MASTER'S THESIS ACEX30

Advances in Neural Networks for Optimizing Drinking Water Pipeline Management

A Comprehensive Literature Review and Practical Application in Network Calibration with Roughness Analysis

Master's Thesis in the Master's Program Master's Infrastructure and environmental engineering MSc

DAVID MÜHLFELD

Department of Architecture and Civil Engineering
Division of Division Name
Research Group Name
CHALMERS UNIVERSITY OF TECHNOLOGY

Göteborg, Sweden 2025

Advances in Neural Networks for Optimizing Drinking Water Pipeline Management
A Comprehensive Literature Review and Practical Application in Network Calibration with Roughness Analysis

Master's Thesis in the Master's Program Infrastructure and environmental engineering MSc

DAVID MÜHLFELD

Examensarbete ACEX30
Institutionen för arkitektur och samhällsbyggnadsteknik
Chalmers tekniska högskola, 2025

Department of Architecture and Civil Engineering
Division of Division Name
Research Group Name
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: + 46 (0)31-772 1000

Cover:
Department of Architecture and Civil Engineering
Göteborg, Sweden, 2025

Advances in Neural Networks for Optimizing Drinking Water Pipeline Management
A Comprehensive Literature Review and Practical Application in Network Calibration with Roughness Analysis

Master's Thesis in the Master's Program Infrastructure and environmental engineering MSc

DAVID MÜHLFELD

Department of Architecture and Civil Engineering
Division of Division Name
Research Group Name
Chalmers University of Technology

ABSTRACT
The aim of this Master's thesis is to make use of machine learning methods for the purpose of optimizing the management of drinking water pipelines. It targets the automation of a water distribution network calibration process in an aim to attain efficiency while eliminating sources of error. In this respect, a literature review forms part of modern applications of AI to water management and covers a practical case study about network calibration.
The theoretical part describes the research status quo regarding AI and Neural Networks for water management, in general, and a bit more concretely towards network calibration. In that respect, the practical section covers the implementation of an artificial neural network to proceed with the automatic calibration for a real water supply network. Methods based purely on AI do not hold great hopes for network calibration. Therefore, further research is needed to test approaches such as Physically Informed Neural Networks or hybrid methods.
Key words:

Key words: Neural Networks, Drinking Water Pipeline Management, Network Calibration, Roughness Analysis, Artificial Intelligence in Water Systems, Hydraulic Modeling

Deep learning techniques and algorithms are emerging as a disruptive technology with the potential to transform global economies, environments and societies.

[…]

[I] hope this review will inspire research and development that can harness the power of deep learning to help achieve sustainable water management and digitalize the water sector across the world.

(Fu et al., 2022, p. 1)

1 Introduction

1.1 Background

Aging infrastructure, frequent leaks, and burst pipes present major challenges for cities worldwide. Drinking water networks are not only crucial for human life itself, but also for guaranteeing a certain quality of life. A failure in these networks can lead to significant economic losses and health risks. In the U.S. alone, for instance, more than 240,000 incidents of burst pipes are recorded annually, resulting in the loss of over 2 trillion gallons of treated drinking water (Dawood et al., 2020, p. 1). The need for rehabilitation of these networks thus is high. Capelo et al. (2021) highlight that in many industrialized countries, continuous maintenance effort is required such as constant monitoring and optimization in order to ensure the functionality of water supply systems. This includes predicting pipe breaks and optimizing maintenance activities – a point where modern technologies could be applied (Capelo et al., 2021, p. 1).

In recent years, neural networks and other AI approaches have proven to be powerful tools for automation and optimization. They hold strong potential for improving efficiency, particularly in modeling nonlinear relationships for leak detection and network calibration (Dawood et al., 2020, p. 6). However, large-scale AI application in water management is still in its early stages. Most current technologies are still at the laboratory or pilot project stage (Bertolini et al., 2021, p. 22).
Krishnan et al. (2022) further confirm the limited implementation of AI technologies in the water sector. Despite their potential, AI-based solutions have primarily been applied in specific areas, such as water quality prediction, with only a few implementations in broader processes (Krishnan et al., 2022, p. 5).
In their paper on artificial intelligence (AI) in urban water infrastructure Fu et al. (2023) underline the need for greater use of AI technologies in water management to enhance the performance of urban water infrastructures. They point out that current applications are often fragmented and lack systematic integration (Fu et al., 2023, pp. 2–3).

As noted, scientific research in this field shows that digitization and automation are key to addressing the challenges of aging infrastructure and improving water supply resilience (Fu et al., 2023, pp. 2–3). Given the importance of the topic and the utility of AI technologies in water management, this report builds on this background and focuses on the application of artificial intelligence for the calibration of drinking water networks.

1.2 Aim and Objective

This report tries to apply AI in an optimum manner with the goal of optimizing the management of drinking water pipelines, in particular regarding the automation of network calibration, which is a resource and cost intensive procedure. It is divided into two parts: a literature review and a case study in collaboration with

RBS wave GmbH, an engineering company based in Stuttgart, Germany dealing with water distribution network management. The first part informs about the state of the art in research involving AI applications on WDS, particularly related to network calibration. This provides a theoretical background for the second part, in which a feasibility test of applying neural networks to automate the network calibration process will be performed under real conditions. Its goal is to show the reductions in time, labor, and costs that neural networks can achieve in calibration, as opposed to traditional calibration by hand.

The literature review focuses on research done on the use of AI in WDS management, specifically neural networks. More recently, AI and machine learning (ML). ML have attracted a great deal of interest in support solutions concerning the leak detection, the deterioration in pipes, and the calibration of the network. Yet, although there have been numerous reviews on water pipe deterioration, failure risk, and monitoring, there is still a lack of publications strictly devoted to ML and AI application in the management of water systems. Most existing practices still depend on either physical or statistical modeling methods. It is in this regard that Dawood et al. (2020) argue that a literature gap exists in relation to the practical implementation of AI into real-world WDS, which this thesis aims to fill (Dawood et al., 2020, p. 2).

While theoretical progress has been witnessed, the gap between laboratory-developed models of AI and their practical application in reality is still huge. Abdelmageed et al. (2022) note that most AI models have only been tested on synthetic data or within controlled laboratory conditions, and few pieces of research have used real field data collected from actual water systems. This is one of the major reasons why there is a lack of effective, real-world validation, which in turn undermines the reliability of AI models applied to real problems in WDS (Abdelmageed et al., 2022, p. 14). Therefore, aim of this thesis is to use neural networks to show how automation in network calibration can be carried out using real-world data from a municipal water distribution network and adding to the quickly expanding knowledge based on practical AI applications involving the water sector.

The WDS calibration process involves the adjustment of model network parameters in order to arrive at improved alignments between the model predictions and the observed data. These may include pressure drops at several points within the system. Traditionally, this is a rather laborious and iterative process of trial and error where, after every model run, engineers would readjust parameters by hand in order to match the observations. According to Zhao et al. (2022), this is an extremely time-consuming process since adjustments have to be made repeatedly until the desired degree of accuracy has been reached. The convergence often happens very slowly and does not really ensure the best result in those cases where the gaps between the model predictions and real-world data are substantive. This shows how manual calibration is inefficient, especially for large and complex networks (Zhao et al., 2022, p. 2).

The nonlinearities present in the WDS models can be modeled by neural networks, reducing manual interference; hence, the calibration process will also be faster and more reliable. Basically, this thesis will be based on the development of an AI model to automate the calibration process using real-world data from a municipal

water distribution network. That is, this model would utilize real data in an attempt to provide better efficiency and accuracy in calibration.

The second part of the thesis then focuses on the practical application of AI in the context of a case study based on real-world water distribution network data. Nowadays, as mentioned above, the calibration is performed manually by tuning the model to match pressure drops observed at different points within the system. This process is labor-intensive and consumes time that is no longer affordable and hard to find due to the shortage of skilled engineers. Therefore, the company would like to investigate the use of neural networks in order to meet these challenges by automating the calibration process.
The practical part of this paper involves the elaboration and testing of a neural network model that can be trained with data sets previously gathered from the water distribution network of the company. Thus, it might be able to automatically accommodate network parameters according to real-world observations and decrease the calibration time and manpower required. This, in return, would allow the company to utilize its resources much more effectively. The optimal outcome is a decreased reliance on human labor and, consequently, corporate operations with increased efficiency in general.
Additionally, based on the case study, it will be examined how well the neural network model achieves calibration results under real-world conditions.

In this regard, the contribution of this thesis is not limited to updating the academic literature, but is rather intent on finding a practical implementation for AI in WDS. With identifying key gaps in academic literature and introducing a case study on applied AI in the WDS to automize calibration, reduce labor costs and increase operational efficiency, this paper will bridge the gap between academic research on theoretical aspects and practical applications. In this way, it provides insights for both academia and practitioners in the water management sector and aims to foster AI technology adoption.

2 Theoretical Background and Overview

2.1 Definition and Historical Development of Artificial Intelligence and Neural Networks

Artificial Intelligence has a long development history and the sector's growth can be directly related to increases in computing capacity. In general, AI can be understood as a collection of methods enabling machines to carry out typical human tasks. One aspect of AI is ML (Das et al., 2015, pp. 31–32).

ML refers to a collection of methods that find patterns in data and, based on these patterns, make predictions about the future, or help make decisions in case of uncertainty. ML systems develop and evolve through learning over time, as more data becomes available; this makes them dynamic systems. The broad role of ML methods is iterative enhancement in performance through learning, which again is crucial to deal with intricate decision-making processes. A very important notion concerning ML is its relation to Supervised Learning (SL): both are supposed to model and interpret difficult data. SL, coming from statistics, also aims at deciphering complex relationships in data in a manner pretty similar to ML. Both disciplines are adaptive; their models adapt either to new incoming data, or to changing conditions in the environment (Bertolini et al., 2021, p. 2).

In that respect, several learning paradigms have been developed in ML, each suited for a different kind of task. Indeed, SL provides a mapping between input data and desired outputs from labeled training data. It is highly instrumental in applications that depend on accurate predictions based on prior knowledge obtained from experimental or observational data, for instance. Supervised Learning is especially suitable when there is an explicit relation between inputs and outputs that could be modeled (Bertolini et al., 2021, p. 3).

On the contrary, Unsupervised Learning (UL) does not use labeled data, and attempts to draw an unapparent pattern from within. Unlike expected variables of certain outcomes, UL focuses on finding structures or correlations within the data. Commonly, UL is used in clustering, density estimation, and dimensionality reduction tasks that simplify data without prior knowledge of the results to be expected (Murphy, 2012, pp. 9–13).

Another method is Reinforcement Learning (RL), which is unique in that its focus is on learning through interactions with the environment. In RL, an agent performs actions in a given state and receives feedback in the form of rewards or penalties. This feedback helps the agent to adjust its behavior in order to maximize cumulative rewards over time. Whereas SL and UL map inputs to outputs, RL maps states of the environment to actions, making it particularly useful for sequential decision-making tasks (Bertolini et al., 2021, p. 3).

In comparison to conventional statistical methods, particularly AI has different advantages in modeling and analyzing complex nonlinear relationships among variables. Unlike other common statistical models, AI handles the issues of

uncertainty, imprecision, and ambiguity much more effectively and thus offers better solutions to real-world problems. Its flexibility in dealing with these issues makes AI strong across various fields (Dawood et al., 2020, p. 1).

However, due to their advantages, ML models face critical voices. The main criticism of ML models is related to their lack of transparency and quantification of uncertainty. Most of the ML models give results neither providing any clue about the uncertainty related to the predictions made, nor what specific inputs are contributing most to the results. This lack of interpretability might especially be problematic in applications that require accountability and trust. Besides, ML-based models are mostly narrow, performing well in the domain of training, but failing when it comes to generalization in newer datasets (Garzón et al., 2022, p. 17).

The introduction of Artificial Neural Networks (ANN) was one of the big bounds in AI. From the original proposition of McCulloch and Pitts in 1943, ANNs have evolved into much more sophisticated systems that can learn complex patterns from an enormous set of data. Each ANN consists of hierarchical layers of interconnected neurons, which feed their computed outputs into subsequent layers. A neuron (also called a perceptron in early neural networks) takes multiple input values, applies weights to them, sums them up, and passes the result through an activation function to produce an output. With increased computational resources, more sophisticated architectures were invented: Convolutional Neural Networks (CNNs), Long Short-Term Memory Networks (LSTM networks), Graph Neural Networks (GNNs), autoencoders, and Deep Reinforcement Learning models. These architectures have revolutionized many fields, including image recognition, natural language processing, and time series analysis (Abdelmageed et al., 2022, p. 9; McCulloch & Pitts, 1943, p. 99) .

A review of neural network architectures is offered by Fu et al. (2022), including details on the purpose of each neural network. For example, multi-layer perceptron networks are supposed to be used with nonlinear activation functions to model complicating relationships, while autoencoders are meant for unattended compression and reconstruction of data.
Figure 1 illustrates the structure of a typical multi-layer perceptron (MLP) as used in this thesis. It shows a fully connected network with an input layer, two hidden layers, and an output layer. Each node in one layer is connected to every node in the subsequent layer. The additional bias units are also included, represented by nodes with constant input (1) at each layer.
Such multi-layer perceptrons form the basis of many modern neural network architectures and are especially suitable for supervised learning tasks where complex, nonlinear mappings between inputs and outputs need to be learned.

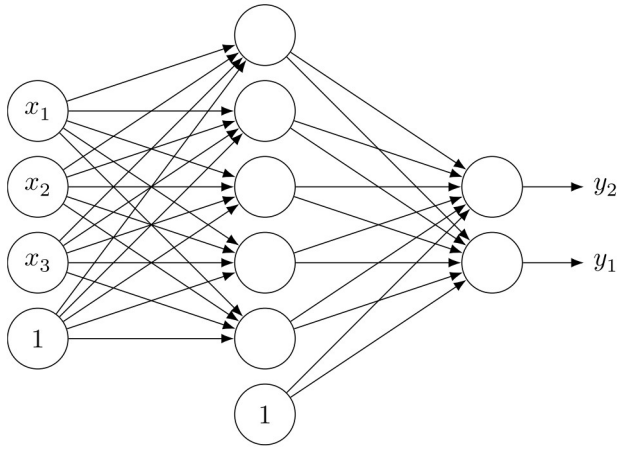

Figure 1: Structure of a two-layer perceptron with bias units (Mitschke, 2022, p. 14)

LSTM networks then are more suitable for tasks involving memory of past inputs—for example, learning long-term dependencies in time series. CNNs excel in recognizing features of images through convolutional and pooling layers. Deep Reinforcement Learning (DRL) combines deep learning with reinforcement learning to let an agent optimize decision making by interacting with the environment. GNNs power applications that have graph-structured data where the relations between the nodes are important. All in all, specialized architectures have come to enable AI to solve such complex problems with increased precision and speed. Given the great improvements in deep learning based on ANN, AI is performing outstandingly in applications initially thought to be very difficult or even impossible.

This includes applications such as autonomous driving systems and strategic games like AlphaGo, a computer program developed by DeepMind that defeated world champions in the board game Go. AlphaGo combines deep neural networks with a Monte Carlo tree search and is considered one of the most impressive demonstrations of the capabilities of deep learning (Silver et al., 2016, pp. 487–488). These systems take advantage of more complex architectures to achieve performances even superior to human capabilities (Fu et al., 2022, pp. 3–4).

Their versatility is exercised in many fields, water quality modeling being one of them. There are several advantages to this approach: ANNs do not require physics-based algorithms, which accelerates model construction and provides more flexibility. Besides, it is able to model nonlinear relationships, and the incorporation of user expertise into the modeling process is allowed (O'Reilly et al., 2018, p. 2). These benefits make ANNs very valuable in complex modeling tasks and thus relevant for different industries.

Implementation of ANNs and other ML models does not, however, remain without challenges. After training, most of the models are narrowly specialized, hence limiting their flexibility when applied to new or evolutionary data patterns. In addition, the lack of transparency of ML models leads to difficulties in interpreting the results, especially in those applications where reliability and interpretability are indispensable. In the recent past, the growth of enabling technologies like sensors, open-source software, public datasets, and cloud computing has accelerated the adoption of AI and ML. These have become widely available and

more affordable; hence, they have enabled many industries to apply AI in their businesses. Moreover, various governments have been encouraging investments in AI through incentives and funding and, in this way, foster the growth of AI-driven innovations (Garzón et al., 2022, p. 17).

Over the last decades, there has been a sea change in AI and its subfields, especially ML and ANNs. This has been facilitated by an increasing computational power combined with the development of specialized architectures, which have helped AI to solve complex tasks in various industries. Although challenges such as transparency and specialization remain, the continuous improvement in these models and the growth of enabling technologies surely promise even greater developments ahead (Bertolini et al., 2021, p. 2).

2.2 Overview of Hydraulic Modeling, Network Calibration, and the Impact of Roughness on Pipe Systems

The accuracy of hydraulic models is crucial for the management of drinking water networks. Uncertainties in water demand and infrastructure conditions increase the complexity of drinking water systems, making accurate predictions essential. Calibration improves the accuracy of the models and adjusts the model parameters to the actual measured data. This process minimizes the differences between measured and simulated data (e.g. pressure or flow) (Meirelles et al., 2017, p. 4339) (Zanfei et al., 2023, p. 1). In general, a WDS consists of nodes, pipes, reservoirs, tanks, pumps, consumers and many other components. Good calibration ensures high reliability of the components and, consequently, the networks. (Ashraf et al., 2024, p. 21905)

In their beginnings, the first approaches pursued a manual strategy. With the help of hydrant tests, very high flows were generated artificially in the networks. Such high loads lead to higher pressure differences in the networks, making the difference between measured and calculated data more visible. The next step in development was the introduction of non-linear optimization algorithms for calibrations - a substantial progress (Wéber & Hős, 2020, p. 2).

Walski 2000 emphasizes that calibrations tested under average demand conditions may not reveal errors that arise in scenarios with very high flow rates and thus result in very high-pressure losses. According to him, exact calibration is not possible due to a number of uncertain factors and can only ever be an approximation (Walski, 2000, p. 94).

All in all, calibration methods can be divided into three categories. Trial-and-error methods, explicit methods, and optimization techniques. Trial-and-error methods are iterative approaches that update the parameters, re-simulate the networks, and then repeatedly show the differences between the models and the real data. Explicit methods solve advanced steady-state equations using available measurement data. However, these methods require very high computational

resources. Optimization techniques use algorithms to estimate network parameters that minimize the differences between the models and the real measurements (Zanfei et al., 2023, p. 2).

2.3 Digitization in the Water Sector

Among other advantages, digitization holds enormous potential for efficiency enhancement, sustainable development, and resiliency in the sector of water supply. Advanced technologies, such as the Internet of Things and AI, make it possible to monitor and manage water resources much more precisely than before. This has great implications given the challenges of climate change and rapid urbanization (Daniel et al., 2023, p. 2).

The main factor driving digitization processes in water industry is a growing demand to reach further efficiency in the consumption of resources while water losses should be reduced. In this scenario, IoT and AI provide sophisticated tools for monitoring and control of water supply systems. Energy consumption would decrease while workflows would be optimized correspondingly (Dada et al., 2024, p. 1374).
These aspects, among others, have been the driving force behind the ever-increasing adoption of digital technologies in the water sector due to, in most part, economic benefits, regulatory pressures, and hydrological constraints. However, there can be a number of challenges, including high capital investment at the outset that is beyond reach for small-scale water suppliers. Moreover, since digitization greatly enhances computerized capabilities, it makes networks prone to various severe cyberattacks, such that stringent IT security frameworks must be deployed (Adedeji et al., 2022, p. 4).

Digitization within the sector of water management leverages a range of technologies, such as Advanced Metering Infrastructure (AMI), IoT devices, and AI-driven analytics. These technologies facilitate detailed assessments of water consumption patterns, enabling both optimized usage and increased operational efficiency (Gohil et al., 2021, p. 64085).
An essential conceptual framework that represents the comprehensive digitization of water infrastructure is Water 4.0. This paradigm aims to improve the operational efficiency of WDS through the implementation of cyber-physical systems as well as information and communication technologies (ICT) that enable real-time monitoring and automated control (Adedeji et al., 2022, p. 4).

The integration of data from multiple case studies has provided valuable insights into urban water management. Empirical data from various regions support the fact that sensor implementation in wastewater infrastructure coupled with real-time data analytics has increased both water quality and responsiveness during flooding.
A particularly relevant example shows how the implementation of real-time control systems with integrated sensor networks and data analytics can greatly improve urban water quality. This achievement emphasizes the importance of

collaboration between different stakeholders in achieving digital transformation within the water sector (Schellart et al., 2021, pp. 417–418).

All in all, it can be said that digitization holds much promise for efficiency, sustainability, and resilience to climate change in the water sector. It can address serious technical, financial, and organizational barriers. A successful digitization strategy will therefore bring technological innovation and, additionally, motivate public and private stakeholders to collaborate on solutions to the important issues of long-term security and reliability of water supply systems (Daniel et al., 2023, p. 6).

3 Literature Review - Artificial Intelligence in Water Pipe Management

For this thesis, an extensive literature review has been performed, which shall shed light upon the current state of research in the field of drinking water supply by means of AI. In order to provide a solid basis for implementation in practice in the field of network calibration, this review is to point out the various areas where AI can be used with special focus on the AI methods that have already been applied.

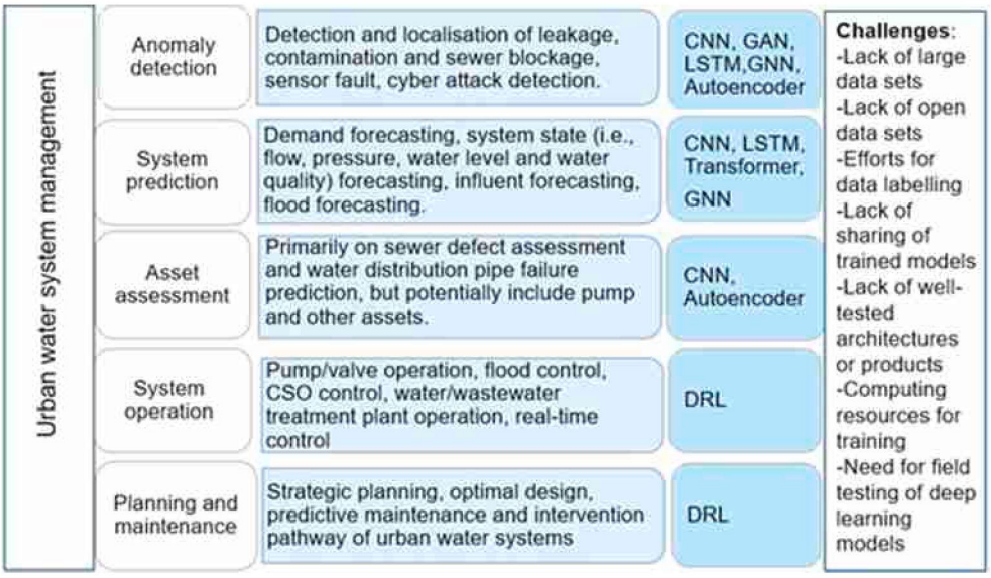

Figure 2: Key application areas in urban water management and relevant architectures of deep learning (Fu et al., 2022, p. 3)

Some of these different application areas of AI in the management of urban water are represented above in a structured form in Figure 2. Anomaly detection, system prediction, asset assessment, system operation, and planning and maintenance stand out among the primary application areas. The various AI methods that have been applied respectively include Convolutional Neural Networks (CNN), Generative Adversarial Networks (GAN), LSTM networks, GNNs, and Deep Reinforcement Learning (DRL). The diagram also underlines some key challenges, such as large datasets, which are scarcely available; open datasets, which are rarely provided; the labeling of data being a very tedious process; field testing; and high computational power also being required for training. Areas where AI has already been successfully applied in drinking water supply and which hold potential for future developments will be discussed more extensively later on.
The following review chapters aim to provide the result of what can be called a systematic investigation into challenges and opportunities arising from the application of AI in water management.

3.1 Identification Approaches used in Scientific Literature to analyze System Failures in Drinking Water Pipelines

The use of AI to identify system failures in drinking water networks is becoming increasingly important. In particular, the prediction of pipe bursts taking into account various influencing factors such as pipe age, pipe diameter and pipe material shows promising approaches. Dawood et al. (2020), for example, describe a model that was created from input data such as the age of the pipes, the diameter and length of the pipes, and the annual breakage rates as target values. The model learns to recognize the relationships between the data and the output values in order to make predictions for new data samples.
One implementation of such models is the use of ANN, as used in a study in a Polish city. The multilayer perceptron algorithm (MLP) was implemented to predict the failure rates of house connection and distribution lines. The input variables were the material, length, diameter and year of installation of the pipes, while the output values represented the failure rate. The best model performance was achieved after 39 learning cycles for house connections and 118 learning cycles for distribution lines. The correlation between the actual and predicted data in the learning process was acceptable, with a correlation coefficient of $R^2 = 0.95$ (Dawood et al., 2020, p. 5). This shows that AI models are able to make accurate predictions if the input data is sufficiently precise.

Another example of the successful use of AI in the area of drinking water networks is the E.L.M. (Extreme Learning Machine) model developed for the Greater Toronto Area. This three-layer model is able to identify the patterns between the input variables (such as pipe diameter, length, material and previous failures) and the output values (time to next failure). Particularly noteworthy is the high accuracy of the E.L.M. model, which is due to its ability to analyze complex nonlinear relationships in a short time. Furthermore, the model shows a high tolerance to disturbances and uses a simple calculation method because the weighting is calculated in a hidden forward layer, as opposed to the complex backpropagation methods of other models (Dawood et al., 2020, p. 5).
AI models have also been used in the risk analysis of water systems. One example is the application of geographic information systems (GIS) together with ANNs or fuzzy inference systems. A risk map of the main urban water pipes was created for a city in France using an ANN methodology in combination with a GIS (Dawood et al., 2020, p. 7).

3.2 Predictive Maintenance Techniques for Drinking Water Pipeline Management as described in Literature

Academic literature describes approaches that improve the management of piping systems by predicting pipe bursts and thus enabling targeted maintenance measures.
Lijuan et al. (2012) used ANN with a Radial Basis Function (RBF) to predict the year of the next failure of drinking water pipes. This prediction can help to target future on-site inspections more easily and thus make maintenance work more

efficient (Lijuan et al., 2012, p. 453). The ability to accurately predict the timing of a pipe burst is a great advantage because inspections and repairs can be carried out on demand, optimizing both the lifespan of the pipes and the use of resources. Another approach was investigated by Sattar et al. (2019), who applied an Extreme Learning Machine Neural Network (ELM-ANN) for predicting burst pipes. This approach exploits the ability of ELM-ANN to respond quickly and accurately to large amounts of data and to make predictions based on historical pipe bursts (Sattar et al., 2019, p. 167).
Snider and McBean (2020) compared the performance of a Gradient Boosted Tree (GBT) algorithm with a survival analysis-based Weibull Proportional Hazard Model (Weibull-PH) for predicting pipe bursts. They used a 45-year data set to train the model and tested it with a 10-year data set. The results show that both models are able to predict pipe bursts with high accuracy, with each model showing different strengths in terms of failure probability analysis (Snider & McBean, 2020, pp. 6–7).
The AI-based methods described in research literature thus offer promising approaches for predictive maintenance management in drinking water networks. In particular, the ability to make predictions about future pipe bursts based on historical data enables managers to take proactive maintenance measures and thus increase the reliability and longevity of water systems.

3.3 Methods and Technologies for Leak Detection in Drinking Water Pipelines

Leakage detection in drinking water networks is a major challenge in water management, as water losses not only cause financial damage but also have a significant ecological impact. In many cases, a substantial portion of the leaked water infiltrates the wastewater pipe system, which can disrupt processes at wastewater treatment plants and lead to increased pollutant discharge into receiving water bodies (Sola et al., 2021, pp. 3–4). However, the identification and localization of leaks is extremely difficult due to the complex structure of water networks, a limited number of installed sensors, and noisy measurement data (Mücke et al., 2023, p. 1).
The most common traditional techniques include acoustic signal analysis, minimum night flow monitoring, and calculation of the water balance in specific network sectors. These methods are widely used, but significant investment is required to implement them efficiently. Moreover, noisy measurement data pose an extra challenge for these systems (Capelo et al., 2021, pp. 1–3).

However, current trends are moving towards AI methods. These technologies offer the possibility of processing large amounts of data and recognizing patterns, which makes them superior to conventional methods (Abdelmageed et al., 2022, p. 2). One important approach is the classification of leaks. The processing of acoustic emission data with ANN has proven to be particularly effective here. Such systems learn to recognize vibrations caused by pressure drops in defective pipes and use this information to distinguish between normal and abnormal conditions. For example, acoustic signals generated by leaks can be processed using the Fast

Fourier Transform or wavelet transforms to improve analysis accuracy (J. Kang et al., 2018, p. 4288; Nam et al., 2021, pp. 3484–3485).

In addition to classification-based approaches, there are prediction-based models based on historical data to predict future leaks. A successful example is the use of LSTM (Long Short-Term Memory) models, which are used to predict water demand and to accurately detect leaks (Fu et al., 2022, p. 5).
Another group of sensors for AI-assisted systems are flow meters, which detect unusual amounts of water and thus indicate a possible leak (Abdelmageed et al., 2022, p. 4). Such sensor data is of central importance for the AI-supported localization and sizing of leaks (Abdelmageed et al., 2022, p. 6). A promising method for leak detection is the combination of different sensor types. Pressure sensors are installed in a network area called the District Metering Area, or DMA, whose data are then processed by AI models to reveal potential leaks. These are then followed by acoustic sensors such as noise loggers or hydrophones, to localize the actual leaks (Abdelmageed et al., 2022, p. 15).

Hybrid AI models that combine different algorithms are particularly powerful. One approach uses support vector machines (SVM), decision trees and Naive Bayes in a joint model. This hybrid model was able to achieve a recognition accuracy of 98.25% in tests (Dawood et al., 2020, p. 9).

