AI Methods for Anomaly Detection in Cyber-Physical Systems: With Application to Water and Agriculture

Md Nazmul Kabir Sikder

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computer Engineering

Feras A. Batarseh, Chair

Luiz Antonio Pereira DaSilva, Co-chair

Vasileios Kekatos

Sook Shin

Jactone Arogo Ogejo

November 25$^{\text{th}}$, 2024

Arlington, Virginia

Keywords: AI Assurance, Cyber Physical Systems, Outlier Analysis, Machine Learning, Deep Learning, Intelligent Water Systems, Context

AI Methods for Anomaly Detection in Cyber-Physical Systems: With Application to Water and Agriculture

Md Nazmul Kabir Sikder

(ABSTRACT)

In today's interconnected infrastructures, Cyber-Physical Systems (CPSs) play a critical role in domains including water distribution, agricultural production, and energy management. Modern infrastructures rely on a network of cyber-physical components—mechanical actuators, electrical sensors, and internet-connected devices—to supervise and manage operational processes. However, the increasing complexity and connectivity of these systems amplify their vulnerability to cyberattacks, necessitating robust cybersecurity measures and effective Outlier Detection (OD) methods. These methods are essential to prevent infrastructure failures, reduce environmental waste, and mitigate damages caused by malicious activities. Existing approaches often lack the integration of multiple operational metrics and context-driven techniques, hampering their effectiveness in real-world scenarios. In large CPSs—comprising hundreds or thousands of sensors, actuators, PLCs, IoT devices, and complex Control and Protection Switching Gear (CPSG)—the challenge of ensuring data quality, security, and reliability is costly.

Cyberattacks frequently appear as outliers or anomalies in the data and are launched with "*minimum perturbation*," making their detection significantly challenging. This dissertation proposes a novel framework, multiple pipelines, and AI-based methods to develop context-driven, data-driven, and assurance-focused OD solutions. Emphasis is placed on water and agricultural systems, illustrating the proposed framework's effectiveness, particularly through enhanced decision-making, operational efficiency, and cybersecurity measures.

A comprehensive survey of OD methods that employ Artificial Intelligence (AI) techniques establishes the foundational understanding of OD. This survey underscores that successful OD depends on domain knowledge, contextual factors, and assurance principles. Synthesizing these insights, the dissertation leverages synthetically generated SCADA data and GAN-produced poisoned data, as well as real-world SCADA data from Wastewater Treatment Plants (WWTPs), to identify outliers and address critical problems—such as forecasting tunnel wastewater overflows under extreme weather conditions—by applying Recurrent Neural Network (RNN)-based Deep Learning (DL) methods. Additionally, an AI-based decision support tool is introduced to detect anomalies in complex plant data and optimize operational set-points, thereby aiding Operation and Maintenance (O&M) in Water Distribution Systems (WDSs).

Similarly, in Agricultural Production Systems (APSs), which traditionally rely on reactive policies and short-term solutions, integrating advanced AI-driven OD methods provides farmers with timely, data-informed decisions that account for contextual changes resulting from outlier events. Machine Learning (ML) and DL methods measure associations, correlations, and causations among global and domestic factors, aiding in the accurate prediction of agricultural production. This contextual awareness helps manage policy, optimize resource utilization, and support precision agriculture strategies.

The main contributions of this dissertation include introducing a novel framework that integrates OD techniques with AI assurance and context-driven methodologies in CPSs; developing multiple pipelines and DL models that enhance anomaly detection, forecasting accuracy, and proactive decision support in WDSs and APSs; and demonstrating measurable improvements in cybersecurity, operational efficiency, and predictive capability using real-world and synthetic data. These efforts collectively foster more trustworthy and sustainable CPSs. Experimental results are recorded, evaluated, and discussed, revealing that these contributions bridge the gap between complex theoretical constructs and tangible real-world applications.

AI Methods for Anomaly Detection in Cyber-Physical Systems: With Application to Water and Agriculture

Md Nazmul Kabir Sikder

(GENERAL AUDIENCE ABSTRACT)

Recent unprecedented AI and sensor technology advancements are transforming all domains, including Water Distribution Systems (WDSs) and Agricultural Production Systems (APSs). With Industry 4.0, WDSs and APSs are undergoing a significant digital transformation to enable data-driven monitoring and control of utility operations. Incorporating cyber elements—such as sensors, actuators, data transmitters, receivers, Programmable Logic Controllers (PLCs), and Internet of Things (IoT) devices—aims to make these Cyber-Physical Systems (CPSs) more effective in Operation and Maintenance (O&M). However, this progress comes with a trade-off, as CPSs become increasingly vulnerable to security and safety threats. For example, in 2013, hackers seized control of a small Florida dam, releasing unprocessed water into nearby communities. Furthermore, on February 5th, 2021, a Florida water treatment plant (in Oldsmar, FL) was compromised when the hacker altered the levels of sodium hydroxide ($NaOH$) in the water—a chemical that would severely damage human tissue. Recent targeted attacks on infrastructure in Ukraine also highlight the risks facing critical infrastructures worldwide, including WDSs. These events suggest that current control operations are largely exposed, necessitating sophisticated learning algorithms that can estimate system states, detect anomalies, and mitigate the harm caused by such intrusions. Technology has fundamentally transformed agriculture as well, significantly impacting this domain. Agriculture, a vital occupation in numerous countries, now faces increasing global population pressures. The United Nations (UN) projects the population to reach 9.7 billion

by 2050, intensifying the strain on limited arable land. With only a 4% increase in cultivable land expected by 2050, farmers must do more with less. Traditional methods are insufficient to meet the soaring demands, as a 60% increase in food production is needed to feed an additional two billion people. This necessity for enhanced productivity and reduced waste drives the integration of AI into the agricultural sector. AI adoption not only accelerates efficiency but also increases production volumes, shortening the time from farm to market. This dissertation proposes novel, data- and context-driven Deep Learning (DL)-based methods and decision-support tools to enhance cybersecurity and anomaly detection within WDSs and APSs. Focusing on these critical infrastructures demonstrates how AI-driven strategies can effectively address real-world challenges and improve resilience, operational efficiency, and overall trustworthiness. The contributions of this dissertation include a framework and pipelines that incorporate contextual insights and AI assurance principles to improve anomaly detection and cybersecurity in these domains; the development of DL models tailored for identifying complex outliers and providing actionable decision-support, thereby optimizing resource allocation and ensuring sustainable operations; and validation of these approaches through experimental evaluations using real-world and synthetic data. Collectively, these efforts highlight significant improvements in reliability, efficiency, and scalability for critical infrastructure management, bridging the gap between theoretical advances in AI-driven anomaly detection and their practical application in WDSs and APSs.

Dedication

This dissertation is dedicated to my beloved son, Nehan Ishraf, my wife, Israt Zahan, and my parents. Your unwavering love, encouragement, and steadfast support have been the bedrock of my journey. To Israt, your reminders of perseverance and resilience have been a source of constant inspiration. To Nehan, your pure and unconditional love has fueled my ambition and kept my spirit alive through every challenge. To my parents, your sacrifices, wisdom, and unshakable belief in me have provided the foundation upon which I have built this academic milestone.

I also extend this dedication to my mentors and teachers. Your guidance, encouragement, and passion for knowledge have profoundly shaped my intellectual pursuits. The lessons you imparted have not only deepened my understanding but also instilled in me a lifelong commitment to learning and discovery.

This dissertation is a reflection of the collective support, sacrifices, and beliefs of everyone who has walked this path with me. Each of you has played an indispensable role in my growth, and for that, I am eternally grateful.

Acknowledgments

I extend my heartfelt gratitude to all those who have contributed to completing this dissertation. First and foremost, I am deeply thankful to my advisor, Professor Dr. Feras A. Batarseh, for his invaluable guidance, unwavering support, and insightful feedback throughout this journey. His mentorship and expertise have been instrumental in shaping my research and academic growth. I am also indebted to my committee members, Professor Luiz A. DaSilva, Professor Vassilis Kekatos, Professor Sook S. Ha, and Professor Jactone A. Ogejo, for their constructive critiques and valuable suggestions that have greatly enriched the quality of this work.

I am fortunate to have collaborated with exceptional colleagues and fellow researchers from the A3 lab[1], who have inspired and challenged me. Their diverse perspectives and collective efforts have significantly contributed to the research presented in this dissertation.

My gratitude extends to the faculty and staff of the ECE department, whose dedication to fostering a stimulating academic environment has been an essential part of my educational experience. I am also thankful to the Commonwealth Cyber Initiative (CCI), whose financial support has enabled me to carry out this research with the necessary resources.

Finally, none of this would have been possible without the unwavering support of my family and friends. Their encouragement, patience, and understanding have sustained me through the highs and lows of this academic journey. I am deeply thankful for their love and belief in my potential.

This dissertation is a testament to the collective efforts of all those who have supported and guided me along the way. Their contributions have left an indelible mark on my academic and personal growth, and I am profoundly grateful for that.

[1] https://ai.bse.vt.edu/

Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
AIA	Artificial Intelligence Assurance
ALSP	Adversarial Logging Scoring Pipeline
APS	Agricultural Production System
BATADAL	BATtle of the Attack Detection ALgorithms
CAI	Secure AI
cP_2O	Context-driven Prediction, Protection, and Optimization Framework
CPS	Cyber-Physical System
CPSG	Control and Protection Switching Gear
CSO	Combined Sewer Overflow
DAGMM	Deep Autoencoding Gaussian Model
DBN	Deep Belief Network
DeepAg	Deep Learning Framework for APS
$DeepH_2O$	Deep Learning Pipeline for WDS Cybersecurity
DL	Deep Learning
DSS	Decision Support System

EAI	Ethical AI
FAI	Fair AI
FWS	Florida Water Supply
GAN	Generative Adversarial Network
GNN	Graph Neural Network
HBOS	Histogram-Based Outlier Detection
HCAE	High Confidence AutoEncoder
INFLO	Influenced Outlier Factor
IoT	Internet of Things
KNN	K-Nearest Neighbor
LIME	Local Interpretable Model-agnostic Explanations
LOCI	Local Correlation Integral
LOF	Local Outlier Factor
LoOP	Local Outlier Probabilities
LSTM	Long Short-Term Memory
MAA	Model Agnostic Assurance
ML	Machine Learning
MM-LSTM	Multivariate Multistep Long Short-Term Memory
OD	Outlier Detection

P_2O	Prediction, Protection, and Optimization Framework
PDP	Partial Dependence Plots
PLC	Programmable Logic Controller
RADS	Real-time AI-based Decision Support
RBWU	Riviera Beach Water Utility
RTU	Remote Terminal Unit
SAI	Safe AI
SCADA	Supervisory Control And Data Acquisition
SHAP	SHapley Additive exPlanations
SVM	Support Vector Machine
TAI	Trustworthy AI
TGCN	Temporal Graph Convolutional Network
VAE	Variational AutoEncoder
WDS	Water Distribution System
WWTP	Wastewater Treatment Plant
XAI	Explainable AI

Chapter 1

Motivation, Background, and Contributions

This chapter introduces AI methods and their relevance in CPSs. It outlines the significance of using AI in WDSs and APSs, highlighting its pivotal role in today's critical infrastructures.

1.1 An Introduction to AI and CPS

CPSs are- *Intelligently networked systems with embedded sensors, processors, and actuators that are designed to sense and interact with the physical world (including human users), and support real-time, guaranteed performance in safety-critical applications*" (Wang et al. [9], DHS [10]). CPS is a multi-dimensional and complex scheme incorporating industrial components and IoTs to construct advanced and automated production environments (Mansour [11]). The systems mostly comprise networking modules, sensors, and actuators that are appropriate in the automation, power, civil structure, medicine, and development field (Liu et al. [12]). In general, it's a scheme where cyber applications and external operations are supported in an integrated manner.

To elaborate further, CPSs operate concurrently through physical and cyber layers to achieve enhanced operational performance. Figure 1.1 presents a basic schematic diagram of a CPS. It consists of two major layers for operations: a physical layer and a cyber layer. The physical

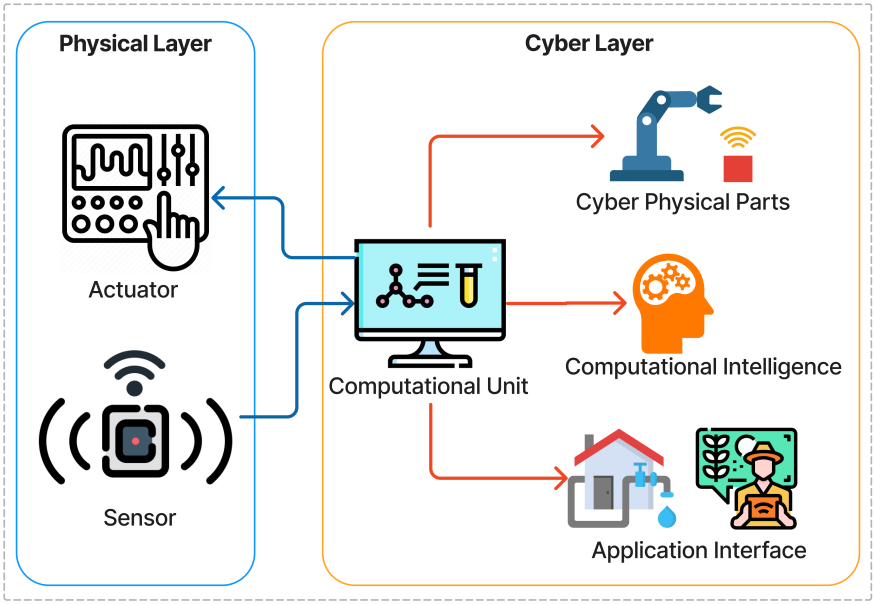


Figure 1.1: A Typical CPS System Schematic Diagram

layer typically encompasses a wide array of components, such as sensors and actuators, which play a pivotal role in various domains. In the context of WDSs, these sensors monitor factors like water flow rates, pressure levels, and quality parameters. In APSs, they might track soil moisture, temperature, and crop health. These sensors continuously gather data from the real-world environment and transmit it to the cyber layer, which is often a networked infrastructure. Within the cyber layer, sophisticated algorithms and control systems process this data in real-time, making intelligent decisions. For instance, WDSs can optimize the flow of water to ensure efficient distribution while minimizing waste. In agricultural settings, it can guide irrigation systems to provide crops with the precise amount of water they need for optimal growth. The network layer then communicates the appropriate responses back to the physical layer's actuators through predefined protocols. In WDSs, this might involve adjusting the flow of water pumps or opening and closing valves. In APSs, it could trigger actions like activating irrigation equipment or adjusting the position of solar-powered trackers for optimal sunlight exposure. CPSs integrate the physical and cyber layers to enhance the

efficiency, reliability, and sustainability of various systems, including WDSs and APSs.

The rapid advancement of technology is leading to intelligent devices, enabling monitoring of sophisticated tasks such as water flow and quality estimation in water systems, improving clean water supply management, and reducing environmental waste (Sikder et al. [5]). Similarly, in agricultural systems, integrating smart components such as sensors to monitor soil moisture, weather conditions, and crop health aids farmers in making data-driven decisions for improved productivity (Kulkarni et al. [3]). While the interconnection of devices presents opportunities for technological progress and automation, it simultaneously amplifies vulnerabilities to cyber threats and compromises data privacy and security. This complexity also diminishes system explainability to the naked human eye. As data flow among nodes in the network, safeguarding the confidentiality, integrity, and authenticity of sensitive information becomes of utmost importance. The intricate network structure may introduce vulnerabilities, raising the risk of cascading failures or disruptions with significant implications for essential operations (Jeffrey et al. [13]). Network components offer potential entry points for adversaries to exploit sensitive data and undermine the critical infrastructure's functionality. Critical applications, such as water, agriculture, and energy systems, are being targeted by intentional cyber threats and hacking state and non-state teams (Hassanalieragh et al. [14]). Over the recent years, special consideration has been given to improving CPS monitoring and security.

1.1.1 Definition of AI and Its Applications in CPSs

AI encompasses reasoning, planning, learning, processing, and the ability to manipulate objects (de Fine Licht and de Fine Licht [15], Kammerer [16], Sun et al. [17]). It involves the integration of cognitive architectures capable of human-like performance, encompassing

aspects like motivation, emotion, and personality (Sun et al. [17]). In 1950, Alan Turing introduced an acceptable operational interpretation of AI called The Turing Test (Turing [18]). This test determines whether a computer has AI capabilities by assessing whether a human interrogator can differentiate between written responses from a person or a computer after posing questions. Turing deliberately designed the test to exclude direct interaction between the interrogator and the computer, as simulating a person's physical presence is unnecessary to gauge intelligence.

The name AI was coined in 1956 by c computer scientist John McCarthy (McCarthy [19]), and work started in earnest soon after World War II. AI encompasses various subfields, ranging from the general (learning and perception) to the specific, such as playing chess, proving mathematical theorems, writing poetry, driving a car on a crowded street, and diagnosing diseases. AI is relevant to any intellectual task; it is truly a universal field. Also, it is becoming more prevalent in every aspect of my life (Figure 1.2), especially aiding CPSs to solve real-world problems across different application domains such as water, agriculture, and energy systems (Russell [20]). Following are some examples of how AI is assisting CPSs to address challenges, improving efficiency in real-world applications:

1. **WDS Management:** AI-powered CPS is revolutionizing the management of WDS in urban areas. Water distribution networks are complex, and ensuring efficient O&M while minimizing intentional anomalies and losses is a significant challenge (Batarseh and Kulkarni [21]). AI methods can accurately analyze data from sensors in the distribution network, including pressure, flow, and water quality data, and provide necessary insights about the system's health.

2. **APS Management:** AI-driven CPS in agriculture uses data from sensors, drones, and satellite imagery to optimize irrigation, fertilization, and crop management (Gurrapu

et al. [2], Dharmaraj and Vijayanand [22], Gurrapu et al. [23, 24]). By applying AI methods, farmers can make data-driven decisions to enhance crop yield, reduce resource wastage, and improve sustainability.

3. **Energy Optimization:** AI methods can optimize energy consumption in buildings, manufacturing plants, and transportation systems (Kulkarni et al. [3], Zahraee et al. [25]). Also, it can optimize energy usage to achieve energy efficiency and cost savings by analyzing sensor data and external factors like weather conditions.

4. **Predictive Maintenance:** AI-powered CPS can analyze sensor data from machines and equipment to predict potential failures or maintenance needs Kulkarni et al. [3], Achouch et al. [26]. By detecting anomalies in the data, O&M can be scheduled proactively, reducing system downtime after failure and minimizing costly breakdowns.

5. **Environmental Monitoring:** AI-powered methods can analyze environmental data, such as air quality, water levels, and weather patterns, to monitor ecological health (Himeur et al. [27]). This information helps predict and manage environmental risks and address pollution and natural disasters.

6. **Healthcare Monitoring:** AI-powered tools in healthcare utilize wearable devices and IoT sensors to monitor patients' health conditions continuously (Alshamrani [28]). AI methods analyze the data to detect early signs of health issues and provide timely alerts to healthcare professionals, facilitating better patient care and timely interventions.

7. **Smart Transportation:** AI is revolutionizing the transportation domain by optimizing traffic flow, predicting traffic congestion, and managing public transportation schedules (Khawar et al. [29]). For autonomous vehicles, AI methods can enable real-time decision-making to navigate safely and efficiently through traffic.

8. **Smart Grids:** AI can aid in managing power distribution and consumption in smart grids (Omitaomu and Niu [30], Usman et al. [31]). AI methods analyze data from various sources to balance energy demand and supply, optimize grid operations, and integrate renewable energy sources effectively.

9. **Supply Chain Optimization:** AI-powered applications optimize supply chain operations by analyzing data from various stages, including manufacturing, inventory management, logistics, and customer demand (Pournader et al. [32]). This optimization reduces costs, improves delivery times, and enhances overall efficiency.

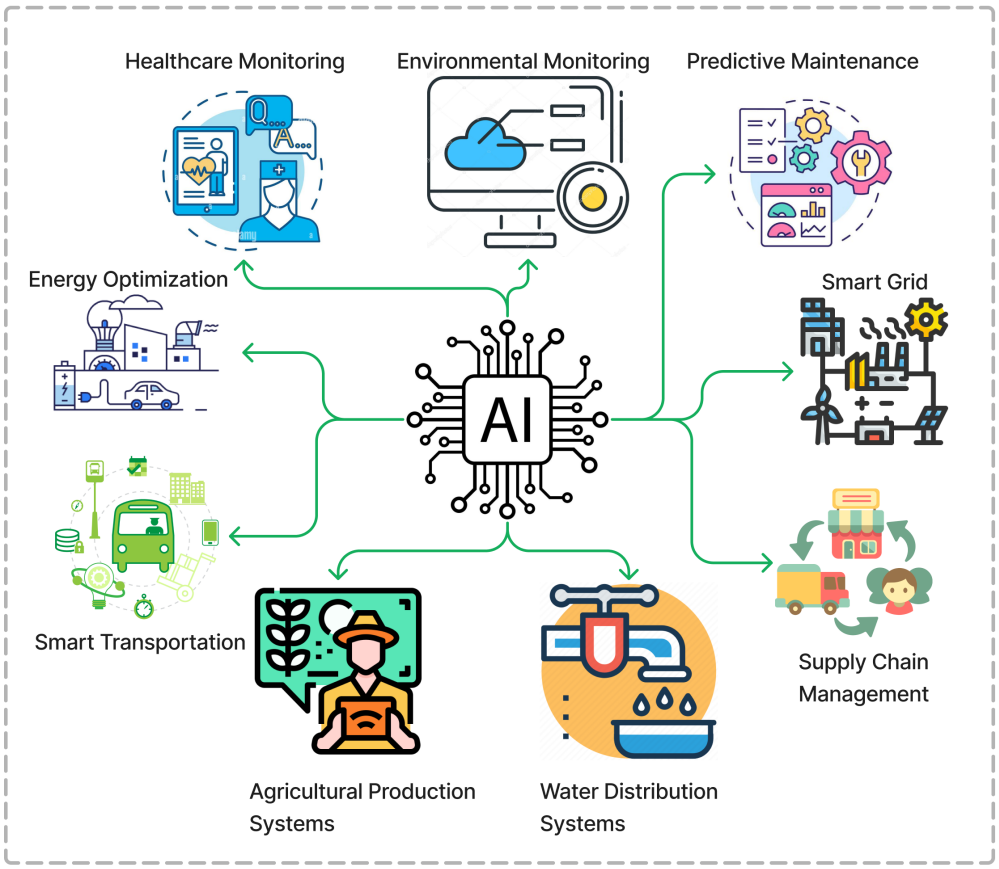


Figure 1.2: AI Applications in CPS

Water and agricultural systems are crucial among all infrastructures and are directly linked to public health, safety, and food security. With Industry 4.0, they are becoming complex,

and their interconnected nature makes them susceptible to cyber threats and anomalies, necessitating the application of AI solutions for enhanced cybersecurity (Gurrapu et al. [2], Kulkarni et al. [3]). The potential consequences of cyberattacks on WDSs can go beyond the infrastructures to mass problems. For example, disruptions of clean water distribution can lead to public health issues, while poor decisions on agricultural production pipelines can disrupt food production and have significant economic impacts. AI-driven solutions can safeguard these infrastructures against cyber threats, mitigating risks to public health and the economy (Sikder et al. [5]).

1.1.2 Introduction to AI for Outlier Detection

An outlier or anomaly can be defined as an abnormality, deviant, or discordant data point from the remaining dataset in data science literature. According to Hawkins [33], *"an outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism."*

According to Aggarwal [34], in data mining literature, normal data are also known as *"inliners"*. Often, in real-world applications, such as fraud or intrusion detection systems, outliers are sequential, not single data points within a sequence. For instance, a network intrusion is an event in a sequence intentionally caused by an individual. Correctly identifying the anomalous event helps to handle those sequences. In most conventional cases, OD methods have two outcomes: binary labels and outlier scores (Aggarwal [34]). Outlier scores impose each data point's level or degree of *"outlierness"*. Scores naturally rank outlier points and provide various information about the methods. However, they don't represent a concise summary with small group sizes. Binary labeling represents whether a data point is a strong outlier or an inliner. OD methods can provide outlier scores, which can then be converted to

binary labels for learning purposes. For that, a threshold is selected based on the statistical distribution of the dataset.

Binary labels provide less information regarding the degree of outlierness; however, in most applications, it is the desired outcome for decision-making. Defining how much deviation is sufficient from a normal data point for an outlier is a subjective judgment. Datasets from real applications might contain embedded noise; however, analysts might not be interested in keeping such noise. Therefore, investigating significant deviation is a prime decision for OD methods. To comprehend this problem clearly, Figure 1.3(a) and 1.3(b) illustrate two-dimensional feature spaces. It is evident that clusters are identical in both figures. However, considering a single data point "A" in Figure 1.3(a) seems different from the rest of the data points. Therefore, "A" in Figure 1.3(a) is clearly an outlier. However, point "A" in Figure 1.3(b) is surrounded by noise, and it's quite difficult to say if it is noise or an outlier. When designing OD algorithms, normal and outlier boundary conditions must be precise and specific to application requirements.

