, RGCN, GAT, SGC, PPNP, APPNP, and Cluster-GCN. This replication will focus on detecting leak locations and predicting leak and blockage situations in pipeline systems with gaseous fluids.