3.4 Water Quality Prediction Models with Artificial Intelligence

AI-enabled approaches can detect potential contamination at an early stage. ANN and similar methods have mostly been targeted at finding hidden patterns in historical data that may go undetected by traditional statistical techniques (O'Reilly et al., 2018, p. 2). Dawood et al. (2020) describe a model that, with data collected over seven years, reveals an average performance of 91% in validation. Thus, such models enable water utilities to identify any probable hazard point(s) within their drinking water network well in advance and to take proper action before the system deteriorates (Dawood et al., 2020, p. 7).
Of course, besides predicting contamination, the ANN technique can also be applied to model water quality parameters like residual chlorine and ammonia in chlorinated water systems. In Australia, for example, a Generalized Regression Neural Network was used to serve this purpose for a chlorinated water system situated east of Perth. In this GRNN study, while residual chlorine could be predicted successfully, the parameter of free ammonia was indicated only poorly. This inaccuracy led to wrong input values during the training phase of the model, which generated noise in the data and, in turn, affected the results negatively.
According to Dawood et al. (2020), another problem with the performance related to predicting water quality issues is the imbalance within the data. Specifically, in most circumstances, abnormalities such as contamination barely occur, whereas data from water sensors are quite variable and dynamic. However, training models to detect such rare events reliably is difficult (Dawood et al., 2020, p. 7).
One way out is using long short-term memory-LSTM models trained with a fixed rate of positive samples at 10% per batch of training data. The model with the best

performance, compared to standard LSTM and other machine learning methods, was then selected for anomaly detection in large-scale water datasets (Qian et al., 2020, pp. 4–5). It is likely that ANNs will enhance the processes involved in water treatment through making predictions about the effect of deteriorating water quality on the different water treatment steps (Najah et al., 2013, pp. 199–200; Veerapaneni et al., 2010, pp. 42–43). This might allow sanitation to identify critical parameters and process steps that require closer monitoring or adjustment (O'Reilly et al., 2018, p. 2). In turn, this will help to establish more efficient processes for maintaining consistent water quality, as this capability can reveal certain correlations hidden in data.
Overall, the cases presented here suggest that AI has proved a very powerful tool for water quality modeling. They can allow both the short-term fluctuations and long-term trends in water quality parameters to be identified and, hence, allow for more proactive management of water supplies. Such models can be of particular benefit where the interdependencies between particular parameters are not as yet well understood but are able to identify historical trends and infer likely future development (O'Reilly et al., 2018, p. 2).

3.5 Calibration of Water Distribution Systems

Calibration of drinking water systems is normally a requirement for obtaining reliable simulation results from hydraulic models. This often consists of an iterative procedure whereby model simulation results are compared to field observations and model parameters are changed in an attempt to obtain acceptable agreement between the predicted and measured values of a particular parameter. Some of the important parameters that need to be considered during calibration include pipe roughness, water demand in the nodes, and operational status consisting of pipes, pumps, and valves. Pipe roughness is among the most influential parameters on system uncertainty, although it is cumbersome to measure and to estimate accurately.
This is usually done both manually and automatically, each tactic having considerable advantages and difficulties. Manual calibration is usually performed in a step-by-step process where the model parameters are iteratively adjusted in accordance with the measured values. In any case, this approach is time-consuming and at least some expertise is needed; engineers or modelers sometimes need several years of experience before reliable results can be obtained. Automatic calibration, on the other hand, relies on optimization algorithms to facilitate the process and secure higher accuracy (Do et al., 2016, p. 2; Kang & Lansey, 2010, p. 28; Zhang et al., 2018, 1, 10).

An interesting recent development in the area of automatic calibration involves the incorporation of ANN. For example, Meirelles et al. 2017 present an approach whereby the pressures at the nodes are estimated by using an ANN that was provided with monitoring measurements. The generated estimates then become the input to an optimization routine, seeking to minimize the differences between the calculated and the measured pressures with the help of PSO. This hybrid approach enhances the calibration accuracy by reducing the degrees of freedom of the problem.

However, this method also has certain risks. With an excellent calibration process, errors in the ANN model can spread easily, thus increasing the discrepancies at some nodes. Nonetheless it is important to note that, despite its limitation, this approach is still an important improvement in calibration techniques, especially for networks that have scarce monitoring data (Meirelles et al., 2017, p. 4349).

The most general categorization of the automatic calibration method can be an optimization-based and numerical approach. Optimization-based methods comprising those based on evolutionary algorithms minimize discrepancies between simulated and observed hydraulic parameters through variation in the values of variables affecting pipe roughness and nodal demands. While these methods are indeed powerful, they can also become computationally intensive for large-scale systems. Numerical methods have the additional advantage of being able to calibrate several parameters simultaneously such as pipe roughness coefficients and nodal demands in a more efficient and less complex way. Zhang et al. (2018) propose a numerical approach wherein the calibration objective function can be posed as an equation of least squares that allows determination of the parameters by efficiently iterating forward in updates until convergence criteria are satisfied. In their study, this method worked well: a model-simulated value was almost in accordance with the field observation (Zhang et al., 2018, 1–2, 10).
While powerful, optimization techniques intrinsically are not AI in the larger sense. Of course, by integrating ANNs into network calibration, a wholly new level of sophistication is achieved and can thus perhaps be considered a form of AI. ANNs allow the model to learn from early data while optimization algorithms tune the parameters to increase the overall accuracy and efficiency of the calibration process. As shown in the work of Meirelles et al. (2017), this approach opens a pathway to achieve significant improvements, although careful training of the ANN model remains important in order to avoid compilation issues in error values in the results, and to ensure reliability.
In general, model calibration of drinking water networks is performed in both manual and automatic ways. In particular, recent hybrid ANN methods combined with optimization methodologies such as Particle Swarm Optimization (PSO), an algorithm inspired by the social behavior of bird flocks or fish schools, have shown great potential to enhance the accuracy and efficiency of calibration efforts. Such approaches still require the utmost care about data collection and model training to afford acceptable error rates in network calibration. It remains to be seen whether manual, optimization-based, or numerical methods are to be used by taking into account the demands of the network and the availability of data or computational resources.

3.6 Digital Twins with Artificial Intelligence

Recently, the so-called Digital Twin concept has gained much attention with regard to monitoring and managing drinking water networks. As Bonilla et al. (2022) show, gapless monitoring of WDS remains a very challenging task; however, it is one of the necessary measures to keep the systems working (Bonilla et al., 2022, p. 1). This is where the concept of the digital twin plays a role. It

facilitates the virtual representation of the physical system throughout its lifecycle, drawing upon real-time data for better understanding of the system and further supporting learning processes and informed decisions (Fu et al., 2022, p. 12).

According to IBM, the digital twin is not only a virtual representation of the physical system that is continuously updated via real-time data streams, but it can also be coupled with mathematical models that can either be physically based, data-driven, or hybrid. The most interesting aspect however is that these models are able to reflect in real time the current state of the actual system. In addition, the digital twin can predict future states of the water system and allow for the analysis of "what-if" scenarios to review potential solutions. Consequently, a central characteristic of the digital twin is that strategies identified on it can be applied directly to the physical system—be it in planning, maintenance, or operation (Bonilla et al., 2022, p. 10; Fu et al., 2022, p. 12; Tao et al., 2019, pp. 2405–2406).
Real-time data from sensor networks integrated into the mathematical models of the digital twin allow for an accurate mapping of the current state of the system and therefore facilitating well-founded decisions. Moreover, the development of additional models is based on the methods of AI, such as Convolutional Neural Networks and Long Short-Term Memory-LSTM networks. They understand and predict the behavior of systems (Bonilla et al., 2022, p. 6; Tao et al., 2019, pp. 2407–2409). Hence, a digital twin enables autonomous decisions regarding identification and implementation of optimal interventions within the physical system, integrating said algorithms into deep reinforcement learning (Fu et al., 2022, p. 12; Grieves & Vickers, 2017, pp. 92–101).

This improvement, however, includes big challenges. Designing highly intelligent and autonomous digital twins demands a large amount of intensive, long-term research and development work in many disciplines. These are challenges that need to be overcome if the full potential of this technology is to be realized and they are to become an integral part of system monitoring and control (Fu et al., 2022, pp. 12–14; Grieves & Vickers, 2017, pp. 100–102; Tao et al., 2019, pp. 2408–2409).

3.7 Estimation of Values in Water Pipe Networks

Estimation of values within water pipe networks is a fundamental challenge in water supply system management. A real water network may contain up to thousands of nodes that represent points of water intake, customer connections, and joints of pipes. At the same time, due to infrastructural limitations and cost-related constraints, only some of these nodes are equipped with sensors and regularly maintained. This creates the need to integrate more data and sensors in order to train a high-quality pressure estimation model. Particularly, during the training process, a model with, for example, deep learning architecture - learns the estimation of the pressure in unknown nodes driven exclusively by measurements provided by only a limited number of sensors (Truong et al., 2023, p. 2).

One of various promising approaches working with water distribution networks is the estimation of pressure using GNNs. GNNs are appropriate approaches in this context since the water supply systems can be modeled as graphs. In this way, the networks under observation can utilize structural and topological information of graphs to model the relationships between nodes and make predictions. This approach can, for example, provide an estimate using early data of the pressure in nodes within the water network, where the density of measurement data is very low in order to achieve better management of water supply systems (Z. Wu et al., 2021, p. 4).
Following the results of research conducted by Hajgató et al. (2021) it is evident that GNNs created on the basis of spectral filtering methods are able to restore the nodal pressures in a few water supply systems with an error rate of at most 5%, given that the observation fraction is at least 5%. Moreover, the results prove that GNNs surpass traditional approaches such as interpolated regularization. They are also satisfactory in the case of shallow neural network architectures, provided the number of filters per layer is big enough to model the underlying physics of the water network (Hajgató et al., 2021, p. 10).
The continuous integration of real time data and the generalization capability of GNNs will allow the pressure estimation models to be applied for different water networks even when the topology of the network was not fully observed during training. This also means that the model can be applied to new scenarios and network topologies with very minimal reconfiguration (Truong et al., 2023, p. 5).
Consequently, GNNs will have a great impact on the estimation of values in water pipe networks and, therefore, will allow for finer monitoring and control of the network even when limited sensor data is available. Continuous model parameter optimization and incorporation of additional sensor data can further enhance the estimates, thus supporting efficient management of water supply systems.

3.8 Water Demand Prediction with Artificial Intelligence

Forecasting water consumption is an essential task in water supply management. Accurate demand predictions enable water utilities to operate more efficiently and to optimize pumping operations at lower costs. While classical state-of-the-art forecasting approaches like ARIMA models have given good results until recently, they are limited by nonlinear water consumption patterns. Especially AI and ML have presently served as promising alternatives for surmounting such challenges. Some of them will be presented in the following.

Gated Recurrent Unit (GRU) networks appeared to gain prominence in recent times - especially thanks to their capability of capturing temporal dependencies in the data. One of the key merits of GRU networks lies in their ability to reduce cumulative prediction errors. Studies prove that, at extreme points, the introduction of virtual data points to reduce nonlinearity can result in an error rate increase of up to 30%, though this causes an increase in the training time. In this model, new features are used by unsupervised classification methods to further improve the accuracy of the prediction (Salloom et al., 2021, pp. 8–10).

Long Short-Term Memory (LSTM) networks store relevant information over long periods of time, making it possible to make long-term predictions. A study showed that LSTM models are particularly suitable for short-term demand forecasts where the forecast period is less than one day (Fu et al., 2022, p. 4).

Another significant example is the application of LSTM models in smaller water utilities that have limited access to early data. Here it was shown that LSTM models working in an online learning setting can provide reliable predictions even with a short training phase. This method offers significant advantages, especially for small utilities with limited resources (Kühnert et al., 2021, p. 2).

Combining Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks can use attention mechanisms to capture both spatial and temporal features of the data and ignore irrelevant information. This approach improves prediction accuracy, especially in urban water supply systems that need to account for a large number of variables (Zhou et al., 2022, 5-6, 13).

There have already been successful applications of ANN and SVR models in water demand forecasts, especially for hourly predictions. Amongst others, a study undertaken by Herrera et al. (2010) indicates that ANN and SVR models outperform the per statistics traditional model ARIMA in terms of better accuracy in the prediction of water demand. This is particularly true when many data sources are considered, such as weather and consumption data, in order to create a reliable forecast (Herrera et al., 2010, pp. 146–148).

Thus, it can be said that non-linear prediction techniques have been found to hold useful for water supply systems. Especially the new AI methods pointed out for water demand forecasting, such as GRU networks, hybrid CNN-LSTM models, and neural networks, raise the accuracy of forecasting substantially due to improved efficiency. In this way, it is easier for utilities to plan the operation of their systems, and, in this context, to rule out or expect changes in demand due to sudden events.

4 Methodology

4.1 Description of the Drinking Water Network

The case presented in this thesis relates to the network of a medium-sized municipality located in south-western Germany near the city of Heidelberg.
The network supplies approximately 1,755 inhabitants and several small businesses. All in all, the network consists of 360 pipelines connected by 342 nodes. The total volume of the network is 129.4 m^3, referring to the water currently held in the system. On average, 8,6 m^3/h of water flows in the network to maintain pressure and secure the supply (Anonymized Municipality 2024).

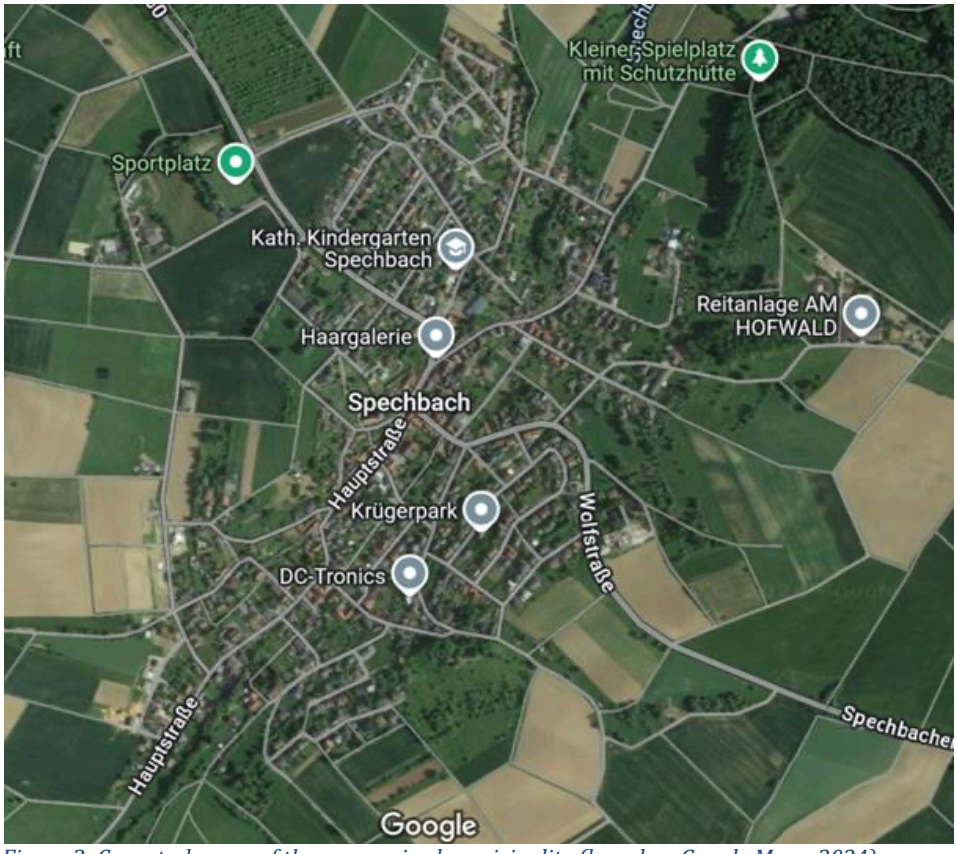

Figure 3: Case study area of the anonymized municipality (based on Google Maps, 2024)

The satellite picture in Figure 3 shows the geographical distribution of the city, which corresponds with the drinking water network. It has a rather elongated shape from north-east to south-west; the high point of the network is in the western, the low point in the southern area.

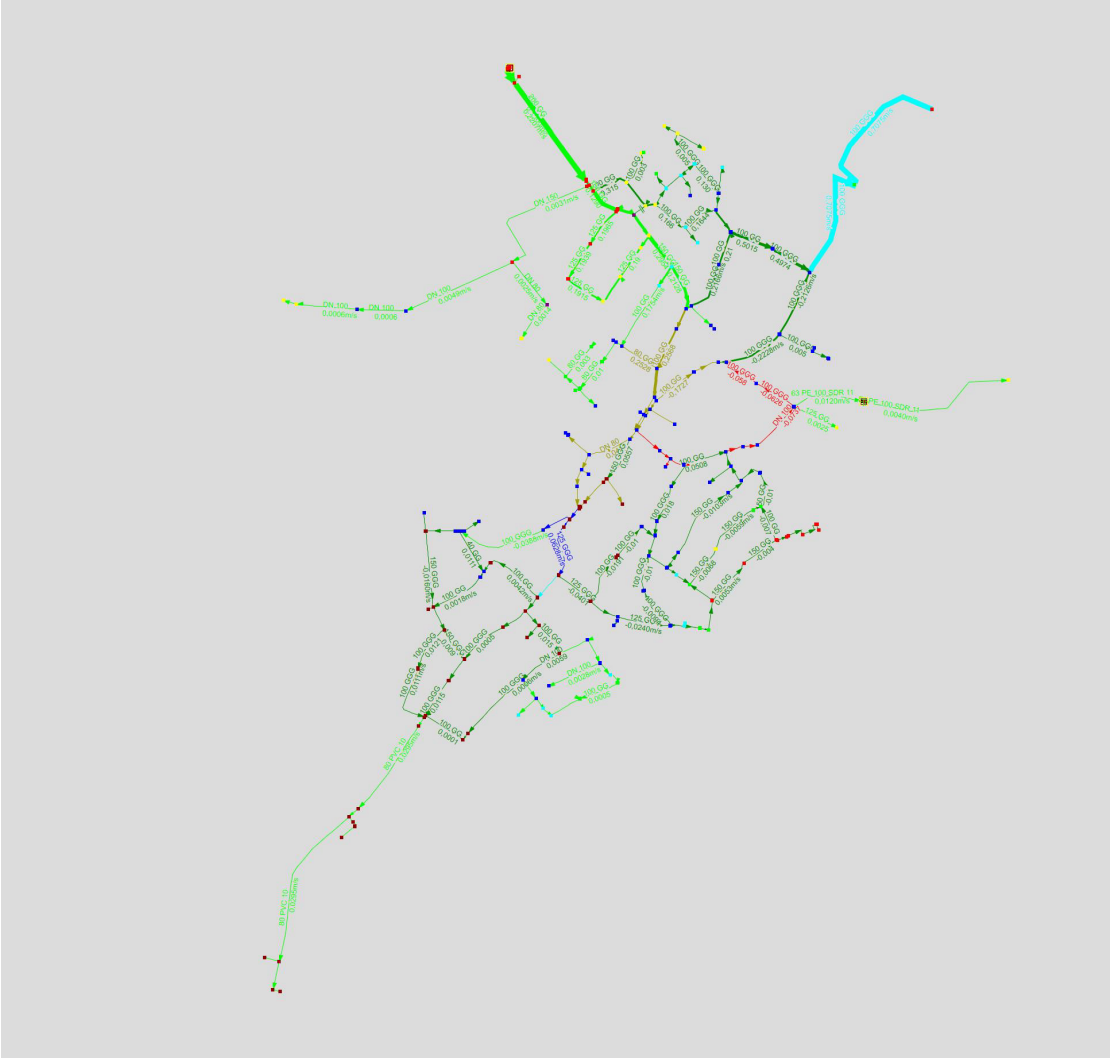

Figure 4: Pipe network of the anonymized municipality (RBS wave GmbH, 2024)

Figure 4 shows the pipe network map of the city, created with the software STANET. The thickness of the pipes reflects their flow rate. This figure depicts the water flows recorded during a measurement with an artificially high load at the north-eastern end of the network. With undertaking measurements during such extraordinary conditions, the pressure differences in the network can be recorded more easily.

The figures shown originate from internal analyses by RBS wave GmbH and therefore cannot be verified by public sources. The author assures their accuracy. Likewise, their accuracy is not of the utmost importance for the actual analysis, so their verification is not of central relevance.

4.2 Current Calibration Process at RBS wave GmbH

The calibration of drinking water networks is an important process to ensure that hydraulic models deliver accurate simulation results. At the engineering office, this process is carried out in several iterative steps and is performed manually.

The variable unit during calibration is the roughness of the pipes. It is important to note that valves in the network can also be inadvertently closed. Since most consumers have a dual-feed supply for redundancy, this can stay undetected for some time. The loop connections assure that all customers receive water even in the event of an accidentally closed valve. Thus, the error may not be readily evident. Yet, a closed valve can still significantly affect the hydraulic behavior of the network. Thus, the possible event of undetected closed shut-off valves should always be kept in mind during calibration.
First, a model of the pipe network in question is created by the STANET software based on the customer's data. This model already contains the correct lengths, diameters and geometry of the pipes as well as the number and arrangement of pumps, pressure regulators and valves. The consumption data such as the water withdrawals per household are also transferred to the model. Since the aim of the calibration is to find out the specific roughness values of the pipes, no real roughness values can be included in the model. Pipe roughness changes over time due to deposits, which is why the standard values for new pipes can be used as initial values, but do not represent reality (Do et al., 2016, pp. 1–3). However, these initial values can be used for first calculations in the model to calculate the nominal condition of pipe pressure.
Then, in order to collect real data, pressure loggers are installed at various nodes in the pipe network to record the pressure curve in the pipe. An artificial load is then generated at defined points in the network by high water withdrawals. Walski (1983) already described this method of using high flow rates for data collection during network calibrations. Due to these especially high flow rates, the difference in pressure is very high, so the results obtained are more solid and measurement errors have less impact (Walski, 1983, pp. 362–364, 2000, 95, 97). By evaluating the data gathered by the pressure loggers, the different pressures at the nodes - with and without load - can be specified and compared.
In the model, the pressures without and with load are calculated under the same conditions and with the same water flow rates, but with the standard roughness. Since the standard values do not reflect the actual roughness, the calculated pressures differ from the real pressures. By adjusting the roughness values in the model, the calculated pressure differences from the model are iteratively approximated to the measured pressure differences. This method makes it possible to adapt the model in such a way that it reflects the actual conditions in the network and indirectly enables the determination of pipe roughness.
It is important to note, however, that the topographical data (elevation) of the individual nodes cannot be assumed to be exact. This is also reflected in the hydraulic head, which combines elevation and pressure. Since absolute pressure varies with elevation, pressure differences are used instead for calibration. To avoid this source of error, the calibration is not conducted with the absolute pressures, but with the pressure differences.

The drinking water network model can be regarded as sufficiently calibrated if it meets all the accuracy standards as stipulated by the German Association for Gas and Water. Following them, two most important requirements that are believed to verify the accuracy of the model are (Deutsche Vereinigung des Gas- und Wasserfachs e.V., 2006, p. 16):

- The discrepancy between measured and calculated pressure differences (dp) at any node of the network should be limited to 2% of the hydrostatic pressure of the node, not more than a maximum of 0.2 bar.

- The average deviation U should be within a defined range which may be given by $a/\sqrt{n}$, where 'a' is permissible deviation and 'n' is the total number of pressure loggers. Thus, the range may be mathematically expressed as: $-a/\sqrt{n} \leq U \leq +a/\sqrt{n}$. U represents the average deviation between the measured and simulated pressure differences across all nodes. It quantifies the overall fit of the model.

If both of these criteria are met, the model can be considered calibrated within the permissible deviation limits. If these conditions are not fulfilled, or if there is suspicion of hidden critical issues, further research on the network into non-discerned system faults may be needed.

4.3 Computing Environment and Software Versions

In general, AI applications require huge computational resources, especially within the hydraulic modeling domain. The architecture of most modern GPUs is designed to provide a great deal of the required processing. Among the major suppliers of the currently most high-performance GPUs, NVIDIA can be considered as the best in the industry because of its advanced hardware architecture and comprehensive support of AI frameworks (Boutros et al., 2020, p. 17; Nikolic et al., 2022, p. 6).

At the beginning of the project for this thesis, an office PC with an Intel i5 processor and 8 GB RAM was used. Although sufficient for the initial development of AI models, this very quickly turned out to be sparse for the growing computational needs of this kind of applications. From early on, the search for improved computational capacity has been pursued, and an collaboration has been established with RBS wave GmbH, Chalmers University of Technology, and the University of Stuttgart. Even though this collaborative network seemed promising, owing to issues of data protection, the idea of using cloud-based solutions for the computations was rejected. For similar reasons, neither of the two universities' computing facilities were utilized.
In September 2024, a high-performance PC was put into operation, configured especially for AI computation. The machine is equipped with an Intel i9 processor of 14th Generation, with 64 GB of RAM, and an NVIDIA GPU with 12 GB of memory. This computer is located in Stuttgart and will be reached by remote access over Citrix Workspace, allowing the flexible but secure use of the computer's computational capacity.

In order to use the NVIDIA graphics card for training the AIs, additional drivers had to be installed: Compute Unified Device Architecture (CUDA) and CUDA Deep Neural Network Libery (cuDNN). CUDA is required for the parallel execution of

processes on the graphics card, while cuDNN enables convolution, pooling and normalization of individual training processes.
Moreover, two additional packages were integrated into the Python environment: PyTorch and TensorFlow. In order to perform the calculations on the graphics card, it was necessary to coordinate both the different installations and their versions, as well as the different drivers.

Once the Python-based frameworks came into play, including PyTorch and TensorFlow, it was crucial to align versions with CUDA and cuDNN to ensure compatibility and stability.
After a thorough analysis of configurations, the final setup of the system was configured to run Python 3.11, CUDA 11.8, and cuDNN 8.6 using matching versions of TensorFlow (2.17.0). This was the best current combination of technologies that met the project's computational requirements in terms of robustness and stability for the development and execution of models.

STANET, a software developed by Fischer-Uhrig, an engineering consultancy based in Berlin, Germany, specializing in hydraulic modeling software, was used for the calculation of the hydraulic conditions. For many years, STANET has been the standard tool used by the company for analyzing pipe networks and hydraulic computations. In this project, the software version 10.3.31 was used, complemented by an extended license which allowed operating STANET via external programs.
This functionality is very important for creating synthetic data sets, which are the backbone of AI-driven analysis and thus allowing smooth and seamless interaction between AI models and hydraulic simulations. The extended license facilitates automating data generation and manipulation, one of the key needs for the AI framework in order to efficiently carry out large-scale simulations.

Applying these computational facilities and settings has helped the project establish an adequate infrastructure that could address various challenging tasks related to AI-based hydraulic modeling. The selected system components and software match each other in such a way to maximize computational efficiency while ensuring the protection of data, processed according to institutional policies.

4.4 Synthetic Data

4.4.1 Use of Synthetic Data in Optimization and Calibration of Drinking Water Networks: Opportunities, Challenges, and Research Perspectives

Using synthetic data is an already established practice in research on how to optimize the management of drinking water networks. It is a possible alternative for when there is a lack of or difficult access to real data sets: For example, one of the key problems arising for researchers when developing models able to predict network conditions and enabling leak detection by calibration of the network parameters is a lack of data. Especially at the beginning of research, this is a

considerable problem they are facing. Here, synthetic data could be of great help (Abdelmageed et al., 2022, p. 6).

It can be concluded that synthetic data is most valuable for and central to training, model validation, and model evaluation. Synthetic datasets can be created with hydraulic simulations or by computer-based modeling using software such as EPANET. Simulated datasets have been used in various scientific studies to support AI model development and training. Accordingly, Abdelmageed et al. (2022) identified that 27 out of 60 considered studies actually made use of synthetic data while training, testing, and validation of models based on AI, underlining their relevance and wide usage within scientific research (Abdelmageed et al., 2022, p. 6).

However, using synthetic data is not without challenges. Garzón et al. (2022) mention that synthetic data cannot represent all processes contained in real data. For example, certain complex interactions and stochastic processes that can occur in a real dataset are difficult to generate. These limitations could affect the generalizability and validity of those models developed with synthetic datasets (Garzón et al., 2022, p. 15).

In this context, Truong et al. (2023) add that the assumption of similar distributions between the training and evaluation datasets may not always be true in practical scenarios. In practice, this ideal case rarely happens, which makes the estimation of a model more difficult regarding its reliability. Nevertheless, working with synthetic data has become an essential tool to support deep learning model development considerably (Truong et al., 2023, p. 3).

Fu et al. (2022) underline the fact that such an approach allows for the exploitation of high-resolution models developed during the last decades in the water sector for generating large data sets to train deep learning models. This can increase the size of the training datasets considerably and, at the same time, can reduce the risks concerning data security, since no real, sensitive data is required for the process (Fu et al., 2022, p. 10).

Another strong motivation for shifting towards the use of synthetic data is that, in most cases, real data sets are incomplete. Capelo et al. (2021) state that a complete characterization of pipe bursts in real water supply systems is seldom possible due to either non-existent integrated data management systems, or the fact that the installed sensors are not enough to cover the entire network. Often, the only remaining option is an artificial generation of such data using a robust and well-calibrated network model (Capelo et al., 2021, p. 3).

Henceforth, synthetic data has become an essential tool for meeting the challenge of data availability in water network research. Based on the scientific findings presented in the preceding chapters, the following sections of this thesis present the development of a code for generating synthetic data sets for the purpose of water network calibration. In this context, the goal is to set up a database as comprehensive and realistic as possible, forming the basis for further development of the code concerning the calibration with respect to roughness.

4.4.2 Description of the code

To generate training data for the AI models, a Python-based code was developed to create synthetic data sets from a calibrated water network in STANET. The goal was to simulate different network conditions for both error detection and roughness prediction. The process includes two variants: one simulating network errors (e.g., closed gate valves) and another with randomized roughness values.
The code is structured in three modules: the first selects positions for gate valve closures or roughness variation; the second modifies the input file accordingly; and the third performs hydraulic simulations in STANET and exports the results. For each case, data is generated both with and without artificial network load to reflect real calibration scenarios. This ensures that the models are trained on realistic pressure differences.
The overall process is fully automated and modular. It uses Python's subprocess to run STANET externally and generate a large number of synthetic data sets efficiently. A more detailed description of the code structure and logic can be found in Appendix A: Technical Implementation of the Synthetic Training Data Generator.