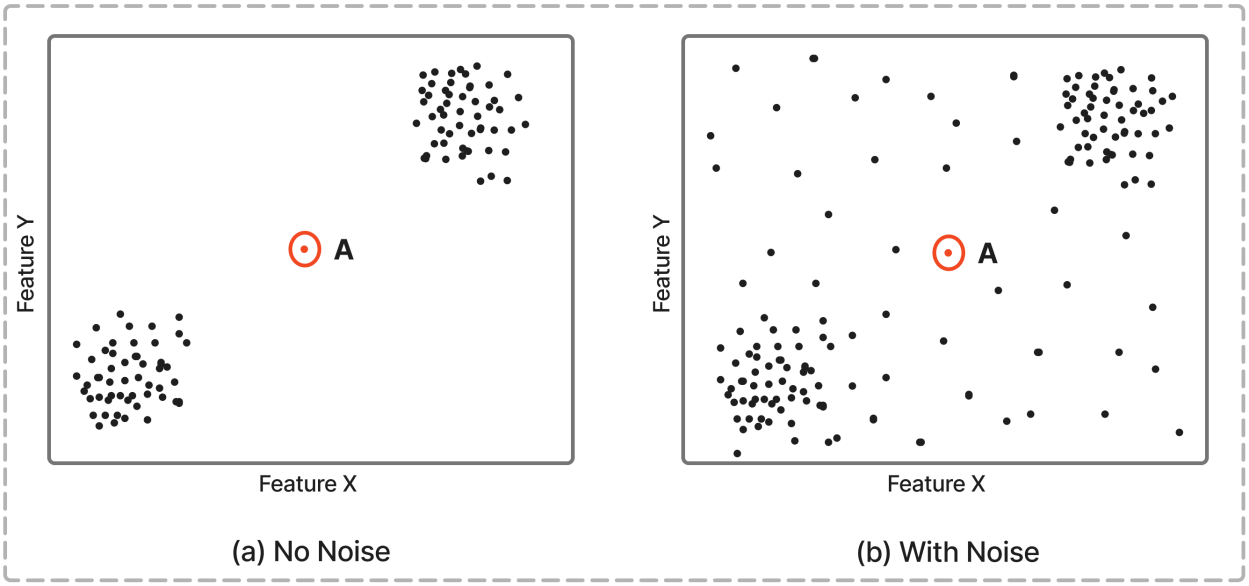


Figure 1.3: Anomalies and Noise in Data

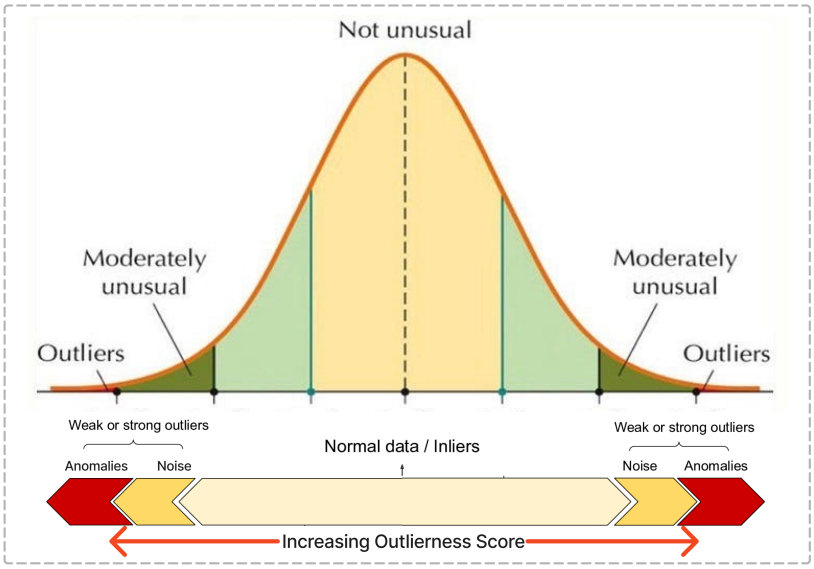


Figure 1.4: A Typical Data Spectrum With Noise and Outliers

In unsupervised learning methods, noise is defined as weak anomalies that don't meet the criteria of being an outlier. For instance, data points close to the boundary are mostly considered noise (as presented in Figure 1.4). Often, the separation criteria of these data points are subjective and depend on the interest of application-specific demands. Real data points generated from noisy environments are difficult to detect using scores. That is because noise represents deviated data points and requires domain experts to select the threshold between noise and outliers to satisfy application requirements. Success in OD depends on data modeling, where every application has its unique data management requirements. Evidently, the OD technique needs to process the attribution in the data and be sensitive enough to understand the underlying data distribution model. By properly examining the data model, contextual outliers can also be achieved. Aggarwal et al. [35] proposed a concept of linkage outlier by analyzing social networks. Here, nodes that don't show any connection with each other are likely to be outliers; therefore, data distribution models play an important role in designing OD algorithms.

AI excels in outlier/anomaly detection (Sikder and Batarseh [36]), continuously monitoring

system data, and establishing baseline behavior. During the development of AI-based applications, data are created by several generational processes or observations collected from one or multiple entities. in a CPS, outlier instances are generated when one or a collection of entities behave in an unusual manner. Therefore, it is essential to understand the behavior of outliers to diagnose a system's health and predict potential system failures. Some of the most popular OD applications are intrusion detection methods (Aggarwal et al. [35]), credit card fraud detection (Porwal and Mukund [37]), medical diagnosis (Gebremeskel et al. [38]), sensor events in critical infrastructure, precision agriculture, earth science, and law enforcement (Bordogna et al. [39]); detailed discussions and examples are provided in Chapter 2. One of the recently successful applications of OD is credit card fraud identification, where AI-based OD algorithms are used to find if sensitive information, such as customer identification or a card number, is fraudulent or stolen (Porwal and Mukund [37]). In this context, unusual buying patterns are observed, especially large transactions or irregular buying activities.

1.1.2.1 Introduction of Cybersecurity for CPS

According to Aslan et al. [40], "*The term cybersecurity refers to a set of technologies, processes, and practices to protect and defend networks, devices, software, and data from attack, damage, or unauthorized access*". The complexity of cybersecurity has grown due to the rapid proliferation of interconnected devices, systems, and networks, compounded by advancements in digital infrastructure and the economy. This has led to a notable surge in cyberattacks, many of which have severe consequences. Table 1.1 paints a picture of United States cyber security statistics from the latest studies and reports by comparitech[1].

Cyber adversaries continuously evolve and are associated with nation-states and criminal groups, deploying increasingly sophisticated attack methods to target even the most knowl-

[1]https://www.comparitech.com/blog/information-security/us-cyber-crime-statistics/

edgeable targets (Chithaluru et al. [41]). This ongoing evolution has amplified the scale and impact of cyberattacks, prompting the need for intelligence-driven cybersecurity strategies to counter evolving threats and effectively manage the abundance of data. Esteemed bodies like the National Institute of Standards and Technologies (NIST) are promoting proactive and adaptive approaches, advocating real-time assessments, continuous monitoring, and data-driven analyses to identify, prevent, detect, respond to, and document cyberattacks to mitigate future security breaches (Umezawa et al. [42]).

Table 1.1: Thirty-five United States Cyber Security Statistics (Source: omparitech.com)

Statistic	**Details**
1. Almost 89.7% of United States organizations saw at least one successful attack over a one-year timeframe	A 6.7% increase from 2020, higher than Mexico, Spain, Germany, Colombia, and China.
2. Ransomware affected almost 78.5% of United States organizations within a year	The Second most impacted country, behind Australia.
3. United States organizations upped security budgets by almost 4% in 2021	3.8% increase, spending 13.7% of IT budgets on security.
4. Nearly 89% of United States businesses prefer using security products that utilize ML and AI	Moderate to strong preference; Saudi firms (98%) and German companies (71.6%).
5. The United States endures the largest portion of ransomware Trojan attacks	Highest share of attacked users, followed by Kazakhstan, Iran, and China.
6. Almost 59% of organizations were hit by ransomware in 2020 and dropped to 51% in 2021	This made the United States the second most attacked country (up from 6th the year before) behind India (68%) and Austria (57%)
7. Attacks were stopped before data were encrypted in 25% of cases	20% less success in encrypting data during ransomware attacks in 2021.
8. One-quarter of United States organizations paid the ransom	25% paid, more than six times higher than Spain.

Continued on the next page

Table 1.1 – Continued from previous page

Statistic	**Details**
9. United States companies paid an average of $620,000 in remediation costs	Average remediation cost increased by over 50% in 2021.
10. About 9 in 10 organizations have cyber security insurance	90% have cyber insurance policy, and 75% have ransomware coverage.
11. Almost 12% of users tried to open a phishing link in 2020	11.82% attempted to open a phishing link in 2020.
12. The United States was the third-largest source of spam	Russia was the worst offender, with 21.27% of spam originating in the country. Germany (10.97%) was in second and the United States (10.47%) in third.
13. The United States tops the list of the most COVID-related malicious file detections	Over 16 million detections since December 2020.
14. The United States ranks 45th out of 75 for cybersecurity performance	The United States scored 19.69, Denmark's top scorer and Tajikistan's lowest.
15. The United States has the highest portion of firms qualifying as cyber experts	25% qualify as cyber experts, 27% considered novices.
16. About 18% of firms had to pay a substantial fine as a result of a breach	This was well over the global average of 11%.
17. Only 33% have standalone cyber insurance	Hiscox found this number was unchanged from 2020.
18. The United States is the third most affected country by stalkerware	Despite the high prevalence of stalkerware, 86% of adults are unaware of its existence or the danger that someone in their household may be snooping on them.
19. Almost 75% of United States organizations experienced phishing attacks	35% of those affected experienced immediate financial loss, twice the global average..
20. United States firms faced many and varied social engineering attacks	81% of United States firms had faced smishing attacks in 2020, 77% had experienced vishing schemes, and 80% had dealt with weaponized USB drives.

Continued on the next page

Table 1.1 – Continued from previous page

Statistic	**Details**
21. Only 52% of United States workers know what phishing is	The global average was 63%. The UK performed the best, with 69% knowing the correct definition.
22. Only 54% know the definition of malware	This was well below the global average of 65%. In its 2020 study, Proofpoint found that 30% of United States workers think malware is a type of wifi-boosting hardware.
23. ALMOST 75% give family members and friends access to work-issued devices	Vulnerabilities associated with checking emails, reading news, using social media, and shopping online.
24. Only 28% of United States businesses use Multi-Factor Authentication (MFA)	Denmark (46%) heading the list and Italy (20%) at the bottom.
25. The average employee has 75 passwords	United States employees were about average. Employees in Belgium have to manage 115 passwords, and those in Sweden, just 50
26. American company Google was issued the largest GDPR fine to date	Google was fined $50,000,000 for not observing principles around transparency, the sufficiency of information, and the presence of legal basis
27. The United States had the highest data breach costs averaging $9.05 million	In the US, the average cost of a data breach rose from $8.64 million per incident in 2020 to $9.05 million in 2021. This is by far the highest, with the Middle East in second place with $6.93 million, followed by Canada with an average cost of $5.4 million. World averages were up 10% year on year.
28. Almost 24% of breaches are the result of human error	The largest cause of breaches is malicious attacks, behind 54% of incidents. System glitches cause a further 22%.

Continued on the next page

Table 1.1 – Continued from previous page

Statistic	**Details**
29. It takes United States companies an average of 186 days to identify a data breach	Average identification time is 207 days, and the time to containment is 73 days; United States firms do a little better here.
30. The IC3 received over 790,000 complaints in 2020	Since 2016, there have been a total of 2.2 million complaints resulting in losses of $13.3 billion.
31. More than 70 top cyber criminals conspired against the United States in 2020	These crimes include espionage, identity theft, wire fraud, computer intrusions, and more.
32. There was a shortage of 377,000 IT security jobs in 2021	ISC reported a staffing gap of 377,000 jobs in the United States alone in 2021. Globally, the shortage of IT security roles has reached 2.1 million.
33. Supply chain ransomware was the biggest threat in 2021	The ransom of $4.4 million was paid to the hacking group, who supplied a tool to restore the systems to their original state, though the process took several hours to complete.
34. DDoS Attacks in the United States increased by 7% in 2021	DDoS attacks grew by 11% in the first half of 2021 versus the first half of 2020. DDoS attacks were the most significant in the United States in Q1 2021, contributing to 7% of the reported attacks.
35. The United States continues to host the most botnet-controlled servers	36% of the hosted botnets were in America, while 24% were hosted in unidentified locations.

1.1.2.2 AI for CPS Cybersecurity

AI presents intriguing solutions that can offer insights and intelligence to counter the constantly evolving landscape of cyber threats (Wirkuttis and Klein [43]). AI can predict and proactively address potential issues by rapidly analyzing massive volumes of events and monitoring diverse cyber risks. Consequently, AI is increasingly becoming an integral part of

cybersecurity efforts, finding applications in various scenarios to automate security tasks or augment human security teams. The fusion of the cybersecurity domain with AI has attracted considerable research attention, resulting in numerous studies that tackle challenges related to identifying, safeguarding against, detecting, responding to, and recovering from cyberattacks. One of the key applications of AI in CPS security is intrusion detection and prevention (Jamal et al. [44]). AI-driven OD algorithms can detect and prevent intrusion attempts in real-time by analyzing network traffic patterns and system behavior. They can identify unusual activities and recognize patterns indicative of potential cyberattacks (Sikder et al. [5]). This enables rapid response to mitigate security breaches and protect CPS from unauthorized access. AI can leverage threat intelligence data to recognize known outliers, attack patterns, and signatures (Al-Hawawreh et al. [45]). By cross-referencing historical data and current behavior, OD algorithms identify and could help defend against known attack vectors, minimizing the impact of recurrent cyber threats. Also, they can perform behavioral analysis to understand normal interactions between system components (Sikder et al. [5]). Any deviation from the learned behavior can indicate a potential cyber threat, enabling swift response and proactive defense mechanisms.

Similarly, AI could also forecast potential cyber threats and vulnerabilities by analyzing historical attack data and emerging trends (Thakkar and Lohiya [46]). This predictive capability enables organizations to proactively strengthen their cybersecurity posture and implement preemptive measures to prevent potential attacks. Moreover, AI techniques are also used to develop adversarial ML models that can identify and neutralize malicious activities and data poisoning (Goldblum et al. [47]). These models continuously learn from new attack patterns to improve detection accuracy.

1.1.3 Water Distribution and Agricultural Production Systems

A WDS is a CPS that encompasses both physical and cyber processes (Figure 1.5). It consists of one or more physical processes, including interconnected tanks, pipes, pumps, storage reservoirs, valves, and hydraulic components, meticulously designed to efficiently transport potable water from water treatment plants to numerous points of use within communities and urban areas. These physical processes are controlled and monitored through computing systems (Lee [48]), often referred to as PLCs and Remote Terminal Units (RTUs), which are interconnected via communication networks. This cyberinfrastructure, together with SCADA monitoring systems, facilitates the effective control and management of the physical processes (Bobat et al. [49]), ensuring the reliable distribution of safe and clean drinking water to residential, commercial, industrial, and institutional consumers, encompassing homes, businesses, schools, and healthcare facilities (Alperovits and Shamir [50]).

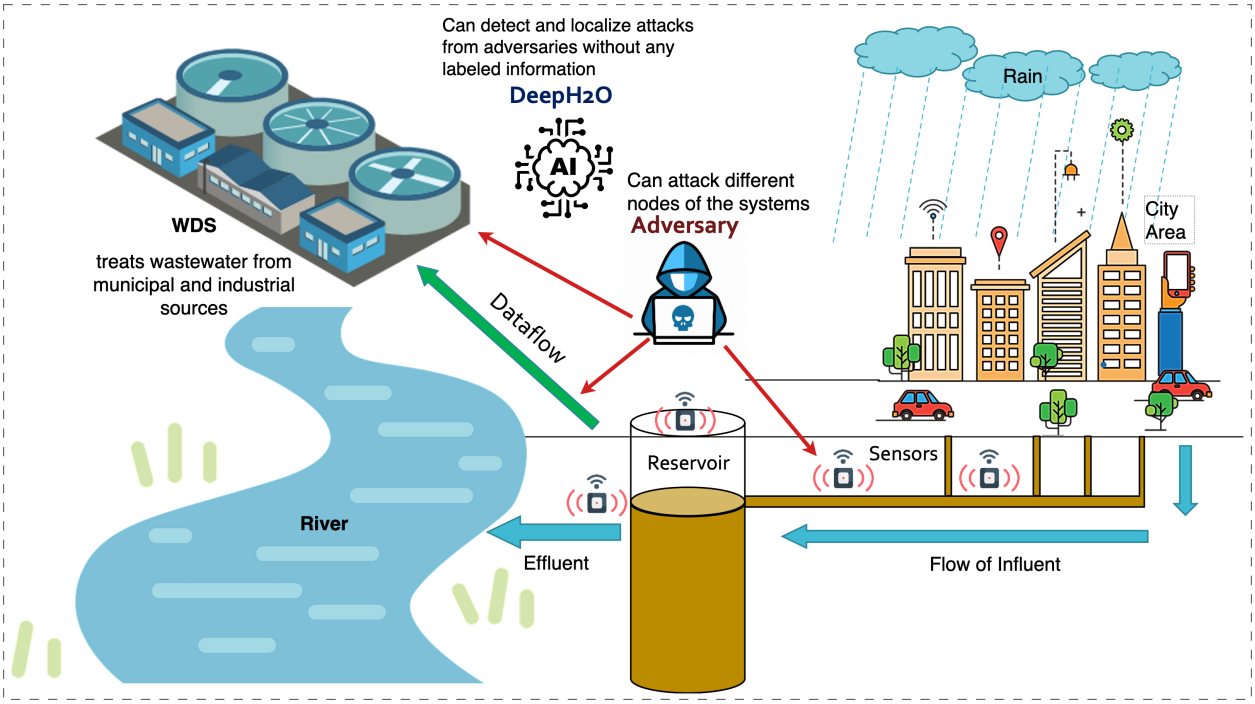


Figure 1.5: AI for Water Distribution Systems

Unfortunately, the same networks expose the system to adversaries, as depicted in Figure 1.5. Water distribution networks are often geographically spread and require automatic control to operate. Automation makes the water distribution network vulnerable to cyber-physical attacks (Slay and Miller [51]). Instances of malicious cyber activity targeting water and wastewater systems have underscored the vulnerabilities within critical infrastructure. The United States Department of Homeland Security (DHS) designates the water and wastewater sector (WWS) as one of the primary targets for cyberattacks among the 16 lifeline infrastructure sectors (DHS [52]). Ensuring its protection from cybersecurity threats is recognized as a matter of utmost national importance (WH [53]). Within the timeframe of 2012 to 2015, the WDSs underwent the highest count of evaluations carried out by the Industrial Control Systems Cyber Emergency Response Team (ICS-CERT [54]) of the Cybersecurity and Infrastructure Security Agency, which regularly conducts on-site cybersecurity assessments for various critical infrastructure sectors. The singular exception occurred in 2014 when the number of evaluations slightly surpassed those in the energy sector. According to ICS-CERT's report in 2016 (ICS-CERT [54]), there were 25 cybersecurity incidents reported by water utilities in 2015, ranking WDSs as the third most targeted sector. The fact that the United States houses more than 151,000 public water systems (USEPA [55]) might suggest that the cybersecurity risk within WDSs is low and most systems are adequately secure. Nevertheless, the reality is quite different, as numerous cybersecurity incidents either remain unnoticed and consequently unreported (Walton [56]), or they remain undisclosed due to the potential damage they could inflict on the victim's reputation, customer trust, and ultimately, financial gains (Cava [57]; Rubin [58]). Furthermore, the severity and ramifications of cyber-initiated incidents can rival those arising from operational technology incidents. WDSs have embraced the digital era, but the absence of dedicated cybersecurity intelligence to offer tailor-made security protocols, ensure system security and train employees remains a prominent challenge.

CISA [6] presented a few more cyber attacks (Table 1.2) in the United States WDSs in recent decades. These real-world events vividly depict the high stakes in securing WDSs against cyber threats, highlighting the pressing need for advanced protection mechanisms.

Similarly, The proliferation of emerging digital technologies has instigated a profound shift towards digital transformation across various economic sectors, including agriculture. The agricultural landscape and farming practices are undergoing substantial modifications due to the assimilation of Information and Communication Technologies (ICT) and the growing integration of the IoT. This phenomenon has fostered the emergence of the concept of *"Smart Agriculture"* (Ratnaparkhi et al. [59]; Colizzi et al. [60]). Forecasts from the United Nations (UN) project that the global population will exceed 9 billion by 2050, precipitating an expected surge of approximately 70% in food production demands (McKenzie and Williams [61]). This anticipated escalation in food production rates intensifies competition within the agricultural sector and heightens finite resources like land and water utilization.

Table 1.2: Examples of Cyber Attacks on WDSs (Source: CISA [6])

Date	Attack Type	Target	Location	Impact
August 2021	Ghost variant ransomware	Wastewater system facility	California	Ransomware message on SCADA servers
July 2021	ZuCaNo ransomware	Wastewater SCADA computer	Maine	Manual operation until recovery
March 2021	Unknown ransomware variant	WDS facility	Nevada	SCADA and backup system affected
September 2020	Makop ransomware	WDS facility	New Jersey	Detection of potential infiltration
March 2019	Unauthorized access	WDS facility	Kansas	Former employee exploitation

Consequently, a demand arises for a novel agricultural paradigm harnessing advanced tech-

nologies to cater to this exigency. Smart agricultural systems encompass intricate multi-sensor configurations and Decision Support Systems (DSS) (Chichernea [62]; Gondchawar et al. [63]) capable of capturing, analyzing, and processing extensive volumes of farm data. This analytical capability empowers farmers to optimize product quantity and profits through informed decision-making. These systems are poised to progressively enhance operations, competitiveness, and profitability in the agricultural sector. Recent studies (Demestichas et al. [64]) underscore the application of ML techniques for identifying data anomalies, commonly known as outliers. The integrity of the decision support system relies on uncorrupted sensor data; hence, the objective is to identify and preclude anomalies from entering the DSS, as anomalies could prompt the DSS to suggest actions detrimental to crop and livestock health.

Due to the unique focus of the research work in the agricultural domain, this dissertation introduces the term *"APS"*. The following definition comprehensively encapsulates the essence of the agricultural paradigm that this work examines and aims to address:

"An APS is a technological integration of agriculture's physical and digital aspects that aims to empower farmers to make AI-driven decisions, improving product quality & quantity and maximizing resource efficiency by providing data-driven insights and eventually enhancing their competitiveness and overall sustainability."

The importance of WDSs and APSs, their complexity, real-time requirements, and potential impact on public health and the economy highlight the significance of AI solutions for cyber threat detection and anomaly monitoring. However, the current focus of AI research on WDSs and APSs is imbalanced towards the AI application spectrum. This is primarily due to the perception of lower potential economic revenue and higher complexity in generalizing algorithms for these systems, given their inherent differences and limited transferability of knowledge between them. However, in light of these systems' unique challenges and

increasing significance, the study of AI in the context of WDSs and APSs has become critically needed.

1.1.4 Introduction to Context-driven AI for CPS

The management of wastewater treatment, including Wastewater Treatment Plants (WWTPs), is becoming increasingly complex due to factors such as urban population growth, climate change, and aging infrastructures (Gheisi et al. [65]). Operators face numerous challenges, including the optimization of de-watering pump schedules, controlling energy and chemical consumption costs during extreme weather conditions, and accurately interpreting sensor data for water quality treatment (Malviya and Jaspal [66]). Events like heavy rainfall can strain these systems beyond their capacities, leading to the overflows of untreated water that pose significant environmental and public health risks, potentially violating the United States Environmental Protection Agency (EPA) standards and regulations (Bastian et al. [67]).

Municipalities are increasingly turning to sensor technology and Artificial Intelligence (AI) for operational improvements, inspection, and data analysis (Chang et al. [68]). AI is revolutionizing wastewater management by addressing operational challenges, mitigating risks, and contributing to environmental sustainability (Matheri et al. [69]).

One key application of AI in this sector is predictive maintenance. AI algorithms monitor equipment conditions to predict maintenance needs, reducing downtimes and emergency repairs (Kulkarni et al. [3], Matheri et al. [69], Liu et al. [70]). Another significant application involves optimizing treatment processes. AI analyzes sensor data to optimize chemical dosing, energy consumption, and overall system efficiency, leading to cost savings and a reduced environmental footprint (Sreng [71], Alam et al. [72], Safeer et al. [73]). Additionally,

AI enables real-time monitoring and alerts by continuously monitoring wastewater quality, detecting anomalies, and providing early warnings to prevent contamination and ensure compliance with environmental regulations (Flores et al. [74], Nishan et al. [75], Mohanty et al. [76]). Furthermore, pipeline inspections are enhanced through AI-powered drones and sensors, which detect issues such as cracks, corrosion, or stageages more efficiently and accurately than traditional methods (Aitken et al. [77], Sousa et al. [78], Rayhana et al. [79]). Despite these significant advancements in AI applications, accurately forecasting short-term fluctuations in complex WWTPs remains a considerable challenge (Kulkarni et al. [3])

1.2 Motivation and Research Background

This section further discusses the challenges, real-world applications, and AI solutions of WDSs and APSs against cyber threats; it also comprehensively explains the necessity of cyber-attack detection and OD methods.

1.2.1 AI For Water Distribution Systems

Ongoing transformations within water systems encompass a broad array of critical infrastructures, such as reservoirs (Bobat et al. [49]), WDSs and WWTPs (Spellman [80]), and smart water networks – characterized as CPS. Smart water networks are built upon the interplay between physical water assets and networked devices engineered to monitor, operate, and supervise all aspects of the distribution system. These devices consist of sensor networks (Ostfeld et al. [81], Hart and Murray [82]), mobile sensors (Gong et al. [83]), and smart meters (Cominola et al. [84]). Integral components of smart water networks include PLCs and SCADA systems. PLCs are embedded devices linked to sensors and actuators

for data management and process control, while SCADA systems are centralized computers responsible for overseeing infrastructure operations, storing real-time process data, and conducting analyses.

While enhancing the reliability, autonomy, and efficiency of modern WDSs, these networked devices simultaneously expose both the physical and cyber infrastructures to cyber-physical attacks (CPAs), as highlighted in a recent editorial (Rasekh et al. [85]). Such attacks encompass a spectrum from accessing private consumer or operational information to intentionally damaging physical water assets such as pumps, valves, and tanks, leading to reduced water supply and even compromising water quality. The pivotal role of WDSs in ensuring safety renders them alluring targets for terrorism and cyber warfare (Lewis [86], Horta [87], Moyer et al. [88]), elevating concerns regarding their vulnerability and potential impact on economies and local communities.

1.2.1.1 Challenges in Water Distribution Systems

In a modern WDS, human observers cannot detect all anomalies; even when they become aware of an anomaly, they often misinterpret it. The same human shortcomings in interpreting complex data are evident in after-action reports about the Deepwater Horizon oil spill, which, among other things, recommend increased use of AI to prevent future spills (Board et al. [89]). Modern WDSs are too complex to monitor effectively without AI. For instance, many sewage spills result from leaks in antiquated underground systems that can go undetected. In July 2020, an old pipe broke in New Haven, Connecticut. No one noticed until a citizen saw raw sewage on the street the next morning and called it in. Two million gallons of untreated sewage spilled into Long Island Sound over the next several days[2]. This incident highlights the broader importance of AI in addressing various challenges in

[2] www.nhregister.com/news/article/Save-the-Sound-investigating-after-15396322.php

infrastructure monitoring and public safety.

Additionally, the Maroochy Water Services incident in 2000 (Queensland, Australia) marked one of the initial attacks in the water supply sector. A disgruntled contractor targeted the SCADA of a sewage system, releasing nearly 1 million liters of wastewater into waterways and parks (Slay and Miller [51]). Since then, instances of cyber-physical attacks have been consistently on the rise. According to the United States Industrial Control Systems Cyber Emergency Response Team (Taormina et al. [90]), multiple cyber-physical attacks have already been perpetrated against United States water utilities. Remedial measures are being undertaken both at national and international levels; the United States Environmental Protection Agency (EPA) has been proactively addressing cyber threats for at least five years (Taormina et al. [90]), while international collaborations between water and environmental agencies have been initiated recently (Taormina et al. [90]).

Despite the recent advancements in computing technology, WDS has security flaws because of its dependency on decade-old network devices. Therefore, an attacker can easily eavesdrop on the communication between the network and the central control system. Additionally, adversaries can send malicious attacks by spoofing sensor measurements, concealing the intended attacks from the operators' sight (Garcia et al. [91]) - also known as concealed attacks. Another popular cyber-attack, replay attack (Mo and Sinopoli [92]), occurs when a cybercriminal eavesdrops on secure network communication, intercepts, and delays signals to misdirect the receiver. Fortunately, such attacks have a digital and physical footprint in the network, such as sensor value deviation from the norm or water flow rate changes during high-demand hours. Statistically, these abnormal events can be considered anomalies, which a suitable learning algorithm can detect. In concealed attacks (Teixeira et al. [93]), since attackers conceal the physical layer, it becomes difficult to detect these attacks by using ML models. For example, (Taormina and Galelli [94]) presented difficulties associated with ML

algorithms to detect concealed attacks in WDSs compared to DL algorithms. This concludes that DL algorithms are superior in representing such complex ecosystems.

Similarly, WWTPs process the wastewater collected from cities, households, factories, and more before discharging it (effluent) for reuse in some cases (such as reclaimed water or for agriculture) or dumping to a river or another water body (Corominas et al. [95]). These WWTPs are complex systems that utilize advanced network devices to improve O&M. WWTPs use large connected tunnels for storing sanitary and wet-weather flows for treatment (Owolabi et al. [96]). In the plant, the decisions on pumping the stored wastewater from tunnels need to be made in a short time because the wastewater cannot exceed the tunnel's safe levels, which can cause the overflow of the untreated water (Corominas et al. [95], Schütze et al. [97]) or overuse of chemicals. This makes calculation time a critical issue (Schütze et al. [97]). The United States Environmental Protection Agency (EPA) reports between 11,400 and 37,900 million liters of wastewater annually- of overflowing untreated wastewater in the environment (Date et al. [98]). This overflowed wastewater harms the soil, air, and rivers (Owolabi et al. [96]). It additionally leads to public health issues, such as gastrointestinal outbreaks (Sojobi and Zayed [99]). It has also been noted that wastewater treatment consumes about 12.6% of the total energy by public utilities (Sanders and Webber [100]), which makes up about 30% of the total operation and maintenance costs in a WWTP (Date et al. [98]). This makes it essential to have a solution that predicts the overflow of the wastewater while minimizing the potential overflow risks and greatly helping critical decision-making processes (such as pumping and adding chemicals). This can be achieved using AI for different downstream tasks to provide sophisticated decision support at WWTPs.

1.2.1.2 Context-driven Deep Learning for Wastewater Management

There is an increasing demand for accurate short-term forecasting tools in WWTPs to support real-time decision-making and enhance emergency preparedness (Fu et al. [101]). Industry experts and major utilities in the United States have noted that predictive models with a forecasting horizon of 4 to 6 hours could substantially improve WWTP resource management and operational efficiency (Kulkarni et al. [3]). Despite this demand, conventional statistical methods and many modern ML and DL solutions struggle to effectively capture the complex non-linear behaviors and seasonal patterns inherent in WWTP data (Kulkarni et al. [3], Yang et al. [102], Sathya et al. [103]).

Traditional ML models, such as Exponential Smoothing (ES) (Gardner [104]), Auto-Regressive Integrated Moving Average (ARIMA) (Arora and Taylor [105]), XGBoost (Chen and Guestrin [106]), often fail to capture temporal relationships in multivariate time series data due to a lack of mechanisms, such as recurrence, to model dependencies between sequential data points (Gardner [104], Drucker et al. [107]). Additionally, these models typically require extensive preprocessing steps, such as decomposition or deseasonalization, which add complexity to the forecasting process (Kontopoulou et al. [108]). Traditional ML models also face limitations in capturing long-term and seasonal dependencies, given their restricted receptive fields (Lai et al. [109]). Furthermore, separate feature selection procedures are often necessary, resulting in inefficient training processes (Kulkarni et al. [3]). A notable limitation of many existing models is their focus on point forecasts, which restricts their ability to assess predictive uncertainty (Yan et al. [110]).

The limited incorporation of external contextual factors further restricts the practical effectiveness of these models (Boussif et al. [111], Murugesan et al. [112], Solomon et al. [113], Boyle and Ravenscroft [114], Miao et al. [115], Wang et al. [116], Stein and Gonzalez

[117], Unger et al. [118]). Few models explicitly address seasonality, and existing solutions frequently lack mechanisms to manage forecast bias, compromising reliability (Palmer and Anderson [119]). In typical WWTP operations, external factors—such as weather conditions, river flows, demographic shifts, and economic activities—exert significant influence yet remain unmonitored by internal systems. For instance, real-time weather data can predict inflow surges from heavy rainfall, while demographic trends provide insights into monthly or weekly water usage patterns.

In this study, I integrate these external variables into a forecasting model to enhance the predictive accuracy of critical WWTP variables, such as wastewater tunnel levels (L_{T}) and nitrate concentrations ($L_{\mathrm{NO_3}}$). Incorporating external context allows for more accurate short-term forecasting, supports proactive operational decisions, improves system resilience, and reduces operational costs.

1.2.1.3 Real-World Applications

Cyber-attack and anomaly detection models using AI require data representing the physical structure and temporal behaviors of the WDSs. Despite the stochastic nature of the WDS operational processes, many AI algorithms can help with early attack prediction or anomaly detection (Taormina et al. [8]). In most cases, the models are developed using data streams from SCADA systems to classify if the system is running safely or not. SCADA collects real-time distributed field data measurements, including water flow rates, pump status, and pressure sensor readings, and then transmits them (measurements) to a central server. Due to the complex interdependencies among different nodes, DL models are better suited to computationally represent the system (Bengio et al. [120]). ML models, including ensemble learning models, are a good choice for small and simple networks (Sikder and Batarseh [121]); however, the increasing number of nodes in a network (such as in WDSs) creates non-

linear relationships amongst them. According to (Sikder and Batarseh [121]), DL models can capture non-linear relationships in a distributed network system more effectively when compared to ML models. Therefore, the dissertation aims to address the security of WDSs by building assured, context-based, and generalized DL models.

DL algorithms can be trained on large amounts of data to identify patterns and anomalies that may indicate a cyber-attack. For example, A DL algorithm can be trained to recognize network traffic patterns typical of a denial-of-service attack and then use this information to block similar traffic in the future Amin et al. [122]. DL intrusion detection systems (IDS) are also used to detect and prevent cyber-attacks on WDSs. These systems can use a combination of DL algorithms and rule-based systems to detect unusual activity in the network. Moreover, AI can optimize the security of WDSs by automating many of the tasks currently performed manually. For example, security configurations and patch vulnerabilities can be automatically updated using AI-based DSS, reducing the risk of successful attacks (Tuptuk et al. [123]). Furthermore, monitoring and interpreting WWTP data are crucial for operational decision-making while ensuring a water facility's safety, security, and efficiency (Tuptuk et al. [124]). These reasons constitute a need for a solution that forecasts the wastewater level, detects potential cybersecurity threats, and utilizes these insights to optimize the processes in WWTPs (Radanliev et al. [125])

It is important to note that the use of AI and DL in cybersecurity research for WDSs and WWTPs is still in its early stages (Tuptuk et al. [123]), and more research is needed to fully understand these technologies' capabilities and limitations. However, these algorithms' potential to enhance WDS security is significant, and their application is expected to continue to grow.

1.2.2 AI for Agricultural Production Systems

According to the United Nations, the world's population is expected to increase by two billion persons in the next 30 years, from 7.7 billion (current) to 9.7 billion in 2050, and could peak at nearly 11 billion around 2100. To feed this growing population, a similar increase in food production must also be achieved (Kamilaris and Prenafeta-Boldú [126]). Several challenges exist in agriculture -with declining productivity of resources such as land, the environmental footprint of production practices, and the ensuing need for sustainability– limiting human abilities to scale up production to meet the global demand. Integrating technology into the agricultural ecosystem is considered an important pathway for providing adequate nutrition to the world and ensuring the sustainability of the resources for the benefit of future generations (Liakos et al. [127]). Figure 1.6 illustrates how data-driven analysis of anomalies in economic and weather data can enhance agricultural commodity production and provide timely insights for policymakers to formulate optimized policies.

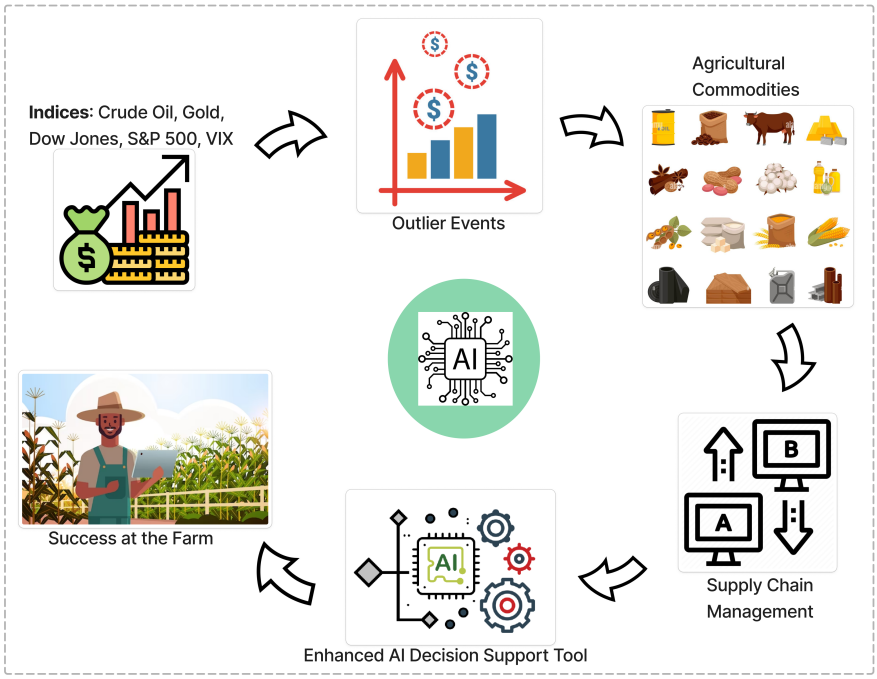


Figure 1.6: AI for Agricultural Production Systems

APS uses various technologies, such as sensing, information technologies, and mechanical systems, to manage different field parts separately (USDA, 2018). Adopting such practice and applying it to day-to-day farm procedures is known as Precision Farming. Precision Farming provides stability amidst conditions such as weather and market demands that are natural actors within agriculture at the local and global level, protecting one's commodities and maximizing economic yield in the long run. Although farmers grow accustomed to such conditions, there are instances where outlier events occur that overwhelm current monitoring and forecasting tools, prohibiting farmers from making sound decisions. Therefore, this work demonstrates how outlier data from contextual variables can be leveraged to predict agricultural production data, offering valuable decision support for APS.

1.2.2.1 Challenges in Agricultural Production Systems

Precision farming tends to optimize complex multivariate farming practices by continuously monitoring, measuring, and analyzing several variables such as weather, soil, and crop type, enabling precise targeting and care for each specific agricultural commodity at a scale that was impossible in the 20th century (Gebbers and Adamchuk [128]). However, with agriculture highly susceptible to outlier events, e.g., floods, drought, and trade wars, predicting the future while accounting for possible outlier events remains a major challenge (Gopinath et al. [129]). During the COVID-19 pandemic, for instance, many farmers and producers were struggling with the forecasts provided to them using traditional econometrics because the models used to create such predictions don't account for outlier events.

Formal acknowledgment of economic fluctuations is insufficient to understand how and why the extremities of outlier events vary and occur. Instead, precision agriculture requires the intersection of policy and economics to enable data scientists and public policymakers to make more informed decisions. It is known that political events directly or indirectly

affect the economy of the Volatility Index (VIX) (Shaikh [130]). COVID-19, which began at the end of 2019, is an outlier event resulting in a vast disruption in the United States economy and financial markets, which was unforeseeable for many (Brown et al. [131]). Consumer consumption increased as states were advised to lockdown, which strained retailers nationwide. The relationship between agriculture and this particular outlier event will be a recurring example throughout this chapter because, for many, this obscure event is the most relevant and well-known outlier in recent memory.

1.2.2.2 Real World Applications

Big data analytics in precision farming demonstrates the importance of recognizing and extracting insights and trends from historical agricultural data to better guide commodity production decisions and policy-making based on context (Storm et al. [132]). As the quantity of data generated in the agricultural ecosystem continues to increase, ML and DL provide accurate predictive insights and guidance on operational decisions with real-time data (Wolfert et al. [133]). ML and DL allow the machine to learn from the available data without being explicitly programmed, thus revealing more insights than what is normally possible through traditional data analytics. DL extends classical ML by adding more complexity to the models with a large learning capacity, as it has a strong advantage in feature learning. This makes DL models flexible and highly adaptable for various complex tasks. That notion allows it to excel at classification and prediction problems in many domains (Kamilaris and Prenafeta-Boldú [126]). The application of DL in agriculture is relatively recent and can be a promising technique considering the impact and potential it has demonstrated in other domains (Kamilaris and Prenafeta-Boldú [126]). Most studies and applications of APS today are localized to the farm environment without much consideration of the impact of external variables, specifically outlier events (Gurrapu et al. [24]).

Returning to the example of the COVID-19 pandemic, it can be classified as a global and political event directly influencing the production of goods. For instance, the distribution of vaccinations, a relatively recent development aimed at curbing the spread of the coronavirus, is intricately linked to the well-being of agricultural operations. The Purdue Food and Agriculture vulnerability index estimates nationwide that- "*over 496,000 agriculture workers have tested positive for coronavirus, with over 3000 in New York State alone*". The management of their fields and crop production was jeopardized alongside their health. The Purdue Food and Agriculture vulnerability index[3], in collaboration with Microsoft, served as a baseline for establishing the scholarly work that is already available and identifying what can be improved upon. Purdue University combined data on the number of COVID-19 cases in each United States county with the county's total population, the United States Department of Agriculture data on the number of farmers and hired farm workers in each county, data on agricultural production of each county, and lastly was able to estimate the share of agricultural production at risk. Visualizing loss of production within various states was useful in developing a deeper understanding of the struggles within the agriculture industry, specifically during an outlier event. Though the loss of production impact for a given commodity is an aspect of agriculture research, it's not useful for predicting the other outlier events considered in this work and their relationship with economic indices. In this sense, this work can be distinguished from Purdue University and other existing scholarly work.

1.3 Summary of Contributions

This section provides a concise overview of the contributions in the fields of WDS and APS achieved through the utilization of ML and DL algorithms. The comprehensive summary

[3]ag.purdue.edu/department/agecon

of this dissertation's contributions is illustrated in Figure 1.7, showcasing three primary AI components: a well-defined model agnostic AI assurance framework; several AI methods applied in the context of WDSs & APSs; and their corresponding pipelines, algorithms, and applications. This dissertation, in Figure 1.7, introduces the *ALSP* framework, comprising three AIA techniques encompassing DL/ML methods, context-aware OD techniques, and predictive modeling approaches. While this section briefly outlines each component's high-level contribution, the intricate details are reserved for Chapter 4.

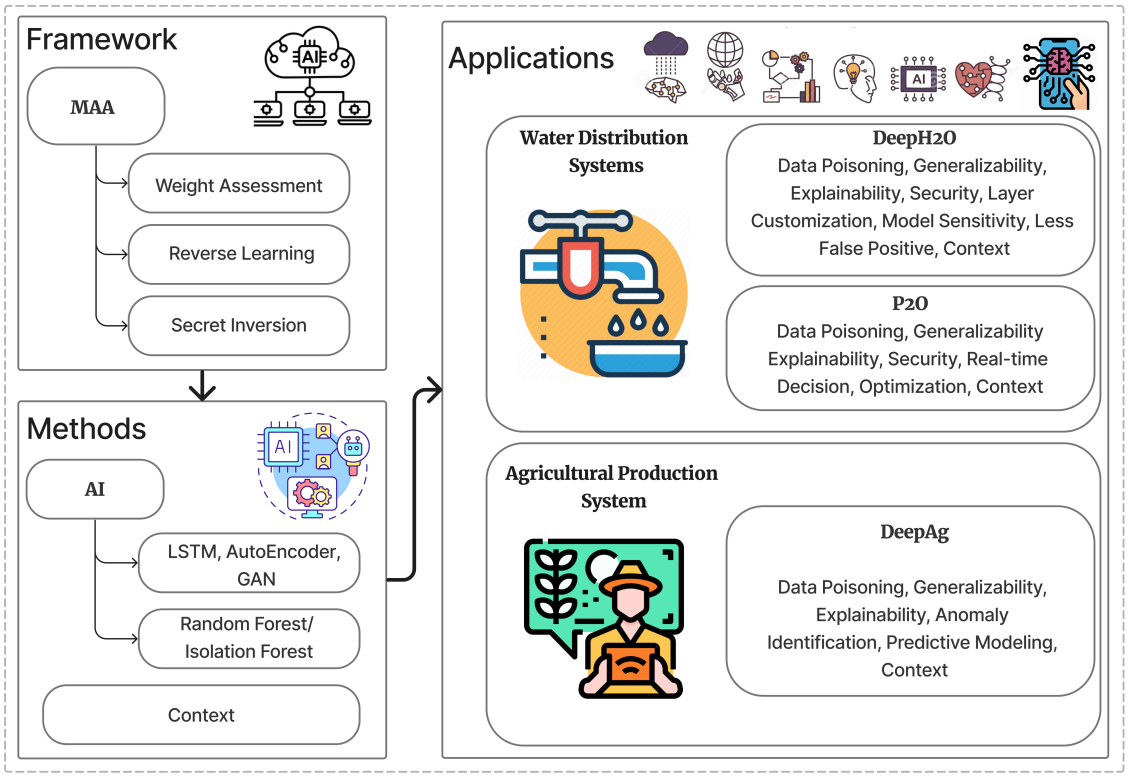


Figure 1.7: Summary of Contributions

This dissertation explores various AI concepts, including framework, pipeline, method, technique, and algorithm. I define them as follows: An AI *framework* is a software library designed to support AI and ML development. It provides tools and predefined modules for easier algorithm implementation. Examples include TensorFlow, PyTorch, and scikit-learn. An AI *pipeline* is a structured sequence of tasks guiding AI application development, from

data collection to deployment. It streamlines the end-to-end process, ensuring efficiency and reproducibility. AI methods are high-level strategies for problem-solving, guiding the overall approach. For instance, using RNNs in NLP is a common method. *Techniques* are specific processes, tools, or procedures used to achieve goals within AI methods. Examples include data augmentation and regularization techniques like L1 and L2. AI *algorithms* are precise step-by-step procedures or formulas for computations. They are fundamental in AI and ML, such as classification algorithms or DL neural networks.

1.3.1 AI Assurance for CPS

In a recent review work on AI Assurance (AIA) Batarseh et al. [134], assurance is defined as *"a process that is applied at all stages of the AI engineering lifecycle ensuring that any intelligent system is producing outcomes that are valid, verified, data-driven, trustworthy, and explainable to a layman, ethical in the context of its deployment, unbiased in its learning, and fair to its users".* This subsection explores the necessity and contribution of Model Agnostic Assurance (MAA) framework.

1.3.1.1 Definition of Assurance Goals

AIA goals can be achieved by either a model-specific or model-agnostic approach. A model-specific approach manages a domain-specific AI algorithm, such as assurance of fairness-aware OD (Mathew et al. [135]), whereas a model-agnostic approach is a generic and universal approach that facilitates verifying AI algorithms irrespective of the domain of study. This dissertation introduces measures to quantify scores for three AIA goals (Batarseh et al. [134]), including explainability, fairness, and security. The goals are, however, quasi-mutually exclusive, and trade-offs are often enforced when choosing amongst them. In literature, it

has been highly arguable whether to make an AI model highly explainable and less safe. The trade-off between goals depends on the requirements of each specific application. It is subjective whether to compromise a model's safety to achieve other assurance goals.

1.3.1.2 The Need for Model Agnostic Assurance

MAA aims to define empirical methods for evaluating subjective measures that are commonly domain-dependent, such as fairness and explainability (Sikder et al. [136]). For instance, an AI model used for recruiting might exhibit bias towards specific candidates, but quantifying that bias is very challenging. Measuring fairness would help ensure that the model is suitable for use within organizational and legal constraints. Another critical example of AIA is in APS. Trusting an AI model to make agricultural production decisions that could have health-related consequences is a process that requires a high level of explainability and trust in the model. Through the *MAA* framework presented in this work, one can assess the explainability of an AI model applied in a specific domain. Additionally, one can investigate security measures using the framework. The framework works better with smaller datasets and simple ML and DL algorithms; otherwise, representing such subjective measures in a quantified manner is undeniably complicated in real-world applications. The presented framework serves as a proof of concept for quantifying assurance goals for small data-driven models.

1.3.1.3 Contribution to AI Assurance Goals

Regardless of the challenges, considering a few trade-offs, this dissertation provides tools for AI engineers to manage the three goals of assurance: explainability, fairness, and security. Accordingly, the contribution is the *MAA* for domain-independent applications through the

Adversarial Logging Scoring Pipeline (*ALSP*) framework. ALPS includes three algorithms: Weight Assessment, Reverse Learning, and Secret Inversion. It leverages game theory, DL techniques, and action logging of an ML algorithm to provide different AIA goals. In this dissertation, multiple empirical outcomes are presented that are deemed successful for AIA goals. One use case presented is for a critical infrastructure: SCADA system in WDS.

1.3.2 Outlier Detection Methods for CPS

Traditional Industrial Control Systems (ICS) and SCADA setups have not fully embraced the extensive connectivity introduced by Industry 4.0 (Alhaidari and Al-Dahasi [137]). Additionally, security protocols within ICS often tend to be secondary or lower priority. This phenomenon can be attributed to the outdated assumption that the ICS environment remains isolated within a secure, air-gapped network (Hewage [138]), a notion that is no longer accurate. The increased integration with potentially hostile networks has resulted in a significant rise in malevolent infiltrations in CPS, leading to considerable financial repercussions and endangering human safety (Hewage [138]). In a CPS, outlier events are subjective and depend on the unique properties of the CPS. It is important to investigate these events as carefully as possible to avoid infrastructure failures because they can cause minor to severe damage to the expensive infrastructure if they go unnoticed. This subsection discusses categories of OD methods using AI and the proposed contributions.

1.3.2.1 Importance of Outlier Detection in CPS

The widespread use of CPS in the modern world has led to a concerning rise in malicious attacks carried out by adversaries. These attacks have been occurring more frequently, especially on critical infrastructure, making them even more susceptible to vulnerabilities.

To emphasize the seriousness of this issue, Table 1.1 provides essential statistics regarding historical cyber threats aimed at United States infrastructure. It is crucial to actively prevent further attacks that could put both national and global infrastructure systems at risk. Notably, cutting-edge applications of OD in the following domains are highly important:

1. **IoT and critical infrastructure operations**: IoT devices utilize wireless sensors to collect various information on architecture, including smart grid, power distribution system, water supply system, and healthcare diagnostic system. It's crucial to know correct and effective data are collected from IoT devices (Jeffrey et al. [13]). If the data are being polluted with outliers because of a sensor fault or a cyber-attack, that should be identified for securing the critical infrastructure. Additionally, OD algorithms need to be trained against attack concealment.

2. **Database and sensor network monitoring**: Sensor networks require continuous monitoring for effective wireless operations. Detecting outliers in sensor networks Abid et al. [139], Feng et al. [140], body sensor networks Zhang et al. [141], and target tracking environments Shahid et al. [142] ensures flawless operations with proper routing in the network.

3. **Fraud and intrusion detection**: Intrusion detection is performed to check if a computer network has any unauthorized access by observing unusual patterns Singh et al. [143]. Additionally, detecting outlier instances is extremely important to secure and safe a network.

4. **Data streams monitoring**: Research studies in Zhang et al. [141], Tamboli and Shukla [144], Shukla et al. [145], Tran et al. [146]; Gupta et al. [147], Cateni et al. [148] showed OD for data streams and time series datasets. Detecting outliers in data streams is important because any abnormality may hinder applications' fast computa-

tional and estimation processes.

5. **Surveillance and security**: Security is an important aspect of computer administrative networks. Cybersecurity is where researchers ensure methods for safe access and proper authentication. An exciting and practical research in cybersecurity is surveillance video OD (Xiao et al. [149]).

6. **Data logging and data quality**: Logging and processing data for commercial purposes can go wrong because of unwanted concealment processes, which, if not detected, might result in irrecoverable loss. Automated data mining models are applied to search for abnormalities while processing large logs (Ghanbari et al. [150]). Proper anomaly identification algorithms need to be applied to enhance data quality (D'Urso [151]; Chenaoua et al. [152]).

1.3.2.2 Categories of Outlier Detection Methods using AI

OD is a creative process; many researchers have been trying to answer the question of how to correctly identify outliers as they provide important information about a system. It is crucial to understand datatypes before applying OD methods; for instance, data can be univariate or multivariate and need a different approach to begin with. In statistical analysis, careful observation regarding feature selection needs to be considered to achieve the feature to represent the data distribution models for both non-parametric and parametric analysis. Moreover, during OD, one must make analytic arguments and intuitions before making any conclusions (Ranshous et al. [153], Braei and Wagner [154], Lai et al. [155]). Besides, real-world applications require context-aware and purpose-based detection because the outcome of the result should benefit the requirements of outlier analysis in any given domain. Research communities are trying to bring forward many innovative and novel algorithms for OD

(Aggarwal [34];Hadi et al. [156]). According to Sikder and Batarseh [36], OD methods can be classified into six categories: Statistical, Density, Clustering, Distance, Learning, and Ensemble-based OD methods.

1. **Statistical OD Methods**: These methods identify outliers based on their statistical characteristics, such as mean, standard deviation, or distribution (Yang et al. [157], Hido et al. [158], Goldstein and Dengel [159]). Common approaches include the Z-score, where outliers are data points that fall outside a certain number of standard deviations from the mean, and the Grubbs test, which detects outliers based on the maximum or minimum value deviating significantly from the mean.

2. **Density-based OD Method**s: These methods identify outliers based on the density of data points in the dataset (Breunig et al. [160], Tang et al. [161], Kriegel et al. [162]). Outliers are typically identified as data points in regions with low data density. One popular density-based method is the Local Outlier Factor (LOF), which measures the density of a data point relative to its neighbors, identifying outliers with low local densities.

3. **Clustering-based OD Methods**: These methods first group data points into clusters and then identify outliers as data points that do not belong to any cluster or belong to small, sparse clusters (MacQueen et al. [163], Ester et al. [164], Karypis et al. [165]). One well-known algorithm is the K-means clustering algorithm, which assigns data points to clusters based on their distance to cluster centroids.

4. **Distance-based OD Methods**: These methods identify outliers based on their distance to other data points. Outliers are typically data points that are far away from the majority of the data (Zhang et al. [166], Huang et al. [167]). The k-Nearest Neighbors (k-NN) algorithm is commonly used, where outliers are detected based on their

distances to the k nearest neighbors.

5. **Ensemble-based OD Methods**: These methods combine multiple OD algorithms to achieve better performance and robustness (Campos et al. [168],Rayana and Akoglu [169]). They leverage the diversity of different algorithms to improve OD accuracy. One common ensemble method is the Isolation Forest, which constructs random decision trees to isolate outliers by their ease of separation from normal data points.

6. **Learning-based OD Method**s: Learning-based OD methods use ML techniques to build models that can distinguish between normal and outlier data points (Dutta et al. [170], Aggarwal and Yu [171]). These models are trained on labeled data, where outliers are marked as such. Common learning-based approaches include Support Vector Machines (SVM) and Neural Networks.

Each OD method has its strengths and weaknesses and is suited for different datasets and applications, which will be further discussed in Chapter 2. Choosing the appropriate method depends on the specific characteristics of the data and the nature of the outliers being sought.

1.3.2.3 Contribution to Outlier Detection Methods for CPSs

This dissertation presents several OD methods that investigate anomalies in a model-specific approach in WDS and APS, including DL and ML-based algorithms (Sikder et al. [5, 136]). Key contributions are listed as bullet points as follows: (1) Introduction of the secret inversion method, employing exhaustive feature comparisons via Autoencoder reconstruction; utilization of reconstruction errors (r) to determine AIA scores, particularly focusing on SAI and CAI goals. (2) The incorporation of AIA techniques, including custom hidden layers and constraints, for a robust and generalizable cyber outlier detection model called a high-confidence autoencoder ($HCAE$). (3) Development of Multivariate Multi-step LSTM ($MM-LSTM$)

for accurate anomaly detection from WWTP. (4) Development of an anomaly detection model for APS called Isolation Forest to investigate economic and weather data outliers.