4.5 Pipe Failure Prediction

4.5.1 Problem Definition and Objective

In the process of network calibration in terms of roughness, the initial challenge is to identify potential structural or hydraulic errors within the network itself.
An operational failure in the form of a pipe closure within a network that remains unnoticed is known as a network error. These errors can occur when valves that were used to disconnect a subnetwork are not reopened once construction work has been completed. It is quite common for parts of a network to be isolated during construction or maintenance activities to ensure safe and efficient working. This isolation is typically done by closing valves to cut off the flow of water to specific areas. However, once the work is completed, these valves must be reopened to restore normal network operations.
Unintentional closures can occur due to human error, such as forgetting to reopen the valve, miscommunication between workers, or incorrect documentation of the valve status.
Sometimes, a given valve might appear to be opened when it is still partly or completely shut, especially for old or unmaintained valves. Mechanical failure or problems of sediment accumulation in the valves contributes to incomplete re-opening of these valves. When not detected, all these closed valves create what is called network error. As networks are invariably designed with redundancies, there are usually two routes for the water to reach most consumers. Thus, if a pipe or part of the network is inadvertently closed, the supply of water can often be maintained via an alternative route. For this reason, these conditions may not be immediately apparent. Still, the network performance as a whole will be negatively affected (Berardi et al., 2014, p. 2588; Liu et al., 2017, p. 4).
Consequences include increased pressure drops, nonuniform flow distribution, and higher energy consumption by pumps, which work harder to maintain

adequate pressure throughout the network. This reduced efficiency can also lead to an increased risk of pipe bursts or leaks elsewhere in the network due to uneven distribution of pressure. Furthermore, consumers may experience lower water pressure or variability in supply, especially in peak demand situations. Such issues can be avoided by regular inspection, valve maintenance, and proper communication among the maintenance teams.

Given that a network exhibiting structural or hydraulic errors behaves significantly differently from a correctly configured system, the initial step in the calibration process is to determine whether such errors are present. The detection of network errors represents an important preliminary task in the overall calibration workflow and can provide valuable insights for operational decision-making. In order to meet these requirements, the present work employs a two-pronged approach. In the initial (and subsequent) phase, an algorithm is developed and described that focuses on identifying network faults.

4.5.2 Descriptions of Modifications to the Real-World Model

The drinking water network that is object to this thesis has been simplified as much as possible to be still close to reality for scientific research. A number of changes have been made compared to the original network. One of the major simplifications is the removal of the private pressure boosting system. This system was originally designed to serve different altitudes. However, it is excluded in this work because it would complicate hydraulic behavior unnecessarily without adding value to the research questions. The intention was to make the model easy to work with, but not to lose the basics of hydraulics. Another adjustment that was made is that the lines in front of containers were removed, as they didn't make a great difference in hydraulic modeling. They connected only the storage tanks themselves without influencing supply security or pressure conditions elsewhere in the network. Removing them made the network simpler yet retained the relevant aspects of water supply.
With this approach, a simplified model could be created that is still representative of the drinking water network. In fact, all the fundamental hydraulic properties were maintained while the model complexity was reduced from an analytical point of view. This will strongly reduce variables and will simplify the running of simulations - a great advantage for modeling and calibration.

4.5.3 Data Preprocessing

To ensure reliable training of the fault prediction model, a comprehensive data preprocessing pipeline was developed. This process involved cleaning inconsistent simulation data, extracting node and pipe features, and constructing a graph representation of the water network enriched with physical and spatial attributes. Feature scaling and positional encoding were applied to improve model learning. The full methodology is described in Appendix B: Data Preprocessing for Graph-Based Fault Prediction. .

4.5.4 Neural Network Model Design

4.5.4.1 Model Selection

GNNs were chosen because they are a good application for pipe networks. Specifically, GATs are highly suitable for the simulation and prediction of data in pipe networks due to their ability to effectively model complex spatial relationships and dynamic interactions within graph-structured data. Drinking water networks can be naturally represented as graphs, where nodes represent junctions or other key elements, and edges represent pipes connecting these elements. GATs leverage this graph structure to provide a highly effective predictive modeling approach.

GATs excel at representing the topological complexity of water networks. Pipe networks, such as those in drinking water systems, are inherently complex due to their topological structure. Traditional ML models struggle to capture the dependencies between nodes (e.g., junctions) and edges (e.g., pipes), whereas GATs can model these relationships directly. The attention mechanism in GATs assigns different levels of importance to different nodes, allowing the model to learn which connections within the network are most critical for prediction tasks. According to Fu et al. (2023), GNNs are particularly effective for managing complex water distribution networks, as they maintain good hydraulic performance and are resilient to emergencies (Fu et al., 2023, pp. 2–3).
This ability to differentiate between the importance of nodes and edges is crucial when predicting faults or changes in a WDS. For example, a particular node's pressure might depend more heavily on one neighboring node than on others. GATs can learn these nuanced relationships through their attention mechanism, making them exceptionally well-suited for modeling the physical interactions present in pipe networks. Truong et al. (2023) also highlight that GNNs effectively leverage graph topology to estimate pressure in WDS with a significant reduction in prediction error compared to traditional models (Truong et al., 2023, pp. 4–5).

One of the most significant advantages of GATs is their use of attention mechanisms to learn spatial relationships. In the context of a pipe network, not all nodes contribute equally to the conditions at a particular point in the system. GATs apply attention weights to different nodes, allowing the model to focus on the most influential ones for a certain prediction task.
For instance, when predicting the probability of a pipe failure, certain features such as flow rate or pressure from neighboring nodes may be more important than others. GATs dynamically adjust these weights during training, which improves the model's ability to learn complex dependencies within the network. This focused attention allows for more accurate and interpretable predictions compared to other GNNs that treat all proximal nodes equally. Ashraf et al. (2024) demonstrate that GNNs using physics-informed learning can emulate hydraulic simulators with high accuracy, making them suitable for complex infrastructure like WDS (Ashraf et al., 2024, p. 21911).

Drinking water networks span large geographic areas, and the geospatial positioning of nodes and edges is crucial for understanding the behavior of the

system. GATs are capable of integrating geospatial information directly into the model through node and edge attributes, enabling the model to use this information in its predictions.
The use of positional encodings allows GATs to incorporate spatial coordinates (`XRECHTS`, `YHOCH`), which enhances the model's understanding of geographic relationships within the network. This is particularly important for predicting spatially correlated events, such as pressure drops or flow changes, that depend on the physical layout of the network.
The scalability of GATs makes them well-suited for large drinking water networks that may include thousands of nodes and edges. By processing parallelly and using simple graph representations, GATs can efficiently scale to accommodate large datasets without a significant increase in computational costs.
Additionally, GATs can be adapted to include new nodes or edges as the water network expands. This adaptability is key in real-world scenarios where infrastructure is continuously evolving. GATs are inherently designed to handle these incremental changes, allowing for continuous learning and improving the model's robustness over time.

The primary goal of predictive modeling in drinking water networks is to identify potential faults, such as leaks, pressure anomalies, or pipe bursts. GATs excel at this due to their ability to learn complex interactions within the network. By considering both direct and indirect influences between nodes, GATs can identify patterns that may indicate an impending fault. For example, a sudden drop in pressure at a node might be influenced by changes several nodes away. Traditional models may miss such dependencies, but GATs can capture them effectively through multi-hop attention mechanisms. This makes GATs particularly powerful for detecting anomalies, as they can identify subtle changes in network behavior that may be precursors to larger issues. Savic (2019) points out that GNNs can be calibrated to closely match real-world hydraulic conditions, which is essential for accurate fault detection and network analysis (Savic, 2019, p. 12).

Thus, it can be concluded that GATs offer a powerful approach for modeling and predicting data in drinking water networks. This is particularly due to their ability to capture complex spatial relationships, handle irregular and dynamic graph structures, and integrate geospatial data effectively. Their scalability and adaptability make them ideal for real-world applications, where the network structure is subject to continuous changes. With attention mechanisms, GATs can focus on the most relevant parts of the network, leading to more accurate and insightful predictions, which is crucial for maintaining the reliability and efficiency of drinking water systems.

4.5.4.2 Model Architecture

The GAT model described here predicts conditions in drinking water pipe networks by analyzing node and edge features to detect faults such as closed pipes. It visualizes the network as a graph, nodes and edges representing different components and their connections. By using attention mechanisms, it assigns

varying levels of importance to nodes, enabling the model to focus on the most critical relationships. It is particularly designed to identify failures that affect the network's functionality.

The key components of the GAT Model are:

- Graph Attention Convolution Layers (GATConv): These layers propagate and refine node information to improve prediction accuracy. The first layer identifies important neighboring nodes, and the subsequent layers enhance feature complexity.
- Batch Normalization: This is applied after the first two GATConv layers to stabilize training and ensure consistent feature scaling—essential for graph networks, where feature distributions can vary widely.
- Dropout: Dropout layers are included to prevent overfitting, making the model more robust when dealing with new data.
- Edge Attribute Processing: Edge features are processed using a multi-layer perceptron (MLP) to create embeddings. These are then combined with node embeddings from the connected nodes.
- Fully Connected Layer for Edge Prediction: This layer analyzes combined node and edge features to estimate the likelihood of faults for each edge.

The forward pass of the GAT model involves several steps:

- Node Feature Propagation: Node features are passed through GATConv layers to aggregate information from neighboring nodes. These features are updated based on attention weights that indicate their individual relevance.
- Edge Feature Processing: The edge MLP processes edge features to produce embeddings.
- Combining Node and Edge Features: Node embeddings and edge representations are connected, creating a comprehensive feature set that describes both the nodes and their connections.
- Edge Prediction: The combined features are passed through a fully connected layer, which predicts faults on the edges.

By making use of attention mechanisms, GATConv layers, and edge attribute processing, this model provides an accurate representation of interactions within a drinking water network. Techniques like dropout, batch normalization, and adaptive optimizers enhance its reliability, ensuring that it performs well in detecting and predicting network faults.

4.5.5 Training

Training of the GAT model is a very important step to ensure high-quality predictions of network faults in drinking water pipe systems. This section

describes the setup for training, the splitting of data, and other challenges that were faced during the process.
The dataset used in this project was divided into training (60%), validation (20%), and testing (20%) sets (Zanfei et al., 2022, p. 8). This split ensured a balanced approach to training the model while maintaining suitable data for validation and testing, allowing for proper evaluation of model performance during training.
In machine learning, one of the tasks is to come up with training and testing datasets that are representative of the conditions that the model is going to face during real-world applications. Ideally, both the training and testing datasets should represent these conditions. However, this over-similarity between training and testing due to identical simulation settings could lead to overfitting. As Truong et al. (2023) point out, by generating the training and testing datasets with identical simulation settings, generalization by the model is compromised. The model might only learn to reconstruct previously demonstrated conditions. However, when the algorithm enters the application phase to make predictions for real-world data, the accuracy of these predictions is insufficient. This is because the model is overly trained on the dynamics and characteristics of synthetic datasets, leaving it inadequately trained to handle the significantly more complex and diverse real-world data. (Truong et al., 2023, pp. 2–3).

The training procedure for the GAT model used for this thesis was a binary classification setup where the model was trained on network faults, specifically at instances of flow and velocity equal to zero, which correspond to closed pipes. This training procedure involved validation techniques:

1. Optimizer: Since gradients were sparse, the optimizer used was AdamW. It adapts the learning rates for every parameter so that the training becomes stable.
2. Loss Function: A class-weighted binary cross-entropy loss function (BCEWithLogitsLoss) was used here in order to handle the imbalanced data. More precisely, one positive weight (pos_weight) weighted the pipe closure scenarios due to their scarcity (Garzón et al., 2022, pp. 10–12).
3. Learning Rate Scheduler: The ReduceLROnPlateau scheduler resolves the learning rate with respect to the validation loss w.r.t convergence and without overshooting the optimum solution.
4. Early Stopping: Early stopping was introduced to prevent overfitting. It works by stopping the training in case the loss on its validation does not improve for several consecutive cycles. This helps saving unnecessary computation and keeping the model from memorizing the training data.

One of the problems that were faced during the training process was class imbalance. Since pipe closure rarely happens, the dataset contained far fewer positive examples than negative ones. To handle the issue of class imbalance, a weighted loss function was used that allows a higher penalty on misclassified positive examples and encourages the model toward correctly identifying closed pipe conditions.

4.5.6 Testing

Aside from training, testing is another important process in the GAT model development. The purpose of testing is to evaluate the model's performance on hidden data, ensuring its robustness and ability to generalize effectively beyond the datasets used for training and validation.

The model was tested on a test dataset sized 20 % of the total data that was kept back in training and validation. This test case could be applied to real-world scenarios because many examples included in this dataset would be faced by the model for the very first time.

The evaluation metrics used to assess the GAT model's performance included accuracy, ROC AUC score, precision-recall curve, and confusion matrix. Each metric provided a different perspective on the model's ability to predict pipeline faults accurately and reliably.

- **Accuracy**: It is the proportion of correct predictions made by the model. It gives a general view of the performance of the model, but may not be adequate in the case of an imbalanced dataset.
- **ROC AUC Score**: The Receiver Operating Characteristic – Area Under the Curve (ROC AUC) score represents the ability of the model to differentiate between faults and non-faults. The higher the score, the better the discrimination between the positive and negative classes. The ROC AUC score summarizes the model's ability to distinguish between fault and non-fault cases across different classification thresholds.
- **Precision-Recall Curve**: The precision-recall curve, along with the average precision score, gives insight into how well the model performs at correctly predicting the positive class, i.e. fault conditions. This metric is highly relevant in the case of imbalanced datasets where the instances of the positive class are less frequent.
- **Confusion Matrix**: A confusion matrix further specifies the model's predictions into true positives, false positives, true negatives, and false negatives to give detailed insight into the types of errors the model is making, and where improvements should be made.

During the testing phase of the case study for this thesis, the test dataset was put into the already-trained GAT model to predict each sample. Then, the predicted results were collected to measure the performance against the actual labels of the samples. The classification of output was done by applying a sigmoid activation function to the logits. Then, thresholding was applied to these continuous outputs to convert them into binary predictions that showed the presence or absence of a pipeline fault.

4.6 Roughness Prediction

4.6.1 Problem Definition and Objective

The goal of this chapter is to propose a procedure for estimating pipe roughness in the drinking water network, considered an important parameter for hydraulic model calibration, which seeks to attain accuracy in the simulation of the network. Precise determination of pipe roughness reduces discrepancies between measured and simulated pressures, hence resulting in better calibration and operational efficiency. The objective is to evaluate the performance of three machine learning models for this task, ANN, XGBoost, and GAT. Since objectives for this section somewhat matched those of Section 4.5, Pipe Failure Prediction, extra effort has been done to avoid redundancies so that this would fall on the roughness prediction while Section 4.5 addresses concerns of network fault detection.

4.6.2 Descriptions of Modifications to the real-world Model

In the initial attempts, a drinking water network was used for analysis as described in Section 4.5.2. However, initial analyses of the results showed that the prediction quality for all networks was not satisfactory. Therefore, the network was simplified step by step until it was possible to make predictions for roughness using algorithms.
The results presented in the roughness prediction section are based on a very simplified network, which barely resembles the original network. Only the reservoir and the main supply line were retained. The networks investigated are now very simple examples of drinking water networks, increasing in complexity to demonstrate the limits of the methodology regarding its application.
The networks consist of a progressively larger number of pipes (5, 20, 50, 100, and 200), as well as an increasing number of loops in the networks (0, 1, and 2).

4.6.3 Data Preprocessing

In this section, the data cleaning process is revisited first. Then, follows an explanation of the data preprocessing for the GAT algorithm, which is very similar to the one described in Section 4.5.3 Following that, the data preprocessing for the ANN and XGBoost algorithms will be discussed, as it differs from that for the GAT. The procedures for data cleaning are identical to the approach described in Section 4.5.3.

The data preprocessing for the GAT is similar to the procedure described in Section 4.5.3. However, there are certain differences. Building upon the previously described preprocessing workflow for training a GAT model on drinking water networks, the procedures for the roughness prediction code differ in several aspects while still retaining many of the core approaches outlined in the first

description. Therefore, the following part will not come back to the contents that remain unchanged, but focus on the differences.

While both codes apply feature scaling, the second implementation places greater emphasis on ensuring consistent scaling across multiple datasets by deriving scaling parameters from a broader training data base. These are collected through matching node and edge files from a specified directory. The scaling parameters that are obtained hereby are then saved in order to be reused in both the training and inference stages. The difference between this code and the first one is that, while the first code uses a specific file for the scaling adjustments, this code uses a complete set of available training data. The reasoning behind such a choice is that, in order to predict the pipe roughness, a broader understanding of the whole dataset is required to capture the realistic variability in the pipe conditions.

In terms of graph representation, the core idea remains the same: nodes represent junctions or endpoints, and edges pipes. The handling of positional encoding is similar, too, utilizing sine and cosine transformations to help the model recognize spatial relationships between nodes. Then, positional encodings are injected to make the spatial attributes of nodes properly represented. This careful inclusion of geographical data enhances the model's ability to comprehend variation in space in a more nuanced way. This second version also allows for dynamic splitting of the loaded datasets into training, validation, and test sets. Previously having been set-in-stone, the method get_loaders() now allows the split to be dependent on a ratio chosen by the user, adding flexibility at the time of dataset usage. Such splitting ensures that the training, validation, and test sets are indeed representative of the whole dataset, adding to the generalization capability of the model. This includes explicit consistency checks of the features as well as feature scaling, on top of just the presence of positive examples in this second version, making it much more robust concerning training the model.

The ANN and XGBoost algorithms take a different route for preprocessing compared to the GAT model.
The FeatureEngineer class handles the entire feature engineering process: first, it combines multiple datasets into one combined DataFrame. The combined data provide a better overview of the network and allow both the ANN and XGBoost models to be effectively trained.
After combining, polynomial features are added using the PolynomialFeatures transformer with a degree of 2. This improves the model's ability to capture nonlinear relationships between features. It is essential to save the transformer on the hard drive to ensure consistency in case the same transformation needs to be applied to new data during prediction.
Following the application of the PolynomialFeatures transformer, i.e. the addition of polynomial features, the data is split into a division of 80/20 for training and testing, respectively. This approach ensures that the models receive enough data to generalize well on hidden data during both training and evaluation.
The next step, feature scaling, uses the StandardScaler, which normalizes features to have a mean of zero and a standard deviation of one. This step is critical for algorithms sensitive to feature scales, such as ANN. The scaler is saved on disk to ensure consistency during the prediction phase.

In brief, the preprocessing that would be carried out on ANN and XGBoost consists of combining the datasets, generating polynomial features, splitting the data into training and testing sets, and scaling the features. Such steps put the data into an optimal form for training and prediction. Thus, both ANN and XGBoost models might learn appropriately from the data and generalize well on possibly new data.

4.6.4 Neural Network Model Design

4.6.4.1 Artificial Neural Network

The ANN model utilized in this work is based on an architecture designed to predict the roughness parameters in drinking water networks. Although there are several other options, a neural network model has been chosen due to its well-adapted characteristics to grasp nonlinear relationships between input features of the pipe network and targeted roughness coefficients. These include several hidden layers that involve batch normalization, activation function, and dropout in each, generally helping to avoid overfitting and to achieve a better generalization of the model. In total, five such hidden layers were used to create highly complex models.
The decision to use neural network architecture stems from its capability to accomplish complex nonlinear relationships that hydraulic systems usually face. These can be visualized, whereby increased complexity in the architecture allows the model to capture deeper features. This, in turn, enhances accuracy in the network calibration.

The architecture of the ANN model in question is based on an input layer that has the same number of neurons as there are features in the dataset. This is completed by five hidden layers with a progressive decrease in neurons, first starting with 256, then 128, 64, 32, and 16 in that order. Each of these hidden layers is further followed by batch normalization and a dropout rate of 30%, which prevents overfitting of the model and ensures that it does not get overly dependent on particular neurons. Finally, the final layer consists of one neuron that uses linear activation predict the roughness value.

The model used was then compiled using the Adam optimizer with a learning rate of 0.001. The loss function chosen for this study is Huber, which is very robust to outliers and perfectly fitted for datasets with big dispersions (*Adam Optimizers*, 2025; Bartz et al., 2023, pp. 64–66). All these measures were undertaken to measure the quality of training. During the process, MAE and MSE were calculated.

In this training, the validation process ends when the validation loss stops improving. This is another approach to save time in training and avoid overfitting. In the training dataset, the data is split into a training set and a validation set, using 20% as a validation set. This model was trained up to 1000 epoch with a batch size of 32, with the training stopping automatically if the validation loss did not improve within 10 successive epochs.

Afterwards, the model was saved on the disk to enable future predictions without needing to train it again. In the proposed methodology, the model was saved and the saving path was dynamically generated based on the project directory. Also, a method was developed which loads the trained model and thus it can be applied easily in real-life. Hence, it can be used efficiently and effectively for roughness prediction in drinking water networks.

4.6.4.2 XGBoost

The XGBoost model presented in this paper is designed with the aim to forecast roughness parameters in drinking water networks. It has been implemented with GPU acceleration in order to enhance its performance. The selected architecture exploits the gradient boosting framework provided by XGBoost that seems particularly effective for the detection of complex patterns in data and is realized with an iterative refinement of weak models. In this section, the design choices, the strategies for hyperparameter tuning, the training process, and the evaluation methodology adopted for this model are explained.

XGBoost is particularly suited to predict values in drinking water networks for several reasons: it is outstandingly capable of handling big datasets and a high number of features.
It can capture complex nonlinear relationships, usually characteristic of water distribution systems, thanks to its gradient boosting approach that combines multiple weak learners into strong predictive models. Moreover, XGBoost applies regularization techniques—specifically L1 (Lasso) and L2 (Ridge) regularization—which help to prevent overfitting by penalizing overly complex models. L1 encourages sparsity by driving less important feature weights to zero, while L2 discourages large weights by spreading the influence across features. This is particularly valuable when working with real-life data that may be noisy or highly variable. The algorithm can also handle missing values effectively, which makes it well suited for drinking water networks, where incomplete datasets are a common challenge (*XGBoost Documentation*, 2025).
Moreover, XGBoost makes parallel and distributed computing possible. Hence, it is quite efficient for training with big datasets and especially suitable for this case study. Using GPU acceleration during the training process has further reduced computation time notably. The chosen parameters are a learning rate of 0.05, a maximum depth of 5 for the trees, and both subsampling and column sampling to avoid overfitting. The model architecture is designed to include the tree method 'hist' for faster training and GPU acceleration (CUDA) for further computational gains. This allows XGBoost to be applied to the high-dimensional and possibly sparse dataset used for this application.

The architecture for the XGBoost model was defined by a set of parameters selected in such a way as to optimize performance and efficiency. The model uses an objective function of 'reg:squarederror' that is highly suitable for regression tasks including the prediction of roughness coefficients. Besides, the chosen tree-building method 'hist' is highly efficient and fast in producing optimal splits, making it ideal for high-dimensional data.

In order to improve the model's performance, GPU acceleration is facilitated by using CUDA. Other than methods relying solely on CPU, those with GPU are able to process data faster thanks to parallel processes enabled by the GPU.
The key parameters for the training process include a maximum depth of 5, which helps control the complexity of the trees and prevents overfitting. A learning rate of 0.05 ensures that updates during training are constant, thus allowing the model to converge effectively without overshooting the optimal solution. Also included are 400 estimators to provide sufficient boosting rounds for the model to learn from the data. A subsampling set to 0.9 introduces randomness into the process by using a subset of the data for each boosting round, while a column sampling by tree set to 0.7 guarantees feature variability per tree, further reducing the risk of overfitting.
The parameters listed above were chosen to equilibrate the model's complexity over its ability to generalize and draw results even from new, unseen data.

In general, the training procedure of XGBoost is efficient and robust. Training starts by initializing the model with certain parameters. Further, data is divided into training and validation sets in order to estimate the performance metrics of the model in each training iteration. Acceleration on a GPU was enabled to reduce the training time of the model considerably, hence making it practical even for large-size datasets to be trained within a reasonable time.

The model is fitted by minimizing the mean squared error (MSE), which is a standard loss function in regression problems. During training, it iteratively changes its parameters with the intention to reduce this loss to a minimum level and thus make its predictions more accurate. Secondly, GridSearchCV is used for hyperparameter tuning, which can go through a range of different combinations as a means of identifying an appropriate setting for the model.
It watches metrics such as root mean squared error (RMSE) and R-squared (R^2) through training, which informs us about both magnitudes of error and the capability of the model to capture variance in the target variable. This careful monitoring helps to ensure that the model fits well on the training data and effectively generalizes to new data.

In this study, the model was saved after training in JSON format to be easily reusable and widely deployable. The memory path was determined dynamically with respect to the structure of the project to make it consistent so that it would be easy to load the model for further evaluation or application in real-world scenarios. The model can be reloaded and applied for new data so that predictions may be possible without a need for retraining, thus ensuring practicability of the model.

4.6.4.3 Graph Attention Network

This section builds on the previous use of Graph Attention Networks (GATs) described in Chapter 4.5.4. To avoid redundancy and provide a fresh perspective,

we focus on the similarities and differences between the current GAT model and the one described earlier.

Considering the reasons listed earlier, GAT is adequate and predestined for prediction processes regarding drinking water networks as it makes appropriate usage of topological relations associated with complex relationships that represent, in fact, a complex entity for GNNs. Consider a simplified pattern of a drinking water network represented as a graph. The graph node instances are sensors and joints, while edges stand for the pipe connections of given nodes. It then draws on the attention mechanisms within the network to evaluate neighboring nodes and weigh them differently against those considered critical.
The motivation for using GATs results from their capacity to incorporate geographic information concerning nodes, using positional encoding XRECHTS and YHOCH. This accounting for spatial dependencies would help the present model yield more precise predictions about events where space is considered an important factor, some of them corresponding with pressure drops.

In the following, some of the important innovations in the current GAT model are presented (Brody et al., 2021):

1. Graph-Based Imputation: Contrary to the previous model, this proposes a graph-based imputation methodology that tries to fill in the blanks with the help of surrounding nodes and their already existing relationship, hence improving the quality and making the prediction more robust.
2. Stronger Architecture: The current model uses six layers of GATv2, each with 16 attention heads. Such a heavier architecture makes the model capable of capturing deeper node and edge information and hence captures subtler details regarding the complex interaction between various entities in the network.

The training approach for the current GAT model also includes some differences compared to the earlier setup:

- Objective: While the previous model had a major objective in finding faults such as closed pipes, the objective in this model will lie in the estimation of the roughness of pipes within the drinking water network. Therefore, since the roughness of the pipes is a continuous variable, the handling of node and edge attributes is different.
- Training Methodology: The training is conducted using much more sophisticated optimizers, including a learning rate schedule. Preprocessing techniques include Graph-based imputation and Batch normalization in order to further augment data quality and maintain stability over the model during the process of training. In an effort to avoid overfitting and, in this way, improving generalization, there's a dropout.

4.6.5 Applied Optimization Techniques to Improve Model Performance

To enhance the predictive performance of the models used in this study, several targeted optimization strategies were applied. These improvements were implemented across all three modeling approaches: Artificial Neural Networks (ANN), XGBoost, and Graph Attention Networks (GAT). The aim was to better capture the complex and often nonlinear relationships within the simulated water distribution network data, particularly in the presence of incomplete or noisy datasets.

The ANN was extended beyond a simple baseline configuration. Instead of relying on a single-layer model, a deeper network was constructed by stacking three dense (fully connected) layers. Between these layers, activation functions were used to introduce non-linearity, and dropout was applied to reduce overfitting. Additionally, normalization techniques were used to stabilize and speed up training. Training was carried out with a commonly used optimizer, and early stopping was applied to prevent overtraining. Multiple architectural variants were tested manually. Despite these enhancements, the ANN model remained limited in its ability to generalize, especially in complex network scenarios with loops (Bartz et al., 2023, pp. 64–66; Goodfellow et al., 2016, pp. 21–25).

XGBoost is a gradient boosting algorithm that builds an ensemble of decision trees. In this study, the model was optimized through a comprehensive grid search covering several thousand unique parameter combinations. Parameters such as tree depth, learning rate, number of estimators, and regularization settings were systematically varied. Each combination was evaluated using cross-validation, and the best configuration was selected based on prediction error. XGBoost showed strong performance even under more complex conditions, which made it particularly suitable for modeling pipe roughness (Bartz et al., 2023, pp. 46–49).

The GAT model belongs to the class of graph neural networks, which are designed for learning on structured data represented as graphs. In the context of this study, the water distribution network was modeled as a graph where nodes represent junctions and edges represent pipes. The GAT model used an attention-based architecture with multiple layers and parallel attention heads. The implementation was based on a specialized graph learning library, and feature scaling as well as spatial encoding techniques were applied to improve learning. Although no automated hyperparameter tuning was used, the model was adjusted manually across different configurations. Despite these efforts, the GAT model remained sensitive to network complexity, particularly in scenarios involving loops and dynamic flow path variations (Bartz et al., 2023, pp. 64–66).
In addition to the three main model types, initial experiments were also conducted using a simplified Physics-Informed Neural Network (PINN), which integrates physical equations directly into the learning process (Raissi et al., 2019).. However, due to unstable training behavior and unsatisfactory predictive performance, this approach was not pursued further in the core analysis. The

potential of PINNs remains promising, but further research and methodological refinement would be necessary.

4.6.6 Testing

Testing is a very important procedure that will help to determine how the models perform under real conditions for pipe roughness prediction in drinking water networks. The performance testing methodology of GAT, ANN, and XGB models on unseen data, the metrics used, as well as key findings from these tests are described in this section.

The testing was conducted as follows. First, the dataset was split into a set for training, and one for validation. In total, 20% of the data were defined for testing. The test set remained completely disregarded during training and validation. This ensured that the model's performance based on this data can be taken as an example, or proof, for its generalizability. Moreover, in order to assess the model's ability to predict pipe roughness under changing circumstances, the test data included various different variables. Circumstances the model might have to work with are, among others, flow rates, pressure conditions, and pipe materials.

Several common regression metrics were used to evaluate the models' performance:

- MSE: A metric to measure the average squared differences between actual and predicted roughness values. The lower the MSE, the more accurate the prediction.
- RMSE: Provides a measure of prediction error in the same unit as the target variable, making it easier to interpret. Lower values signify better performance.
- Mean Absolute Error (MAE): Calculates the average absolute differences between predictions and true values, offering an intuitive and less outlier-sensitive error measure than MSE.
- R-squared (R^2): This metric indicates how well the models' predictions agree with actual data variability. An R^2 close to 1 signifies that the model is able to effectively capture the patterns underlying the data.