1.3.3 AI-based Decision Support Systems for Water Distribution and Agricultural Production Systems

In the context of a WDS, timely information is crucial for informed decision-making that minimizes operational risks and optimizes resource allocation, encompassing tasks like pump scheduling and chemical mixing (Sikder and Batarseh [36]). Leveraging AI techniques, including outlier identification such as detecting tunnel wastewater overflow (Gurrapu et al. [2], Sikder and Batarseh [36]), and tunnel effluent level forecasting, facilitates the comprehension of complex nonlinear relationships among system components such as sensor data, weather information, and reservoir water levels. AI methods can identify anomalies and patterns indicative of overflow events in underground sewer networks. This proactive approach enhances the efficiency of response measures and minimizes the negative impacts of overflows.

Within WDS operational processes, AI's capabilities extend to reducing operational expenses (OPEX) by forecasting energy requirements and optimizing resource allocation (Kulkarni et al. [3]). Over the course of the historical development of hydraulic modeling, optimization methodologies have played a pivotal role in WDSs. AI-powered optimization algorithms hold significant potential across various domains, ranging from energy optimization and optimal pump placement to designing effective monitoring and control networks and managing infrastructure during extreme climatic events (Sebestyen et al. [172]). AI-driven optimization encompasses a diverse array of strategies, including Genetic Algorithms (GAs) and DL methods (Kulkarni et al. [3]). For example, in WDSs addressing nitrogen reduction in the

effluent, utilizing DL or GA optimization approaches has proven advantageous (Batarseh and Kulkarni [21]). The versatility of AI extends across numerous downstream tasks, empowering operators with advanced decision-making support.

Combined sewer overflows (CSOs) represent major water-quality threats to hundreds of cities and communities in the United States that are served by combined sewer systems. CSO events cause the release of untreated stormwater and wastewater into receiving rivers, lakes, and estuaries, causing various environmental and economic problems. However, O&M costs associated with CSO management are expensive. The EPA estimates the costs of controlling CSOs throughout the country are approximately $56 billion (Wise et al. [173]). The optimization of the massive energy consumption of the treatment process facilitates mitigating greenhouse gas emissions, which has been considered one of the biggest global challenges in the 21st century. This is because water treatment requires intensive energy consumption. For example, WWTPs consume up to 20% of the total energy by public utilities and 2-3% of the world's electricity consumption (Longo et al. [174]). The accelerated population growth and urbanization further require more energy input to satisfy the higher water treatment standards. Therefore, the optimization of energy is highly desired. Since biological treatment is still the most common WWTP strategy for pollutant removal, the associated pump operation, chemical addition, and aeration largely contribute to the total energy cost in the WWTP. Understanding the factors that will most affect the energy cost allows for creating a higher energy and cost-efficient wastewater treatment strategy.

Similarly, data-driven DSS is necessary for APSs to leverage advanced data analytics and AI techniques to provide valuable insights and recommendations to farmers and agricultural practitioners (Gurrapu et al. [2]). By collecting and analyzing various data sources, such as weather conditions, soil quality, crop health, and historical production data, DSS can help identify patterns, trends, and potential issues affecting crop yields and overall agricultural

productivity. Using ML and DL algorithms for predictive modeling, DSS can offer context-based and real-time guidance on optimal production schedules and other crucial aspects of agricultural management (Khan et al. [175]). As a result, farmers can make informed decisions, enhance resource efficiency, reduce environmental impact, and maximize yields, ultimately leading to sustainable and prosperous agricultural practices.

1.3.3.1 Contribution to AI for water Distribution and Agricultural Production Systems

Within the realm of WDS, this dissertation offers compelling solutions to many real-world challenges. For instance, identifying intense rainy days in a WWTP is essential to forecast tunnel wastewater levels. Since overflowing water in the tunnel leads to higher operational costs, it can pollute the river or water body (Batarseh et al. [176]). By correctly predicting water levels, the operator can make decisions ahead of time for resource allocation, i.e., pumps, and minimize chemical consumption. Similarly, the forecasted data can help optimize energy consumption, saving up 25-30% of total operation and maintenance (O&M) costs (According to the USEPA). By forecasting tunnel wastewater levels, the operator can take the most cost-efficient action to save energy. DL-based prediction models (including LSTM and GRU) can represent systems using historical data from a group of nodes or a single node in a WWTP. Using LSTM, forecasting can be produced in real-time for the next multiple hours to assist with capacity and pumping plans. A typical WWTP requires a 4-hour multistep forecast to predict tunnel water level using LSTM architecture so that action can be taken in a timely manner (Kulkarni et al. [3]); operators require around 3 to 5 hours to operate such decisions efficiently. Summary of contribution to data-driven DSS for a typical WWTP (Figure 1.8 presents a schematic diagram of DSS in WWTP) are as follows:

- Development of data-driven DSS pipeline for WDS cybersecurity and optimization

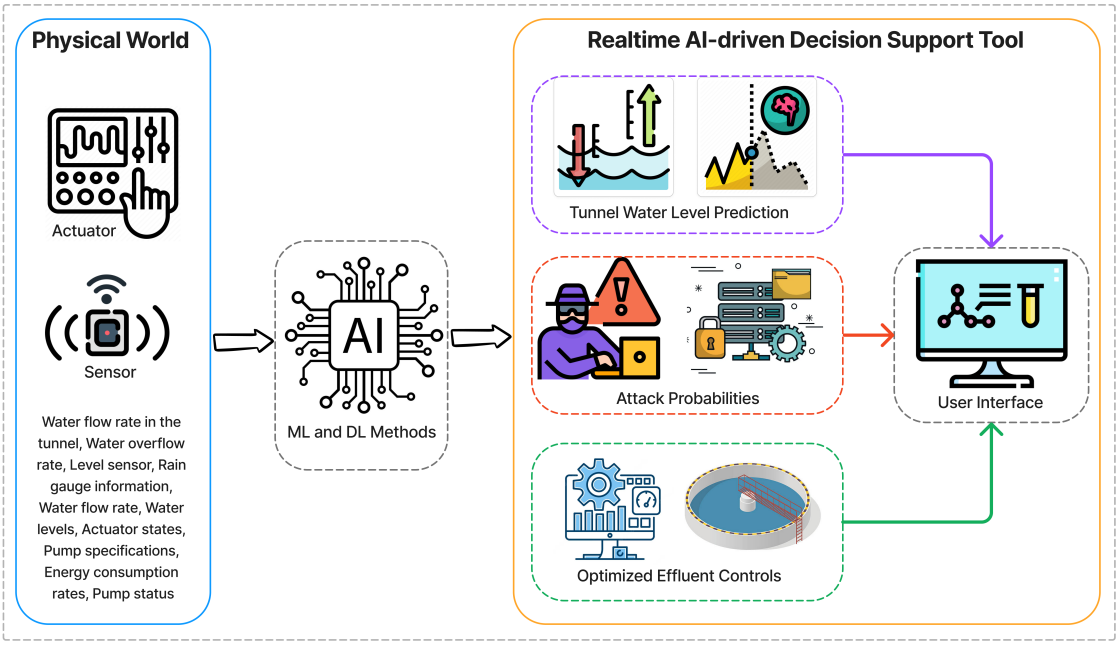


Figure 1.8: AI-based Decision Support System for WDS (P_2O): Three AI Components including Prediction, Protection, and Optimization.

(Kulkarni et al. [3], Batarseh et al. [176]) called Real-time AI-based Decision Support (*RADS*) (Figure 1.8). *RADS* comprises three major AI-driven methods: a protection module ensuring data security, a prediction module forecasting system state variables, and an optimization module providing actionable recommendations (Kulkarni et al. [3], Batarseh et al. [176])

- Development of security method for cyber attack intentionality detection; forecasting method for tunnel wastewater levels, aiding resource management; optimization method for suggesting pump operation strategies, ultimately enhancing the system's efficiency (Kulkarni et al. [3], Batarseh et al. [176]).

Summary of contribution to data-driven DSS for APS (Figure 1.6): (1) Development of a novel pipeline *DeepAg*, an approach using DL to measure the impact of outlier events on agricultural production and predict future patterns (Gurrapu et al. [2]). (2) Collection of

various contextual data, such as financial indices, weather information, and major policy changes related to agricultural commodities, gathered over two decades (Gurrapu et al. [2]). (3) Development of OD and predicting models- isolation forests and LSTM models; understanding commodities production and their causation and correlation with financial indices (Gurrapu et al. [2]).

1.3.4 Cybersecurity in Water Distribution Systems

Given the digital nature of CPSs, they can be vulnerable to different kinds of cyber threats, especially in cases where adversaries can conceal the state of the attack. If an adversary (state or non-state actor) successfully compromises a WDS, that could result in major destructive consequences to water quality, public health, and agricultural irrigation. This dissertation presents empirical AI-based methods for detecting such concealed attacks in WDS.

1.3.4.1 Challenges in Water Distribution Systems and Events of Cyber-attacks

WDSs safeguarding is a national priority (Trump [177]). Adepu and Mathur [178] noted that cyber attacks have increasingly targeted WDSs in recent years, and they are ranked third in the ICS CERT vulnerability report. Ilyas et al. [179] provide two reasons for this pattern: (i) due to the expansion of the IoT and (ii) the proliferation of AI in the decision-making processes. Further, Hassanzadeh et al. [7] presented fifteen disclosed, documented, and malicious cybersecurity incidents in the water sector (Table 1.3), among which some of the most recent incidents are the Florida Water Supply (FWS) hack in 2021 Robles and Perlroth [180] and the Riviera Beach Water Utility (RBWU) attack in 2019. In the FWS hack, the hacker gained remote access to the PLC unit that controls *sodium hydroxide* ($NaOH$) levels in the water. The hacker increased the amount of sodium hydroxide content in the water by

110-fold, but fortunately, the attack was mitigated before the toxic levels of chemicals were diffused into the distribution network. In the RBWU incident, ransomware, a common type of cyber-attack, was launched, which paralyzed the computer systems controlling pumping stations, water quality testing, and payment operations. The government authorities paid 65 bitcoins - approximately $600,000 – to the attacker in a few days, but still, after two weeks, water pump stations and water quality testing systems were partially available.

Table 1.3: Summary of Cyber Incidents in the Water Sector (Source: Hassanzadeh et al. [7])

Number	Location	Year	Target System	Primary Impact
1	Australia	2000	Wastewater	Environmental pollution
2	Pennsylvania	2006	Water treatment	Data breach
3	California	2007	Irrigation	Water theft
4	Illinois	2011	Water plant	Cry-wolf effects
5	Florida	2012	Wastewater	Data breach
6	New York	2013	Dam	Data breach
7	United States	2013	Water utility	Data manipulation
8	United States	2016	Water utility	Control manipulation
9	United States	2016	Water utility	Data breach
10	United States	2016	Water utility	Bandwidth theft
11	UK	2017	Water supplier	Financial impact
12	Europe	2018	Water utility	Resource theft
13	North Carolina	2018	Water utility	Data loss
14	Colorado	2019	Water District	Denial of access
15	Florida	2019	Water utility	Data loss

Further, on January 15, 2021, an intrusion happened on the water treatment plant that served parts of the San Francisco Bay area Collier [181]. The hacker had the username and password of an employee's Teamviewer account. The hacker tried to poison the drinking water by deleting the programs that treat the drinking water. It took one day to discover this hack, and then the authorities acted by changing the password and reinstalling the programs. The systems were breached, yet the authorities could notice the intrusions only after investigating traffic and data flow. The incident exposes the vulnerability of WDS infrastructures and its high relevance to public safety.

Cyber-attacks on WDSs can take many forms. According to survey work by Tuptuk et al.

[123], some common types of cyber attacks on WDSs include:

1. **Ransomware Attacks**: In a ransomware attack, hackers gain unauthorized access to the water system's computer network and encrypt critical data, effectively locking out legitimate users. The attackers then demand a ransom to provide the decryption key, threatening to disrupt the water supply if the ransom is not paid (Mohurle and Patil [182]).

2. **Denial-of-Service (DoS) Attacks**: These attacks aim to overwhelm the WDS's network or servers, causing disruptions and preventing legitimate users from accessing the system (Yan et al. [183]). This can lead to service interruptions, hampering water supply management and control.

3. **Data Manipulation and Tampering**: Cyber attackers may alter or manipulate data within the WDS, leading to inaccurate measurements, erroneous decisions, and potential damage to the infrastructure. This manipulation can cause the system to distribute incorrect amounts of water or fail to respond appropriately to changing conditions (Dong et al. [184]).

4. **Remote Access Attacks**: Attackers may exploit vulnerabilities in remote access points or weak authentication mechanisms to gain unauthorized access to the WDS's control systems. Once inside the network, they can exert control over critical infrastructure components (Brumley and Boneh [185]).

5. **Insider Threats**: Malicious insiders or disgruntled employees with access to the WDS's network can intentionally cause harm by compromising security measures or sharing sensitive information with external threat actors (Probst et al. [186]).

Water utilities and authorities must adopt robust cybersecurity measures to address these

threats. This includes implementing firewalls, encryption, intrusion detection systems, and continuous monitoring to detect and respond to potential cyber threats promptly. Regular security audits, employee training, and information sharing within the water industry can bolster cyber resilience and protect WDSs from cyber attacks.

1.3.4.2 AI-based Solution for Cyber Attack Detection in Water Distribution Systems

In modern WDSs, the sophistication of cyberattacks has escalated, utilizing subtly altered signals that often evade detection by human operators or traditional expert systems (Adepu and Mathur [187]). Ensuring the security, reliability, and functionality of WDSs is paramount, given their role as critical infrastructure for delivering water for various purposes. Consequently, adopting intelligent algorithms, such as DL, becomes imperative for robust cyberattack detection (Batarseh et al. [188]). Taormina et al. [8] presents a range of AI algorithms that prove beneficial for predicting attacks early or detecting anomalies, given the stochastic nature of WDS operational processes.

Typically, AI models are constructed using data streams from SCADA systems to investigate whether the system is operating securely or facing potential threats. Given the intricate interdependencies among system nodes, DL models offer a more effective computational representation (Bengio et al. [120]). DL models capture these non-linear relationships within distributed network systems compared to ML models (Sikder and Batarseh [121]). Therefore, the dissertation presents WDS security through the development of robust and generalized DL models.

1.3.4.3 Contribution to AI-based Cyber Attack Detection in Water Distribution Systems

This dissertation presents DeepH_2O (Figure 1.9), a DL-based cyber attack detection pipeline (Sikder et al. [5]) for WDS, consisting of two main methods: (1) TGCN with attention and (2) $HCAE$. The supervised model, TGCN with attention, in collaboration with a co-author[4], performs well with time series samples and offers contextual anomaly detection where sensor relations in the distribution system require comprehensive monitoring. Additionally, it is

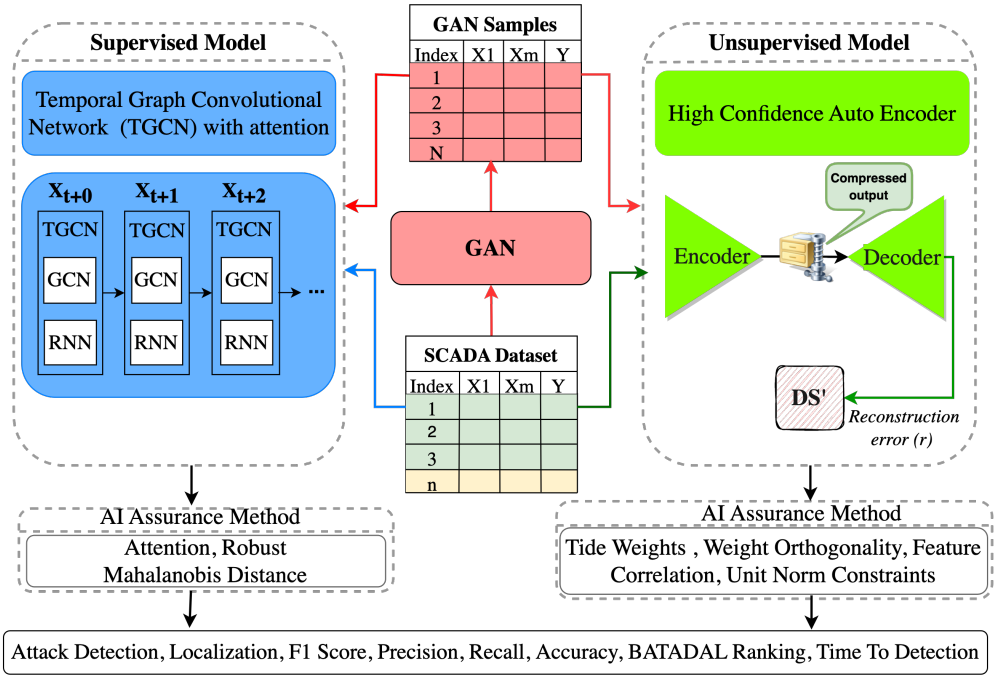


Figure 1.9: DeepH_2O for Cyber Attack Detection in WDSs

expected that a WDS operator might require a method that considers the data samples as non-time series for specific application requirements (for example, missing data). Thus, DeepH_2O includes an unsupervised model $HCAE$ that works well with non-sequential data samples and performs better in multiple evaluation metrics compared to the supervised model.

[4]https://ai.bse.vt.edu/People.html

The dissertation addresses challenges related to DL model decision-making within black-box environments or non-deterministic contexts (Sikder et al. [5]). Summary of contributions to the cyber-attack detection model for a typical WDS is as follows: (1) Development of robust and generalizable cyber attack detection model, assessed synthetically generated poisoned data samples using GANs Goodfellow et al. [189]. (2) Adoption of AI assurance methods such as layer customization and constraints to reduce false positives and improve other performance metrics such as F1-score, precision, and recall to validate $HCAE$'s efficacy in accurately detecting attacks in WDSs. (3) Localization of attacked nodes and sensitivity analysis (Sikder et al. [5]) of $HCAE$.

1.3.5 Context-Driven Deep Learning for Water Systems

In this work, I introduce cP_2O, a context-driven forecasting model designed for WWTPs, which leverages a novel hybrid DL architecture to accurately predict key WWTP variables. The model integrates a dynamic context extraction stage with hierarchically dilated (Chang et al. [190]) Long Short-Term Memory (LSTM) cells, enabling it to capture both short-term fluctuations and long-term dependencies from exogenous variables. Additionally, an internal attention mechanism dynamically weighs contextual information alongside utility data, enhancing the model's sensitivity to important input features and allowing cP_2O to address missing context within utility data.

The primary contributions of this work are as follows:

1. Hybrid Architecture with Context Integration: I developed a hybrid model that combines dynamic context extraction with dilated LSTMs and an attention mechanism. This architecture processes raw WWTP time series data without the need for extensive preprocessing, enabling it to capture both short-term and long-term dependencies

effectively.

2. Dynamic Context Extraction: The model introduces a context extraction stage that incorporates exogenous variables—such as weather, river data, and demographic trends—generating dynamic context vectors. These vectors enhance the predictive capability by incorporating external influences that affect WWTP operations.

3. Attention Mechanism for Feature Weighting: An internal attention mechanism enables the model to dynamically assign weights to input features based on their relevance, reducing the need for manual feature selection. This feature allows the model to focus on the most impactful variables, thereby enhancing forecasting accuracy.

4. Multi-Step Ahead Forecasting with Uncertainty Estimation: The model provides both point forecasts and predictive intervals for multiple time steps ahead (4 to 6 hours). This feature equips decision-makers with insights into forecast uncertainty, crucial for effective risk assessment and real-time operational planning.

5. Bias Reduction through Quantile Loss Function: To reduce forecast bias, particularly during peak or extreme events, I employ a quantile loss function. This approach ensures more balanced predictions, mitigating the influence of outliers—such as abrupt water level surges—on model performance.

I validate the effectiveness of cP_2O through two key experiments conducted on real-world data:

1. Tunnel Wastewater Level Forecasting: This experiment focuses on forecasting influent water levels in tunnels and reservoirs at the Blue Plains Advanced WWTP, operated by DC Water[5]. Accurate short-term forecasting is essential for managing the facility's ex-

[5] https://dcwater.com/

tensive infrastructure, optimizing pump operations, and preventing system overloads. By providing 4 to 6-hour ahead predictions for tunnel wastewater levels, cP$_2$O facilitates improved pump scheduling, which reduces energy consumption and mitigates the risk of untreated water overflow into the environment (Kulkarni et al. [3]).

2. Chemical Variable Prediction: This experiment targets the prediction of critical chemical variables—specifically, pH, ammonia (NH_4), and nitrate (NO_3) levels—at AlexRenew[6]. Effective monitoring of these variables is vital for nutrient removal and compliance with water quality standards essential for ecosystem protection. Using cP$_2$O, I develop a predictive model for chemical sensor values, offering a cost-effective and reliable solution for continuous monitoring that enhances process control and supports regulatory compliance (Sreng [71]).

In both experiments, cP$_2$O outperforms traditional models, reducing Mean Absolute Percentage Error (MAPE) by up to 22% compared to existing models and achieving lower Root Mean Squared Error (RMSE). These results demonstrate the potential of cP$_2$O to enhance resource allocation, drive energy savings, and bolster the resilience of WWTP operations, particularly under extreme weather conditions.

1.4 Overview of the Dissertation

The remainder of this dissertation is organized as follows:

Chapter 2 provides a comprehensive literature review on AI assurance methods, focusing on OD techniques using AI. This chapter explores various categories of OD methods, including Statistical and Probabilistic-based, Density-based, Clustering-based, Distance-based,

[6] https://alexrenew.com/

Ensemble-based, and Learning-based techniques. Additionally, it discusses AI-based Support Systems in WDSs and APSs.

Chapter 3 outlines the problem statement, research questions, and hypotheses that guide the experimental framework. The chapter highlights key challenges and gaps in achieving AI assurance for water and agricultural systems. These hypotheses lay the foundation for subsequent experimental design and analysis.

Chapter 4 presents the core methodologies proposed in this dissertation. It introduces the *MM-LSTM* model and its corresponding pipeline, P_2O, for anomaly detection and optimization in WDS. Additionally, it details the Isolation Forest-based *DeepAg* pipeline for decision support in APS. The chapter also explores the High Confidence AutoEncoder (*HCAE*) pipeline for cybersecurity applications in WDS, including synthetic data generation through GANs. Furthermore, the Model Agnostic Assurance (*MAA*) framework, *ALSP*, is introduced, comprising Weight Assessment, Reverse Learning, and Secret Inversion methods. An enhanced version of P_2O, called cP_2O, is proposed for context-driven forecasting in water systems.

Chapters 5 through 8 focus on experimental design, results, and real-world deployments. Chapter 5 explains the experimental setup for each proposed methodology, including DeepAg, $DeepH_2O$, and cP_2O. Chapter 6 provides an in-depth analysis of experimental results, comparing the proposed models against various baseline methods. Chapter 7 discusses the results and their implications, drawing key conclusions about the effectiveness of the proposed approaches. Chapter 8 demonstrates real-world deployments, particularly at DC Water, showcasing the models' practical utility in addressing real-time challenges.

Chapter 9 explores future directions and broader implications of the research. This chapter discusses potential applications of the proposed methods in other domains of CPSs and

outlines pathways for enhancing scalability, robustness, and generalizability. Additionally, it highlights open research questions and possible advancements in AI for assurance, anomaly detection, and cybersecurity in critical infrastructures.

The appendix provides supplemental materials, including detailed descriptions of datasets, additional experiment details, and hyperparameter tuning processes. It contains attack descriptions for C-Town datasets, hyperparameter selection for DeepH_2O, and extended evaluation metrics. Mathematical formulations, supplementary plots, and tables are also provided to offer deeper insights into the methodologies and experimental results.

Chapter 2

Literature Review

This chapter explores literature and state-of-the-art on model-agnostic AI assurance, OD methods utilizing AI techniques, AI-based decision support systems in WDS and APS, cybersecurity reviews, and cyber-attack modeling in WDS.

2.1 AI Assurance Methods

A model-agnostic approach explains a model in a post-hoc fashion (hindsight) by accepting that the model is as a black-box; with the inner workings of the model hidden from sight, and then attempting to approximate its behavior (Wachter et al. [191]). One such example of a model-agnostic approach is Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al. [192]). This approach attempts to provide local explanations in the form of linear approximations of the model, accurate in small regions of the space. It is also practical when explaining, for example, why a particular individual has been denied a mortgage application. Other post-hoc model-agnostic algorithms provide explanations for ranking features, even when an underlying model is not linear. This includes InterpretML (Nori et al. [193]), SHapley Additive exPlanations (SHAP) (Lundberg and Lee [194]), and Partial Dependence Plots (PDP) (Friedman [195]). Each of these algorithms takes a distinct approach to the process of determining the important contributors to an ML model.

2.1.1 AI Assurance Goals

Albeit MAA is rare, there is no scarcity of model-specific assurance; this dissertation presents a brief review by AIA goal, with the most related study found in the literature for each.

1. **EAI**: In a study conducted by Rossi and Mattei [196] on EAI, a combination of symbolic and logic-based approaches, a data-driven approach, as well as consideration of a rule-based and data-driven hybrid approach is employed. This last approach is proposed with the question of how to determine when preference breaches ethics as a breach in ethics can make the model not viable to use. This dissertation demonstrates a method that weighs each assurance goal in each model to attempt to combat this issue.

2. **FAI**: Another work by Aghaei et al. [197] investigates how biased and unbiased data can both result in AI models that treat certain inputs unfairly when compared to others. A framework is constructed that helps to prevent discrimination between inputs in AI models. The proposed research looks not just at how this dissertation can assure outputs, but also inputs (the dataset) to aid in preventing such biases.

3. **SAI and CAI**: In work presented in Loeser and Iwasaki [198], SAI is investigated, measuring it using states and the reachability of what is defined to be a secure state. This algorithm is used to obtain a quantitative score for security. While this dissertation employs various techniques, such as an auto-encoder, to detect outliers in the data and obtain reconstruction errors to measure a security score, these researchers used the reachability of the output of the AI model.

4. **TAI**: In a study conducted by Kuter and Golbeck [199], SUNNY, "a new algorithm for trust inference in social networks using probabilistic confidence models," is constructed. This algorithm outperformed another trustworthiness measure relevant at

the time in their testing. The models in this dissertation differ in that they employ the measurement of feature contribution with game theory and Shapley (Shapley [200]) and causal values to obtain a score for trustworthiness.

5. **XAI**: In a study conducted by Islam et al. [201], various formulas involving cognitive chunks are employed to determine a quantitative score for XAI. It is mentioned that "explainable decisions from commercial AI systems are going to be a standard imposed by regulators to eliminate bias and discrimination, and ensure trust." XAI is undoubtedly the most studied (Batarseh et al. [134]) out of the AIA goals. The next section presents both pipelines and how this dissertation addresses the six goals.

The book "Towards Trustworthy, Explainable, Safe, and Ethical AI" (Batarseh and Freeman [202]) offers comprehensive guidance and methods for developing and assuring AI systems across various domains. It caters to researchers, scientists, students, and policymakers, emphasizing the importance of valid, explainable, and ethical AI in today's technological landscape.

2.2 Outlier Detection Methods Using AI

This section discusses six OD categories using AI: Statistical and Probabilistic-Based, Density-based, Clustering-Based, Distance-Based, Ensemble-Based, and Learning-Based OD Methods.