After training, the GAT, ANN, and XGB models were applied to the test dataset. Additionally, scatter plots comparing actual versus predicted values were generated. These plots offered a visual representation of each model's performance, highlighting differences and patterns that might not be evident through numerical metrics only. The scatter plots further included MSE, RMSE, MAE, and R^2 metrics and were saved for further analysis and documentation.
In order to test their capability to generalize, the models were tested across various scenarios. This involved varying node and edge conditions to simulate a selection of operational states in the water network, such as pressure drops or increased flow rates. The same test conditions were applied to all models to ensure adequate comparison of the results.

4.7 Testing of the code

Testing was an essential part of this work to verify the correct functioning of key components in the data processing and machine learning pipeline. Unit tests were used to evaluate modules for data handling, feature engineering, and model training. The full testing approach and tested functionalities are described in Appendix C: Code Testing and Validation Procedures.

5 Results

5.1 Results of Network Fault Prediction

For the network fault prediction, a binary classification was used. The performance of the model was evaluated with a number of key metrics. Focus was put on accuracy, recall, sensitivity and precision. The findings are as follows:
The model's accuracy was 98%, indicating that it correctly classified most of the data. Also, the ROC-AUC (Receiver Operating Characteristic - Area Under the Curve) score of 0.98 signifies that the model has done a very good job in classifying positive and negative cases. Recall in this model is 81%, while sensitivity is high, meaning that most actual network faults could be found by the model. Precision, on the other hand, is low at 13%, with considerable numbers of false positives.
The following figures bellow illustrate these findings.
Figure 5: Precision-recall curve showing the balance between precision and recall.
The average precision was 0.30, reflecting the low precision of the model, while the high recall reflects that most positive cases were found. The receiver operating characteristic (ROC) curve represents the discriminative capability of the model and is illustrated in Figure 6. The AUC value is 0.98, reflecting its overall strong performance.
Figure 7 shows the confusion matrix in detail, illustrating the model's prediction performance. The upper-left box represents the True Positives (TP) — in this case, 692 correctly identified network faults. The lower-left box shows the False Negatives (FN), which are 158 actual faults the model failed to detect. On the right side, the False Positives (FP) are shown in the upper-right box: 4,748 non-faults incorrectly classified as faults, indicating false alarms. Lastly, the lower-right box displays the True Negatives (TN) — 239,842 correct identifications of non-fault cases. This distribution shows that while the model detects most faults correctly, a relatively high number of false positives still exist.

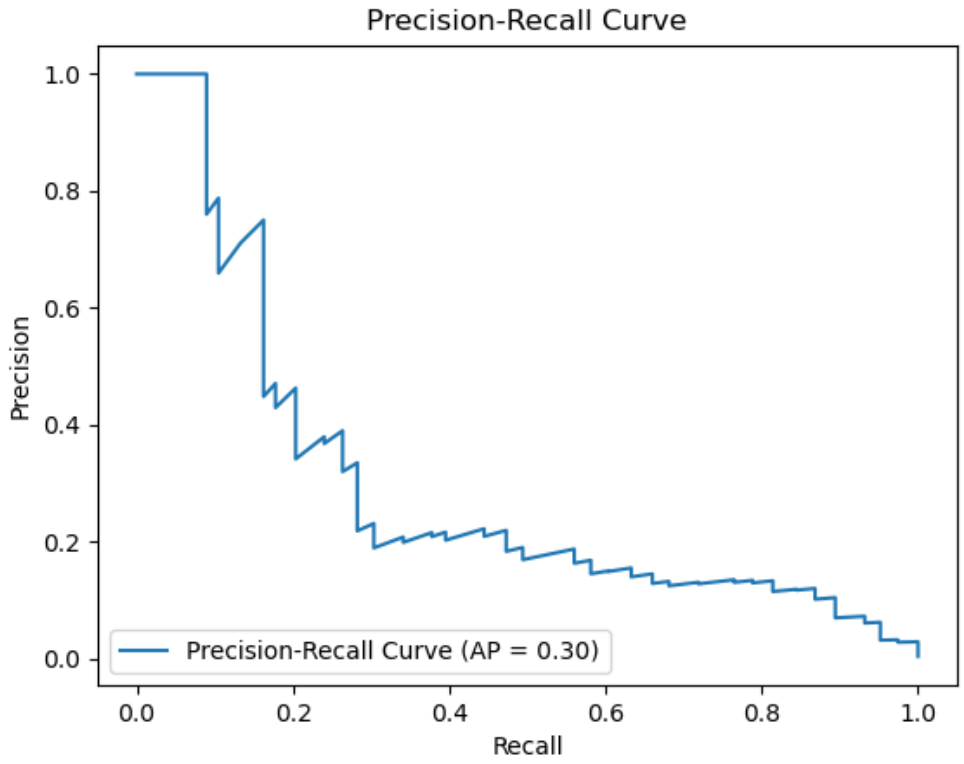

Figure 5: Precision-Recall Curve. Source: Own illustration.

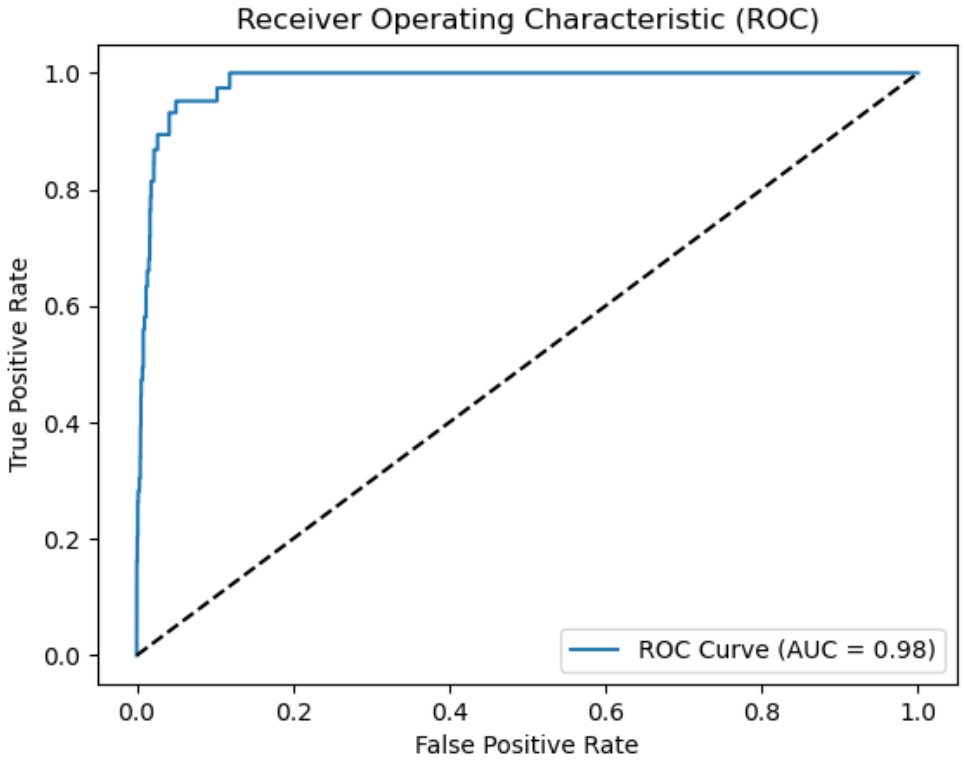

Figure 6: ROC Curve. Source: Own illustration.

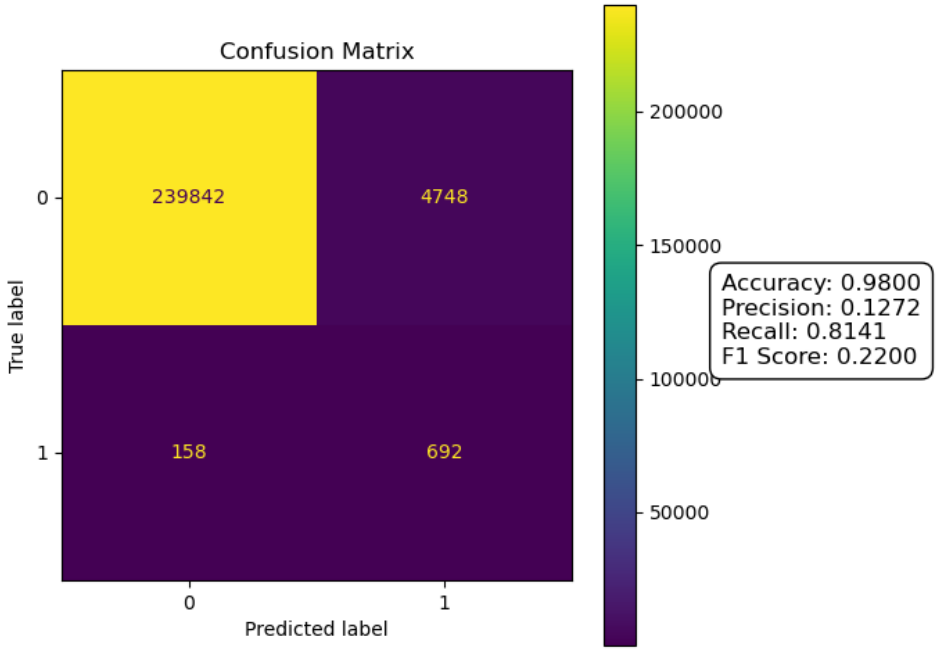

Figure 7: Confusion Matrix: Source: Own illustration.

5.2 Results Roughness Prediction

Additional to network faults, this study was interested in roughness prediction. In chapter 5.2, we will come back to the different models used in the case study and the respective outcomes for roughness prediction. More precisely, the results for the Artificial Neural Network model, the XGBoost, and the Graph Attention Networks model. The graphical representation of the results for the different network configurations and algorithms can be found in Appendices D to F.

Table 1 provides a numerical overview of the R^2 values achieved by the three models (ANN, XGBoost, and GAT) across different network configurations

Model	Pipes	Loops	R^2
ANN	20	0	-593.1866
ANN	20	1	-424.0149
ANN	20	2	-396.8687
ANN	50	0	-394.2446
ANN	50	1	-393.3950
ANN	100	0	-506.2375
GAT	5	0	0.1431
GAT	20	0	0.3376
GAT	20	1	0.3643
GAT	20	2	0.1450
GAT	50	0	0.2339
GAT	50	1	0.0198
GAT	100	0	0.0012
GAT	200	0	0.0003
XGBoost	5	0	0.1440
XGBoost	20	0	0.4501
XGBoost	20	1	0.4638
XGBoost	20	2	0.4291
XGBoost	50	0	0.7629
XGBoost	50	1	0.7227
XGBoost	100	0	0.8693
XGBoost	200	0	0.9284

Figure 8: R^2 values of all three models across different network configurations.

5.2.1 Results of Artificial Neural Network (ANN) - Roughness Prediction

This section will present the results for the roughness prediction using the ANN model. During the evaluation process of this model, a synthetic dataset was used. Furthermore, the predicted results were compared to actual values of roughness. In general, the performance of an ANN model is estimated using the metrics of Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and the coefficient of determination R^2. In order to show the relationship of the predicted values with the true values of pipe roughness for the different synthetic datasets, graphs are plotted. The pipe roughness values range between 0 and 2, which is important for interpreting the MAE.

In the case study, the dataset with 20 pipelines and 0 loops resulted in poor model performance, with a Mean Squared Error (MSE) of 178.34, a Mean Absolute Error (MAE) of 10.32, a Root Mean Squared Error (RMSE) of 13.35, and a coefficient of determination (R^2) of -593.19. A negative R^2 value indicates an extremely poor model fit — worse than simply predicting the mean of the target values. For context, a perfect fit corresponds to $R^2 = 1.0$, while $R^2 = 0$ indicates that the model does not explain any variability in the target variable. The spread of predicted values in this case suggested increasing difficulty in accurately capturing the roughness, implying that the ANN's predictive capability decreases significantly with smaller datasets.

Also, with a synthetic dataset of 50 pipelines and 0 loops, the model did not capture the true values well - as indicated by an MSE of 118.91, MAE of 8.21, RMSE of 10.90, and an R^2 value of -394.24. This negative R^2 value reflects poor predictive power, with the spread of the predicted values suggesting high variance and inability of the ANN to closely align with the actual roughness values.
On the dataset with 100 pipelines and 0 loops, the performance continuously declined: MSE=152.97, MAE=9.39, RMSE=12.37, and R^2=-506.24. That shows that many of the predicted values had significant margins of error, highlighting the model's struggle to learn underlying patterns.
However, with a dataset of 20 pipelines and 1 loop, the model performed marginally better with an MSE of 127.59, an MAE of 9.18, an RMSE of 11.30, and a negative value of R^2 as -424.01. Once again, the poor result indicated a lack of sufficient predictive power, as this model struggled to give trustworthy predictions.
When running the ANN model for 50 pipelines and 1 loop dataset, it showed an MSE of 118.81, MAE of 8.44, RMSE of 10.90, and R^2 value of -393.40. This depicts that the model's prediction is not well correlated with the actual values, and a negative R^2 signifies that the ANN failed to generalize from the dataset.
The last dataset, with 20 pipelines and 2 loops, yielded relatively low error metrics: 118.57 for MSE, 9.20 for MAE, 10.89 for RMSE, and -396.87 for R^2. However, even though the MSE and RMSE are lower here, this negative R^2 suggests that even this model could not describe the variance of the data very well, indicating that roughness was poorly predicted.

The negative values of R^2 that are consistent for all datasets indicate poor generalization of the model and its inability to capture underlying correlations in the data. Furthermore, the relatively high MSE and MAE values show that the model's predictions were rather far from the true values. This can be due to overfitting or underfitting, quality issues in synthetic data, or poor feature engineering. A negative R^2 value also implies that the model performs worse than a simple mean predictor. In some configurations, the R^2 values were strongly negative, which suggests that the model failed to learn any meaningful patterns from the data. These results should be interpreted with caution, as they may also point to instabilities in the training process or insufficient regularization.

5.2.2 Results of the XGBoost

Regarding the XGBoost model, the results analysis was conducted as follows. For evaluating the XGB model, synthetic datasets were used, and the predicted values were compared against the true roughness values. In order to describe the model's performance, several measures were used, including MSE, MAE, RMSE, and R^2. For each synthetic dataset, graphs were generated to visualize the variation of the predicted values against the actual values of the roughness of the pipe. The outcomes per dataset are as follows:

For the dataset with 5 pipelines and 0 loops, the XGB model yielded an MSE of 0.2580, an MAE of 0.4302, an RMSE of 0.5079, and an R^2 value of 0.1440. This

indicates that the model was able to capture certain hidden patterns within the data, but the low R^2 value shows limited predictive capability.
With the dataset containing 20 pipelines and 0 loops, the XGB model showed an improved performance, reflected in an MSE of 0.1651, an MAE of 0.3014, an RMSE of 0.4063, and an R^2 value of 0.4501. The increase in the R^2 value signifies that the model captured more variability in the data compared to the smaller dataset.
For the dataset with 50 pipelines and 0 loops, the model performance improved further, achieving an MSE of 0.0713, an MAE of 0.1651, an RMSE of 0.2671, and an R^2 value of 0.7629. The greater R^2 value indicates that the model explained a significant portion of the variance.
In the testing cycle with the dataset containing 100 pipelines and 0 loops, the model yielded an MSE of 0.0393, an MAE of 0.1104, an RMSE of 0.1983, and an R^2 value of 0.8697. The XGB model performed strongly, capturing the relationships within the data effectively.
For the dataset with 200 pipelines and 0 loops, the XGB model achieved an MSE of 0.0216, an MAE of 0.0803, an RMSE of 0.1468, and an R^2 value of 0.9284. The very high R^2 value indicates that almost all variance in the data could be explained by the model, demonstrating excellent predictive capability with larger data volumes.
With the dataset with 20 pipelines and 1 loop, the model's performance remained consistent, achieving an MSE of 0.1610, an MAE of 0.3021, an RMSE of 0.4012, and an R^2 value of 0.4638. The slight increase in R^2 value indicates that the model made effective use of the additional information provided by the loop.
For the dataset with 50 pipelines and 1 loop, the XGB model achieved an MSE of 0.0835, an MAE of 0.1840, an RMSE of 0.2890, and an R^2 value of 0.7227. While the MSE and RMSE values were slightly higher, the model demonstrated good performance with a high R^2 value.
Finally, the XGB model for the dataset with 20 pipelines and 2 loops yielded an MSE of 0.1701, an MAE of 0.3151, an RMSE of 0.4125, and an R^2 value of 0.4291. Although the R^2 value was slightly lower for this dataset compared to the set with 1 loop, the model generally performed well in capturing most general trends in the data.

Overall, results show that with increasing size of datasets, the performance of the XGB model increased consistently. The progressively increasing R^2 values highlight that, with larger datasets, the model was more capable of generalizing and capturing the underlying relationships in the data compared to smaller datasets. For detailed information on and graphic visualization of the results for each dataset, please see the appendix.

5.2.3 Results of the Graph Attention Networks (GAT)

Finally, the results of GAT for pipe roughness prediction should be presented. As with other models, GAT model was evaluated on synthetic data sets and compared with the actual values of roughness. The key evaluation metrics here are MSE, MAE, RMSE, and R^2. Graphs showing in detail the relationship of true versus predicted values can be found in the appendix. Results per dataset are described in the following.

For the dataset with 5 pipelines and no loops, the GAT model had a mediocre performance. The MSE for the GAT model was 0.2634, MAE equalled 0.4346, RMSE was 0.5132, and R^2 was 0.1431. This suggests that the model succeeds only to a limited degree in correctly predicting the roughness values for this small and relatively simple dataset.
Besides, the performance of the model slightly improved when increasing the dataset size to 20 pipelines without loops. GAT reported an MSE of 0.2014, an MAE of 0.3608, an RMSE of 0.4488, and an R^2 of 0.3376. Hence, it is a best fit, though there is room for improvement.
In the case of a dataset of 50 pipelines without loops, the model further showed continued performance but in a moderate way: MSE = 0.2324, MAE = 0.3981, RMSE = 0.4820, and R^2 = 0.2339. It slightly declined when the size increased to 100 pipelines without loops: MSE = 0.3004, MAE = 0.4744, RMSE = 0.5481, and R^2 about zero at 0.0012. For the largest dataset size, with 200 pipelines and no loops, the performance was comparable. It displayed an MSE of 0.3013, an MAE of 0.4755, an RMSE of 0.5489, and an R^2 of 0.0003, hence showing that the model's performance decreased in capturing the complexity as the dataset size grew.
Furthermore, loops added more complexity to the network, and this had varying effects on the model's performance. For the dataset with 20 pipelines and 1 loop, the GAT model showed an MSE of 0.1914, an MAE of 0.3410, an RMSE of 0.4375, and an R^2 of 0.3643. This slight improvement in performance might indicate that the model was able to handle a small degree of network complexity.
However, when the number of pipelines reached 50 with 1 loop, the performance decreased slightly, giving an MSE of 0.2950, an MAE of 0.4683, an RMSE of 0.5432, and an R^2 of only 0.0198. This could be interpreted as that the model was failing to keep or even increase accuracy as the dataset was getting more complex.

In another round, the GAT model could predict the dataset with 20 pipelines and 2 loops, giving an MSE of 0.2569, an MAE of 0.4170, an RMSE of 0.5068, and an R^2 of 0.1450. It appears that the added loops make it more difficult for the model to achieve a high degree of accuracy.

A complete summary of all evaluation metrics for GAT, XGBoost, and ANN models across different network configurations—including MSE, MAE, RMSE, and R^2—is provided in Appendix H: Overview of Evaluation Metrics for All Models and Network Configurations. This overview complements the graphical figures and offers a concise tabular comparison of model performance.

6 Discussion

The state of research presented here shows an increase in the development of AI in drinking water supply, covering a wide spectrum that includes, among others, anomaly detection, system prediction, and maintenance. However, as things stand now, one may even question whether the AI models applied fully meet the manifold complex requirements demanded in practice. This is because most of the architectures described, like CNN, GAN, LSTM, and GNN, do require large amounts of high-quality data and are mostly unavailable. This is a common bottleneck hindering practical realization. Hence, it is necessary to improve data using methods or strategies other than usual.

Nonetheless, for predicting system failures, AI has shown promising results, particularly using risk factors such as pipe age or material. Still, one may ask whether those models are generalizable across different regions or networks. As an example, the E.L.M. model developed for Toronto showed very good accuracy of prediction, but to what extent this will adapt to other networks is an open issue. This underlines the need to investigate the robustness of the models and their proper adaptation to different network conditions. Predictive maintenance approaches can be enabled using ANNs and ELM-ANNs for possible reductions in unplanned failures.

However, the academic literature shows that these models strongly depend on the quality of historical data. Another aspect is that the comparison of the GBT algorithms and traditional methods shows that the AI model reaches higher accuracy, but then again might be more prone to distortions in data. Therefore, the question arises of how predictive models can be made more robust to provide stable predictions even with incomplete or faulty data. The main problems related to water management concern the detection of leaks. Traditional approaches, based for example on the analysis of acoustic signals, have proven very expensive and error-prone. In fact, AI-based solutions ensure significant advantages in terms of fast and accurate processing; in particular, the use of ANN brings great benefits in classifying leaks. On the other hand, few systems have so far been deployed because of the prohibitive complexity of data processing and the need to install wide sensor infrastructure. This means that the use of hybrid models where traditional methods are combined with those of AI seems to be the most promising approach so far.

Similarly, the ANN and LSTM models for water quality prediction have the ability to recognize hidden patterns in such data, whereby events of potential contamination can be identified. Still, some challenges remain - mainly in forecasting rare events, as the available data used most of the time in developing models is not representative.

This raises the question of how AI models can be prepared in a better way to improve in these kinds of predictions of rare events that are so important. An alternative may be the generation of synthetic data targeted to improve training. Automatic calibration of water networks with the aid of ANNs combined with optimization algorithms may result in a notable increase in accuracy and efficiency in calibration. However, risks such as error propagation due to bad ANN estimates remain that might deteriorate overall calibration. This proves that,

despite progress made in the automation of calibration, careful validation and adaptation of the models by their authors are required to ensure their reliability.
One of the most promising ways to monitor and manage water networks in real time is offered by digital twins like described in chapter 3.6. Generally speaking, only through the combination of detailed physical models with AI is such profound analysis and prediction of system behavior possible. Nevertheless, both the computer performance and the data base for developing such systems are high. What emerges from this discussion is that a completely intelligent and autonomous digital twin requires long-term efforts of research and development in order to realize its full potential.
Estimating values like pressure in water pipe networks with the use of GNNs could prove that AI models surpass traditional models in this field, especially at low data density. Generalization into different topologies of the network is still a challenge, and it is not very clear how these models can be adapted for reliable results with different network structures. Such a prediction of water demand by GRU, LSTM, and hybrid CNN-LSTM methods would enable more accurate forecasting and, therefore, more effective system control. Results obtained in this research study indicate that the models still cannot give good predictions under specific conditions, for example in case of sudden changes of demand. Additional external factors might be integrated into the model, like weather data or socioeconomic parameters, to further increase accuracy.
In sum, the literature review has shown that AI can bring potential benefits to water management while it also points out some of the existing challenges. Data availability and model generalizability are still two major obstacles. The hybrid approach of combining physical models with AI serves as the most promising means at present to face these challenges. Further research is needed to improve data availability, enhance robustness in models, and integrate various approaches for an integrated, reliable solution on water management.

Building on the findings from the literature review, this study explored specific challenges and performance evaluations of AI-based models in practical scenarios, such as network fault analysis.
The aspects that this work brings forward are rather important to discuss, not only in view of the performance of the model, but also regarding its practical applicability. The following part of the discussion is dedicated to these aspects, focusing on model accuracy, the trade-off between precision and recall, types of errors, comparisons with existing methods, and potential real-world applications of network fault analyses.
Indeed, as the analysis shows, the model is very accurate with 0.98 ROC-AUC, and quite good in separating positives versus negatives. However, the results should be interpreted with care. While the sensitivity of the model is high, its precision remains quite low, which means tens of thousands of false alarms. This fact raises concerns that the model might not be viable for real-world applications, as false positives could trigger interventions that are not necessary, increasing operational costs.
The main problem with the model performance is the precision-recall trade-off. Based on the precision-recall curve, this model seems to cause false alarm too often. More precisely, while it manages to find the most of possible faults, it also

finds many which are not there. In real conditions, this may become a serious problem during model deployment. The challenge is finding an optimal precision-recall trade-off, which is not just about identifying as many true positives as possible, but also about minimizing unnecessary actions, which can strain resources and reduce the efficiency of the entire system.
A better insight into the model's profile of error might be given by the confusion matrix. The number of false positives is far greater compared to the false negatives—4,748 versus 158—which suggests the model is overly cautious and trips false alarms too frequently. This could be due to any factor or combination of factors, including dataset characteristics or just inherent instabilities in the network. It could also be that the high false positive rate comes from the model's inability differentiate between minute variations signalling actual faults and those that don't. These are some points that an improved feature engineering or an increased quality of training data may help to decrease, in turn reducing the rate of false alarms and making the model more reliable.
Furthermore, it would be more effective if a comparison was made, contrasting the model's results with the results of existing methods. Traditional systems, such as numerical or optimization-based methods of network analytics, are still widely used today, and their comparison with the AI-based approach could be interesting. While the AI model definitely holds certain promise, particularly in its capability to learn from complex data patterns, it may not yet match, in a number of respects, the robustness of traditional methods. Outlining the relative strengths and weaknesses of both approaches may point the way to future development work, perhaps even hybrid methods that exploit the best of both worlds.
Another critical aspect that has to be considered is its practical application in a real-world environment. The model could detect very minute changes in pressure earlier than common approaches, thus averting major failures before they take place. However, one of the most considerable barriers is the high rate of false alarms. The practical sensitivity to early signs of a fault in any situation has to be weighed against unnecessary interventions. Such a balance would be achieved by making the model both effective in fault detection and efficient in resource use.
Although the fault analysis in networking developed here is promising, especially towards early fault detection using the AI model, considerable challenges remain. The main issues are false alarms and practical deployment of the model. As already indicated, future research could be done on improving the model's precision, probably by enhancing data quality, refining feature selection, or using a combination of AI techniques with traditional approaches. Much more reliable solutions could thus be based on hybrid methods, bringing together the adaptability of AI with the robustness of traditional techniques for managing complex water network systems.

To conclude the discussion, it is crucial to examine the comparative performance of different algorithms for the prediction of pipe roughness. This helps to understand the strengths and weaknesses of individual models and highlights avenues for future research.

The XGBoost model showed the best overall performance across all configurations in this study. Especially for larger, loop-free networks, it

consistently delivered stable and accurate predictions of pipe roughness. For instance, in the configuration (200;0), XGBoost reached an excellent R^2 of 0.93, with a very low MAE of 0.08, RMSE of 0.15, and MSE of 0.02. These values indicate that the model captures the nonlinear relationships within the data exceptionally well, and does so consistently as the dataset size increases.
Even in smaller networks, such as (20;0), XGBoost performs reliably, with a R^2 of 0.45, MAE of 0.30, and RMSE of 0.41. When loops are added to the networks, the model's performance remains stable. For example, in (20;1), XGBoost achieves R^2 = 0.46, and in (20;2) still reaches R^2 = 0.43, with only marginal changes in MAE and RMSE. This indicates that XGBoost is able to handle the added complexity introduced by alternative flow paths reasonably well. While loops may increase the variability in flow and pressure distributions, XGBoost does not suffer from significant accuracy loss under these conditions.
In contrast to the GAT model, which struggled especially in looped configurations, XGBoost outperformed GAT in every single network setup across all four evaluation metrics: MSE, MAE, RMSE, and R^2. For instance, in (50;1), XGBoost reached a R^2 of 0.72 compared to only 0.02 by GAT. Likewise, in (100;0), the difference was even more striking: XGBoost R^2 = 0.87, GAT R^2 = 0.001.
These results highlight XGBoost's robustness and scalability. Its architecture, based on gradient-boosted decision trees, allows it to adapt well to increasing data volume and complexity. The model benefits significantly from larger datasets, which occur naturally as the number of pipelines increases. This advantage is compounded when multiple datasets are available, effectively multiplying the training volume and improving the generalization performance.
In summary, XGBoost consistently demonstrated superior accuracy, lower error metrics, and better generalization than alternative models, making it a strong candidate for roughness prediction tasks in drinking water networks, regardless of network complexity.

The GAT model, on the other hand, shows mixed results when applied to datasets of varying complexity. While it performs moderately well on small, loop-free networks, its performance deteriorates as the size or complexity of the network increases, particularly when loops are introduced. For example, for the configuration (20;0), GAT achieves a R^2 of 0.34, a MAE of 0.36, and a RMSE of 0.45, which are within an acceptable range for this task. However, with two loops addedin configuration (20;2), GAT's R^2 drops to 0.15, while MAE increases to 0.42 and RMSE to 0.51. This pattern continues across larger or more complex networks, where the model often fails to generalize well.
In direct comparison, XGBoost consistently outperforms GAT across all evaluated configurations. For instance, in the same (20;2) case, XGBoost achieves a lower MAE (0.32) and RMSE (0.41), and a higher R^2 (0.43). Similar trends are visible across all network types, with GAT never surpassing XGBoost in any of the four evaluation metrics.
These findings suggest that GAT may struggle with capturing dynamic behavior in pipe networks that include alternative flow paths due to varying pressure losses. While GAT is well-suited for modeling static graph topologies, water distribution networks with loops are inherently dynamic systems, where flow paths shift depending on real-time hydraulic conditions. This discrepancy between

model assumptions and system behavior could explain the underperformance of GAT.
Furthermore, the consistent superiority of XGBoost, despite its lack of topological awareness, indicates that GAT's potential is not fully realized in this application.

Among the three compared models, the Artificial Neural Network (ANN) clearly showed the weakest performance across all evaluated configurations. Its prediction accuracy was consistently poor, and in every tested setup, the R^2 value was strongly negative, indicating that the model performed worse than simply predicting the mean of the target variable.
For example, in the small and simple configuration (20;0), the ANN model produced an R^2 of -593.19, a MAE of 10.32, and a RMSE of 13.35. These error values are an order of magnitude higher than those of the GAT and XGBoost models. Even when additional loops were introduced (e.g., in (20;2)), the performance remained poor, with R^2 = -396.87, MAE = 9.20, and RMSE = 10.89. These results suggest that the ANN model was unable to generalize and may have overfit to noise in the training data.
The trend persists even as the datasets grow. In configuration (100;0), where XGBoost achieved R^2 = 0.87 and GAT still maintained a low but positive R^2 = 0.001, the ANN model reached R^2 = -506.24, with a MAE of 9.39 and RMSE of 12.37. The failure to improve with larger datasets is particularly surprising, as ANNs typically benefit from increased data volume.
These outcomes suggest that the ANN architecture used in this study was either too shallow, not sufficiently regularized, or otherwise ill-suited to capture the complexity of roughness patterns in water distribution networks. Additionally, ANNs can be sensitive to input scaling, feature interactions, and data variability—all of which are present in this application. Especially in networks with loops and alternative flow paths, where hydraulic relationships become nonlinear and path-dependent, the ANN failed to extract meaningful predictive structures.

For roughness prediction in general, the best results were achieved with XGBoost, especially within less complicated networks without loops. Considering the GAT model and its ability to handle network topologies, it didn't live up to its potential once the loops increased the structural complexity of the network. The ANN model was not suitable for the problem in its current form and would likely require deeper architectures or entirely different approaches to deliver meaningful results.
Despite extensive optimization efforts—including hyperparameter tuning, architectural adjustments, and regularization techniques—significant improvements in prediction accuracy could not be achieved. To give the models better chances to learn and generalize, the originally complex hydraulic networks used in the early stages of this study were intentionally simplified. The initial networks contained multiple loops and dynamically changing flow paths that better reflect real-world conditions. However, as the models showed poor results in these realistic settings, simpler and more uniform network configurations were used in later experiments to reduce structural complexity and isolate predictive performance.
Even under these simplified conditions, however, the ANN and GAT models failed to generalize adequately, and XGBoost, while comparatively strong, still fell short

of application-level reliability. This suggests that the applied methods alone may not be sufficient to capture the nonlinear and dynamic nature of hydraulic behavior in pipe networks.

All in all, it would appear that even the best-performing results from XGBoost are not yet sufficient to meet the accuracy standards set by the German Association for Water and Gas Networks. For future research, hybrid approaches—particularly those that combine AI-based models with numerical, optimization, or simulation-based techniques—should be considered.
Physics-Informed Neural Networks (PINNs) could also be a promising approach to improving prediction quality, as they incorporate physical laws directly into the learning process. This can be especially helpful in data-scarce scenarios, or when the system behavior is strongly governed by known hydraulic principles. PINNs were briefly tested in the early phase of this work, but did not yield satisfactory results at the time due to unstable training and high computational demand. Therefore, this approach was not further pursued. However, recent research, such as the study by Ashraf et al. (2024), has shown that with more computing power and methodological refinement, PINNs can be successfully applied for parameter estimation in water networks.
Nevertheless, with thorough investigation, this approach could still hold significant potential. Moreover, the literature review indicated that there are no studies on projects using solely AI methods for calibration. Thus, a combination of the different methods mentioned could be more effective in addressing the challenges in predicting pipe roughness in complex water networks.

AI integration into water management has shown great promise but also pointed out critical challenges regarding anomaly detection, network fault analysis, and pipe roughness prediction. The literature review demonstrated the potential of AI in enhancing system monitoring and prediction capabilities, but emphasized hybrid approaches due to limitations in data availability and model generalizability. In network fault analysis, although AI models have shown very good performance by catching faults with high accuracy, an optimal balance between precision-recall has remained the Holy Grail for practical effectiveness.

On the other hand, the evaluation of roughness prediction models showed that their effectiveness is greatly reduced with advanced pipe network complexity. This means that a more hybrid model combining different aspects of AI and a more traditional approach might provide better performance. Or rather, increasing data quality and model robustness will lead towards reaching the full potential of artificial intelligence in water management applications.

7 Conclusion

The potential of AI, particularly neural networks, for optimizing water supply networks is significant. Its most common applications include predictive maintenance, leak detection, and water quality modeling, or network calibration. Even though the adaption of AI in the water management sector still is an incipient business, its significant degree of technological availability displays a promising perspective regarding enhancements that will make the sector more efficient and sustainable.
In this study, different models were able to identify network faults such as undetected valve closures, which is very important for network reliability maintenance. However, accuracy and sensitivity-to-accuracy ratio remain problematic, resulting in a considerable number of false positives. Similarly, artificial neural networks showed partial success in estimating the roughness of the pipe, but without sufficient accuracy. The predictions were reliable for very simple networks. Yet, in practice, these simple networks are infeasible.

A comparison of the various models used showed that XGBoost is the best performing model for both simple, loop-free networks and networks with slightly increasing complexity On the contrary, GAT gave ununiform results with increased network variability, and ANN started slowly to develop weaknesses in recognizing underlying patterns, thus compelling it towards deeper architectures if required.
Future research should focus on the further integration of AI and neural networks within pipeline management and calibration modeling. The robustness of models is considered key factor and needs to be gained with extensive and varied data. A combination of AI and both numerical and optimization methods is thus open for further research. Moreover, a close investigation of methods like Physics-Informed Neural Networks would help improve the prediction quality.
In sum it can be said that AI can offer enormous options in enhancing precision and effectiveness related to drinking water management. However, model precision and practical applicability are mostly out of balance. Hybrid approaches that embody the strengths of both AI and conventional methodologies could go a long way in improving the current working processes in water management.

8 References

Abdelmageed, S., Tariq, S., Boadu, V., & Zayed, T. (2022). Criteria-based critical review of artificial intelligence applications in water-leak management. *Environmental Reviews*, *30*(2). https://doi.org/10.1139/er-2021-0046

Adam Optimizers: Pyhton Packages. (2025, April 9). https://keras.io/api/optimizers/adam/

Adedeji, K. B., Ponnle, A. A., Abu-Mahfouz, A. M., & Kurien, A. M. (2022). Towards Digitalization of Water Supply Systems for Sustainable Smart City Development—Water 4.0. *Applied Sciences*, *12*(18). https://doi.org/10.3390/app12189174

Anonymized Municipality. (2024). Water supply overview. (originally retrieved from a public municipal website).

Ashraf, I., Strotherm, J., Hermes, L., & Hammer, B. (2024). Physics-Informed Graph Neural Networks for Water Distribution Systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, *38*(20), 21905–21913. https://doi.org/10.1609/aaai.v38i20.30192

Bartz, E., Bartz-Beielstein, T., Zaefferer, M., & Mersmann, O. (Eds.). (2023). *Hyperparameter Tuning for Machine and Deep Learning with R*. Springer Nature Singapore. https://doi.org/10.1007/978-981-19-5170-1

Berardi, L., Ugarelli, R., Røstum, J., & Giustolisi, O. (2014). Assessing mechanical vulnerability in water distribution networks under multiple failures. *Water Resources Research*, *50*(3), 2586–2599. https://doi.org/10.1002/2013WR014770

Bertolini, M., Mezzogori, D., Neroni, M., & Zammori, F. (2021). Machine Learning for industrial applications: A comprehensive literature review. *Expert Systems with Applications*, *175.* https://doi.org/10.1016/j.eswa.2021.114820

Bonilla, C. A., Zanfei, A., Brentan, B., Montalvo, I., & Izquierdo, J. (2022). A Digital Twin of a Water Distribution System by Using Graph Convolutional Networks for Pump Speed-Based State Estimation. *Water*, *14*(4). https://doi.org/10.3390/w14040514

Boutros, A., Nurvitadhi, E., Ma, R., Gribok, S., Zhao, Z., Hoe, J. C., Betz, V., & Langhammer, M. (2020). Beyond Peak Performance: Comparing the Real Performance of AI-Optimized FPGAs and GPUs. In *2020 International Conference on Field-Programmable Technology (ICFPT)* (pp. 10–19). IEEE. https://doi.org/10.1109/ICFPT51103.2020.00011

Brody, S., Alon, U., & Yahav, E. (2021). *How Attentive are Graph Attention Networks?* https://doi.org/10.48550/arXiv.2105.14491

Capelo, M., Brentan, B., Monteiro, L., & Covas, D. (2021). Near–Real Time Burst Location and Sizing in Water Distribution Systems Using Artificial Neural Networks. *Water*, *13*(13). https://doi.org/10.3390/w13131841

Dada, M. A., Majemite, M. T., Obaigbena, A., Daraojimba, O. H., Oliha, J. S., & Nwokediegwu, Z. Q. S. (2024). Review of smart water management: IoT and AI in water and wastewater treatment. *World Journal of Advanced Research and Reviews*, *21*(1), 1373–1382. https://doi.org/10.30574/wjarr.2024.21.1.0171

Daniel, I., Ajami, N. K., Castelletti, A., Savic, D., Stewart, R. A., & Cominola, A. (2023). A survey of water utilities' digital transformation: drivers, impacts, and

enabling technologies. *Npj Clean Water*, *6*(1). https://doi.org/10.1038/s41545-023-00265-7

Das, S., Dey, A., Pal, A., & Roy, N. (2015). Applications of Artificial Intelligence in Machine Learning: Review and Prospect. *International Journal of Computer Applications*, *115*(9), 31–41. https://doi.org/10.5120/20182-2402

Dawood, T., Elwakil, E., Novoa, H. M., & Delgado, J. F. G. (2020). Artificial intelligence for the modeling of water pipes deterioration mechanisms. *Automation in Construction*, *120.* https://doi.org/10.1016/j.autcon.2020.103398

Deutsche Vereinigung des Gas- und Wasserfachs e.V. (2006). *Berechnung von Gas- und Wasserrohrnetzen Teil 1::* Hydraulische Grundlagen, Netzmodellierung und Berechnung (GW 303-1).

Do, N. C., Simpson, A. R., Deuerlein, J. W., & Piller, O. (2016). Calibration of Water Demand Multipliers in Water Distribution Systems Using Genetic Algorithms. *Journal of Water Resources Planning and Management*, *142*(11), Article 04016044. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000691

Fu, G., Jin, Y., Sun, S., Yuan, Z., & Butler, D. (2022). The role of deep learning in urban water management: A critical review. *Water Research*, *223.* https://doi.org/10.1016/j.watres.2022.118973

Fu, G., Sun, S., Hoang, L., Yuan, Z., & Butler, D. (2023). Artificial intelligence underpins urban water infrastructure of the future: A holistic perspective. *Cambridge Prisms: Water*, *1.* https://doi.org/10.1017/wat.2023.15

Garzón, A., Kapelan, Z., Langeveld, J., & Taormina, R. (2022). Machine Learning‐Based Surrogate Modeling for Urban Water Networks: Review and Future Research Directions. *Water Resources Research*, *58*(5), Article e2021WR031808. https://doi.org/10.1029/2021WR031808

Gohil, J., Patel, J., Chopra, J., Chhaya, K., Taravia, J., & Shah, M. (2021). Advent of Big Data technology in environment and water management sector. *Environmental Science and Pollution Research International*, *28*(45), 64084–64102. https://doi.org/10.1007/s11356-021-14017-y

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. *Adaptive computation and machine learning*. The MIT Press.

Google Maps. (2024). Aerial view of the case study region [Anonymized]. Retrieved April 2024, from https://maps.google.com

Grieves, M., & Vickers, J. (2017). Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In F.-J. Kahlen, S. Flumerfelt, & A. Alves (Eds.), *Transdisciplinary Perspectives on Complex Systems* (pp. 85–113). Springer International Publishing. https://doi.org/10.1007/978-3-319-38756-7_4

Hajgató, G., Gyires-Tóth, B., & Paál, G. (2021). Reconstructing nodal pressures in water distribution systems with graph neural networks. *ArXiv*, *2021.* https://doi.org/10.48550/arXiv.2104.13619

Herrera, M., Torgo, L., Izquierdo, J., & Pérez-García, R. (2010). Predictive models for forecasting hourly urban water demand. *Journal of Hydrology*, *387*(1-2), 141–150. https://doi.org/10.1016/j.jhydrol.2010.04.005

Kang, D., & Lansey, K. (2010). Demand and Roughness Estimation in Water Distribution Systems. *Journal of Water Resources Planning and*

Management, *137*(1), 20–30. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000086

Krishnan, S. R., Nallakaruppan, M. K., Chengoden, R., Koppu, S., Iyapparaja, M., Sadhasivam, J., & Sethuraman, S. (2022). Smart Water Resource Management Using Artificial Intelligence—A Review. *Sustainability*, *14*(20). https://doi.org/10.3390/su142013384

Kühnert, C., Gonuguntla, N. M., Krieg, H., Nowak, D., & Thomas, J. A. (2021). Application of LSTM Networks for Water Demand Prediction in Optimal Pump Control. *Water*, *13*(5). https://doi.org/10.3390/w13050644

Lijuan, W., Hongwei, Z., & Zhiguang, N. (2012). Leakage Prediction Model Based on RBF Neural Network. In Y. Wu (Ed.), *Advances in Intelligent and Soft Computing. Software Engineering and Knowledge Engineering: Theory and Practice* (Vol. 114, pp. 451–458). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-03718-4_56

Liu, H., Walski, T., Fu, G., & Zhang, C [Chi] (2017). Failure Impact Analysis of Isolation Valves in a Water Distribution Network. *Journal of Water Resources Planning and Management*, *143*(7), Article 04017019. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000766

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133. https://doi.org/10.1007/bf02478259

Meirelles, G., Manzi, D., Brentan, B., Goulart, T., & Luvizotto, E. (2017). Calibration Model for Water Distribution Network Using Pressures Estimated by Artificial Neural Networks. *Water Resources Management*, *31*(13), 4339–4351. https://doi.org/10.1007/s11269-017-1750-2

Mitschke, N. (2022). *Konvolutionäre neuronale Netze in der industriellen Bildverarbeitung und Robotik.* https://doi.org/10.5445/KSP/1000146397

Mücke, N. T., Pandey, P., Jain, S., Bohté, S. M., & Oosterlee, C. W. (2023). A Probabilistic Digital Twin for Leak Localization in Water Distribution Networks Using Generative Deep Learning. *Sensors (Basel, Switzerland)*, *23*(13). https://doi.org/10.3390/s23136179

Murphy, K. P. (2012). *Machine learning: A probabilistic perspective* (4. print. (fixed many typos)). *Adaptive computation and machine learning series*. MIT Press.

Najah, A., El-Shafie, A., Karim, O. A., & El-Shafie, A. H. (2013). Application of artificial neural networks for water quality prediction. *Neural Computing and Applications*, *22*(S1), 187–201. https://doi.org/10.1007/s00521-012-0940-3

Nikolic, G. S., Dimitrijevic, B. R., Nikolic, T. R., & Stojcev, M. K. (2022). A Survey of Three Types of Processing Units: CPU, GPU and TPU. In *2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)* (pp. 1–6). IEEE. https://doi.org/10.1109/ICEST55168.2022.9828625

O'Reilly, G., Bezuidenhout, C. C., & Bezuidenhout, J. J. (2018). Artificial neural networks: applications in the drinking water sector. *Water Supply*, *18*(6). https://doi.org/10.2166/ws.2018.016

Qian, K., Jiang, J., Ding, Y., & Yang, S. (2020). Deep Learning Based Anomaly Detection in Water Distribution Systems. In *2020 IEEE International*

Conference on Networking, Sensing and Control (ICNSC) (pp. 1–6). IEEE. https://doi.org/10.1109/ICNSC48988.2020.9238099

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045

RBS wave GmbH. (2024). Pipe Network of a medium-sized municipality (internal documentation).

Salloom, T., Kaynak, O., & He, W. (2021). A novel deep neural network architecture for real-time water demand forecasting. *Journal of Hydrology*, *599.* https://doi.org/10.1016/j.jhydrol.2021.126353

Sattar, A. M. A., Ertuğrul, Ö. F., Gharabaghi, B., McBean, E. A [E. A.], & Cao, J. (2019). Extreme learning machine model for water network management. *Neural Computing and Applications*, *31*(1), 157–169. https://doi.org/10.1007/s00521-017-2987-7

Savic, D. (2019). Artificial Intelligence-How can water planning and management benefit from? *IAHR White Papers*.

Schellart, A., Blumensaat, F., Clemens-Meyer, F., van der Werf, J., Mohtar, W. H. M. W., Ramly, S., Muhammad, N., Bonneau, J., Fletcher, T. D., Costelloe, J. F., James, R., Burns, M., Poelsma, P., Ochoa-Rodriguez, S., Bourne, D., Hancock, Z., Wallwork, G., Hale, J., Nikolova-Peters, N., . . . Ebi, C. (2021). Data collection in urban drainage and stormwater management systems – case studies. In J.-L. Bertrand-Krajewski, F. Clemens-Meyer, & M. Lepot (Eds.), *Metrology in Urban Drainage and Stormwater Management: Plug and Pray* (pp. 415–469). IWA Publishing. https://doi.org/10.2166/9781789060119_0415

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489. https://doi.org/10.1038/nature16961

Snider, B., & McBean, E. A [Edward A.] (2020). Improving Urban Water Security through Pipe-Break Prediction Models: Machine Learning or Survival Analysis. *Journal of Environmental Engineering*, *146*(3), Article 04019129. https://doi.org/10.1061/(ASCE)EE.1943-7870.0001657

Sola, K., Bjerkholt, J., Lindholm, O., & Ratnaweera, H. (2021). What Effect Does Rehabilitation of Wastewater Pipelines Have on the Share of Infiltration and Inflow Water (I/I-Water)? *Water*, *13*(14). https://doi.org/10.3390/w13141934

Tao, F., Zhang, H., Liu, A., & Nee, A. Y. C. (2019). Digital Twin in Industry: State-of-the-Art. *IEEE Transactions on Industrial Informatics*, *15*(4), 2405–2415. https://doi.org/10.1109/TII.2018.2873186

Truong, H., Tello, A., Lazovik, A., & Degeler, V. (2023). Graph Neural Networks for Pressure Estimation in Water Distribution Systems. *ArXiv.* Advance online publication. https://doi.org/10.48550/ARXIV.2311.10579

Veerapaneni, S., Budd, G., Bond, R., & Horsley, M. (2010). Using Neural Networks to Predict Treatment Process Performance. *Journal AWWA*, *102*(4), 38–44. https://doi.org/10.1002/j.1551-8833.2010.tb10083.x

Walski, T. M. (1983). Technique for Calibrating Network Models. *Journal of Water Resources Planning and Management*, *109*(4), 360–372. https://doi.org/10.1061/(ASCE)0733-9496(1983)109:4(360)

Walski, T. M. (2000). Model calibration data: the good, the bad, and the useless. *Journal AWWA*, *92*(1), 94–99. https://doi.org/10.1002/j.1551-8833.2000.tb08791.x

Wéber, R., & Hős, C. (2020). Efficient Technique for Pipe Roughness Calibration and Sensor Placement for Water Distribution Systems. *Journal of Water Resources Planning and Management*, *146*(1), Article 04019070. https://doi.org/10.1061/(ASCE)WR.1943-5452.0001150

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C [Chengqi], & Yu, P. S. (2021). A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4–24. https://doi.org/10.1109/TNNLS.2020.2978386

XGBoost Documentation: Python Packages. (2025, April 9). https://xgboost.readthedocs.io/en/release_3.0.0/

Zanfei, A., Brentan, B. M., Menapace, A., Righetti, M., & Herrera, M. (2022). Graph Convolutional Recurrent Neural Networks for Water Demand Forecasting. *Water Resources Research*, *58*(7), Article e2022WR032299. https://doi.org/10.1029/2022WR032299

Zanfei, A., Menapace, A., Brentan, B. M., Sitzenfrei, R., & Herrera, M. (2023). Shall we always use hydraulic models? A graph neural network metamodel for water system calibration and uncertainty assessment. *Water Research*, *242.* https://doi.org/10.1016/j.watres.2023.120264

Zhang, Q., Zheng, F., Duan, H.-F., Jia, Y., Zhang, T., & Guo, X. (2018). Efficient Numerical Approach for Simultaneous Calibration of Pipe Roughness Coefficients and Nodal Demands for Water Distribution Systems. *Journal of Water Resources Planning and Management*, *144*(10), Article 04018063. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000986

Zhao, Q., Wu, W., Simpson, A. R., & Willis, A. (2022). Simpler Is Better—Calibration of Pipe Roughness in Water Distribution Systems. *Water*, *14*(20). https://doi.org/10.3390/w14203276

Zhou, S., Guo, S., Du, B., Huang, S., & Guo, J. (2022). A Hybrid Framework for Multivariate Time Series Forecasting of Daily Urban Water Demand Using Attention-Based Convolutional Neural Network and Long Short-Term Memory Network. *Sustainability*, *14*(17), 11086. https://doi.org/10.3390/su141711086

Appendix A: Technical Implementation of the Synthetic Training Data Generator

As already indicated, this sub-chapter focuses on the development process of a code for creating synthetic data for water network calibration. First, the generation of synthetic data for network errors is explained, followed by the creation of synthetic data sets for different roughness values.

The goal of the code is to automatically create a number of synthetic data sets from an already calibrated drinking water network in STANET. These data sets are supposed to represent a variety of possible conditions in the network and to reflect the actual physical correlations. Since the prediction includes both network errors and the determination of roughness, two variants of the code were developed, generating data sets for network errors as well as different roughness.

An important criterion of synthetic data sets is that data of situations both with and without artificial stress is included. This is necessary because, in practice, RBS calibrates the networks with data drawn from networks under both conditions. Therefore, the final algorithm must take both scenarios into account and generate a version with and without load for each data set. This approach mirrors the real calibration process, where pressure differences are used to iteratively adjust the roughness values to match the measured conditions. By generating synthetic data with and without load, the code ensures that the calibration algorithm can work similarly to how it would be done in a real scenario, using both sets of conditions to determine pipe roughness.

The code is modular and divided into three main modules:

- Module 1: Reading the text file containing the export/import of the network for a closed gate valve. The closed gate valve simulates a network error due to an unknown gate valve that is closed. The text file contains the line ID at which the gate valve is positioned.
- Module 2: Processing and modification of the read-in text file. To create different synthetic data sets in which the closed gate valve is positioned on different, haphazardly selected lines, the line ID in the text file is replaced by another one, randomly chosen. This new text file is then saved.
- Module 3: Reading the new text file into STANET. STANET performs a hydraulic calculation and exports the complete network data as part of the synthetic data set.

Modules and Functions

A central document is `main.py`, the executive code that controls the other components. In the `main()` function, first of all logging is defined and the path to the text file to be read is set. Logging was chosen to ensure a detailed record of the processes, so that the cause of any errors can be easily traced.

The function `process_file()` controls the actual process. First, the module `data_loader.py` is used to read the text file. Then a loop is started that generates several synthetic data sets, with each loop having a unique iteration number.

In each iteration, the text file is adjusted so that a different line ID is stored for the closed valve. This task is performed by the module `data_process.py`. Not all lines of the network can be used. If the closed gate valve is located on a branch line behind which customers are connected to the network, the error would be immediately apparent. Therefore, the gate valve is only positioned in lines that are connected to other lines so that each customer can continue to be supplied. The new text file is saved with the iteration number included in the file name.

Afterwards, the code `stanet_process.py` is activated, which is responsible for importing data into STANET, performing the hydraulic calculation, and exporting network data. Due to technical requirements in STANET, only one import or export can be done per activation.

The code `stanet_process.py` is structured as a class (`class StanetProcess`). The paths to STANET and the individual commands are defined in the instance variables of the class. To start STANET via an external program, the respective command must be added when calling the .exe file. A sample command could look like this:

```
C:\Program Files\STANET\BIN\stanet64.exe“ /N=”path_to_network“
/CONFIG=”path_to_configuration“ /X=”import definition“ /F=”path_to_import_file” /A
/B
```

The command "/N" describes the network to be opened, "/CONFIG" specifies the configuration file, "/X" defines the import parameters, "/F" specifies the path to the file to be imported, "/A" inserts the new values into the existing network, and "/B" starts the hydraulic calculation.

The class `StanetProcess` consists of three functions:

- import_data() – creates the command to import the files.
- export_data() – creates the command to export the files and save the results as a CSV (Comma-Separated Values) file, with the iteration number of the loop included in the file name.
- run_command() – executes the commands via the subprocess package. The functions `import_data()` and `export_data()` start `run_command()` directly after the command has been created.

In order to receive data for each slider position both under load and without load, the entire STANET process is executed twice and in two different networks. The two network files differ only in that in one network, there is an artificial load, while in the other, there is not. This is controlled with a boolean flag that is set when the class is initialized. To reflect these different load conditions, the STANET process

is executed twice with the same but respectively modified import file. The artificial load (situation with load) in the network is defined as 5 m^2/h. As a result, STANET opens four times per run.

The final output of each iteration thus consists of a STANET export without and with load, each clearly marked with the corresponding iteration number of the loop. The differences between these two sets are then calculated just as in the real-life calibration process where pressure differences between loaded and unloaded states are used to determine the correct roughness values.

To also enable the prediction of roughness using synthetic data sets, an additional version of the code was developed using a GitHub forge. In the version for the different roughness values, the original TXT file does not contain the position of the valve, but the roughness for each pipe in the network, which is clearly assigned by the pipe ID. In the file `data_process.py`, the pipe ID is not exchanged, but for each value of the roughness new random values between 0.1 and 2 are generated. These roughness values reflect the range that can occur in reality. They also ensure that, with the artificial load given, the pressure at the beginning of the network is sufficient to supply every consumer. Since STANET performs pressure-oriented calculations, very high roughness values and very high flow rates can result in computed negative pressures, which is not realistic. Therefore, parameters are selected carefully to ensure that the highest pressure drop in the network never leads to negative pressures.

In the file `stanet_process.py`, the import configuration in STANET is adapted to enable the import of the modified roughness data. The rest of the process remains unchanged.

Justification of the Technology Selection

- Python as main language: Python was chosen because of its versatility and extensive libraries. Libraries such as `os`, `subprocess`, and `logging` provide the necessary tools to implement automated processes, file manipulation, and logging of operations. Python is particularly suitable because it can be used for both file processing and interface communication.
- Modular design: The code was designed to be modular to improve maintainability and reusability. Each part of the process is handled in a separate module (`data_loader.py`, `data_process.py`, `stanet_process.py`). This has the advantage that individual modules can be customized as needed without having to redevelop the entire application.
- Use of subprocess: The subprocess package was chosen to control STANET because it provides an easy way to start external programs from Python and to control the commands for import and export. This method allows for seamless integration of Python and STANET.
- Automation of the hydraulic calculation: The decision to control STANET through Python scripts is based on the need to generate and analyze large

amounts of synthetic data. This automation reduces the manual effort significantly and minimizes the sources of error that could occur with manual input.

- Data quality and error handling: Logging was used to ensure detailed logging. Error handling mechanisms are implemented to ensure that the process does not stop when errors occur and that the cause can be easily traced.

A particular challenge in this project was embedding STANET and working with it as an external program. The difficulty lies in the fact that STANET is not inherently designed to be initiated by an external program. Consequently, there are limited functions and minimal documentation in the STANET manual that provide support for such integration, as well as the creation of interfaces and collaboration with the Python code. Since the program itself is not widely used, there are also no forums or communities for discussing errors. For this reason, it took several weeks and many discussions with the software distributor before the creation of synthetic data sets worked smoothly and reliably. Often, very small, insignificant changes in STANET solved the problems. However, these were not apparent from the user's perspective and could only be found with the help of the STANET developers.
This limitation required considerable effort to develop workarounds and to build custom scripts to enable interaction between STANET and the other tools incorporated into the project workflow. Despite these obstacles, through trial and error and extensive testing, a functional solution was eventually achieved.

Appendix B: Data Preprocessing for Graph-Based Fault Prediction

Data preprocessing is a necessary procedure for any ML algorithm in order to be sure that a model trained to predict faults in drinking water networks is of high quality and reliable. This section describes the different methods and procedures applied to transform raw data into an analyzable format, as implemented through various Python scripts.

The first preprocessing step is to read raw data files in CSV format, drawn from the STANET simulation. Data is inconsistent because each row does not have the same number of columns, and there are extra semicolons. Due to these problems, proper cleaning is required. To achieve it, `DataLoader` class has been realized in the `rawdata_load.py` script. It calculates the maximum number of columns used in the dataset and cleans each line to make all rows consistent.
This cleaning process normalizes each row to the same number of columns, including empty fields where appropriate. After cleaning the dataset, it is written into a new CSV. All subsequent processing stages can rely on this CSV to access well-structured data.

Further preprocessing is realised by the class `DataProcessor`, implemented in `rawdata_preprocess.py` after cleaning. This class segregates the data into two broad categories, which are nodes (`KNO`) and pipes (`LEI`). The nodes are the connecting points, while the pipes are the links between the nodes.
This is followed by the `split_data` method, which will pick out rows and extract headers depending on the specific markers `KNO` and `LEI`. Setting headers dynamically guarantees that headers are correctly labelled for an appropriate analysis, avoiding incorrect mappings. The filtered datasets, that is the most relevant columns, are then written into separate CSV files: one for nodes and another for pipes.
Also defined in the same rawdata_preprocess.py, the class DataCombiner will be responsible for combining both Load and NoLoad scenarios. Files are matched to combine corresponding columns from the two data sets. This is important for giving insight into system behaviour under both conditions and also ensures that the data for modeling is complete. The combined dataset contains feature variables such as pipe diameters, flow rates, and pressure drops (see Figure 4 for a schematic map of the network layout).

Additional operations were implemented in the `train_GAT.py` elaboration phase, such as feature scaling, handling missing values, and generating graph-based representations of water networks for the training step of the Graph Attention Networks (GAT) model.
One of the most important preprocessing steps is feature scaling, ensuring that all features are on an equal level with regarding their contribution to the learning process. In this, `StandardScaler` and `MinMaxScaler` are applied to normalize features in order to avoid scaling issues associated with different weighting of features. The scaled features then are used for the GAT model training.

The core of the GAT model is an explicit representation of the water network as a graph. More precisely, the nodes correspond to junctions or endpoints, while the edges correspond to pipes connecting these nodes. Equipped with such a structure, the model can learn meaningful relations between parts of the network and is applicable out-of-the-box for various behaviors of interest for the network. First, the python package 'networkx' was used to create a graph representation of the water distribution network. As mentioned, each node in this graph represents a physical element, either a pipe junction or an endpoint. These nodes are enriched with further attributes of pressure, elevation, and demand since such attributes are important to understand the nodes' behavior within the network.
Edges in the graph represent pipes connecting these nodes and are annotated with features like diameter, length, and flow rate. More importantly, this model allows the representation of not only connectivity, but also the physical properties of each connection. This representation using edge features will let the GAT model learn effectively from the relationships among the connected nodes. Edge features refer to attributes assigned to the connections (edges) between nodes in a graph. In the context of water networks, these may include pipe length, diameter, material, or calculated flow and pressure differences.