2.2.1 Statistical and Probabilistic-based Outlier Detection Methods

Statistical OD algorithms are easy to use and exhibit improved detection rates and run times performance, particularly for quantitative ordinal and real-valued data distributions. However, they may face challenges with high-dimensional feature spaces. Parametric models assume underlying density distributions, which can result in poor performance and unreliable outcomes, especially for managing data streams from complex networks. They also struggle with multivariate feature spaces due to high computational costs. The histogram-based approach is unsuitable for high-dimensional data as it cannot capture interaction between features. According to Sikder and Batarseh [36], specific statistical methods can be adopted to address these gaps, which may lead to longer processing times and misleading data distributions. Research suggestions include using non-parametric methods for unknown data patterns and applying algorithms that can handle data streams and high-dimensional feature spaces for improved scalability. Non-parametric models like KDE are more suitable for most applications, but they can be computationally expensive in noisy environments (Pokrajac et al. [203], Gao et al. [204], Boedihardjo et al. [205]). Despite some limitations, statistical methods remain viable for targeted domains and data streams. Several OD models, such as Histogram-Based Outlier (HBOS) (Goldstein and Dengel [159]) and PCA methods, have shown promising performance in various applications and offer robustness in analyzing outliers. Table 2.1 presents popular statistical and probabilistic-based OD methods using AI techniques.

Table 2.1: Statistical and Probabilistic Based Outlier Detection Methods

Method Name	**Year**	**Source**	**Description**
Gaussian Mixture Models	2009	Yang et al. [157]	Unsupervised Gaussian Mixture Model for OD using Expectation Maximization algorithm.
Guided Density Ratio Approximation for OD	2011	Hido et al. [158]	Novel statistical methodology using guided density ratio approximation for OD.
Histogram-Based OD Technique (HBOS)	2012	Goldstein and Dengel [159]	Univariate feature space model using dynamic and static histogram bin width, scoring data points for outlierness.
Improved Statistical Model with Projections	2015	Tang et al. [206]	Improved statistical model using Gaussian Mixture Model with projections preserving locality.
Non-Linear vs. Linear Regression OD	2017	Dalatu et al. [207]	Comparison of accuracy and misclassification for non-linear and linear regression models in OD.
Kernel Density Estimation (KDE)	2007, 2011, 2013	Pokrajac et al. [203], Gao et al. [204], Boedihardjo et al. [205]	Non-parametric approach using kernel functions to detect outliers.
KDE with Adaptive Estimation of Probability Density	2013	Boedihardjo et al. [205]	Accurate estimation of Probability Density Function using adaptive KDE for time series datasets.
KDE for OD in Power Grid	2015	Boedihardjo et al. [205]	KDE method applied in power grid environment for OD.
Robust Local Outlier Detection (RLOD)	2015, 2015, 2020	Du et al. [208],Campello et al. [209], Li et al. [210]	Pipeline with three stages, capable of detecting local and global outliers, outperforming former OD algorithms.
Copula-Based OD	2020	Li et al. [210]	Effective Copula-Based OD approach.

Continued on the next page

Table 2.1 – Continued from previous page

Method Name	Year	Source	Description
Unique OD Approach with KDE	2019	Qin et al. [211]	OD approach effectively identifies top-N outliers based on KDE on continuous data.
Modified KDE Approach for OD	2020	Ting et al. [212]	Modified KDE approach using Isolation Distribution Kernel to identify similarity between two distributions.

2.2.2 Density-based Outlier Detection Methods

Density-based OD algorithms offer several advantages due to their non-parametric nature. They do not assume any predefined distribution model to manage the dataset, making them highly flexible and adaptable to various data types. These algorithms, such as LOF (Breunig et al. [160]), LoOP (Kriegel et al. [162]), INFLO (Jin et al. [213]), and DWOF (Momtaz et al. [214]), can identify both local and global outliers, making them useful for real-world applications where outliers can occur at different scales. According to Sikder and Batarseh [36], density-based methods often outperform other statistical-based algorithms and provide more flexibility in investigating crucial outliers. They can easily exclude outliers from nearby denser neighbors, enhancing their precision. Another advantage is that density-based algorithms require minimal hyperparameter tuning, making them relatively easy to use. They are efficient at detecting local outliers, further enhancing their usefulness in practical scenarios.

However, density-based methods also come with some disadvantages. They can be computationally expensive and complex compared to certain statistical-based methods. Tuning the hyperparameters, especially the size parameter (k) for k-nearest neighbors, can be challenging and computationally demanding, leading to increased runtime. The shape of neighbors'

density can significantly impact the performance of these methods, and varying densities among neighbors can create complicated models and result in poor performance. Some density-based algorithms, like MDEF and INFLO (Jin et al. [213]), may struggle with complex density estimation, leading to difficulties in defining the outlierness of an object in certain datasets. Additionally, handling high-dimensional time series data can be challenging for density-based models, though recent algorithms have introduced techniques such as pruning and elimination to address these issues.

To improve density-based OD, researchers can address the challenge of sample size in high-dimensional feature space by employing resampling techniques. Proper selection of the hyperparameter k is crucial for evaluating these algorithms effectively. Special attention should be given to computational costs, as some density-based methods can become sluggish with large datasets. For instance, LOCI's (Papadimitriou et al. [215]) complexity increases when applying an extension - radius r, making its computational cost O(n3). Researchers can explore ways to optimize the computational efficiency of these methods. Additionally, addressing the quality of density estimation in specific density-based algorithms, like using connectivity features as seen in COF (Tang et al. [161]), can enhance their performance in certain scenarios, especially when dealing with closely related clusters with varying densities, where INFLO has shown improved outlier scores. Table 2.2 presents a few popular density-based OD methods using AI techniques.

Table 2.2: Density-Based Outlier Detection Methods

Method Name	**Year**	**Source**	**Description**
Local Outlier Factor (LOF)	2000	Breunig et al. [160]	Measures local reachability density to differentiate outliers from normal points in the KNN set.

Continued on the next page

Table 2.2 – Continued from previous page

Method Name	Year	Source	Description
Connective-based Outlier Factor (COF)	2002	Tang et al. [161]	Applies chain distance for density estimation, assuming predefined population distribution.
Local Outlier Probabilities (LoOP)	2009	Kriegel et al. [162]	Uses statistical probabilistic approach to estimate density using distance distribution.
Local Correlation Integral (LOCI)	2003	Papadimitriou et al. [215]	Handles multi-granularity issues for OD and accounts for feature space local density variation.
Relative Density Factor (RDF)	2004	Ren et al. [216]	Prunes data points located in deep clusters using a data model called P-tree for scalability.
Influenced Outlier (INFLO)	2006	Jin et al. [213]	Distinguishes different neighborhoods for accurate neighborhood distribution and OD.
High Contrast Subspace (HiCS)	2012	Keller et al. [217]	Specially designed for large-dimensional datasets, successfully sorts and ranks outliers.
Global-Local Outlier Score from Hierarchies (GLOSH)	2015	Campello et al. [209]	Extends investigation to detect both local and global outliers using statistical interpretation.
Dynamic-Window Outlier Factor (DWOF)	2013	Momtaz et al. [214]	Detects top n outliers by assigning an outlier score called DWOF, deviating from traditional density-based methods.
Algorithms for High Dimensional Data	2014	Wu et al. [218]	Includes RS-forest for efficient handling of high-dimensional data and distributed computing for density estimation.

2.2.3 Clustering-Based Outlier Detection Methods

Clustering-based methods offer distinct advantages for OD, particularly in cases where underlying distribution knowledge is not necessary (Sikder and Batarseh [36]). Being unsuper-

vised models, they are suitable choices when the understanding of the data distributions is limited. Once the models learn about the clusters, they can efficiently detect outliers among additional data points. Their unsupervised nature makes them well-suited for incremental models, as they do not require information about underlying distributions. Clustering-based algorithms are robust and can handle versatile data types effectively. For instance, hierarchical clustering methods for OD are a good choice for different data types, as they produce nested multiple partitions, allowing users to select partitions belonging to specific levels.

However, clustering-based methods have some drawbacks. A major limitation is that they do not assign outliers a score; instead, they rely on binary labeling, indicating whether a data point is an outlier or not (Sikder and Batarseh [36]). The lack of scoring can make it challenging to backtrack model actions, as once the actions are finalized, they cannot be undone. Determining the optimal number of clusters is also a difficult task for most clustering algorithms. Furthermore, clustering algorithms may face difficulties when dealing with datasets containing clusters with arbitrary shapes, as defining the shapes and distributions of multiple clusters can be daunting.

To improve clustering-based models, researchers need to address certain questions when designing the algorithms. They must clarify whether an object defined as an outlier belongs to a cluster or is located outside the cluster boundary. Determining outlier status based on the distance between the object and the cluster centroid is also a crucial consideration. Additionally, clustering-based methods could benefit from addressing how to handle objects that fit into sparse or insignificant clusters.

Despite the drawbacks, clustering-based methods are generally a good choice for many cases. Applying cluster-based algorithms to data streams is an interesting area for further research. For hierarchical and partitioning-based clustering methods, research efforts could focus on speeding up the calculation process and reducing CPU usage for large datasets. Making

the algorithms robust in detecting outliers from lower-density populations or within low-density clusters could further enhance their performance. Table 2.3 presents a few popular clustering-based OD methods using AI techniques.

Table 2.3: Clustering-Based Outlier Detection Methods

Method Name	**Year**	**Source**	**Description**
Partitioning-based Clustering	1967	MacQueen et al. [163]	Assigns weights to features based on their significance to restrain noise effect.
Density-based Clustering	1996	Ester et al. [164]	Models clusters into denser and non-denser groups based on the radius of a cluster.
Hierarchy-based Clustering	1999	Karypis et al. [165]	Partitions the cluster into different levels structured like a tree.
Grid-based Clustering	2005	Zhang et al. [219]	Utilizes grid-based technique to partition the clusters.
High Dimensional Features	2004	Aggarwal et al. [220]	Handles clustering in high-dimensional datasets.
DenStream	2006	Cao et al. [221]	Applies density-based approach for both offline and online OD.
D-Stream	2007	Chen and Tu [222]	Grid-based OD algorithm, identifies outliers using density threshold.
AnyOut	2012	Assent et al. [223]	Computes outliers from data streams anytime using ClusTree topology.
K-Means for Data Streams	2008	Liu et al. [224]	Splits data streams into chunks for OD.
Cluster-Based OD for Big Data	2013	Koupaie et al. [225]	Implements cluster-based algorithm for big data using k-means.
Weighted Clustering Scheme	2014	Bhosale [226]	Combines partitioning and distance-based approach for unsupervised OD.

Continued on the next page

Table 2.3 – Continued from previous page

Method Name	Year	Source	Description
Outlier Beyond Cluster Boundary	2014	Moshtaghi et al. [227]	Updates mean and covariance matrices to detect outliers beyond cluster boundary.
Elliptical Fuzzy Logic	2015	Moshtaghi et al. [228]	Applies fuzzy logic for evolving datasets to identify outliers.
Ensemble Learning	2014	Salehi et al. [229]	Creates multiple clustering models for OD in evolving datasets.
Active Cluster Algorithm	2017	Chenaghlou et al. [230]	Efficient algorithm with lower runtime and memory usage using active clusters.
Cluster Text OD	2016	Yin and Wang [231]	Detects outliers based on low chances of recognizing a cluster.
Self Supervised Detection (SSD)	2021	Sehwag et al. [232]	Framework based on unlabeled distributions outperforming traditional OD algorithms.

2.2.4 Distance-Based Outlier Detection Method

Distance-based methods offer distinct advantages for OD, as they do not rely on the underlying data distributions, making them straightforward algorithms to implement (Sikder and Batarseh [36]). Additionally, they demonstrate superior performance compared to statistical-based methods and scale well for high-dimensional datasets, thanks to their robust architecture.

However, distance-based methods face certain limitations. While they outperform statistical-based approaches in high-dimensional feature spaces, the increasing dimensions can reduce their effectiveness (Sikder and Batarseh [36]). Different objects with unique attributes in a dataset make it challenging for the model to accurately measure distances between such objects. Moreover, using K-Nearest Neighbors (KNN) for distance-based OD can lead to

computational expenses and scalability issues (Knox and Ng [233]). When applied to data streams, distance-based methods encounter difficulties handling data distributions in the local neighborhood and investigating KNN in time series data.

To address these challenges and further improve distance-based algorithms, researchers need to focus on scaling for high-dimensional datasets. The large feature spaces and random attributions of objects pose performance issues, and finding appropriate indexing approaches for assigning neighbors becomes challenging. Models can be enhanced by improving execution time and memory usage. The quadratic complexity of the models can be addressed by employing pruning and randomization techniques or using compact data structures.

Another key challenge is the inability of distance-based methods to detect local outliers, resulting in the calculation of global information instead. To achieve desired scores from KNN algorithms, datasets must be appropriately processed, and selecting proper parameters, including the right k value, significantly impacts model performance. Optimizing these parameters can be challenging but is essential for achieving better results in OD. Table 2.4 presents a few popular distance-based OD methods using AI techniques.

Table 2.4: Distance-Based Outlier Detection Methods

Method Name	**Year**	**Source**	**Description**
K-Nearest Neighbor Models	1998	Knox and Ng [233]	Utilizes distance estimation to identify global outliers.
Recursive Binning and Re-Projection (RBRP)	2008	Ghoting et al. [234]	Improves run time for high dimensional feature space.
Local Distance-based Outlier Factor (LDOF)	2009	Zhang et al. [166]	Manages local outliers and performs similar to COF.

Continued on the next page

Table 2.4 – Continued from previous page

Method Name	Year	Source	Description
Rank-Based Detection Algorithm (RBDA)	2013	Huang et al. [167]	Ranks neighbors based on object proximity.
Reverse Nearest Neighbor Technique	2015	Bhattacharya et al. [235]	Extended study of RBDA using a reverse nearest neighbor.
OD Algorithm in Traffic Data	2015	Dang et al. [236]	Detects outliers in large traffic data in big cities.
Least Spanning Tree for KNN	2015	Wang et al. [237]	Increases searching mechanism for KNN algorithm.
Natural Neighbor Concept	2015	Huang et al. [238]	Modifies KNN technique using a natural neighbor concept.
Heuristic Technique for K Value	2015	Ha et al. [239]	Applies a heuristic technique to achieve k value.
OD in Local KDE	2017	Tang and He [240]	Examines different types of neighborhood information in KDE.
Pruning Techniques	2003	Bay and Schwabacher [241]	Utilizes pruning technique and randomization rule for a nested loop.
Generic Pipeline for Index-Based OD	2007	Angiulli and Fassetti [242]	Detects outliers by pushing data in an index to minimize cost.
Vertical Distance-Based OD	2004	Ren et al. [216]	Implements pruning and labeling techniques for OD algorithm.
Stream Outlier Miner (STORM)	2010	Angiulli and Fassetti [243]	Utilizes distance-based approach with three different algorithms.
Event Detection and Sliding Window	2011	Kontaki et al. [244]	Proposes COD, ACOD, and MCOD algorithms for flexible OD.
Optimized OD for Large Data Volume	2014	Cao et al. [245]	Optimizes range queries to process large data volume.

2.2.5 Ensemble-Based Outlier Detection Methods

Ensemble methods offer notable advantages for OD due to their superior prediction models (Sikder and Batarseh [36]). Notably, Bagging and Boosting algorithms have proven to be robust and less reliant on specific datasets in data mining tasks. These ensemble techniques are especially suitable for handling high-dimensional datasets, which were traditionally challenging for conventional OD algorithms.

However, ensemble methods do have some disadvantages, primarily related to their mathematical robustness and development stage. Feature evaluation may suffer, and selecting appropriate contextual meta-detectors can be challenging. Additionally, combining various algorithms in ensemble methods can lead to a smaller sample space, posing difficulties in handling real data in certain cases.

Addressing these research gaps and improving ensemble analysis is crucial. Ensemble methods demonstrate significant benefits when dealing with streaming data containing noises, where individual classifiers struggle with data quality and processing time (Sikder and Batarseh [36]). Researchers have proposed models to enhance ensemble analysis for OD, and various challenges have been explored, including ranking outliers from different detectors and diversifying principal proposals. Some techniques have eliminated the need for detector selection, which can significantly speed up the identification of unknown outliers. Ongoing research in this area can further enhance the effectiveness and efficiency of ensemble-based OD. Table 2.5 presents a few popular ensemble-based OD methods using AI techniques.

Table 2.5: Ensemble-Based Outlier Detection Methods

Method Name	**Year**	**Source**	**Description**
Bagging	2005	Lazarevic and Kumar [246]	Combines multiple detection algorithms applied on random subsets of features. Each algorithm is assigned a small subset of features to provide an outlier score, improving performance over large datasets.
Boosting	2018	Campos et al. [168]	Leverages strengths of multiple algorithms for robust OD. Components are not independent, as the results of each stage depend on prior executions.
BORE (Bagged Outlier Representation Ensemble)	2016	Rayana and Akoglu [169]	Hybrid and parallel ensemble model for OD, combining bagging and boosting techniques.
XGBOD (Extreme Gradient Boosting OD)	2019	Zhao and Hryniewicki [247]	Applies extreme gradient boosting for sequential OD.
Isolation Forest	2008	Liu et al. [248]	Uses tree-based isolation mechanism for hybrid and parallel OD.
HeDES (Heterogeneous Detector Ensemble)	2010	Nguyen et al. [249]	Combines non-compatible OD methods to form a unified approach for high dimensional datasets.
Ensemble Learning Approach for OD	2014	Zimek et al. [250]	Employs a perturbation technique to account for different diversities in outlier detectors and considers outlier rankings combinedly and distinctively.
Feature Bagging and Subsampling	2016	Pasillas-Díaz and Ratté [251]	Applies both feature bagging and subsampling techniques together for improved performance.
Unsupervised Framework for Outlier Scores	2019	Zhao and Hryniewicki [247]	Dynamically combines and selects outlier scores even if the ground truth is absent.

Continued on the next page

Table 2.5 – Continued from previous page

Method Name	Year	Source	Description
Four Variations of Unsupervised Framework	2018	Zhao et al. [252]	Implements four variations of the unsupervised framework for OD.

2.2.6 Learning-Based Outlier Detection Methods

Learning-based methods offer several advantages for OD. Graph-based approaches provide a comprehensive representation of data interdependencies, aiding in the intuitive understanding of outliers within a dataset. On the other hand, DL methods excel in capturing hierarchical discrimination between features, making them well-suited for large dimensional time series data and enabling effective boundary setting between normal and outlier data. However, these methods also have some drawbacks. Subspace learning, a type of learning-based model, can be computationally expensive (Zimek et al. [253],Dutta et al. [170]). Additionally, not all traditional DL methods perform optimally with increasingly large feature spaces, presenting challenges in OD. To bridge these gaps and enhance the effectiveness of learning-based approaches, further research is needed on specific neural network methods like RNNs, LSTMs, and Deep Belief Networks (DBNs) for OD. Surveys by Kwon et al. [254], Chalapathy and Chawla [255] provide valuable insights and suggestions for advancing deep neural network-based OD techniques. Table 2.6 presents a few popular learning-based OD methods using AI techniques.

Table 2.6: Learning-Based Outlier Detection Methods

Method Name	Year	Source	Description
Subspace Learning Models	2013	Zimek et al. [253]	Appropriate selection of a subset for OD in high-dimensional data.
Subspace Learning Models	2016	Dutta et al. [170]	Sparse subspace learning techniques projecting high dimensional datasets onto low dimensional subspace.
Subspace Learning Models	2005	Aggarwal and Yu [171]	Effective subspace exploration using an evolutionary algorithm for OD.
Subspace Learning Models	2009	Zhang et al. [256]	Method focusing on sparse subspace technique's path and using lattice to denote subspace relationship.
Subspace Learning Models	2016	Dutta et al. [170]	Implementation of sparse encoding to transform objects to multiple linear spaces for OD.
Subspace Learning Models	2013	HHuang et al. [167]	Proposed Subspace OD (SOD) method examining correlations of every object with its shared nearest neighbor.
Subspace Learning Models	2011	Müller et al. [257]	Method emphasizing the relationship between features for OD in contrast to SOD.
Subspace Learning Models	2009	Kriegel et al. [258]	Achieving relevant subspace using Mahalanobis technique through gamma distribution for OD.
Subspace Learning Models	2012	Keller et al. [217]	Identifying subspaces and ranking the outliers using the Monte Carlo method called High Contrast Subspace (HiCS) for OD.
Subspace Learning Models	2016	Van Stein et al. [259]	Using LoOP scores to calculate the degree of outlierness after achieving HiCS for OD.
Active Learning Models	2005	Aggarwal and Yu [171]	Applied ensemble active learning to unveil the reasons for outlier flagging and high computational demand for estimating density in OD methods.

Continued on the next page

Table 2.6 – Continued from previous page

Method Name	Year	Source	Description
Active Learning Models	2014	Görnitz et al. [260]	Alternative active learning method with improved prediction results through repeated learning process and model updates.
Active Learning Models	2019	Das et al. [261]	Proposed Glocalized Anomaly Detection (GLAD) method, combining ensemble outlier detectors and label feedback for active learning in OD.
Active Learning Models	2020	Zha et al. [262]	Deep reinforcement learning-based OD algorithm achieving a balance between long- and short-term rewarding processes for active learning.
Graph Based Learning Models	2008	Moonesinghe and Tan [263]	Proposed "Outrank" algorithm, a graph-based detection framework using Markov random walk on undirected graphs for OD.
Graph Based Learning Models	2018	Wang et al. [264]	Introduced graph-based approach incorporating local information for better OD in comparison to traditional methods.
DL Models	2017	Chen et al. [265]	Utilized deep autoencoder as a semi-supervised model for OD, generating higher reconstruction error for abnormal instances.
DL Models	2017	Du et al. [266]	Developed Deeplog, a deep reinforcement learning-based OD algorithm for online log analysis in critical infrastructure.

2.3 AI-based Support Systems in CPSs

A survey by Sobien et al. [267] addresses the current state of AI assurance for CPSs, with a particular focus on water and agricultural systems. It highlights the growing importance of AI assurance in high-stakes decision-making contexts and suggests future research directions

in applying AI to the CPS field, where AI solutions are currently underutilized. This section discusses related works of AI algorithms in water and agricultural systems.

2.3.1 AI Methods in Water Distribution Systems

Cyber threat detection and outlier identification in WDS is a popular research field. For example, Li [268] made a case for developing and using DL-based models for malware classification and intrusion detection. In a recent work by Jayakrishnan et al. [269], the application of the Soil and Water Assessment Tool (SWAT) is designed for water resources management. Four case studies are presented, showcasing the utility of SWAT in analyzing management scenarios, incorporating radar rainfall data, modeling African watersheds, and addressing water quality issues, particularly in minimizing pollution and potential application in total maximum daily load (TMDL) studies. Further, Hindy et al. [270] used the SMOD dataset to improve security information and event management of water infrastructures. In their study, the authors used six ML models for scenario classification and compared them based on classification accuracy. The authors noted that the k-nearest Neighbors indicated 94% accuracy in detecting anomalies. In another study, Albahar et al. [271] used the SMOD dataset to detect malicious acts from non-malicious ones based on neural networks. The authors compared different models by analyzing the confusion matrix generated from the results. The authors reported greater than 60% accuracy in detecting malicious activities and about 44% accuracy in detecting operational scenarios. Moradbeikie et al. [272] conducted experiments to improve safety via fast and accurate hazard detection. For these experiments, authors categorized data into six classes: Normal data, Transient failure, Permanent failure, Random attack, Stealthy attack, and False alarm; and compared the performance of different ML models for attack detection. The authors used precision, recall, F-measure, false positive rate, and accuracy; they reported about 97% accuracy on hazard detection and

noted that it could reduce about 60% of the time in the system recovery reconfiguration. Sahu et al. [273] proposed a fusion engine that can improve detection accuracy by fusing features to detect cyberattacks in power systems at CPSs. This study utilized F1 score, precision, and recall for evaluating intrusion detection and classification. The authors reported that the fusion engine could improve performance by an average of 15-20% (based on F1 scores). Faramondi et al. [274] used ML techniques for detecting and categorizing threats in CPS using a water distribution testbed. The authors compared four ML techniques based on accuracy, recall, precision, and F1 score. Based on these metrics, the authors reported the highest accuracy (99%) for the Random Forest (RF) model. Lastly, a study conducted by Perrone et al. [275] for threat recognition in critical CPS compared five ML models based on accuracy, precision, recall, specificity, F-measure, and G-mean. The authors reported that the RF model showed the best accuracy (90.2%) for threat recognition compared to other models. Considering these similar studies, my work primarily focuses on DL-based models for detecting and classifying malicious activities (while comparing that to other ML models); DL models proved superior to ML and more scalable than existing state-of-the-art works. To achieve this, two DL models are developed and compared based on accuracy, precision, recall, and F-1 score to select the best threat/OD model.

2.3.2 AI Methods in Agricultural Production Systems

As agricultural ecosystems adopt technology to improve their farming practices, the data collected in the background is increasingly valuable. In Liakos et al. [127], a comprehensive review was conducted on ML applications for APSs. They demonstrated examples of certain precision farming practices, such as crop and soil management, disease detection, livestock management, and water usage, amongst others, that can be improved using ML. An SVM-based methodology was presented by Morales et al. [276] for the early detection of problems

in egg production. The experiment forecasts egg production for up to three days and sends an alert if the production curve displays any anomalies. The results demonstrate that a poultry management system with production forecasting would prove to be useful to assist producers with preventative measures before a problem occurs. Another approach using SVM was shown in Alonso et al. [277] to predict the weight trajectory of livestock given the past evolution of the herd. Additionally, using advanced hardware sensing techniques and artificial neural networks, Pantazi et al. [278] demonstrated an architecture to predict wheat yield production with a high accuracy of 91%.

Agricultural data have also been shown to be useful outside of the farm environment. For example, Gopinath et al. [129] and Gopinath et al. [279] employed deep-learning ML techniques (unsupervised and supervised) to predict trade patterns of seven major agricultural commodities and indicated that unsupervised ML approaches with neural networks provide better prediction fits over the long term. A method in Monken et al. [280] was proposed to measure causal scenarios in trade during outlier events using network-based models, specifically Graph Neural Networks (GNNs) were used to predict outliers effectively and to provide relevant domain explainability. In Batarseh et al. [281], Association Rules (AR) analysis was employed to identify imports and exports associations (*if a then b*) with the trade flows and used Ensemble ML (EML) methods for agricultural trade predictions. In Storm et al. [132], the use of ML for econometric practices was presented and demonstrates the challenges of such simulation models and shortcomings when used for quantitative economic analysis. A fast unsupervised algorithm called Isolation Forest was proposed by Liu et al. [282] for detecting anomalies in continuous data (of all domains). Accordingly, no method is found that could be applied to the production of all commodities considering multiple forms of outliers, my study aims to address that gap Williams et al. [283].

2.4 Cyber Attacks Detection Models in Water Distribution Systems

Given the constantly growing use of CPSs, the uses of AI in CPSs in various applications also grow (Radanliev et al. [284], Gurrapu et al. [2], Gurrapu et al. [23]). This section discusses the DL approaches for CPSs, especially multivariate time series data. Related works for supervised and unsupervised models in CPSs and present adversarial data generation approaches using GANs are discussed. Finally, state-of-the-art methods for water security are presented.

2.4.1 Autoencoder

The use of DL methods for anomaly detection has recently achieved improvements in learning high-dimensional datasets (Mahmud et al. [285]). A deep AE can be a helpful model to eliminate outliers and noise without prior knowledge (Zhou and Paffenroth [286]). A book by Aggarwal [34] on OD discusses how AEs are a natural choice for OD since they are often used to reduce multidimensional datasets. The AE model presented by Zhou and Paffenroth [286], discovers high-quality nonlinear features. This approach includes splitting the input data into two sets to increase the robustness of the model. The work results show good performance since they distinguish between random anomalies and other structured corruptions in CPS data. Sun et al. [287] proposed a novel sparse representation framework that learns dictionaries based on the latent space of Variational AutoEncoder (VAE). This framework addresses the limitations of most existing algorithms that can handle large-scale and high-dimensional data. Their proposed model can obtain hidden information and extract more high-level features by playing the role of dimensionality reduction.

Another work by Zong et al. [288] addresses unsupervised anomaly detection on high-dimensional data. This work aims to address the limitations in existing unsupervised anomaly detection approaches that suffer from decoupled model learning with conflicting optimization goals. This work presented a Deep Autoencoding Gaussian Model (DAGMM) for unsupervised anomaly detection. DAGMM optimizes the deep autoencoder and mixture model parameters jointly to help with the parameter learning of the mixture model. This joint optimization helps the autoencoder further reduce reconstruction errors.

2.4.2 Generative Adversarial Networks

Generative models, including GANs, provide a way to learn deep representations without extensively annotating training data (Goodfellow et al. [189]). The inspiration for this idea comes from the two-player sum game between neural networks, where they balance each other out with gains and losses. GAN consists of a generator and discriminator. The generator captures the potential distribution of real samples to generate new samples, and the discriminator determines which samples are fake by discriminating which of the generated samples are real samples as accurately as possible (Wang et al. [289]). GAN models are necessary for many DL applications, such as security, data augmentation, and privacy preservation. One work by Wang et al. [289] stated that generative models understand data perspective, using real data to fit the distribution parameters and produce new data using the learned distribution. Another work by Goodfellow et al. [290] explained the GAN framework by applying a range of benchmark datasets. They used noise merely on the bottom layer of the generator network. They claimed the samples resulting from their estimation method have somewhat high variance and produce competitive samples compared to the generative models in the literature. Their work did not require interference during the learning, allowing them to incorporate various functions into the model. However, the model has disadvantages.

The discriminator must be synchronized well with the generator to avoid the probability of placing the generator in a small area of data space.

Furthermore, Zhou et al. [291] introduced GAN on the BATtle of the Attack Detection ALgorithms (BATADAL) datasets to create a virtual testbed for WDSs. Their approach computes the membership distance between the dimensions and then divides the dimensions with a small distance into a group. Then, they obtained a larger quantity of attack sample control data by expanding the attack sample. Another work, Shahriar et al. [292], addresses the imbalanced and missing sample data used for Intrusion Detection Systems (IDSs) to defend against CPS attacks. They proposed generating synthetic samples using GANs so the IDS could be trained to use them and the originals. Their results showed improvements in attack detection and model stabilization but did not provide any direction for balancing data classes. In this dissertation, balanced synthetic data are generated using GAN for testing the proposed models' generalizability.

2.4.3 Other AI models for Water Distribution Systems' Security

The security aspects of water distribution have a wide variety of potential solutions. Kadosh et al. [293] presented a one-classifier approach to detect attacks in WDSs. Their approach uses a Support Vector Data Description (SVDD) algorithm to classify normal vs. anomalous behavior. Min et al. [294] proposed an ANN-based DL algorithm to detect cyber attacks. Taormina and Galelli [94] developed an approach that uses AEs to detect and localize intrusion attacks in a WDS Zou et al. [295] proposed an event detection model to detect and mitigate water contamination. In their approach, they proposed a hybrid model that comprises an ANN and a Support Vector Machine (SVM) to detect the contamination events. Bagherzadeh et al. [296] evaluated the effects of different feature selection methods on en-

hancing the model prediction performance of total nitrogen in WWTPs.

Furthermore, they analyzed the importance of different characteristics, namely time, climate, hydraulic flow, and wastewater characteristics, in predicting energy consumption (Bagherzadeh et al. [297]). Their study suggested that the Gradient Boosting Machine algorithm performs better in forecasting energy consumption when compared to other ML algorithms. Mehrani et al. [298] proposed a hybrid model that combines a mechanistic model and ML model to predict the liquid N_2O concentrations; their results suggest that a hybrid model that combines a mechanistic model and an Artificial Neural Network (ANN) model performs better with limited availability of data. Additionally, a significant amount of work has been reported on approaches to detect attacks in CPS used in water treatment plants (Mao et al. [299]).

Furthermore, Yoong and Heng [300] designed an ML framework to detect physical and software-generated anomalies in continuous water treatment plants without false alarms. AAdepu and Mathur [187] designed and developed an expert system, Distributed Attack Detection (DAD), that detects physical anomalies of a plant in real-time operations. This study is a succession of a prior work of Adepu and Mathur [301] where they developed an anomaly detection framework based on physical invariants derived for each plant design stage. Macas and Wu [302] claimed that present water treatment plants are complex, and their spatio-temporal relations need further exploration. The authors presented an unsupervised framework for anomaly detection called Attention-based Convolutional LSTM Encoder-Decoder (ConvLSTM-ED) to capture temporal dependencies. In another study, Zizzo et al. [303] developed an adversarial attacker model to compromise a subset of sensors and validate existing anomaly detection models. In their study, the attacker manipulates the detector by hiding its presence. Similarly, Anthi et al. [304] generated adversarial samples using the Jacobian-based Saliency Map attack and explored how adversarial learning can

target the supervised models. Testing anomaly detection performance using an adversarial attacker model is a popular approach in WDSs; however, based on my search, applying GANs as adversaries for testing the generalizability of the attack detection models in WDSs is a novel study.

2.5 Context to AI for WWTPs: Related Forecasting Studies

Short-term forecasting is essential for optimizing operations in WWTPs, as it directly impacts operational efficiency, resource allocation, and system resilience. However, it remains a critical challenge due to factors such as non-stationarity, high dimensionality, and complex seasonality inherent in time series data (Kulkarni et al. [3]).

2.5.1 Traditional Forecasting Methods

Traditional statistical methods such as ARIMA (Arora and Taylor [105]), ES (Gardner [104]), and Kalman Filtering (Harvey [305]) have long been used for time series forecasting. While these methods are effective for linear data patterns, they often struggle with non-linearities, complex seasonality, and integrating exogenous variables such as weather conditions or market trends (Gheisi et al. [65]). Furthermore, they are limited in capturing long-term dependencies, which is vital for accurate forecasting in modern applications.

As a response to these limitations, ML and DL techniques have gained prominence due to their flexibility in handling complex patterns in high-dimensional time series data. Techniques such as Support Vector Machines (SVMs) (Li et al. [306]), Neural Networks (NNs) (Dudek [307]), and Recurrent Neural Networks (RNNs) (Cho et al. [308]) have shown promising re-

sults. More advanced approaches, such as Convolutional Neural Networks (CNNs) (LeCun et al. [309]) and LSTM networks (Hochreiter and Schmidhuber [310]), have been widely applied for time series forecasting (Sathya et al. [103]). These models have improved the ability to handle complex, nonlinear dynamics and high-dimensional time series data. However, they often require extensive preprocessing and domain knowledge to select relevant features and handle raw time series data effectively (Yang et al. [102]). Furthermore, the absence of mechanisms to dynamically weigh input feature importance limits their capacity to emphasize the most impactful variables at each time step, which may lead to suboptimal forecasting performance (Gao et al. [311]).

To overcome the limitations of single models, hybrid approaches have emerged as a powerful strategy in time series forecasting by combining multiple algorithms to exploit their respective strengths. For instance, (Kim et al. [312]) proposed a hybrid Recurrent Inception CNN model for capturing short- and long-term dependencies, significantly improving forecast accuracy. Similarly, (Massaoudi et al. [313]) employed a hybrid ensemble approach combining LightGBM, XGBoost, and NNs, achieving more accurate and robust forecasts than single models. Despite these advancements, hybrid models encounter challenges that limit their effectiveness in real-world applications, especially in systems heavily influenced by external factors. A primary limitation lies in the insufficient integration of external contextual variables, such as weather patterns, river flow information, or demographic trends (Murugesan et al. [112]). While hybrid models have advanced the ability to capture temporal dependencies, they often fall short in effectively leveraging exogenous variables, which can reduce accuracy in domains such as WWTPs where external influences are critical (Kulkarni et al. [3]). Moreover, hybrid models frequently struggle to adapt to changing external conditions due to a lack of dynamic context extraction mechanisms (Solomon et al. [113]). Their reliance on manual feature selection and the absence of mechanisms to dynamically weigh

input features further constrain their performance (Gao et al. [311]).

2.5.2 Context-Aware Forecasting Methods

Context-aware forecasting models have gained attention as a means to address the limitations of traditional and hybrid methods in integrating external influences. In the context of WWTPs, external factors such as weather conditions, river levels, demographic changes, economic trends, and environmental variables constitute crucial contextual information (Murugesan et al. [112]). Recent studies have highlighted the value of context-aware models in enhancing forecasting accuracy and robustness. For instance, (Solomon et al. [113]) highlighted the role of external data, such as weather and economic indicators, in improving model performance when dealing with systems influenced by multiple factors. Similarly, (Sreng [71]) demonstrated that incorporating real-time weather data into models enhances forecasting accuracy, especially in scenarios where sudden changes in external conditions impact the system's behavior.

Hybrid models that incorporate contextual information are increasingly recognized for their potential to capture both short-term variations and long-term dependencies, thus enhancing forecast reliability (Gao et al. [311]). For instance, (Unger et al. [118]). The absence of dynamic context extraction mechanisms limits the models' adaptability to shifting external conditions and their capacity to capture complex relationships between exogenous variables and the target time series. Additionally, a lack of mechanisms to dynamically weigh input features based on real-time relevance further restricts the efficacy of these models (Wang et al. [116]).

The review of current literature highlights several critical gaps in forecasting models for WWTPs:

- Traditional ML and DL models often struggle to integrate external factors, such as weather data, river conditions, and demographic trends, limiting their applicability in WWTPs where external influences are crucial (Murugesan et al. [112], Kulkarni et al. [3]).
- Many state-of-the-art models require extensive preprocessing and domain-specific knowledge to select relevant features and manage raw time series data effectively (Yang et al. [102]).
- Most existing models do not dynamically weigh input features based on their relevance, resulting in suboptimal performance in systems heavily influenced by changing external variables (Gao et al. [311]).
- A majority of time series forecasting models lack uncertainty estimation capabilities, which are essential for risk assessment and decision-making in critical infrastructure (Piotrowski et al. [314]).
- Many forecasting models struggle to mitigate forecast bias during peak events or extreme conditions, reducing their reliability and deployability in real-world applications (Kim et al. [312]).

By addressing these critical gaps—such as integrating external contextual variables, reducing the need for extensive preprocessing, dynamically weighing input features, estimating uncertainty, and mitigating forecast bias during extreme events—I aim to significantly improve the accuracy and resilience of forecasting models in WWTPs.

Chapter 3

Research Hypotheses

The foundation of scientific inquiry is often built upon hypotheses, which serve as educated assumptions that guide research endeavors. In the context of the work presented, research hypotheses play a pivotal role in directing the course of this study. The subsequent sections present problem statements, research questions, and research hypotheses.

3.1 Problem Statements

The identified problems are categorized into three distinct groups: group one encompasses issues related to AI and AIA, group two pertains to challenges within water and agricultural systems, and group three involves considerations regarding Context to AI for WWTPs. These problem categories are elaborated upon in the subsequent sections as follows:

3.1.1 AI and AIA

This section introduces research statements concerning AI and AIA. It outlines four research statements as follows:

1. DL methods, particularly AEs, used to uncover hidden attacks often exhibit inherent non-deterministic tendencies in their attack detection capabilities (Sikder et al. [5]). This tendency contributes to an elevated risk of false positives. Addressing this issue necessitates

an inquiry into methods for enhancing deterministic characteristics, thereby reducing false positive outcomes. A higher incidence of false positives within a complex system, as applied within a specific domain, implies reduced model accuracy.

2. Cyber attack detection models face the challenge of assessing the generalizability and reliability of DL models within different WDSs (Chandy et al. [315]). To address the problem, this dissertation aims to analyze the impact of poisoned data on DL model performance and subsequently contribute to the enhancement of secure and resilient DL models for WDS applications. This is achieved through an investigation that tests DL models using GAN-generated data, shedding light on the model's ability to withstand and mitigate the effects of malicious inputs, thereby improving its generalizability to unseen data.

3. In traditional cyber attack detection models, particularly in DL models, the localization of features relies on embedded and learned representations within a WDS feature space (Housh et al. [316]). However, their inherent non-determinism often results in an inability to pinpoint the attack nodes accurately. This challenge can be effectively addressed by incorporating AI assurance, leading to the confident identification of attack nodes within a WDS.

4. Deploying DL/ML models in WDS and APS requires assurance in aspects like explainability, fairness, and security (Batarseh et al. [317]). Attaining all these goals is challenging, making a model-agnostic approach essential across domains. This dissertation proposes an empirical, universally applicable model-agnostic AI pipeline for quantifying AIA goals, including explainability, fairness, and security.

3.1.2 Water Distribution and Agricultural Production Systems

This section introduces research statements concerning WDSs and APSs O&M. It outlines two research statements as follows:

1. DL and ML models are extensively studied in APS, but limited data availability often hinders proper model development (Gurrapu et al. [2]). Collecting and investigating context groups could enhance APS models, yet there's no benchmark pipeline. This dissertation explores the impact of incorporating outlier information in precision agriculture, aiming to boost DL-model prediction accuracy and provide a benchmark process.

2. Detecting outliers and optimizing WDS operational processes is a novel and comprehensive study. Despite complex modern networks and extensive data, many utilities still rely on stochastic and probabilistic methods, which struggle with large datasets. AI methods offer a better solution; this dissertation introduces an AI-based decision support framework for anomaly identification and operational cost optimization (Kulkarni et al. [3]) in WDS.

3.1.3 Context for WDS

This section introduces research statements concerning context-aware AI solutions. It outlines two research statements as follows:

1. Context-aware AI solutions can enhance accuracy by minimizing learning and data biases (Wilcox et al. [318]). However, a comprehensive context group selection and framework are lacking in WDSs. This dissertation thoroughly examines how diverse contextual factors—like weather, usage patterns, and population—affect DL model outcomes for specific WDS problems.

2. One challenge of context-based AI solutions is to effectively identify and adapt to relevant contexts for specific applications or subtasks, such as OD or prediction, within a defined time frame (Nobles et al. [319]). This involves leveraging advanced technologies and contextual awareness techniques to develop a contextual AI model that can accurately provide timely and contextually relevant information in dynamic environments.

3.2 Research Questions

The identified questions are categorized into three distinct groups and presented here:

3.2.1 AI and AIA

This section introduces research questions concerning AI and AIA. It outlines four research questions as follows:

1. (AI Assurance): How do AI assurance constraints, such as layer customization, improve models' performance?
2. (Data Poisoning): Can models' generalizability in WDSs be tested using poisoned data generated by GANs?
3. (Feature Localization): How can the two models localize features based on embedded and learned representations in a given feature space (i.e., in a water system)?
4. (MAA): How can developing a model-agnostic AI pipeline, applicable across various domains, be achieved to score AIA objectives such as explainability, fairness, and security?

3.2.2 Water Distribution and Agricultural Production Systems

This section introduces research questions concerning Water and Agricultural Systems operations. It outlines two research questions as follows:

1. (Outlier): How does the incorporation of outlier information contributes to developing a more accurate prediction model in precision agriculture?

2. (Tunnel Wastewater Overflow): How to accurately forecast tunnel wastewater overflow beyond a predefined threshold using a predictive model?

3.2.3 Context to AI for Water Systems

This section introduces the research questions addressed in this work, emphasizing both the theoretical and practical aspects of the proposed context modeling approach for WWTPs. Mathematical formulations are presented using vector notations to describe the model behavior succinctly.

RQ1: Does incorporating contextual data into forecasting models significantly improve the accuracy of short-term predictions in WWTPs compared to models that do not use context?

Given a WWTP dataset $\mathbf{D} \in \mathbb{R}^{T \times N}$, representing multiple time series of WWTP variables (e.g., inflow, water levels), a context matrix $\mathbf{C} \in \mathbb{R}^{T \times M}$, containing exogenous variables (e.g., weather data, river flow), and the true output at time $t+1$ as $\mathbf{y}_{t+1}$, I consider two models:

1. Context-Driven Model:

$$\hat{\mathbf{y}}_{t+1}^{\text{context}} = f_{\text{context}}(\mathbf{D}_t, \mathbf{C}_t; \boldsymbol{\theta}_{\text{context}})$$

2. Without Context Model:

$$\hat{\mathbf{y}}_{t+1}^{\text{no-context}} = f_{\text{no-context}}(\mathbf{D}_t; \boldsymbol{\theta}_{\text{no-context}})$$

The prediction error $\mathcal{E}$ is defined as:

$$\mathcal{E}(\hat{\mathbf{y}}, \mathbf{y}) = \mathbb{E}[L(\hat{\mathbf{y}}, \mathbf{y})]$$

where $L(\cdot, \cdot)$ is a loss function.

RQ2: Is the cP$_2$O model generalizable and effective across different WWTPs, maintaining high accuracy when applied to various datasets?

I assess generalizability by applying the same modeling approach to different WWTPs: For WWTP A, with dataset $\mathbf{D}^{(A)}$ and context $\mathbf{C}^{(A)}$, I train the model with parameters $\boldsymbol{\theta}^{(A)}$; For WWTP B, with dataset $\mathbf{D}^{(B)}$ and context $\mathbf{C}^{(B)}$, I train the model with parameters $\boldsymbol{\theta}^{(B)}$:

$$\hat{\mathbf{y}}_{t+1}^{(A)} = f(\mathbf{D}_t^{(A)}, \mathbf{C}_t^{(A)}; \boldsymbol{\theta}^{(A)})$$

$$\hat{\mathbf{y}}_{t+1}^{(B)} = f(\mathbf{D}_t^{(B)}, \mathbf{C}_t^{(B)}; \boldsymbol{\theta}^{(B)})$$

RQ3: Does the integration of an attention mechanism in cP$_2$O enhance the model's ability to dynamically weigh input features, thereby improving forecasting accuracy?

To evaluate the impact of the attention mechanism, I compare two models:

1. Model with Attention:

$$\hat{\mathbf{y}}_{\text{att},\, t+1} = f_{\text{att}}(\mathbf{D}_t, \mathbf{C}_t; \boldsymbol{\theta}_{\text{att}})$$

2. Model without Attention:

$$\hat{\mathbf{y}}_{\text{no-att},\,t+1} = f_{\text{no-att}}(\mathbf{D}_t, \mathbf{C}_t; \boldsymbol{\theta}_{\text{no-att}})$$

RQ4: Can the quantile loss function employed in cP$_2$O effectively reduce forecast bias during peak events or extreme conditions, and improve uncertainty estimation in multi-step ahead forecasting?

I assess the model's performance by comparing:

1. Model with Quantile Loss Function:
    - Predictions: $\hat{\mathbf{y}}_{\text{quantile},\,t+1}$
    - Parameters: $\boldsymbol{\theta}_{\text{quantile}}$
2. Model with Standard Loss Function:
    - Predictions: $\hat{\mathbf{y}}_{\text{standard},\,t+1}$
    - Parameters: $\boldsymbol{\theta}_{\text{standard}}$

Forecast bias $\mathcal{B}$ is defined as:

$$\mathcal{B} = \mathbb{E}[\hat{\mathbf{y}}_{t+1} - \mathbf{y}_{t+1}]$$

Additionally, the model improves uncertainty estimation, measured by metrics such as prediction interval.

By addressing these research questions, I aim to validate the effectiveness of cP$_2$O in improving forecasting accuracy through context integration, assess its scalability across different WWTPs, and demonstrate the contributions of the attention mechanism and quantile loss function in enhancing model performance.

3.3 Research Hypotheses

Research hypotheses are discussed as follows:

3.3.1 AI and AIA

This section introduces research hypotheses concerning AI and AIA. It outlines four hypotheses as follows:

1. Research Hypothesis: Let $fp_{\text{LayerCustom}}$ represent the false positives of the model with AI assurance constraints, such as layer customization. Let fp_{Baseline} represent the false positives of the baseline model without AI assurance constraints.

The hypothesis can be formulated as follows:

$$H_0 : fp(HCAE)_{\text{LayerCustom}} \geq fp(AE)_{\text{Baseline}}$$

$$H_1 : fp(HCAE)_{\text{LayerCustom}} < fp(AE)_{\text{Baseline}}$$

Where, H_0 represents the null hypothesis, suggesting that the false positives of the model with AI assurance constraints (layer customization) are greater than or equal to the false positives of the baseline model without these constraints. H_1 represents the alternative hypothesis, suggesting that the false positives of the model with AI assurance constraints (layer customization) are less than the false positives of the baseline model without these constraints.

The goal of this hypothesis is to investigate whether AI assurance constraints, such as layer customization, improve the reduction of false positives. The hypothesis testing aims to assess whether the model's performance with AI assurance constraints is significantly better than

the performance of the baseline model without these constraints.

2. Research Hypothesis: Let $\text{HCAE}_{\text{Generalizability}}$ represent the generalizability of $HCAE$ and $\text{AE}_{\text{Generalizability}}$ represent the generalizability of baseline AE on unseen poisoned data generated by GAN.

The hypothesis can be formulated as follows:

$$H_0 : \text{HCAE}_{\text{Generalizability}} \leq \text{AE}_{\text{Generalizability}}$$

$$H_1 : \text{HCAE}_{\text{Generalizability}} > \text{AE}_{\text{Generalizability}}$$

Where H_0 represents the null hypothesis, suggesting that the $HCAE$ tested using unseen poisoned data generated by GANs is less than or equal to the generalizability of the baseline AE model tested using the same data. H_1 represents the alternative hypothesis, suggesting that the generalizability of $HCAE$ tested using unseen poisoned data is greater than baseline AE tested using the same data.

This hypothesis aims to investigate whether the generalizability of the improved model, $HCAE$, can be assessed using unseen poisoned data generated from a GAN. The hypothesis testing assesses whether poisoned data effectively tests $HCAE$'s generalizability.

3. Research Hypothesis: Let F represent the set of extracted features from the water system data, and let E_F denote the set of embedded representations of these features in a lower-dimensional space. Furthermore, let M_1 and M_2 be two DL models trained to localize features based on E_F within the given feature space of the water system.

The hypothesis can be formulated as follows:

$$H_0 : \forall x \in E_F,\ M_1(x) = M_2(x)$$

$$H_1 : \exists x \in E_F,\ M_1(x) \neq M_2(x)$$

Where, H_0 represents the null hypothesis, suggesting that both models M_1 and M_2 yield similar predictions for any given embedded representation x. H_1 represents the alternative hypothesis, suggesting that there exists at least one embedded representation x for which the predictions of models M_1 and M_2 differ.

The goal of this hypothesis is to determine whether the two models M_1 and M_2 are capable of localizing features based on their embedded and learned representations consistently or if there are instances where their predictions diverge. The outcome of hypothesis testing will provide insights into the models' ability to effectively localize features in the water system's feature space.

4. Research Hypothesis: Let D represent the set of all possible domains in which an AI model can be deployed, and let MAA denote the proposed model-agnostic AI framework designed to achieve AIA (Assured AI) objectives, including explainability, safety, and security.

The hypothesis can be formulated as follows:

$$H_0 : \forall d \in D,\ MAA(d) \text{ achieves AIA objectives}$$

$$H_1 : \forall d \in D,\ MAA(d) \text{ does not achieve AIA objectives}$$

Where H_0 represents the null hypothesis, suggesting that the model-agnostic AI framework MAA is universally effective across all possible domains D in achieving AIA objectives. H_1

represents the alternative hypothesis, suggesting that the framework MAA fails to achieve any AIA objectives.

This hypothesis aims to determine whether the proposed model-agnostic framework, $ALSP$, can achieve AIA objectives across various domains or if there are instances where the framework falls short in meeting these objectives. The outcome of hypothesis testing will provide insights into the framework's effectiveness in ensuring explainability, fairness, and security in diverse application contexts.

3.3.2 Water Distribution and Agricultural Production Systems

This section introduces research hypotheses concerning water and agricultural systems operations. It outlines two research hypotheses as follows:

1. Research Hypothesis: Let X represent the set of all possible datasets used in precision agriculture, and let M_{old} denote the existing prediction model without the incorporation of outlier information, and M_{new} denote the new prediction model developed with the incorporation of outlier information.

The hypothesis can be formulated as follows:

$$H_0 : \forall x \in X,\ \text{Accuracy}(M_{\text{new}}(x)) \leq \text{Accuracy}(M_{\text{old}}(x))$$

$$H_1 : \forall x \in X,\ \text{Accuracy}(M_{\text{new}}(x)) > \text{Accuracy}(M_{\text{old}}(x))$$

Where H_0 represents the null hypothesis, suggesting that the incorporation of outlier information in the new prediction model M_{new} leads to an equal or less accuracy compared to the without outlier information model M_{old} in APS. H_1 represents the alternative hypothesis, suggesting that the accuracy of the new prediction model M_{new} is greater than without

outlier information model M_{old}.

The goal of this hypothesis is to determine whether the incorporation of outlier information indeed contributes to the development of a more accurate prediction model in APS. The outcome of hypothesis testing will provide insights into the impact of incorporating outlier information on the model's accuracy, such as R-square and RMSE metrics.

2. Research Hypothesis: Let P denote the set of all possible samples or instances in which water levels are overflowing. Let W_{actual} represent the actual water levels, and $W_{\text{predicted}}$ represent the water levels predicted by the model.

The hypothesis can be formulated as follows:

$$H_0 : \forall p \in P,\ |W_{\text{predicted}}(p) - W_{\text{actual}}(p)| \leq \text{Threshold}$$

$$H_1 : \exists p \in P,\ |W_{\text{predicted}}(p) - W_{\text{actual}}(p)| > \text{Threshold}$$

Where H_0 represents the null hypothesis, suggesting that for all instances p in P, the absolute difference between the water levels predicted by the model $W_{\text{predicted}}$ and actual water levels W_{actual} is less than or equal to a predefined threshold. H_1 represents the alternative hypothesis, suggesting that there exists at least one instance p in P for which the absolute difference between the predicted water levels $W_{\text{predicted}}$ and actual water levels W_{actual} exceeds the predefined threshold.

This hypothesis aims to determine whether the developed predictive model, $MM - LSTM$, can correctly identify instances in which water levels go beyond the threshold. The hypothesis testing will provide insights into the model's accuracy in identifying such instances and whether it meets the desired accuracy criteria defined by the threshold.

3.3.3 Context to AI for Water Systems

This section introduces research hypotheses concerning context-aware AI solutions. It outlines key research hypotheses as follows:

1. Hypothesis on Contextual Factors Improving Model Performance Let C represent the set of all possible combinations of contextual factors, where each combination $c \in C$ includes weather conditions, usage patterns, and infrastructure conditions. Let Performance(c) denote the prediction or classification performance of the DL model for a specific water distribution problem when considering the contextual factors in combination c.