Construction makes sure that nodes and edges line up by providing a unique identifier for each node and each edge, in turn helping to establish and verify the relationships within the graph. The usage of the `add_edge` function from `networkx` provides validation to make sure both nodes exist before adding an edge; otherwise, this could result in free edges or misalignment.

The coordinates (`XRECHTS`, `YHOCH`) enrich the spatial representation of the graph by embedding more geological information. They are used as node attributes for model building, considering the physical layout of the network. This adds spatial information that may be relevant in developing models for the interaction of systems that are geographically dispersed. Positional encodings derived from these coordinates help the model understand the spatial relationship between nodes, something very critical toward making proper predictions of network dynamics across different operational scenarios.
During the program run , the class DataModule uses geographic information on node attributes like XRECHTS, YHOCH, and GEOH. MinMaxScaler scales these geographic features to normalize the values that can then be appropriate for training models. The sinusoidal positional encoding also aims at these geographic features using the add_positional_encoding method. This function creates sine and cosine transformations of the coordinates that enable the GAT model to capture periodic spatial patterns and enhance its capability to learn effectively from spatial relationships.

As indicated earlier, features such as pipe diameter, flow rate, and pressure are very important to understand the dynamics of the network. The graph-based feature engineering methodology helps include those features directly on the nodes and edges of the graph. Lastly, adding geographic positional encodings enhances the spatial awareness inside the graph, hence more context is provided for the model.

However, also processing data that is missing is crucial for any model in order to maintain reliability. Graph-based imputation deals with missing flow rate and pressure records. Coherent with the general network structure, the model fills the missing values using the signals from neighboring nodes in order to make the predictions more accurate. The algorithm, therefore, is trained using data from only a limited number of nodes since, in real-world scenarios, measurement data is usually available for only a subset of nodes. This is imitated in the training of the algorithm by incorporating only the data from specific nodes.

The preprocessing workflow developed in this assessment is considered a critical step in processing raw, inconsistent data about water networks into information that can be used for training graph-based models.

Appendix C: Code Testing and Validation Procedures

In the following section, the essential role of testing in the software development lifecycle will be presented. More precisely, its role in the context of data processing and machine learning. Testing ensures that the applied methodologies function as intended, thereby guaranteeing the resilience and integrity of the model.

For verifying the functionality of various components, the strategy of unit testing was chosen. During this step, dependencies were simulated using Python's `unittest` framework in combination with `unittest.mock`. These unit tests supervised that each module functioned independently, minimizing the risk of errors spreading through the system.

In general, the testing process covered key modules, including data loading, processing, model training, and evaluation. Functions within the `DataLoader` class, such as `find_max_columns` and `clean_line`, were thoroughly tested to handle incomplete rows and ensure consistent data formatting. Additionally, the `DataProcessor` class was evaluated for accurate data splitting into uniform categories.
The so-called `FeatureEngineer` class was applied to assess feature engineering, confirming that datasets had been merged correctly, polynomial features had been generated accurately, and data scaling and splitting had been performed appropriately. Thus, normalized features and an accurate division into training and testing sets could be ensured.
Model training and evaluation were tested using `TestTrainModel.test_train_model` and `TestEvaluateModel.test_evaluate_model` on a `RandomForestClassifier`. These tests verified that the training process was free of errors, producing reliable accuracy metrics.
Furthermore, the functionality of the predictions was examined to confirm that the model could generate accurate classifications. This was validated by `TestMakePrediction.test_make_prediction`. The `save_model` function was tested to ensure that the model could be saved and reloaded without losing its accuracy. This step was important to prepare the model for future use without having to retrain it.

In sum, testing was a crucial part of the research process to verify the functionality of individual components and to ensure the optimal performance of the entire pipeline. However, there is more that should be considered in this regard. Thus, future endeavours will focus on expanding test coverage to include integration testing and incorporating automated testing within a continuous integration framework, thereby maintaining software quality throughout the development lifecycle.

Appendix D: Artificial Neural Network (ANN) - Predicted vs. True Roughness Values

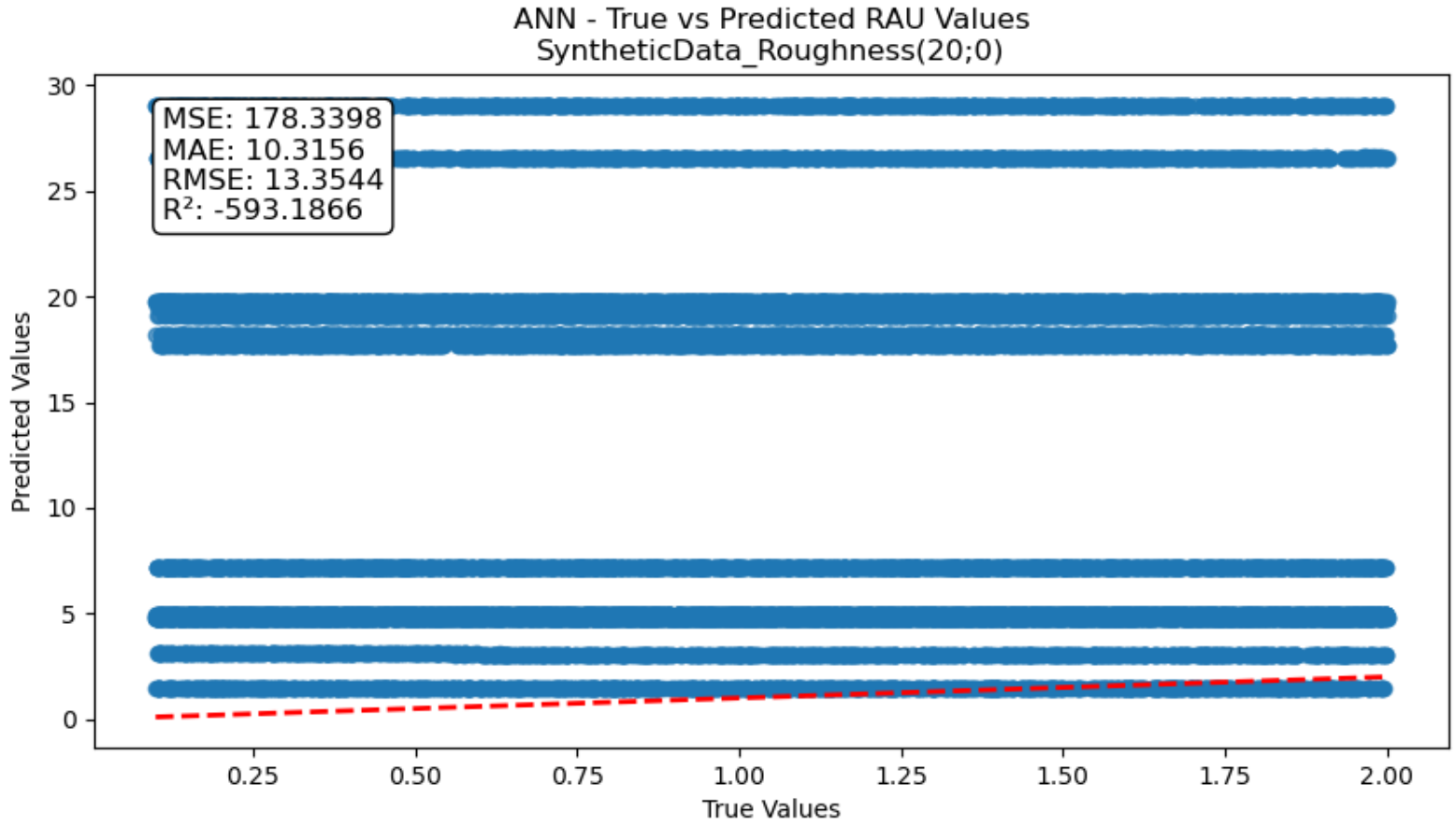

Figure 9: Predicted vs. True Roughness for ANN Model - Dataset with 20 Pipelines, 0 Loops

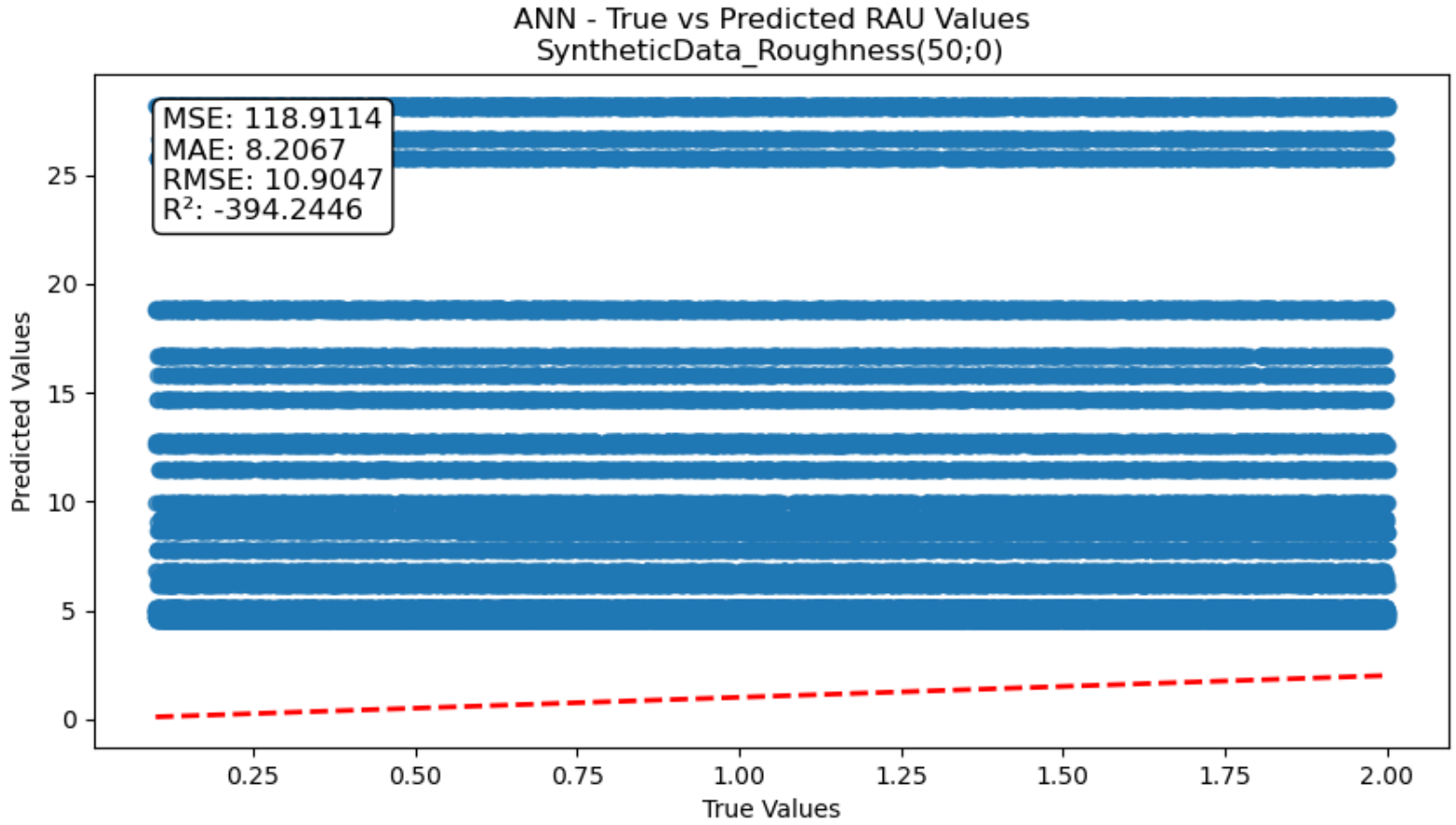

Figure 10: Predicted vs. True Roughness for ANN Model - Dataset with 50 Pipelines, 0 Loops

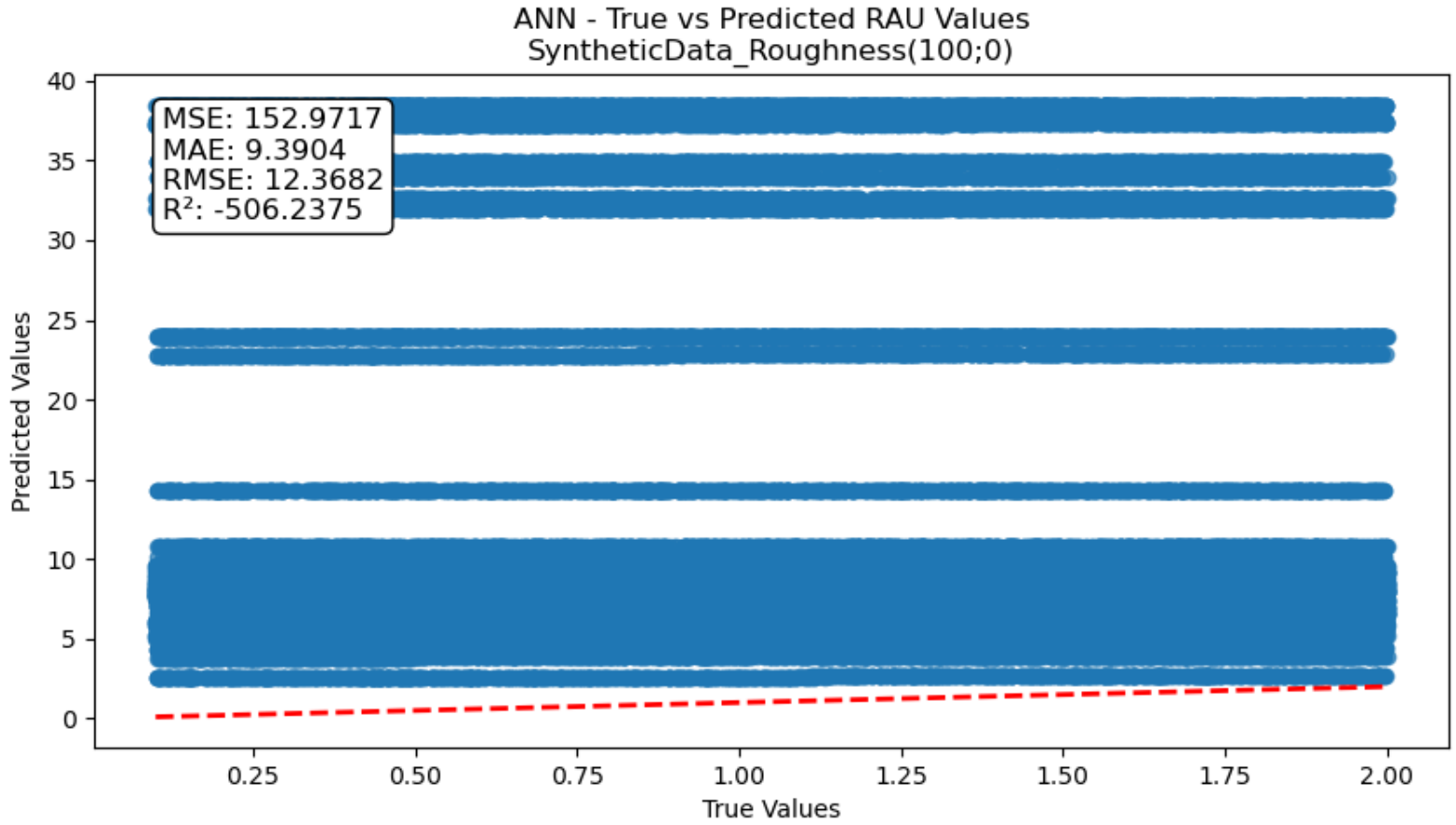

Figure 11: Predicted vs. True Roughness for ANN Model - Dataset with 100 Pipelines, 0 Loops

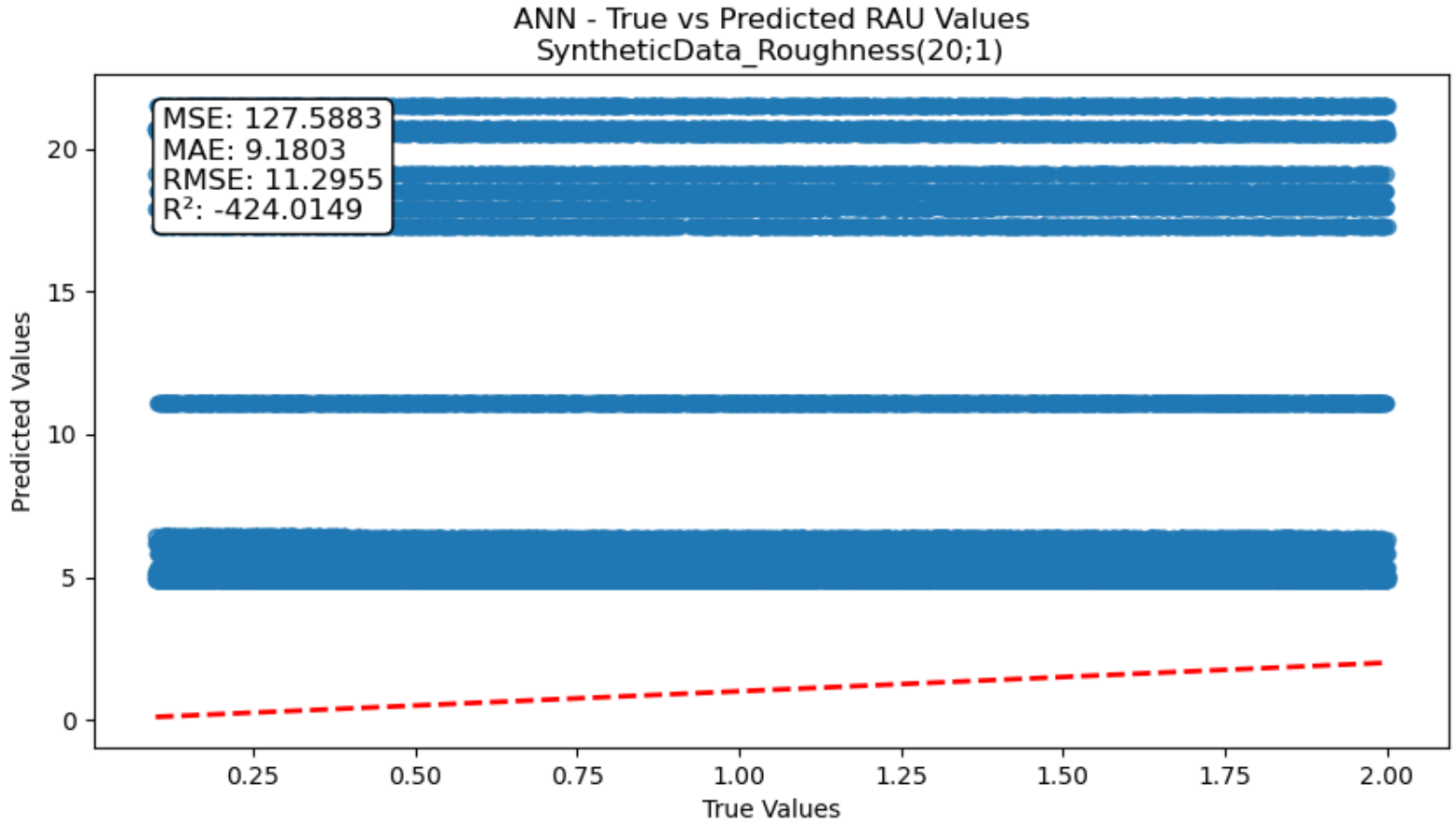

Figure 12: Predicted vs. True Roughness for ANN Model - Dataset with 20 Pipelines, 1 Loop

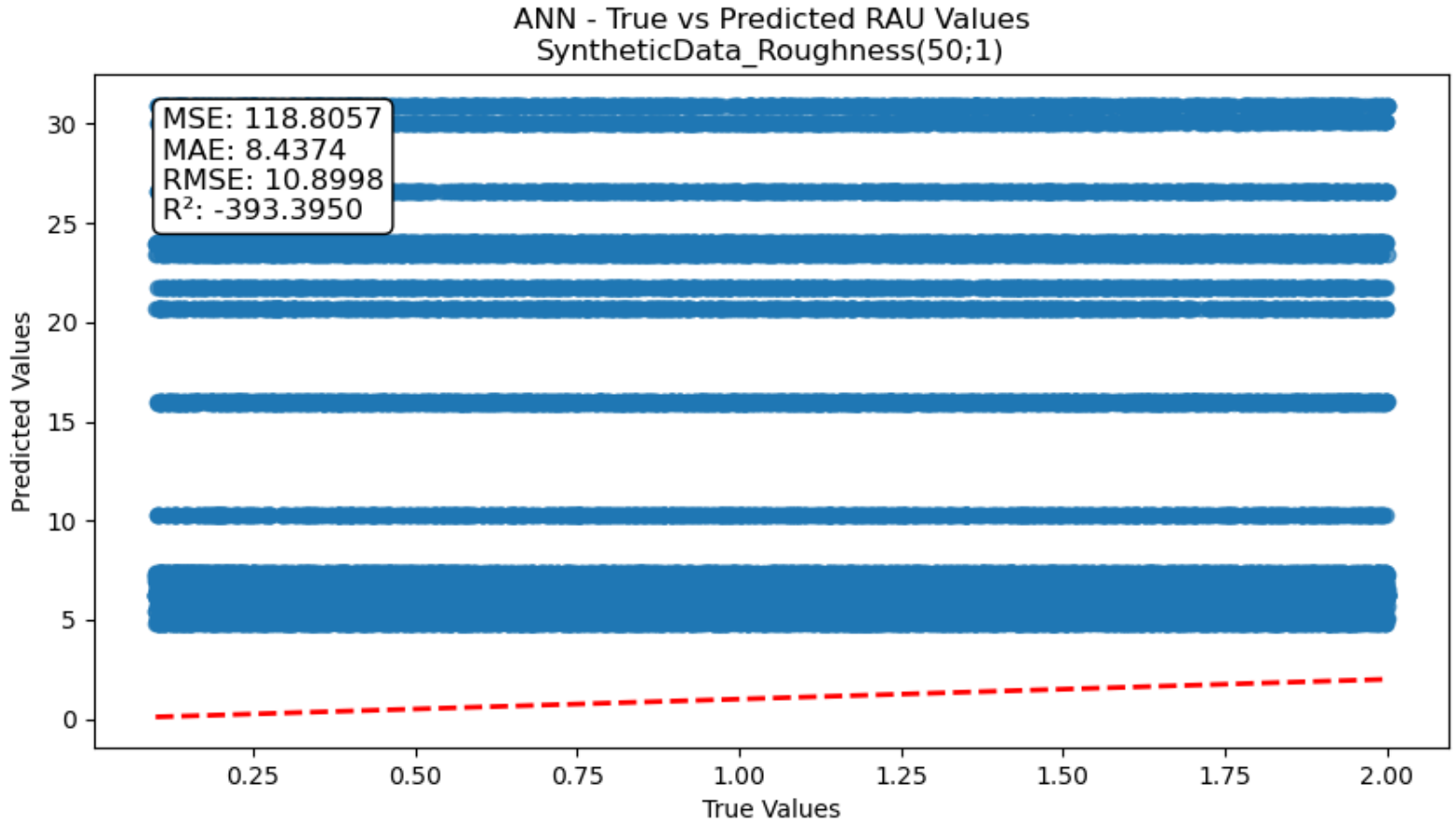

Figure 13: Predicted vs. True Roughness for ANN Model - Dataset with 50 Pipelines, 1 Loop

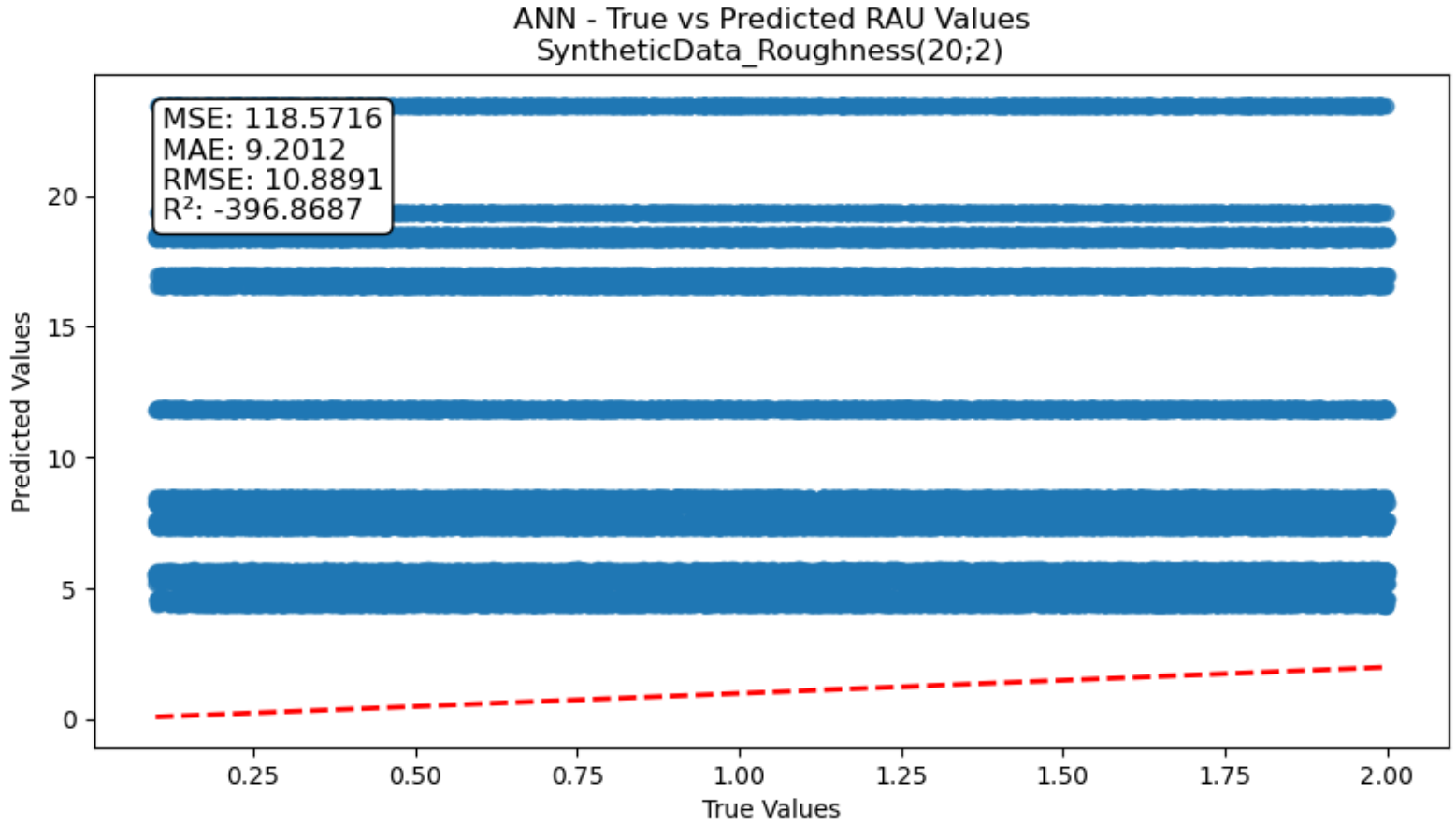

Figure 14: Predicted vs. True Roughness for ANN Model - Dataset with 20 Pipelines, 2 Loops

Appendix E: XGBoost (XGB) - Predicted vs. True Roughness Values

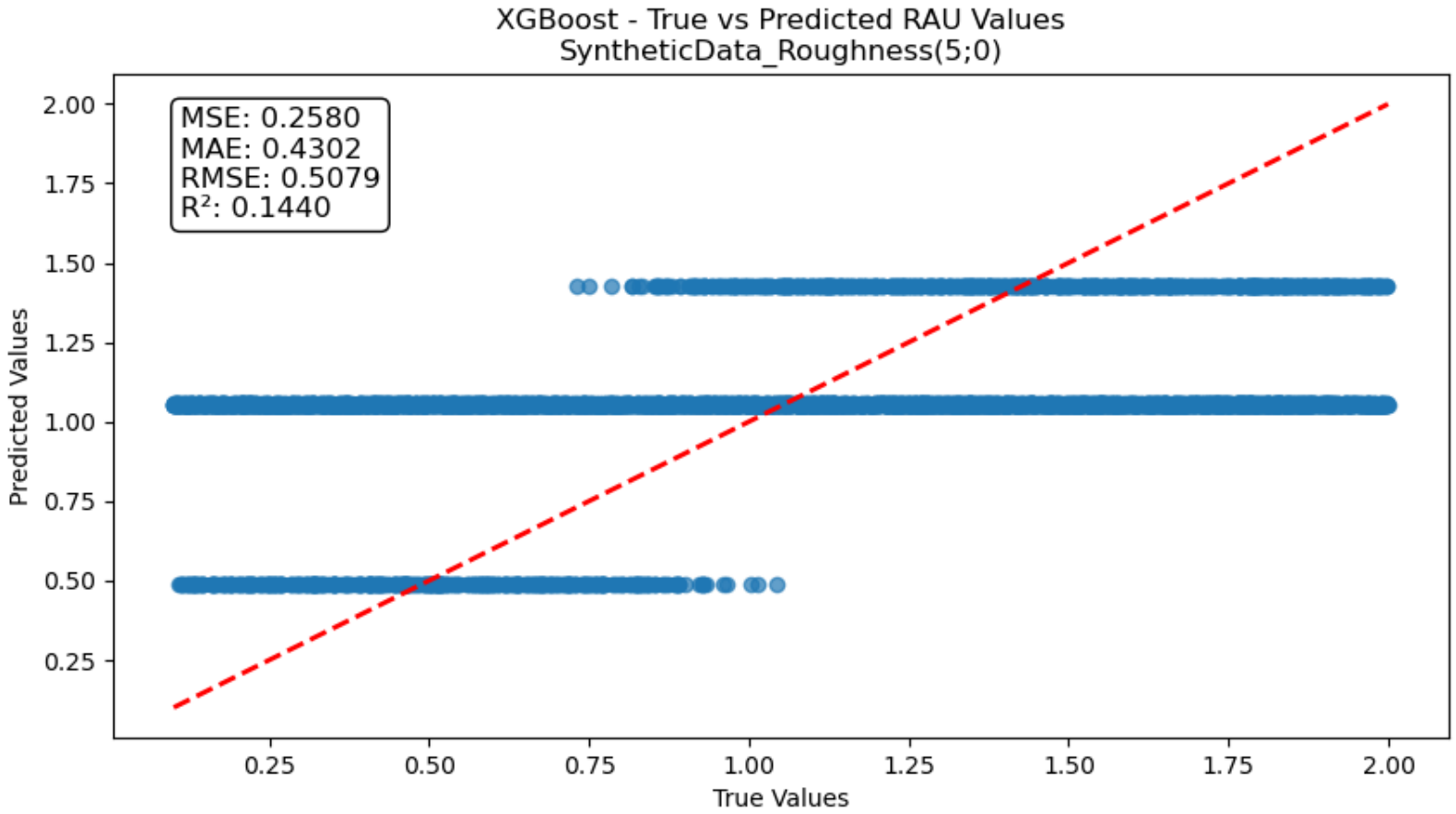

Figure 15: Predicted vs. True Roughness for XGB Model - Dataset with 5 Pipelines, 0 Loops

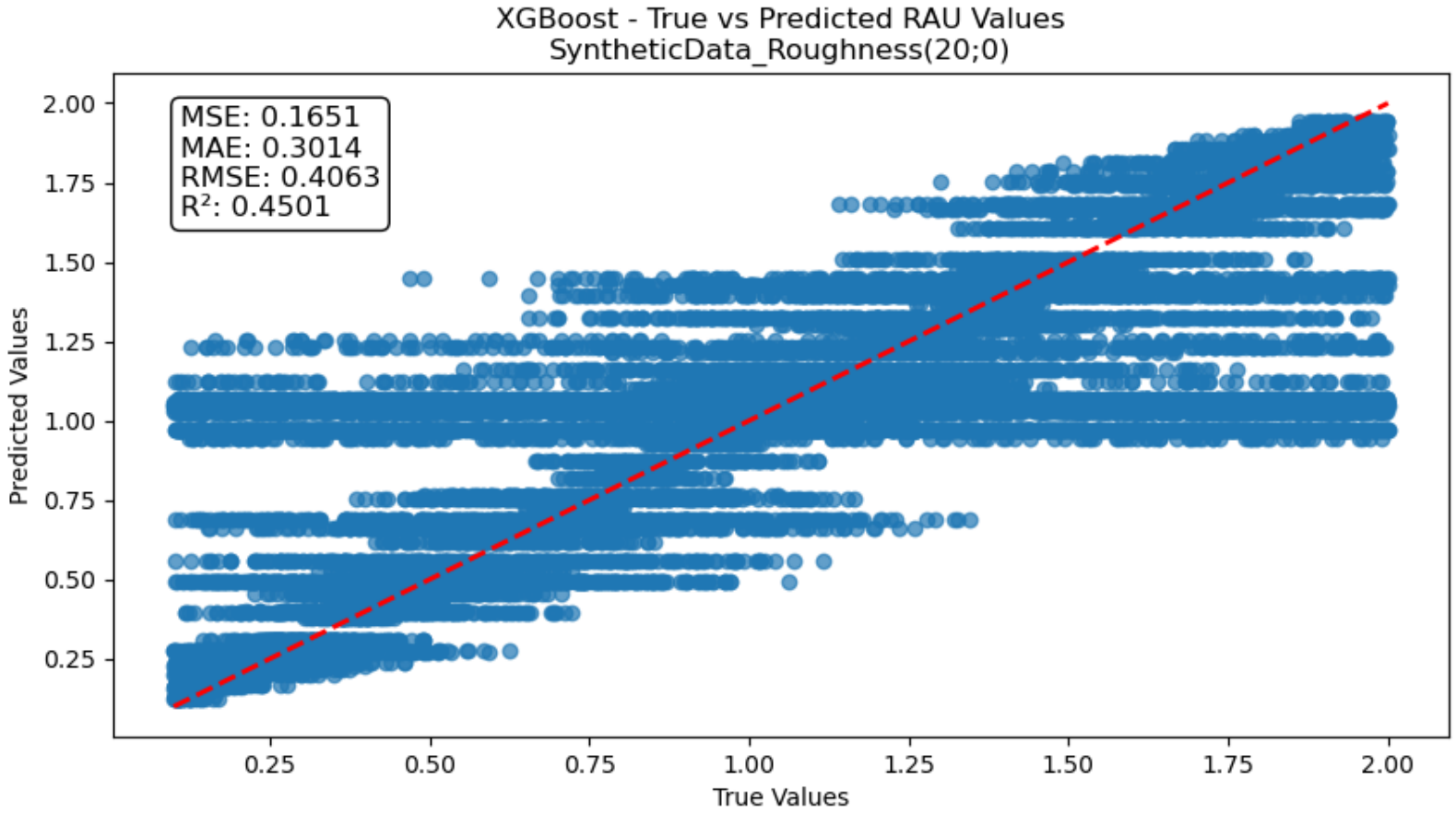

Figure 16: Predicted vs. True Roughness for XGB Model - Dataset with 20 Pipelines, 0 Loops

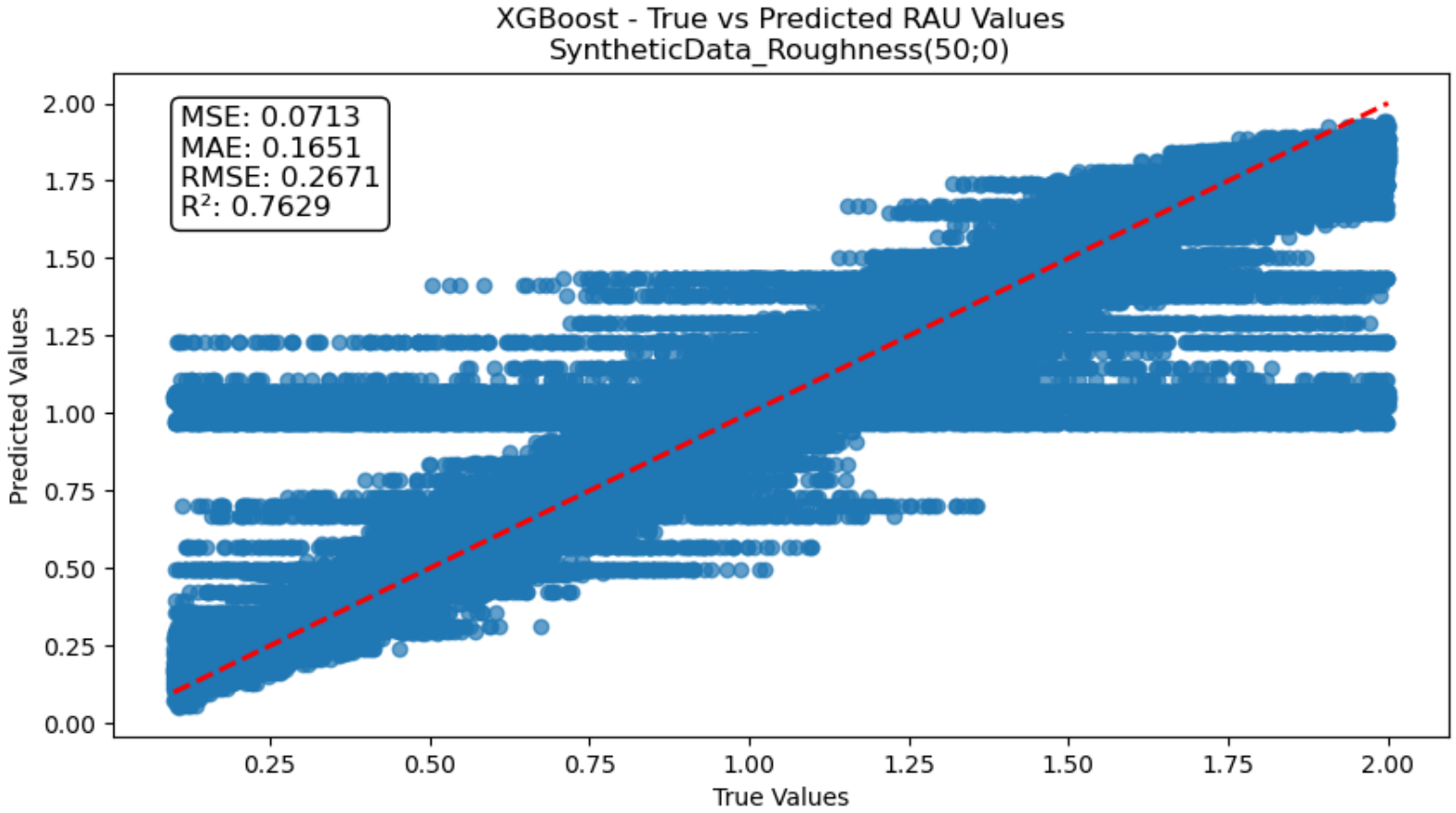

Figure 17: Predicted vs. True Roughness for XGB Model - Dataset with 50 Pipelines, 0 Loops

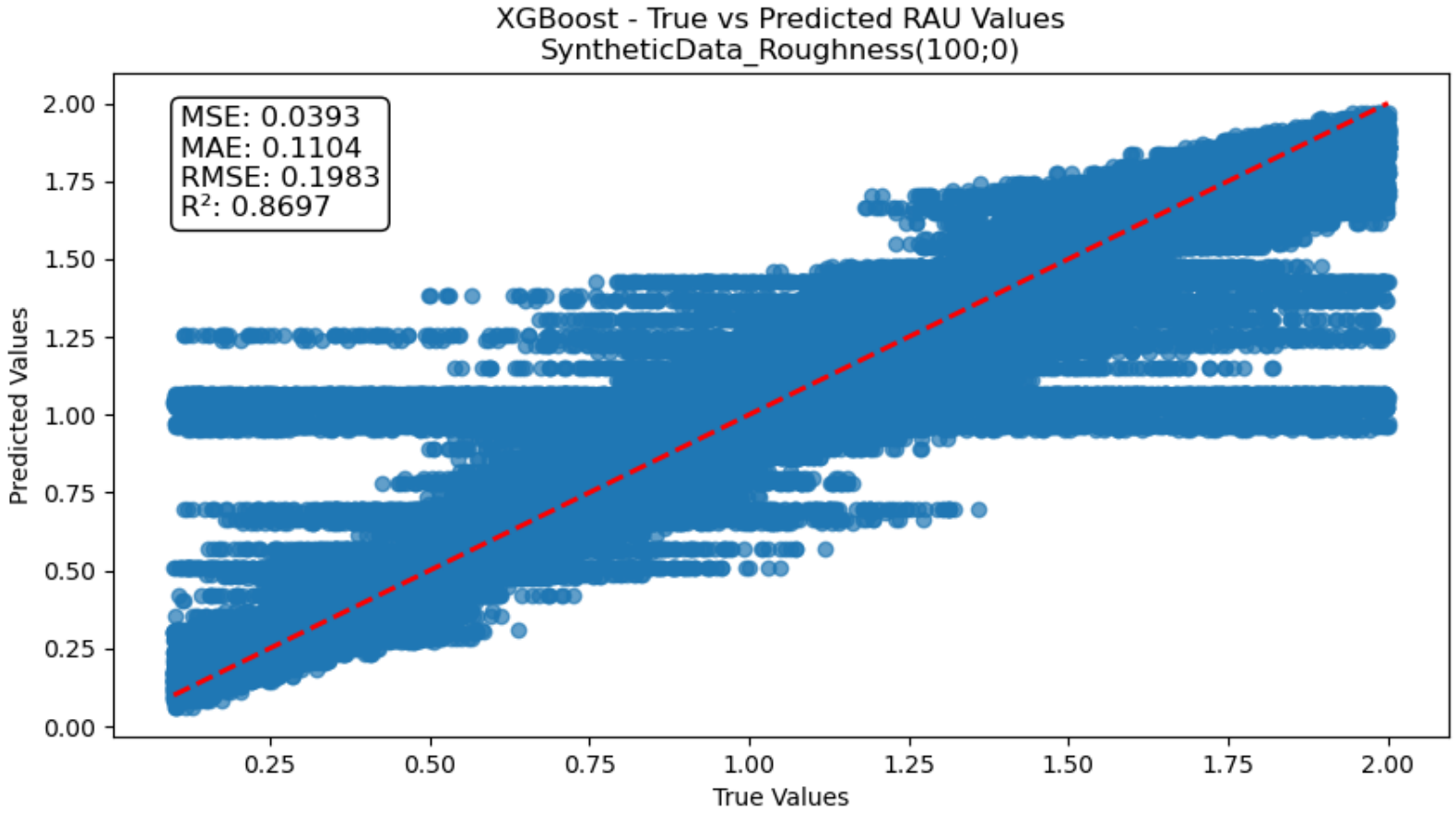

Figure 18: Predicted vs. True Roughness for XGB Model - Dataset with 100 Pipelines, 0 Loops

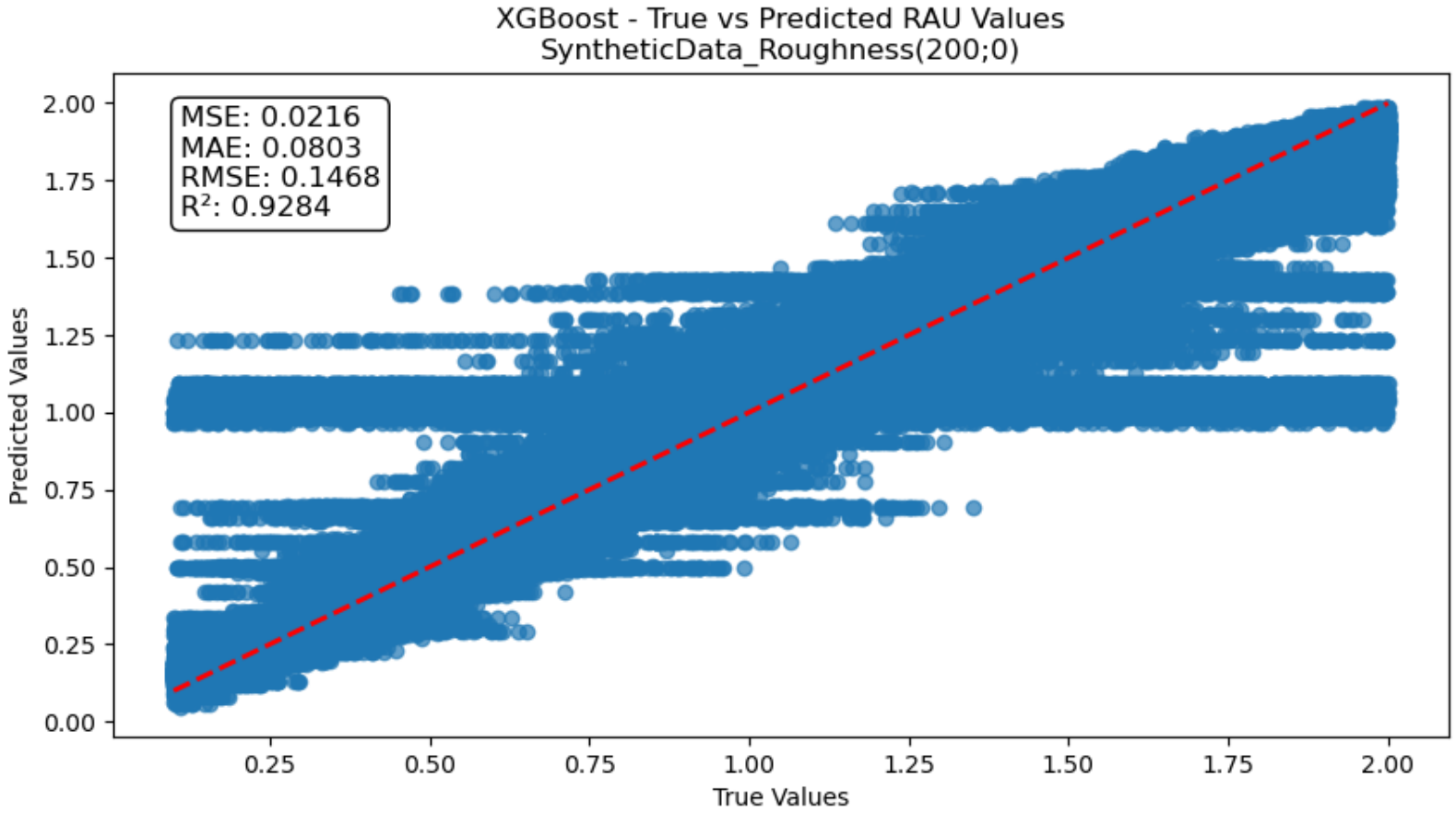

Figure 19: Predicted vs. True Roughness for XGB Model - Dataset with 200 Pipelines, 0 Loops

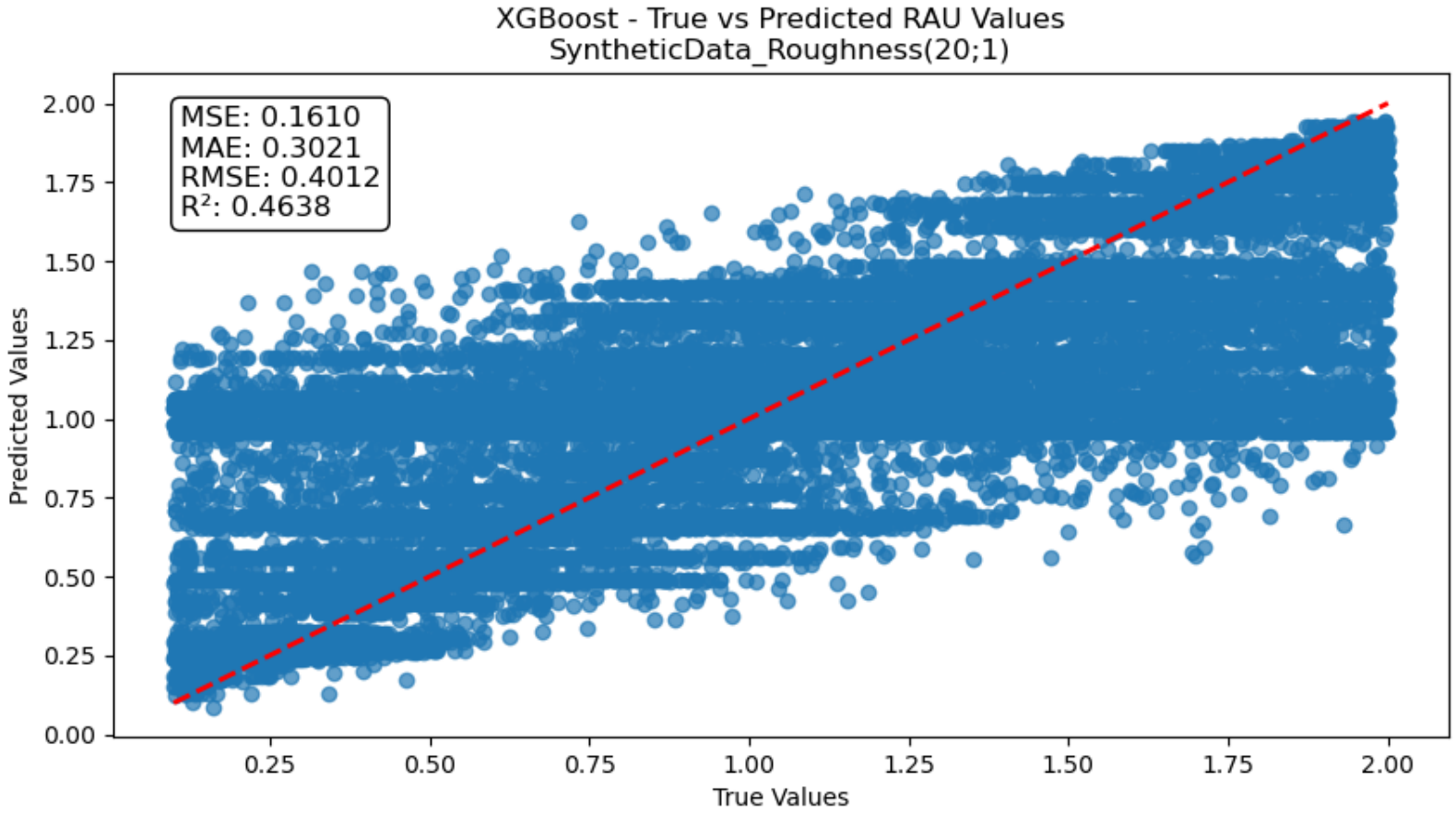

Figure 20: Predicted vs. True Roughness for XGB Model - Dataset with 20 Pipelines, 1 Loop

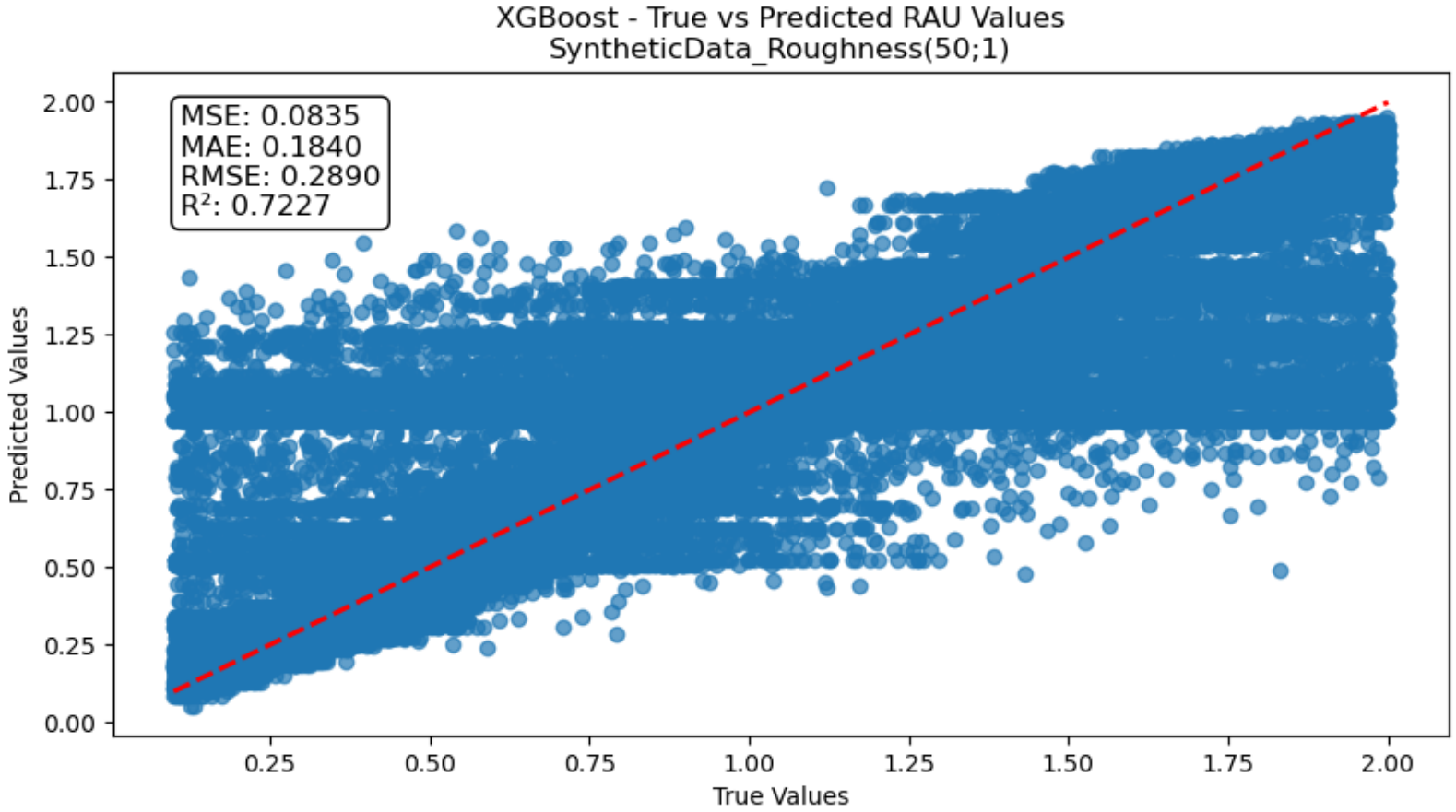

Figure 21: Predicted vs. True Roughness for XGB Model - Dataset with 50 Pipelines, 1 Loop

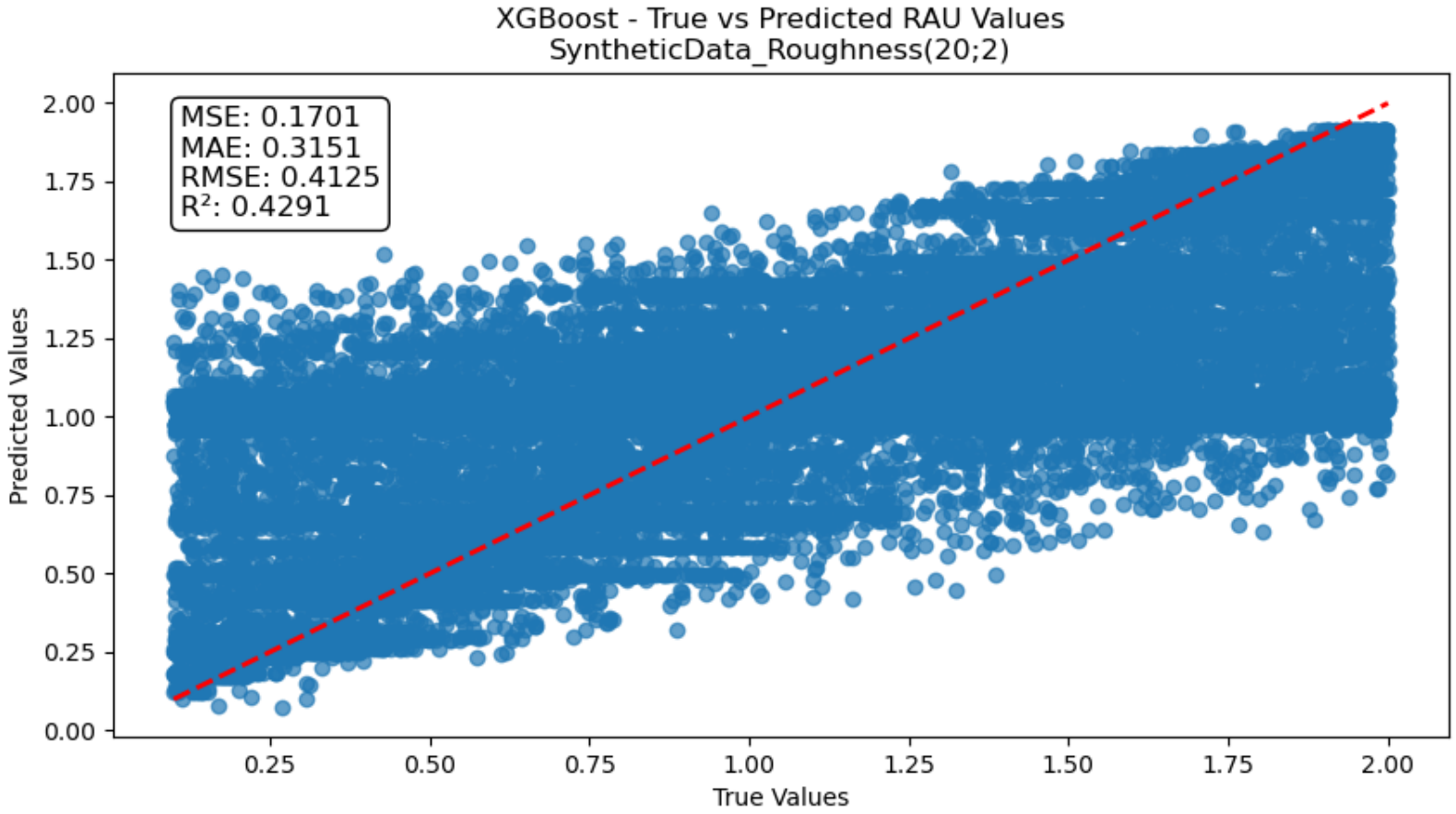

Figure 22: Predicted vs. True Roughness for XGB Model - Dataset with 20 Pipelines, 2 Loop

Appendix F: Graph Attention Network (GAT) - Predicted vs. True Roughness Values

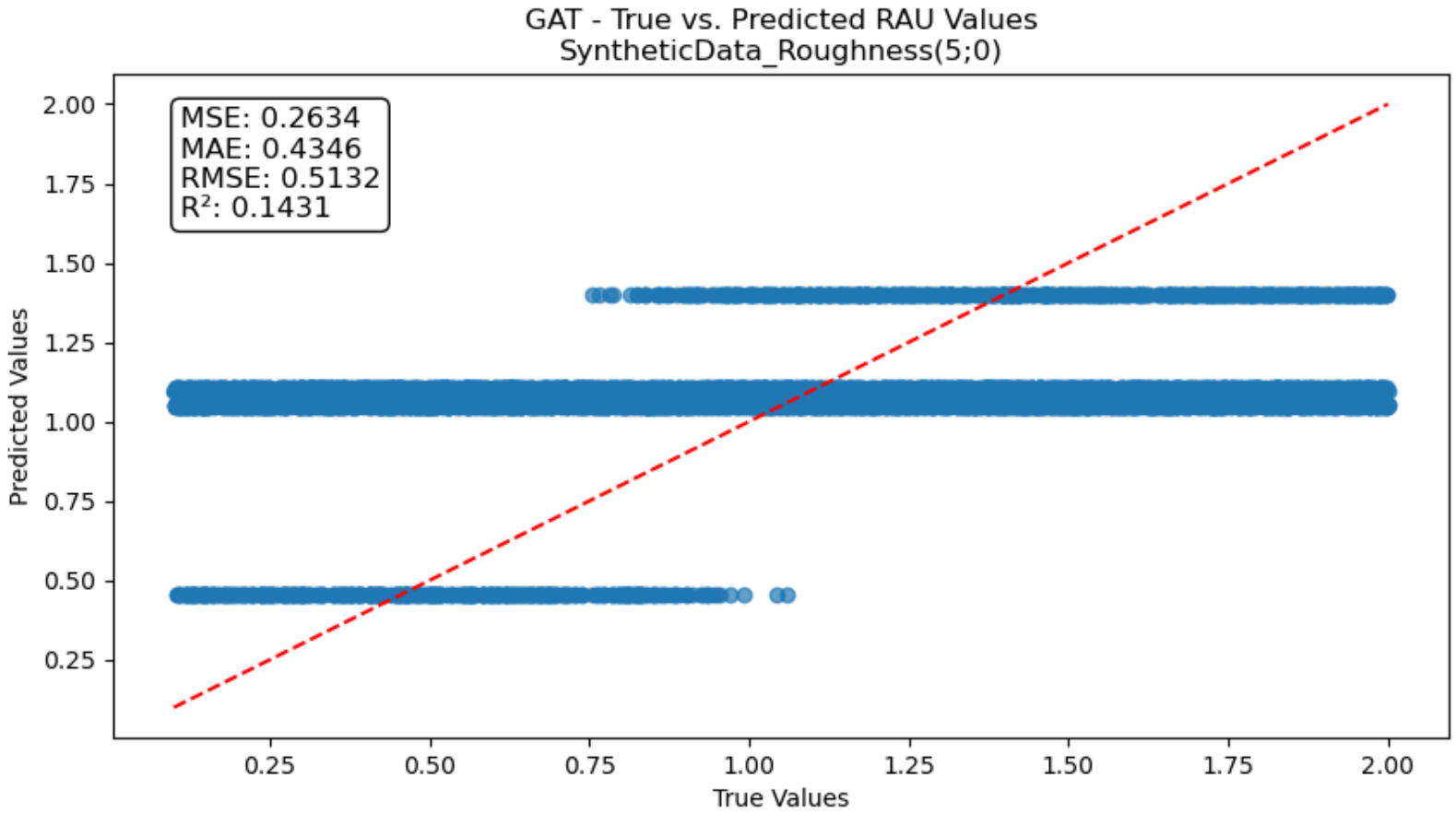

Figure 23: Predicted vs. True Roughness for GAT Model - Dataset with 5 Pipelines, 0 Loops