Formulated hypotheses:

$$H_0 : \forall c \in C, \text{Performance}(c) \leq \text{Performance}_{\text{baseline}}$$

$$H_1 : \exists c \in C, \text{Performance}(c) > \text{Performance}_{\text{baseline}}$$

Explanation: This hypothesis evaluates whether including contextual factors improves the model's performance (e.g., accuracy, F1-score, RMSE) compared to a baseline model that excludes them.

2. Hypothesis on Contextual Model Accuracy Let Context(t) represent the context identified by the model at time t and TrueContext(t) represent the true context.

Formulated hypotheses:

$$H_0 : \forall t \text{ within time window}, \text{Context}(t) \neq \text{TrueContext}(t)$$

$$H_1 : \exists t \text{ within time window}, \text{Context}(t) = \text{TrueContext}(t)$$

Explanation: This hypothesis tests whether the contextual model can correctly identify the true context during various time windows, essential for real-time adaptability in WWTPs.

3. Hypothesis on Incorporating Contextual Data Incorporating contextual data improves forecasting accuracy:

$$\mathcal{E}(\hat{\mathbf{y}}_{t+1}^{\text{context}}, \mathbf{y}_{t+1}) < \mathcal{E}(\hat{\mathbf{y}}_{t+1}^{\text{no-context}}, \mathbf{y}_{t+1})$$

Explanation: This hypothesis examines whether models using contextual data achieve lower error rates compared to those that do not.

4. Hypothesis on Attention Mechanism's Impact The attention mechanism improves forecasting accuracy:

$$\mathcal{E}(\hat{\mathbf{y}}_{\text{att},\, t+1}, \mathbf{y}_{t+1}) < \mathcal{E}(\hat{\mathbf{y}}_{\text{no-att},\, t+1}, \mathbf{y}_{t+1})$$

Explanation: This hypothesis tests whether including an attention mechanism improves the accuracy by dynamically weighing input features based on their relevance.

5. Hypothesis on Forecast Bias Reduction with Quantile Loss The quantile loss function reduces forecast bias during extreme conditions:

$$\mathcal{B}_{\text{quantile}} < \mathcal{B}_{\text{standard}}$$

Explanation: This hypothesis evaluates whether the quantile loss function mitigates bias during extreme events, providing more reliable forecasts.

Table 3.1: Summary of Research Statements, Questions, and Hypotheses

Research Statements	**Research Questions**	**Research Hypotheses**
AI and AIA		
Non-deterministic tendencies in attack detection	How do AI assurance constraints improve model performance?	H_1: $\text{HCAE}_{\text{Generalizability}} > \text{AE}_{\text{Generalizability}}$
Generalizability of DL models in WDSs	Can models' generalizability be tested with GAN-generated data?	H_1: $\text{HCAE}_{\text{Generalizability}} > \text{AE}_{\text{Generalizability}}$
Feature localization in water system	How do models localize features based on embeddings?	H_1: $\exists x \in E_F$, $M_1(x) \neq M_2(x)$
Model-agnostic AI pipeline for AIA goals	How to develop a universally applicable AI pipeline?	H_1: $\forall d \in D$, $MAA(d)$ does not achieve AIA objectives
Water and Agricultural Systems		
Limited data availability in APS	How does incorporation of outlier information impact predictions in agriculture?	H_1: $\forall x \in X$, $\text{Accuracy}(M_{\text{new}}(x)) > \text{Accuracy}(M_{\text{old}}(x))$
OD and optimization in WDS	How can water levels beyond the tunnel threshold be correctly identified?	H_1: $\exists p \in P$, $\lvert W_{\text{predicted}}(p) - W_{\text{actual}}(p)\rvert >$ Threshold
Context to AI for WWTPs		
Context-aware solutions for improved accuracy	How do various contextual factors enhance DL model performance in WDS?	H_1: $\exists c \in C$, $\text{Performance}(c) > \text{Performance}_{\text{baseline}}$
Identifying appropriate context within the time window	Can the contextual model accurately identify appropriate context?	H_1: $\exists t$ within time window, $\text{Context}(t) = \text{TrueContext}(t)$
Impact of attention mechanism on forecasting	Does incorporating attention mechanism improve model accuracy?	H_1: $\mathcal{E}(\hat{\mathbf{y}}_{\text{att},\,t+1}, \mathbf{y}_{t+1}) < \mathcal{E}(\hat{\mathbf{y}}_{\text{no-att},\,t+1}, \mathbf{y}_{t+1})$
Bias reduction during extreme events	Does the quantile loss function reduce forecast bias during extreme conditions?	H_1: $\mathcal{B}_{\text{quantile}} < \mathcal{B}_{\text{standard}}$

Chapter 4

Research Methods, Techniques, and Algorithms

This chapter discusses the proposed AI framework- $ALSP$, pipelines- DeepH_2O, P$_2$O, $DeepAg$; methods, techniques, and algorithms such as $MM-LSTM$, other DL-based methods, GRU, LSTM, AE, $HCAE$, GAN, Random Forest, Isolation Forest, and Context for detecting outliers/cyber threats in WDSs and APSs. Figure 4.1 presents an overview of the framework, methods, techniques, and pipelines discussed in this Chapter. Each AI component is discussed in detail in this chapter.

4.1 Model Agnostic Assurance Framework - $ALSP$

This section presents a framework, $ALSP$, for achieving quantifiable assurance goals, including XAI, FAI, and SAI. The framework validates an AI system by providing quantifiable AIA scores using a combination of both data-driven and AI model-driven approaches. More specifically, $ALSP$ optimizes models using a game theory approach, and it also logs and scores the actions of an AI model to detect adversarial inputs and assures the datasets used for training. It is quite difficult to ensure all six goals for an AI system using a single algorithm; therefore, this dissertation proposes three separate methods in the $ALSP$ framework, including Weight Assessment, Reverse Learning, and Secret Inversion, that are capable of

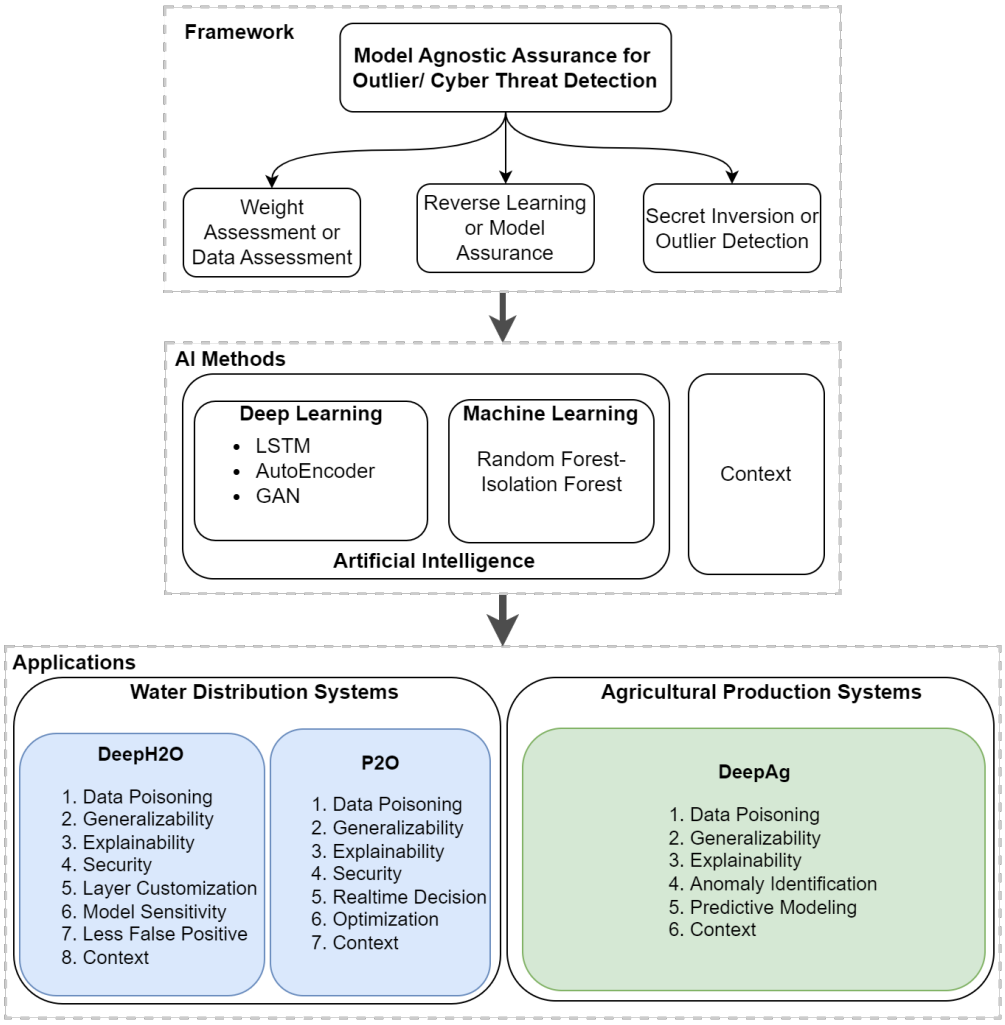


Figure 4.1: Proposed AI Framework, Pipelines, Methods and Models

achieving all three goals, including XAI, FAI, and SAI, for different contextual applications.

Figure 4.2 presents the *MAA* framework for domain-independent applications through *ALSP*. The framework includes three methods: Weight Assessment, Reverse Learning, and Secret Inversion.

A detailed description of all three methods is discussed as follows:

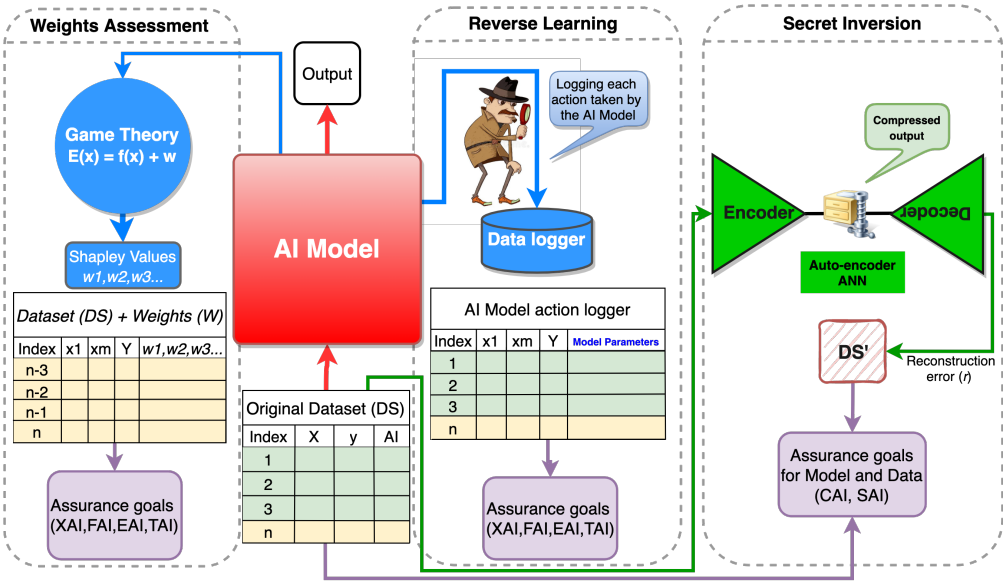


Figure 4.2: Adversarial Logging Scoring Pipeline

4.1.1 Weight Assessment

The Weight Assessment is a method that applies Game Theory to the AI model of particular interest for calculating the Shapley values (Shapley [200]) at every epoch of learning; this method aims to achieve assurance goals, including XAI as a form of quantifiable AIA scores by assigning scores per data point and not as an aggregate score. It is important to note that this dissertation assumes an AI model is required for scoring the given dataset. For the Weight Assessment study, a tree-based algorithm, an Extreme Gradient Boosting Decision Tree (XGBDT), is applied as the baseline model. Shapley values are outcomes of a *game* that assumes cooperation among players and achieves overall gain from alliances. These values represent each player's bargaining power and the payoff that is reasonable to expect in a given context. A characteristic function can represent the alliance in the game. This characteristic function (v) can be mapped as $v : 2^N \rightarrow R$ for a set of N; therefore, each player i gets a *fair* distribution, assuming the game is cooperative and can be represented mathematically as (4.1). Considering a fair cooperation game, a player can expect $v(S \cup \{i\}) - v(S)$ by

averaging the set of possible different permutations in which the alliance was formed.

$$\varphi_i(v) = \frac{1}{n} \sum_{S \subseteq N \setminus \{i\}} \left(\begin{array}{c} n-1 \\ |S| \end{array} \right)^{-1} (v(S \cup \{i\}) - v(S)) \tag{4.1}$$

A unique set of values that indicates the importance of each feature in a given dataset is assigned. Along with the game theory-driven weights, these values also represent a heuristic expectation from an assurance perspective provided by domain experts. Similar to how labels are used as independent variables in a training-testing learning approach, this dissertation introduces the addition of assurance labels: AIA Columns (AIAC). However, AIAC values, as assigned by the domain expert, are not directly provided as inputs to the AI model. For instance, some features in a dataset can be relevant to specific assurance goals such as explainability (consider data on demographics); accordingly, these features are labeled as XAI features. The use cases in this manuscript provide further information on the usability of these labels.

1st block of Figure 4.2 represents steps of the Weight Assessment method. Algorithm 1

Algorithm 1 Weight Assessment

Input: AI model, dataset D, assurance labels $AIAC$
Output: AIA scores
Initialize Shapley values $\varphi_i(v)$ for each feature
Train AI model on D
for i in features of D **do**
    for π in all permutations of features excluding i **do**
        Calculate $v(S \cup \{i\})$ and $v(S)$
        Update $\varphi_i(v)$ using Equation (4.1)
    end for
end for
return AIA scores

presents the detailed steps involved in executing the weight-assessment method. To generate AIA scores, the multiplication of Shapley value weights $\varphi_i(v)$ and AIACs ($AIAC_i$)

is performed. This matrix multiplication generates the AIA score for each row/column. Experimental work provides further information on that.

4.1.2 Reverse Learning

Reverse Learning is a log-based method (2nd block of Figure 4.2) that can trace back assurance issues using a table of recorded learning actions (i.e. reverse engineering). Reverse Learning also accomplishes AIA goals such as XAI. XAI can be enforced if learning details and evolution are available through a log of actions. The *log* records the learning process and indicates points in time during learning where the algorithm's learning accuracy, for instance, has decreased or ceased to improve. That is illustrated in the experimental section. While devising the method, the Gradient Boosting Decision Tree (GBDT) algorithm is employed as a representative instance. However, it's important to note that while GBDT is chosen as the primary algorithm for this example, alternative AI algorithms can also be tested for experimentation. While developing the AI algorithm, the primary focus was to log each learning epoch's actions. The outcome of this algorithm is two-fold: the optimized number of epochs to minimize the loss function and a logged action of each epoch. For GBDT, values including pseudo-residuals r_{im}, gamma γ_{im}, log of odds for the labels $l_m(x)$, probability p_{mi} are saved during each epoch. Equation 4.2 presents a prediction of the GBDT model after each epoch. Equation 4.3 presents the logarithmic loss function of the GBDT model.

$$f_{mi} = \begin{cases} 0 \rightarrow p_{mi} < 0.5 \\ 1 \rightarrow p_{mi} >= 0.5 \end{cases} \tag{4.2}$$

$$L = -\sum_{i=1}^{N} \left(y \log(odds) - \log\left(1 + e^{\log(odds)}\right)\right) \tag{4.3}$$

Algorithm 2 Reverse Learning

Input: AI model, training data D_{train}
Output: Optimized number of epochs, logged actions
Initialize empty logs for each epoch
for i in number of epochs **do**
    Train AI model for one epoch on D_{train}
    Calculate pseudo-residuals r_{im}, gamma γ_{im}, and other values
    Save learning details in logs for epoch i
    Update model weights
end for
Analyze logs to find the optimized number of epochs
Analyze logs for assurance issues
    return Optimized number of epochs, logged actions

Unlike Weight Assessment, Reverse Learning doesn't provide AIA scores; however, it serves as a tool to manually verify and optimize the AI algorithm. Algorithm 2 presents the steps in executing the Reverse Learning method.

4.1.3 Secret Inversion

The Secret Inversion method (3rd block of Figure 4.2) performs exhaustive comparisons amongst features by reconstructing them using an *Autoencoder*. It is assumed that, given the reconstruction errors (r) work using an encoder-decoder mechanism, the AIA scores that are relevant in this case are goals such as SAI and CAI. An AE learns a meaningful pattern of the data model by reducing its feature dimensions. It consists of a feed-forward neural network and works as a self-supervised model where the encoder-decoder can be characterized as an hourglass shape compressor and decompressor. The encoder compresses feature space and translates it into codes that the decoder decomposes. The decoder reconstructs the feature space from the codes. However, the reconstructed signal is not always the same as the input (if it's not the same, it indicates an alteration in the data - which could be a SAI/CAI issue). This difference is represented by the reconstruction errors (r). Let ϕ and

ψ as the encoder and decoder respectively, mathematically an AE can be represented as Equations 4.4, 4.5:

$$\phi : \mathcal{X} \to \mathcal{F} \tag{4.4}$$

$$\psi : \mathcal{F} \to \mathcal{X} \tag{4.5}$$

Here, $X \in \mathcal{X}$ is input for the encoder, and $F \in \mathcal{F}$ represents the code as the input of the decoder. By defining ϕ and ψ, the AE minimizes the reconstruction errors and measures connection weights across the training phase for all elements of input X. It can be defined mathematically as Equation 4.6:

$$\|X - (\phi \circ \psi)X\|^2 \tag{4.6}$$

Where $\circ$ is a composition operator. The reconstruction error can be minimized using either the Adam Optimizer or the Stochastic Gradient Descent (SGD) algorithm. Both optimizers can quickly update the connection weights after a few learning cycles. Algorithm 3 presents the steps involved in executing the Secret Inversion method.

Algorithm 3 Secret Inversion

Input: Dataset D, Autoencoder model, reconstruction errors r
Output: AIA scores (SAI and CAI)
Train Autoencoder model on D
for each feature F_i in D **do**
    Encode F_i using the Autoencoder: $F_i' = \phi(F_i)$
    Decode F_i' using the Autoencoder: $F_i'' = \psi(F_i')$
    Calculate reconstruction error for F_i: $r_i = \|F_i - F_i''\|$
end for
Calculate SAI and CAI based on reconstruction errors
    return SAI and CAI scores

4.2 Outlier Detection Methods for CPSs

This section presents detailed methodologies for *DeepAg* OD methods using ML and DL models in WDS and APS.

4.2.1 Outlier Detection in Agricultural Production Systems - *DeepAg*

This subsection presents OD methods using ML and DL models in APS.

4.2.1.1 Isolation Forest

The attribute of Isolation Forest (Figure 4.3) is that there is a tendency to separate outlier data points from normal data points, as the algorithm randomly selects an attribute and splits values between the minimum and maximum of that attribute. The OD in this work is based on having the economic indices (Crude Oil, Gold, Dow Jones, S&P 500, VIX) as an input to the Isolation Forest algorithms as shown in Figure 4.4. This algorithm is designed for unsupervised anomaly detection. It isolates outliers in a dataset by constructing a binary tree structure. The key idea is that outliers are more likely to be isolated in the early stages of tree construction, while normal data points are likely to appear deeper in the tree. The algorithm assigns an anomaly score to each data point based on the number of splits required to isolate it. As shown in Figure 4.3, the Isolation Forest model constructs an ensemble of isolation trees. Each tree is built as follows:

1. Randomly select a subset of the data points and features to create a sub-sample.
2. Construct a binary tree recursively as follows:
   - Select a feature randomly.

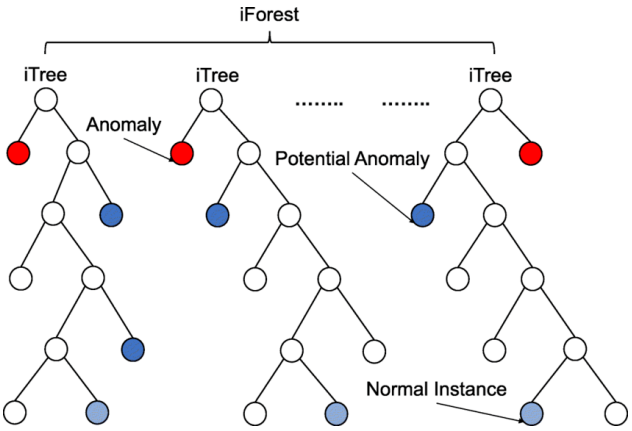


Figure 4.3: Isolation Forest Method for OD (Regaya et al. [1])

- Randomly choose a split value between the minimum and maximum of the selected feature.

- Split the data points into two partitions based on the selected feature and split value.

- Recursively repeat the process on each partition.

The tree construction continues until all data points are isolated or a predefined maximum tree depth is reached; $(T_1, T_2, \ldots, T_n)$- here, T_i represents the i-th isolation tree in the ensemble.

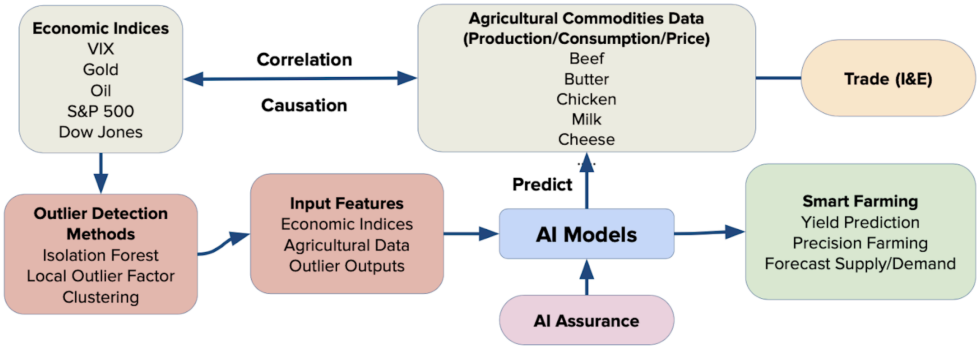


Figure 4.4: *DeepAg* Methodology (Gurrapu et al. [2])

The Isolation Forest algorithm is effective for detecting outliers in high-dimensional datasets and is computationally efficient due to its use of random sub-sampling and binary tree

construction. Random partitioning creates a short path for outliers. Therefore, a sample is more likely to be an outlier if a forest of random trees collectively produces shorter path lengths for particular samples. The algorithm takes several hyperparameters as input; among them, the most important is the contamination rate. It estimates the percentage of outliers that can be approximately predicted from the total data points.

4.2.1.2 Anomaly Detection Thresholds and Contamination Rates

The contamination rate is determined using a popular statistical measure known as the Interquartile Range (IQR) (Figure 4.5). IQR describes the middle 50% of the data distribution. Quartiles slice any Gaussian distribution into four equal groups of 25%. Calculating IQR describes the middle half of the data in the distribution set. These middle data segments are considered normal data points, given that outliers typically reside at the tails of a Gaussian distribution. The representation of the IQR can be expressed as Equation 4.7:

$$\text{Interquartile Range } = Q3 - Q1 \tag{4.7}$$

Where Q1 is the first quartile or 25th percentile and Q3 is the third quartile or 75th percentile.

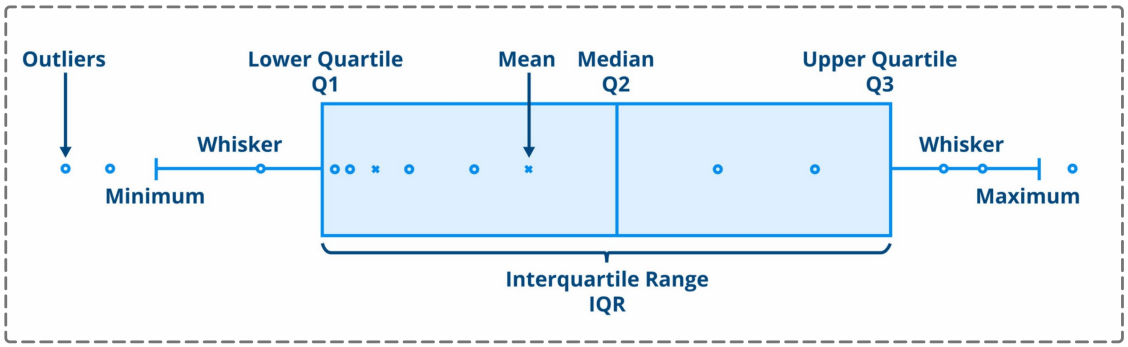


Figure 4.5: Interquartile Range Diagram

Tables 4.1a and 4.1b provide the daily and monthly contamination rate of all financial indices using the IQR method. The Isolation Forest algorithm identifies an estimation of the anomaly score using Equation 4.8, for a given instance x using this formula:

$$s(x, m) = 2^{-E(h(x))/c(m)} \tag{4.8}$$

where E is the average of h trees and c(m) is the average path length of unsuccessful binary searches.

Table 4.1: Contamination Rates for Financial Indices using IQR

(a) Daily Data Contamination (%)

Financial Index	Contamination Rate
VIX	6.559
Gold	5.382
S&P 500	6.008
DOW	6.125
Crude Oil	3.953

(b) Monthly Data Contamination (%)

Financial Index	Contamination Rate
VIX	6.250
Gold	2.232
S&P 500	2.232
DOW	2.232
Crude Oil	6.250

Then, a threshold value T is selected using Equations 4.9 & 4.10 to classify data points as outliers. Data points with an anomaly score below T are considered normal, while those above T are classified as outliers. The threshold values are selected from Tables 4.1a and 4.1b (contamination rates).

$$\text{If } S(x) < T, \text{ then } x \text{ is a normal data point.} \tag{4.9}$$

Algorithm 4 Isolation Forest for Outlier Detection

Input: Dataset X, number of trees T, sub-sampling size S
Output: Outlier scores for each data point
Initialize an empty set of isolation trees: $F = \{\}$
for $t = 1$ to T **do**
    Randomly select S samples from X without replacement to create a sub-sample X_s
    Create a new isolation tree T_t using X_s as follows:
        If X_s contains only one point or maximum depth is reached:
            Create a leaf node with that point.
        Else:
            Randomly select a feature A from the remaining features.
            Randomly select a split value p for feature A within its range in X_s.
            Split X_s into two subsets: X_{left} containing points with $A \leq p$ and X_{right} with $A > p$.
            Create a non-leaf node with feature A and split value p.
            Recursively build the left subtree using X_{left} and the right subtree using X_{right}.
    Add the newly created isolation tree T_t to the set F
end for
Compute the anomaly score for each data point in X as follows:
for each data point x in X **do**
    For each isolation tree T_t in F, traverse the tree to find the depth $d_t(x)$ at which x is isolated.
    Calculate the average depth across all trees: $D(x) = \frac{1}{T}\sum_{t=1}^{T} d_t(x)$
    Compute the anomaly score for x: $S(x) = 2^{-\frac{D(x)}{c}}$, where c is a normalizing factor.
end for
    return Anomaly scores for each data point

$$\text{If } S(x) \geq T, \text{ then } x \text{ is an outlier.} \tag{4.10}$$

Algorithm 4 presents the detailed steps involved in executing OD in economic data using Isolation Forest.

4.2.2 Prediction in Water Distribution Systems - P_2O

This subsection focuses on the Multivariate Multi-step LSTM ($MM - LSTM$) and other AI models for tunnel water level prediction and optimization. A schematic diagram of

the methodology used for this module is shown in Figure 4.6. As shown in the figure, this module has five components: Data Preprocessing, Exploratory Data Analysis (EDA), Model Development, Hyperparameter Tuning, and Model Evaluation & Selection. The details of these components are discussed in the next subsection.