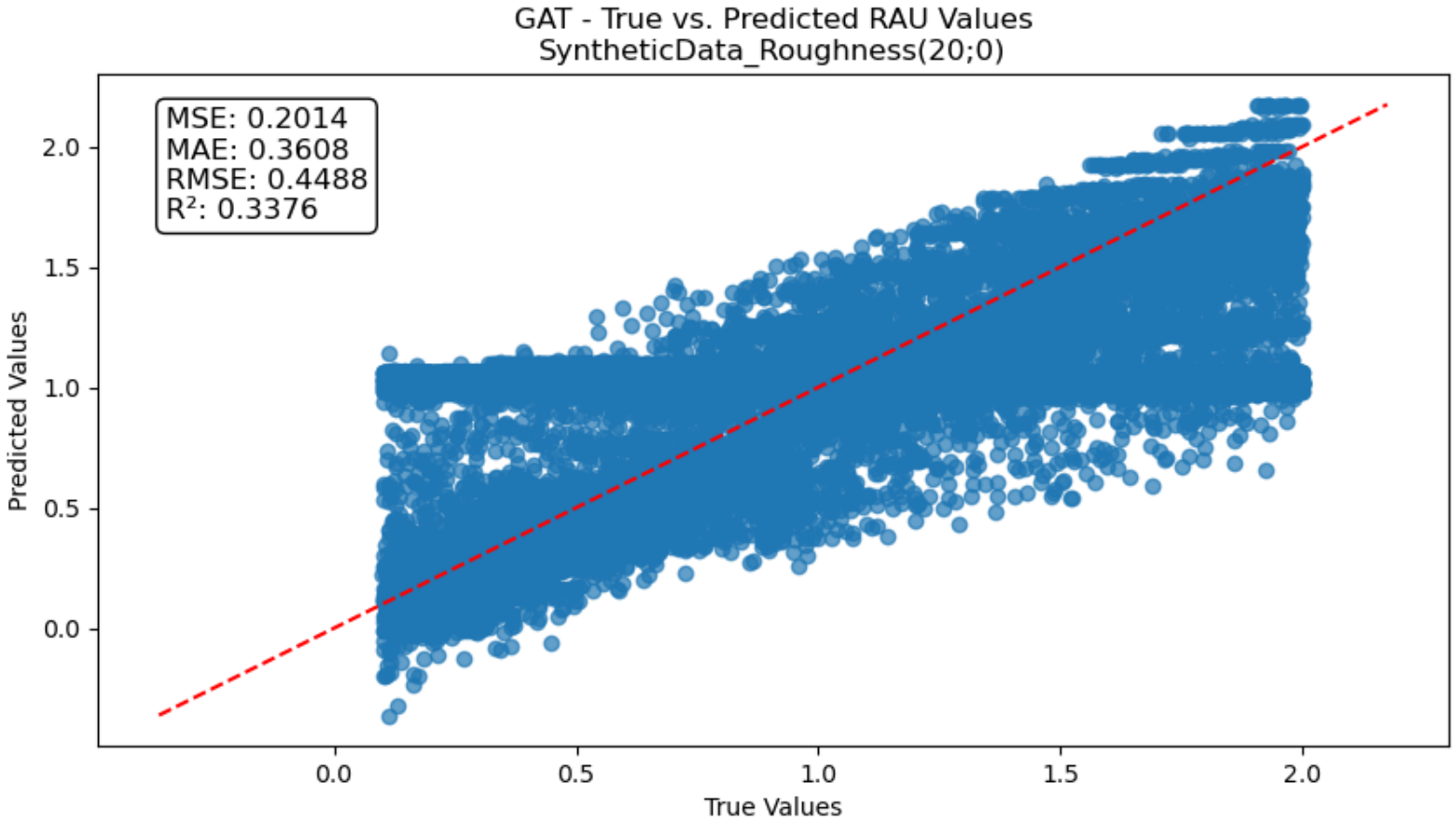

Figure 24: Predicted vs. True Roughness for GAT Model - Dataset with 20 Pipelines, 0 Loops

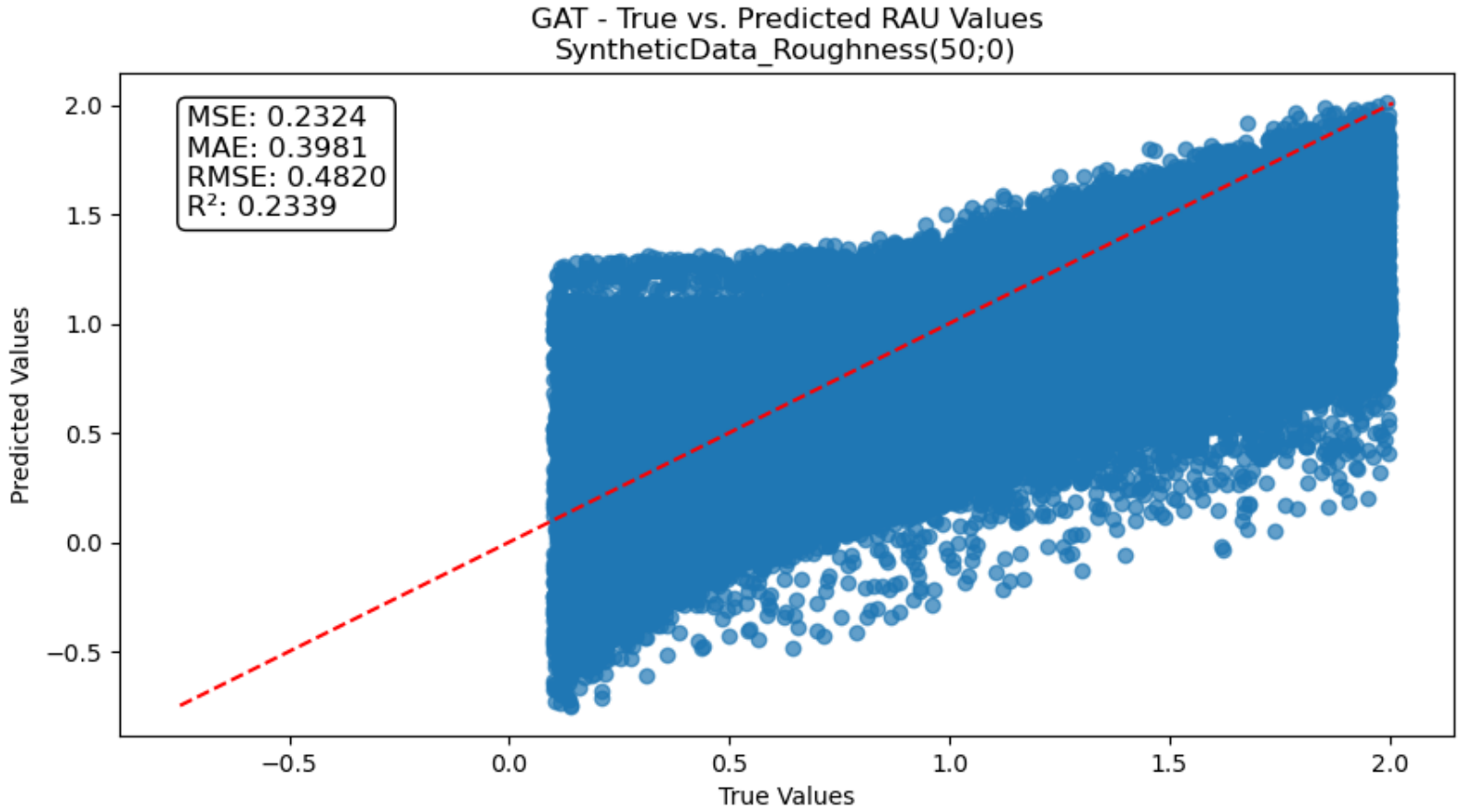

Figure 25: Predicted vs. True Roughness for GAT Model - Dataset with 50 Pipelines, 0 Loops

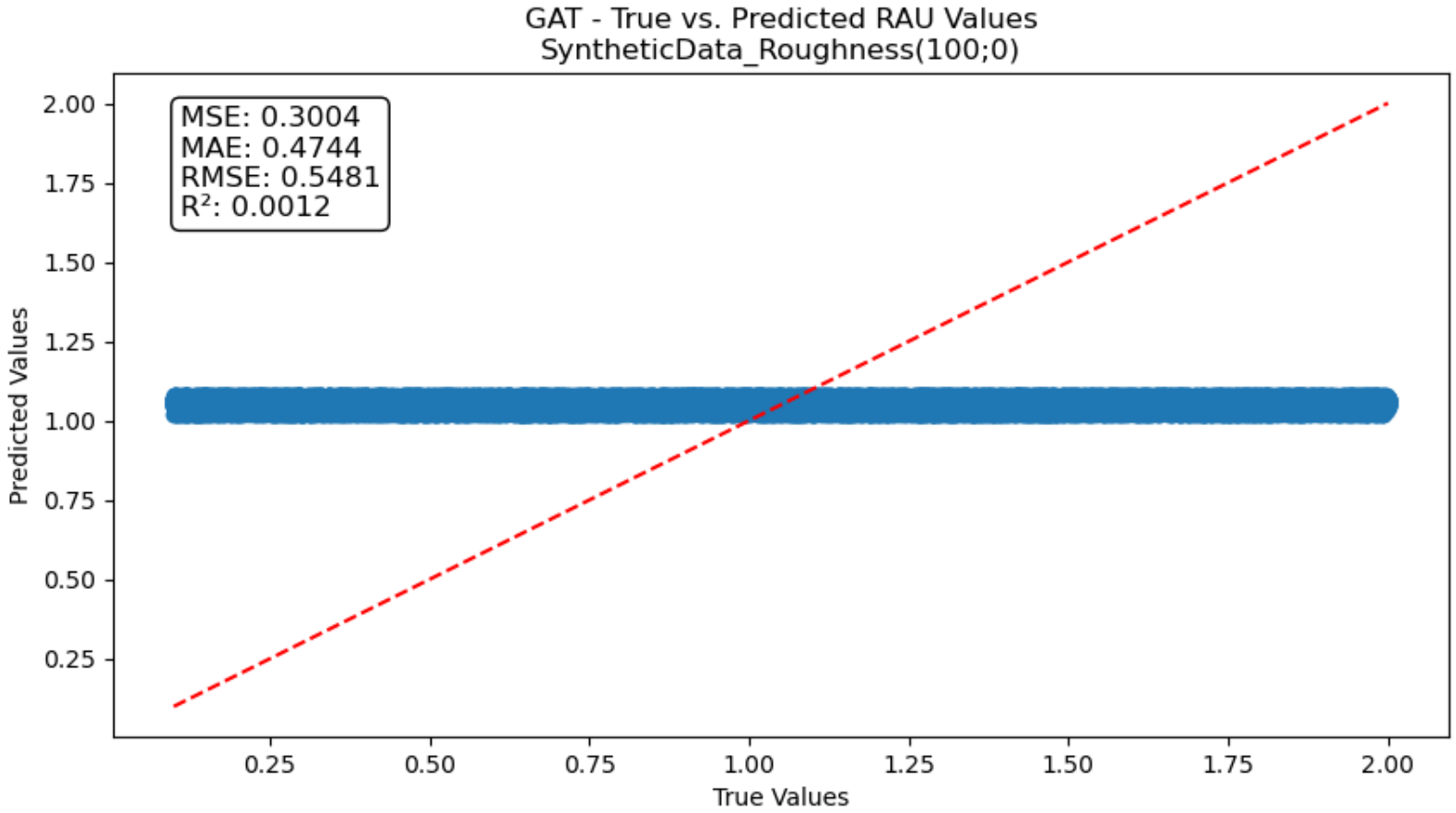

Figure 26: Predicted vs. True Roughness for GAT Model - Dataset with 100 Pipelines, 0 Loops

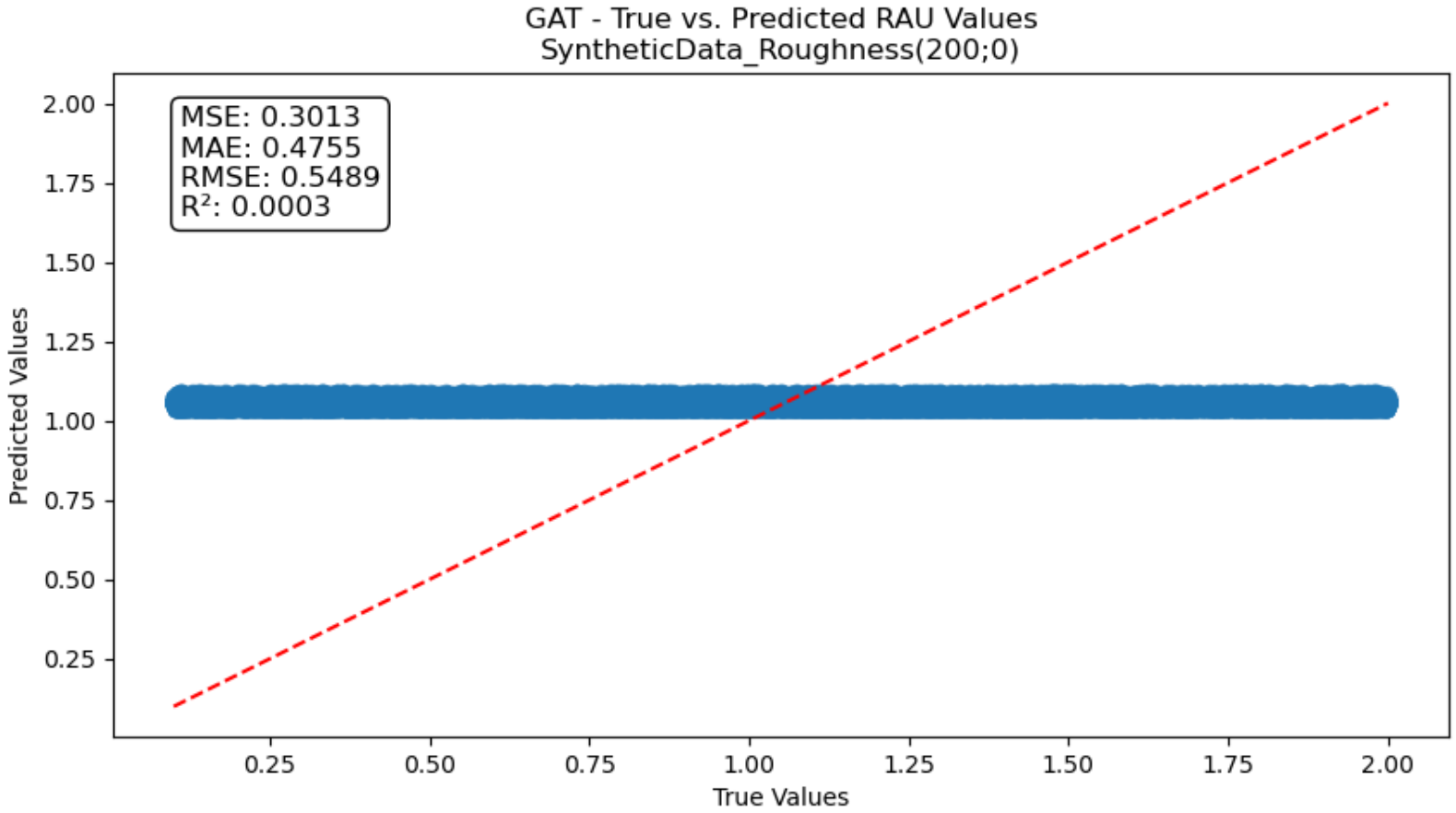

Figure 27: Predicted vs. True Roughness for GAT Model - Dataset with 200 Pipelines, 0 Loops

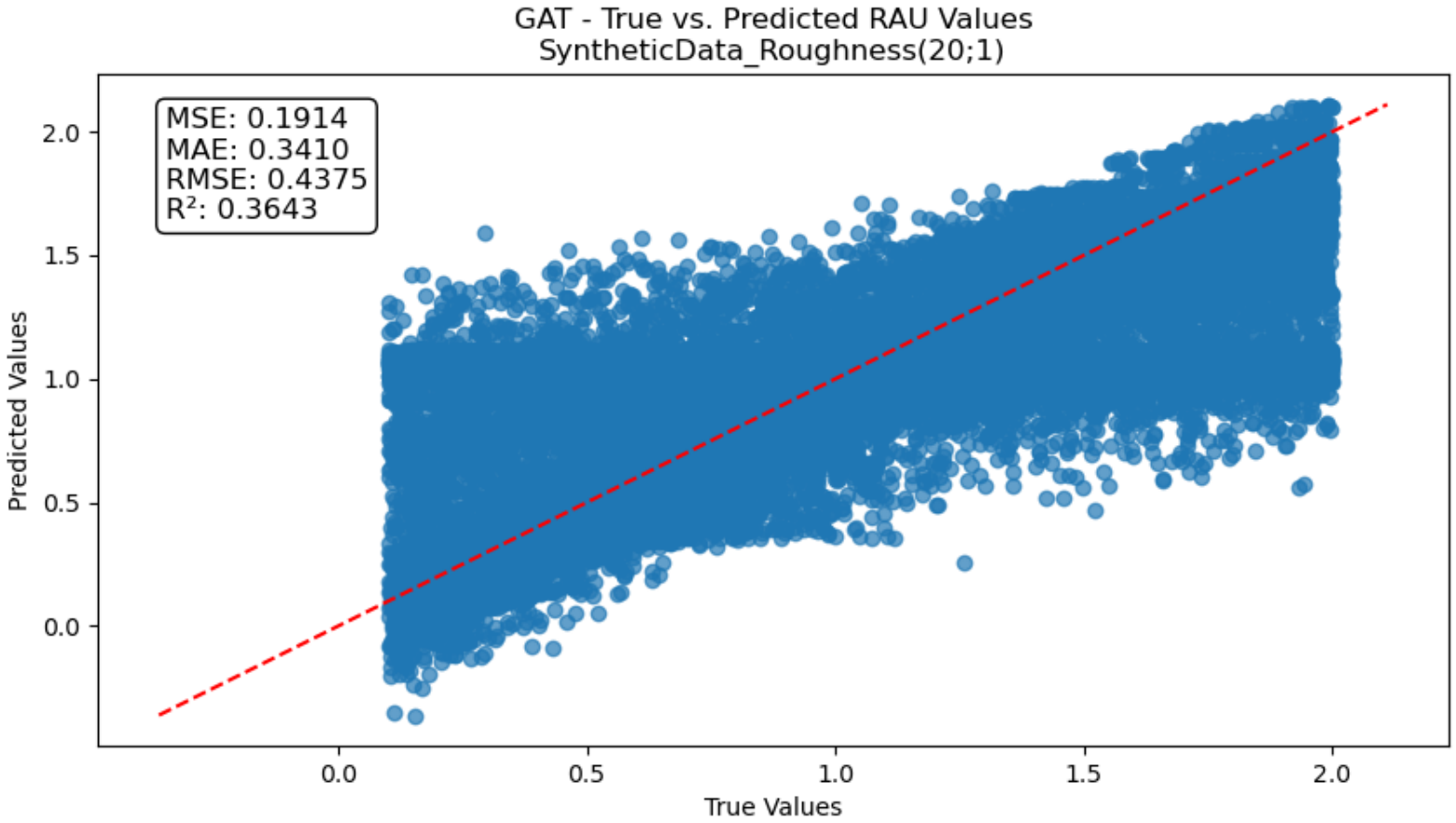

Figure 28: Predicted vs. True Roughness for GAT Model - Dataset with 20 Pipelines, 1 Loops

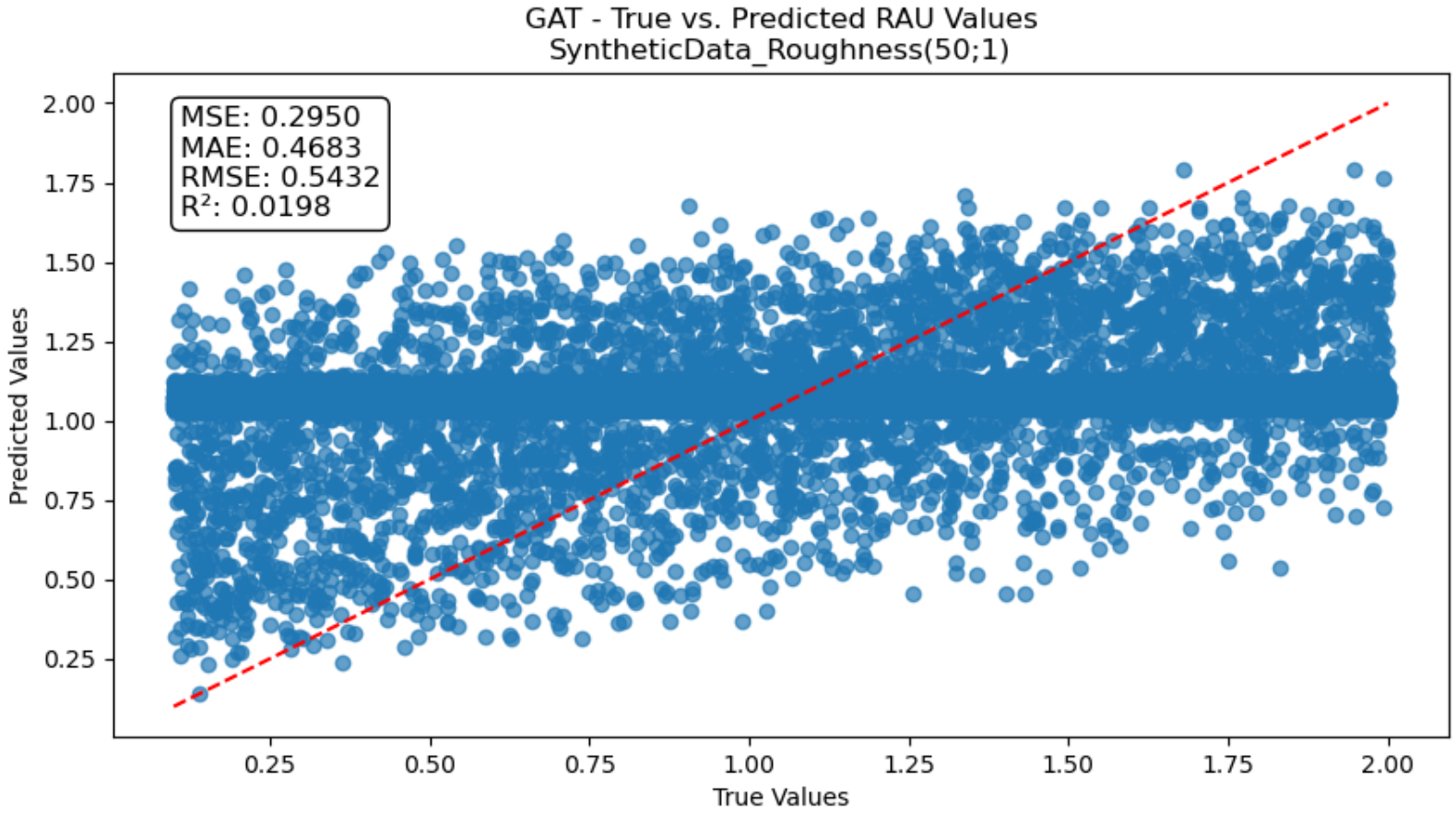

Figure 29: Predicted vs. True Roughness for GAT Model - Dataset with 50 Pipelines, 1 Loops

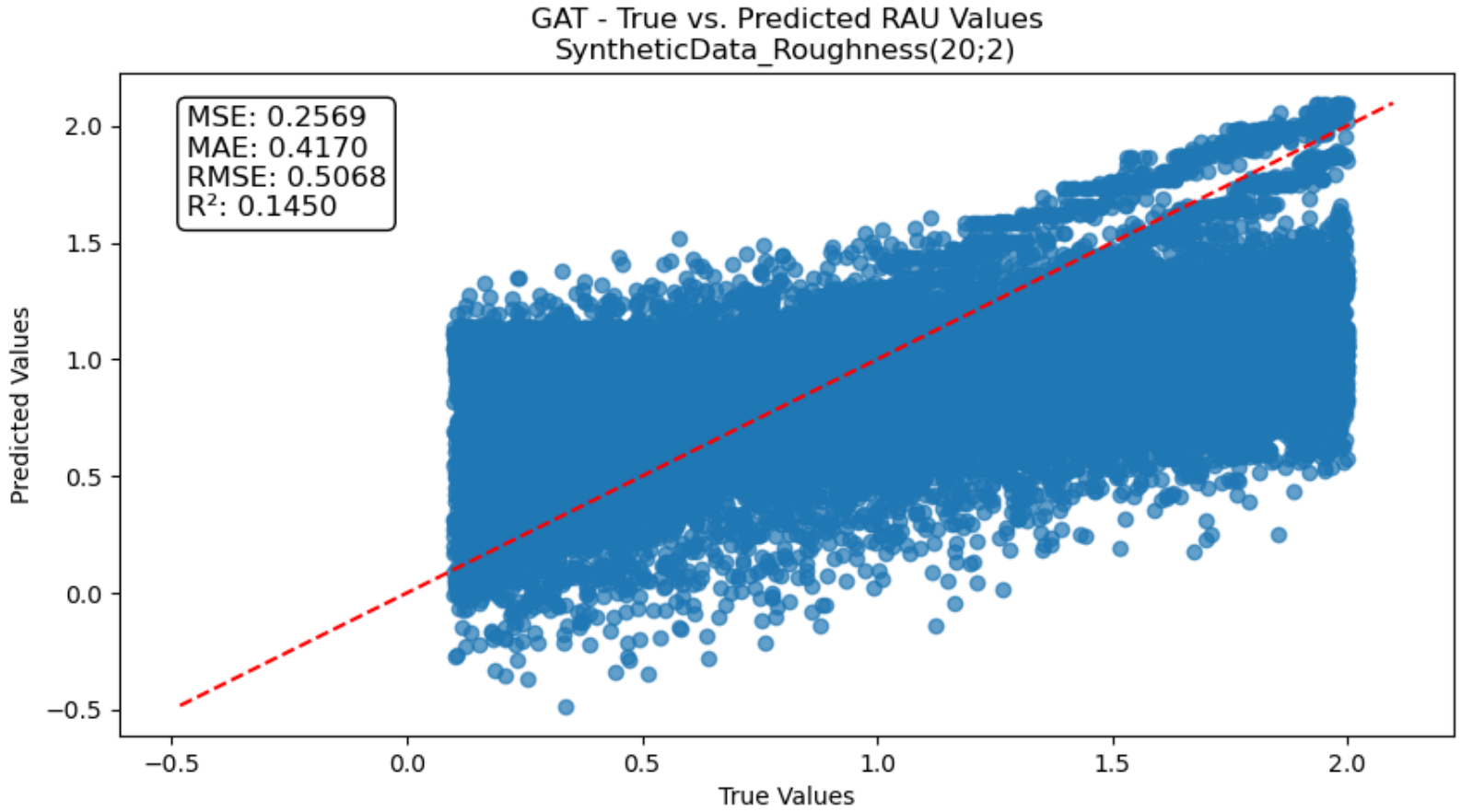

Figure 30: Predicted vs. True Roughness for GAT Model - Dataset with 20 Pipelines, 2 Loops

Appendix H: Overview of Evaluation Metrics for All Models and Network Configurations

Model	Pipes	Loops	R^2	MAE	MSE	RMSE
ANN	20	0	-593.1866	10.32	178.34	13.35
ANN	20	1	-424.0149	9.65	143.61	11.98
ANN	20	2	-396.8687	9.20	118.62	10.89
ANN	50	0	-394.2446	8.69	132.75	11.52
ANN	50	1	-393.3950	8.67	132.57	11.51
ANN	100	0	-506.2375	9.39	153.03	12.37
GAT	5	0	0.1431	0.43	0.26	0.51
GAT	20	0	0.3376	0.36	0.20	0.45
GAT	20	1	0.3643	0.34	0.19	0.44
GAT	20	2	0.1450	0.42	0.26	0.51
GAT	50	0	0.2339	0.40	0.23	0.48
GAT	50	1	0.0198	0.47	0.29	0.54
GAT	100	0	0.0012	0.47	0.30	0.55
GAT	200	0	0.0003	0.48	0.30	0.55
XGBoost	5	0	0.1440	0.43	0.26	0.51
XGBoost	20	0	0.4501	0.30	0.17	0.41
XGBoost	20	1	0.4638	0.30	0.16	0.40
XGBoost	20	2	0.4291	0.32	0.17	0.41
XGBoost	50	0	0.7629	0.17	0.07	0.27
XGBoost	50	1	0.7227	0.18	0.08	0.29
XGBoost	100	0	0.8693	0.11	0.04	0.20
XGBoost	200	0	0.9284	0.08	0.02	0.15