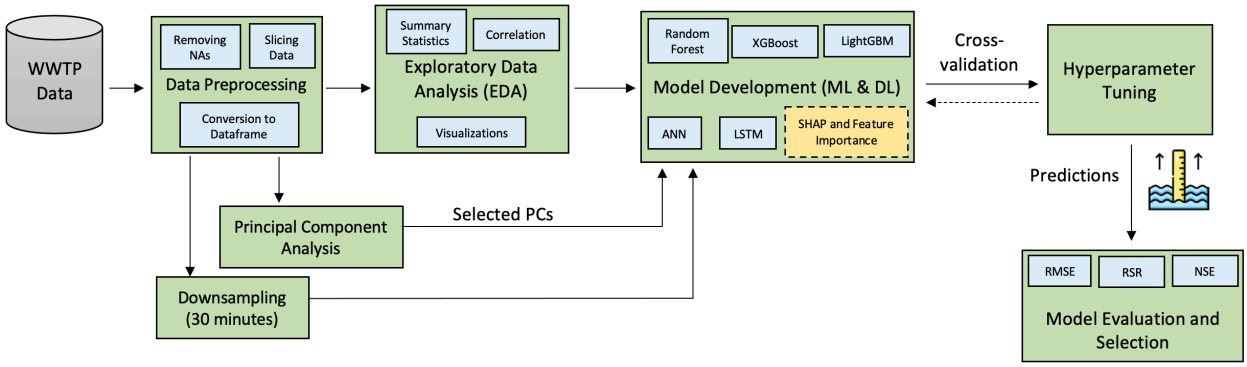


Figure 4.6: A Schematic Diagram of the Methodology Used for Tunnel Water Level Prediction. (Kulkarni et al. [3])

4.2.2.1 Data Preprocessing for Tunnel Wastewater Level Prediction

The data used for this work are from an actual WDS, including 243 columns in each file; thus, the first task was to understand the Not Available (NA) sensor readings in the data. The reason for NAs is due to the format of the data produced by the reporting tool while fetching the data. Thus, NAs were removed from the data, and the output was identified. After preprocessing, selecting output, and combining the data into a data frame, there were 42 columns in the data. This combined data frame also had NAs in the first 60,301 rows, which were removed. Finally, at the end of the preprocessing phase, the data consisted of 42 columns and 367,943 rows.

In the next step, two different versions of the preprocessed data were created - based on Principal Component Analysis (PCA) and Sampling - to understand the effects of data processing techniques and maintain AI assurance. Abdi and Williams [320] provided evidence

that PCA is a widely used technique that provides a set of uncorrelated variables from a set of correlated variables. Thus, considering collinearity in the data, the PCA technique for preprocessing was selected. The second dataset was produced based on downsampling and was performed by selecting the observations based on intervals of 30 minutes. This way, two versions of the data were produced based on the raw data at the end of the preprocessing phase.

4.2.2.2 Multivariate Multi-step LSTM Model Development

A $MM-LSTM$ time series and DL-based model (Figure 4.7) is derived using LSTM architecture, which can take multiple inputs for predicting multistep outputs. Before discussing the modeling steps, an effective objective function (i.e., loss function) is needed to achieve the desired outcome (i.e., accurate prediction of water level peaks or effective pump action generation). The goal is to accurately predict water levels while prioritizing detecting water overflow incidents (anomalous data points). The following considerations are made: choosing the Huber loss as the cost function (Equation 4.11) is made due to outliers in the data caused by extreme weather days and water overflow situations.

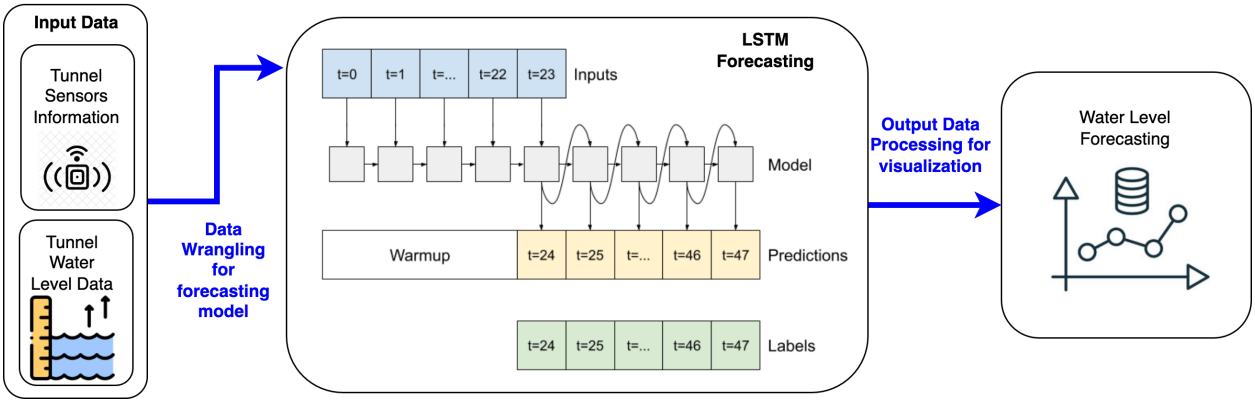


Figure 4.7: Water Level Forecasting LSTM Architecture

Given the presence of significant yet infrequent overflow incidents in the dataset, opting for

the cubic difference as a loss function is a wise decision. Additionally, the primary focus is predicting water level peaks, particularly overflow situations, rather than comprehending the general time series trend. In the context of application, the overflow samples are considered outliers in the data.

The training and evaluation of the proposed multivariate forecasting model (depicted in Figure 4.7) involves incorporating a double LSTM layer and one fully connected layer to predict the subsequent multi-output hours of tunnel water levels. To achieve this, sequences of 24/48/72-hour lengths are created and employed for model training, resulting in a two/three/four/five/six-hour multistep forecast. The optimization of the objective function (Equation 4.11) is accomplished using the Adam optimizer. Following the training phase, model performance is assessed using the test dataset, which comprises 30% of the original dataset.

A subsequent step involves transforming the model's outputs back to the original scale to facilitate the evaluation of forecasted water levels against the actual tunnel water levels. The prediction of peak water levels assumes paramount importance, as the optimization process hinges on these extreme water level predictions to formulate optimal discharge and pumping actions. For peak detection, a threshold of -47ft is chosen empirically; consequently, tunnel overflow is considered to occur when water levels exceed the value of -47ft for the development of the model.

4.2.2.3 Model Optimization and Explainability

The $MM-LSTM$ is optimized using the Huber loss function (4.11), also known as the Huber loss or Huber-M loss, which is a loss function used in regression tasks. It combines the characteristics of the Mean Absolute Error (MAE) loss and the Mean Squared Error

(MSE) loss, making it less sensitive to outliers than MSE while maintaining some of its desirable mathematical properties.

$$L(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{if } |y - f(x)| \leq \delta \\ \delta(|y - f(x)| - \frac{1}{2}\delta) & \text{if } |y - f(x)| > \delta \end{cases} \tag{4.11}$$

where,

- $L(y, f(x))$ represents the Huber loss between the true target value y and the predicted value $f(x)$.

- $|y - f(x)|$ is the absolute difference between the true value and the predicted value.

- δ is a hyperparameter that defines the point where the loss function transitions from quadratic (MSE-like) to linear (MAE-like). It is a non-negative constant.

- When $|y - f(x)| \leq \delta$, the loss function is quadratic, similar to MSE.

- When $|y - f(x)| > \delta$, the loss function is linear, similar to MAE.

The choice of δ controls the balance between the two loss components and determines the smooth transition region between the quadratic and linear parts of the loss function. Smaller values of δ make the loss more robust to outliers. This loss function is commonly used in robust regression tasks where the dataset may contain outliers that can significantly influence the model's performance if a purely quadratic loss (MSE) is used. Algorithm 5 presents the detailed steps in executing forecasting and OD in tunnel water level data using $MM-LSTM$.

Neural networks inherently function as black boxes, posing challenges in their interpretation. To overcome this, this work utilizes the SHAP framework to estimate Shapley values (Shapley [200]), which leverages the game theory rule. This technique evaluates the model's predictions and elucidates the contributions of each feature to each prediction. Specifically, the process, referred to as a deep explainer, dissects each model outcome and backpropa-

Algorithm 5 Multivariate Multistep LSTM with Huber Loss for Tunnel Water Level Forecasting with Anomaly Detection

Input: Multivariate time series data X, LSTM model parameters, prediction horizon H, anomaly threshold T
Output: Forecasted water levels and anomaly labels
Split X into training (X_{train}) and testing (X_{test}) datasets
Train the LSTM model on the training dataset using Huber loss
Initialize model parameters and hyperparameters: $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$
Training Phase:
for i in number of epochs **do**
    for j in mini-batches of training data **do**
        Forward pass: Encode and decode the data,
        $\hat{X} = \text{LSTM}(\text{Encode}(X_{\text{batch}}, \Theta), \Theta)$
        Calculate Huber loss for H-step ahead predictions:
        $L_{\text{Huber}} = \text{HuberLoss}(X_{\text{batch}}, \hat{X}, \delta)$
        Backpropagation: Update model weights to minimize Huber loss,
        $\Theta \leftarrow \Theta - \eta \nabla L_{\text{Huber}}$
    end for
end for
Testing Phase:
for k in mini-batches of testing data **do**
    Forward pass: Encode and decode the data,
    $\hat{X} = \text{LSTM}(\text{Encode}(X_{\text{batch}}, \Theta), \Theta)$
    Calculate Huber loss for H-step ahead predictions for each sample:
    $L_{\text{sample}} = \text{HuberLoss}(X_{\text{batch}}, \hat{X}, \delta)$
    if $L_{\text{sample}} > T$ **then**
        Mark as an anomaly
    else
        Mark as normal
    end if
end for
    return Forecasted water levels and anomaly labels

gates the contribution of all neurons to every feature. It then compares the activation of each neuron to its reference activation, assigning contribution scores based on the resulting differences. Following the computation of multipliers on a representative dataset, feature importance is deduced using a subset of input samples (900 training data samples) and subsequently ranked based on their pronounced contributions to the model's outcomes (Kulkarni et al. [3]). These contribution weights are averaged across all 100 samples.

A DL model's prediction is essential to interpreting the results at the WDS plant. Using this feature importance approach, an operator can get insight into the plant and devise action

plans to promote the desired operational outcome. Furthermore, additional insights into the importance of features help optimize the operational process. For example, essential features dictate when and which pump must start to avoid tunnel water overflow.

4.3 Cyber-attack Detection in Water Distribution Systems - DeepH$_2$O

This section discusses the unsupervised WDSs attack detection model- the mechanism of AE and its revised version, $HCAE$. This work applies AE as a reconstruction-based algorithm that performs dimensionality reduction and reconstructs the original input. The outcome from AE and $HCAE$ are reconstruction errors (difference between output and input data), which identify physical anomalies from the feature space. Figure 4.8 shows a fully connected ANN-based AE and its components. WDS data are fed to the AE and $HCAE$ models. Based on a threshold, the models classify the inputs as normal or anomalous samples.

4.3.1 Auto Encoder

AEs have been a widely adopted DL method for the last couple of decades for both dimensionality reduction and feature engineering (Zhai et al. [321]). This work develops the baseline AE model by adopting an approach from Taormina and Galelli [94].

The AE network is divided into two parts: an encoder function $h = f(X)$ and a decoder function $x' = g(f(x))$. AEs can be generalized as stochastic mappings of $P_{encoder} = (h|x)$ and $P_{decoder} = (x|h)$, where h is a hidden layer $h = f(x)$ that presents a code and is used to characterize the input. Multi-layer perceptrons (MLP) (Chen et al. [322]) form AEs with an input layer, an output layer, and multiple hidden layers. Mathematically, an encoder and

decoder can be written as Equations (4.12 - 4.14):

$$\phi : \mathcal{X} \rightarrow \mathcal{F} \tag{4.12}$$

$$\psi : \mathcal{F} \rightarrow \mathcal{X} \tag{4.13}$$

$$\phi, \psi = \underset{\phi,\psi}{\arg\min} \|\mathcal{X} - (\psi \circ \phi)\mathcal{X}\|^2 \tag{4.14}$$

where Equations 4.12 and 4.13 represent encoder and decoder functionality, respectively; Equation 4.14 represents the proposed loss of the AE, mean squared error.

Input data ($\mathcal{X}$) are transformed into a compressed representation $\mathcal{F}$ and reconstructed as $\mathcal{X}$ again. The objective of an AE is to minimize the reconstruction errors (Equations 4.15 and 4.16), which yields a better reconstruction of the input set $\mathcal{X}$.

$$\mathcal{L}\left(\mathbf{x}, \mathbf{x}'\right) = \left\|\mathbf{x} - \mathbf{x}'\right\|^2 \tag{4.15}$$

$$\mathcal{L}\left(\mathbf{x}, \mathbf{x}'\right) = \left\|\mathbf{x} - \sigma'\left(\mathbf{W}'(\sigma(\mathbf{W}\mathbf{x} + \mathbf{b})) + \mathbf{b}'\right)\right\|^2 \tag{4.16}$$

Reconstruction errors are generally minimized using SGD Bottou [323], a potent optimization tool for many DL applications. However, Adam optimizer, another powerful stochastic optimization method, is applied in this work that outperforms SGD Ruder [324].

An anomaly detection system is expected to produce minimal false alarms, as false alarms are associated with expensive maintenance operations. Figure 4.8 represents an ANN-based AE. Despite having fine-tuned hyper-parameters, AE suffers from non-determinism during training, resulting in a higher reconstruction error. A higher reconstruction error can result in increased false positives, thus affecting the detector's performance (Pang et al. [325]).

As AE algorithms automatically learn features by performing feature engineering for dimen-

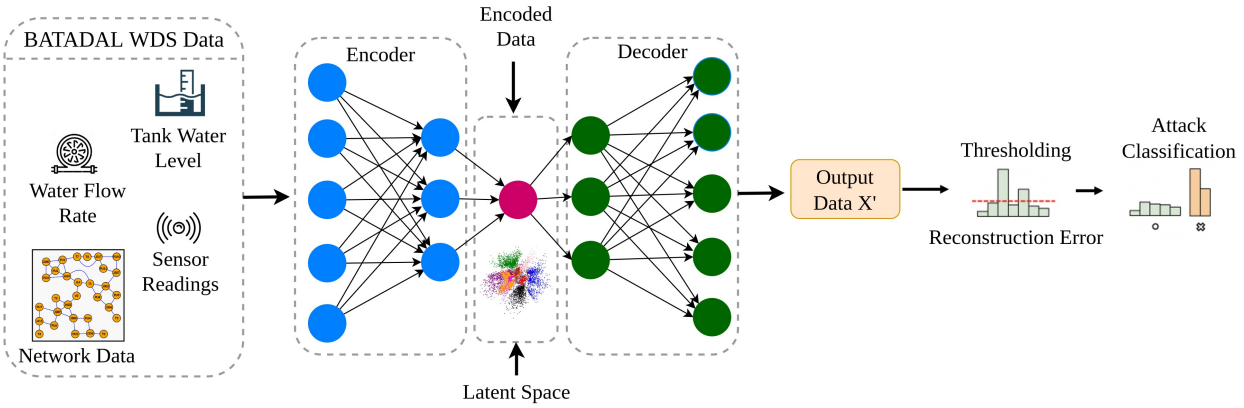


Figure 4.8: Fully Connected ANN-based Autoencoder for WDS

sionality reduction, they tend to learn different features at each time (Zhai et al. [321]). This pattern of learning is suitable for systems where feature importance is unknown (for instance, thousands of sensor values in a WDS, complex and difficult feature space to human perception). However, such a non-deterministic learning pattern might not be suitable for a WDS. It's a natural expectation for a WDS model to provide, if possible, zero false alarms because of expensive maintenance operations. To address these issues, this work revises AE architecture and forms *HCAE*, thus solving the non-determinism problem of AE by further reducing reconstruction errors and improving attack detection performance by reducing false positives.

4.3.2 High Confidence AutoEncoder

HCAE is a modified version of baseline AE; in this work, *HCAE* is developed by applying assurance methods (Equations 4.17-4.20) to AE, improving the attack detection performance compared to the AE (baseline). *HCAE* successfully represents input features in a manifold space that generates minimum reconstruction errors while decoding and recreating the input features.

To minimize the reconstruction errors, the practitioners currently follow a trial-and-error-based method and optimize hyperparameters over multiple iterations. This approach is empirical; reducing the reconstruction errors is time-consuming and computationally expensive in a complex WDS. This work applies a combination of neural network layer constraints for the model and achieves deterministic learning, which results in a manifold representation that yields minimum reconstruction errors. This strategy ensures learning a set of expected features from the training data each time. Resulted in reconstruction errors yielding better feature representation and attack detection performance (Tan and Eswaran [326]). Experimental results are presented in the context of a WDS: how a set of constraints yields minimal reconstruction errors and robust attack detection.

4.3.2.1 AI Assurance Constraints for the Auto Encoder

Recent advancements in DL APIs, including Keras[1], Tensorflow[2], and Pythorch[3], expedite AEs development more than ever. Nevertheless, a lack of a clear understanding of the fundamental properties of dimensionality reduction leads to a complex and inferior model. Thus, it is crucial to understand and adopt the basic properties of dimensionality reduction in AEs. Multiple custom layer constraints are explored and applied to facilitate dimensionality reduction in a WDS. *HCAE* is effective for tuning and optimizing hyperparameters.

In order to improve the AE detection performance, the following set of constraints are applied.

1. Tied Weights: Tied Weights (Alain and Olivier [327]) ensure equal weights for both encoder and decoder. This constraint also ensures easy learning, especially PCA-like

[1]github.com/fchollet/keras
[2]github.com/tensorflow
[3]github.com/pytorch/pytorch

dimensionality reduction and regularization. However, they do not always perform well on complex non-linear models. Again, tied weight constraint is not always necessary to improve the representation continually. If reconstruction errors are reasonable, the coding generates orthogonal latent features for given data. Such representation is helpful in dimensionality reduction and, eventually, for anomaly detection. In a multi-layer AE, weights vectors of layer l from an encoder and a decoder are transposed as Equation (4.17).

$$W_l = W_{-l}^T \tag{4.17}$$

2. Orthogonal Weights: Each weight vector is independent; therefore, the weights of each encoding layer are orthogonal. The orthogonality constraints (Huang et al. [328]) act as regularization for the AE. Mathematically, the orthogonality condition for AE can be presented as,

$$W_{encoder}^T W_{encoder} = I \tag{4.18}$$

On applying, this constraint penalizes non-orthogonal weights. The user can choose either orthogonal or non-orthogonal weights depending on the dataset. Thus, the application of this constraint is conditioned on regularization.

3. Uncorrelated Features: If the output of the encoder is orthogonal, latent representations must be uncorrelated (Kim and Choi [329]). Hence, the output of the AE must meet the condition of Equation 4.19:

$$\text{correlation}\left(O_{\text{encoder }_i}, O_{\text{encoder }_j}\right) = 0 | i \neq j \tag{4.19}$$

4. Unit Norm: The weights of each layer must have unit norms (Douglas et al. [330]).

This property helps to control exploding and vanishing gradients. Unit norm constraint (Equation 4.20) must be allied to all the layers of the AE.

$$\sum_{j=1}^{p} w_{ij}^2 = 1; i = 1, \ldots, k \tag{4.20}$$

These four constraints (Equations 4.17 - 4.20), during model development, ensure the model does not create a sub-optimal decision boundary. They ensure the creation of a well-posed AE while constructing a highly confident cyber-attack detection model for WDS.

Unit norm and orthogonality solve regularization problems, especially for AEs, when AE learns from a training set but does not represent a test set well. Also, tide weight can reduce the number of parameters as a regularization technique. The unit norm constraint addresses the exploding gradients issue by bounding gradients into a finite value. Additionally, orthogonality resolves the vanishing gradients problem by assigning fewer non-zero weights, so only informative weights stand out. Thus, only these non-zero weights flow information during backpropagation and resolve the vanishing gradient issue (Alain and Olivier [327], Huang et al. [328], Kim and Choi [329], Douglas et al. [330]).

When applying these four constraints (Equations 4.17 - 4.20) while designing $HCAE$, a hypothesis is introduced whether $HCAE$ will not converge to a suboptimal point. To test the hypothesis, results before and after using these four constraints are compared; observe if attack detection performances (F1 score and false positives) are improved from the baseline AE (Taormina and Galelli [94]).

Later in the work, adversarial testing is presented using a GAN to observe if the model can detect attacks from synthetically generated poisoned datasets (previously unseen data with a different distribution), thus testing generalizability on unseen poisoned data.

4.3.3 Attack Detection and Calibration Stages of HCAE

HCAE is capable of binary classification, whether the WDSs are under "*ATTACK*" or "*NO ATTACK*" by investigating each sample. Nonetheless, the direct classification technique becomes erroneous with a small and imbalanced dataset because AE requires a large dataset to learn the representation. However, with *HCAE*'s deterministic learning, it's hypothesized that the *HCAE* can detect attacks with minimum false positives. The hypothesis is tested by training both AE and *HCAE* with the same imbalanced data and evaluated with total false positives for each model.

The data streams are presented with non-sequential representation for this work. Next, (X) is defined as $X \epsilon \mathbb{R}^{N \times m}$ which contains N-dimensional observation for m different features; and X_i denotes the systems' values at time i. The attack detection process has two stages (Figure

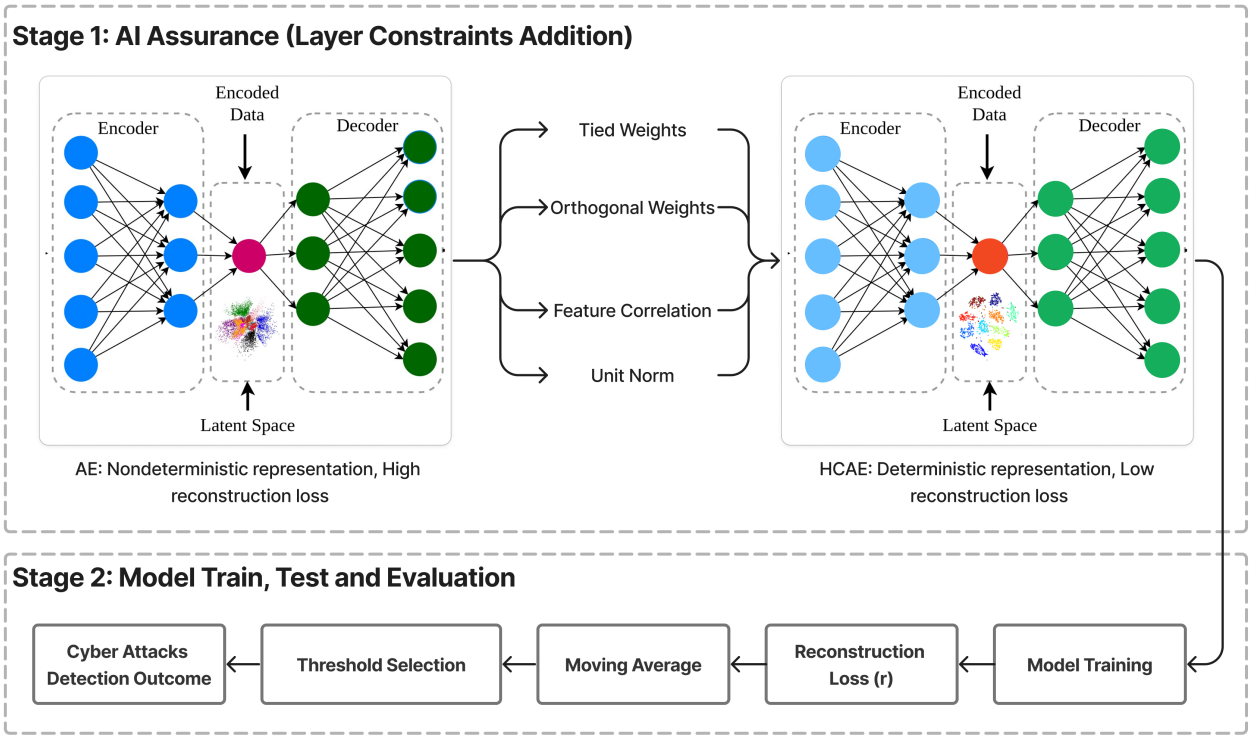


Figure 4.9: *HCAE* Model Development and Attack Detection Workflow

4.9). In stage one, custom layers are created to form $HCAE$ by following Equations (4.17-4.20). Observing the model performance on multiple hyperparameter sets has ensured that the model always yields minimum reconstruction errors and maximum binary classification performance (F1 score). Later in stage two, data are preprocessed and normalized to have the maximal absolute value of each attribute applied to all three provided datasets. After that, $HCAE$ is trained with the normal training dataset labeled "$NO\ ATTACK$", splitting the training dataset into training (X_{train}, Y_{train}) and validation (X_{val}, Y_{val}) set. To avoid learning bias, early stopping is applied, which is another regularization scheme for preventing over-fitting on the training dataset when the model converges.

During each epoch, losses are estimated using Equation 4.17, the squared of the reconstruction errors $r_e = |x - x'|$; minimize them using Adam optimizer. After both AE and $HCAE$ models are well-trained, a threshold θ_{th} is selected in an empirical fashion. For that, the range of average reconstruction errors among all features in a sample is observed and summed up. The threshold is applied based on the final range estimation, as shown in the following Equation 4.21.

$$\theta_{th} = \max \left\{ f(x) : \sum_{m} \frac{|x - x'|}{m} \text{ for } N_{training-samples} \right\} \tag{4.21}$$

The calibration process is crucial to derive a concise threshold θ_{th} for testing the model on new samples. If a test object is classified as "ATTACK," $HCAE$ localizes the features associated with attacked attributes, such as pumps, sensors, and valves, using estimated reconstruction errors.

For the $HCAE$ model, hyperparameters are selected using random search and optimized, resulting in the best model's performance (F1-score and false positives). The objective is to compare the performance of $HCAE$ (AE with constraints) with the AE model (without

constraints). To facilitate a fair comparison between $HCAE$ and AE models, Taormina's AE model is retained using the same hyperparameters of the $HCAE$ model; this refers to the retrained AE model as the baseline AE model. Algorithm 6 presents the detailed steps involved in executing cyber-attack detection in a WDS network data using $HCAE$.

Algorithm 6 Cyber Attacks Detection using $HCAE$

Input: Raw network traffic data X, $HCAE$ model parameters Θ
Output: Detected attacks (binary labels)
Split X into training (X_{train}) and testing (X_{test}) datasets
Train the $HCAE$ model on the training dataset
Initialize model parameters and hyperparameters: $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$
Training Phase:
for i in number of epochs **do**
    for j in mini-batches of training data **do**
        Forward pass:
            $\hat{X} = \text{Decode}(\text{Encode}(X_{\text{batch}}, \Theta), \Theta)$
        Calculate reconstruction loss:
            $L_{\text{reconstruction}} = \sum_{p=1}^{P} \|X_{\text{batch}}^{(p)} - \hat{X}^{(p)}\|^2$
        Backpropagation:
            Update model weights: $\Theta \leftarrow \Theta - \eta \nabla L_{\text{reconstruction}}$
    end for
end for
Testing Phase:
for k in mini-batches of testing data **do**
    Forward pass:
        $\hat{X} = \text{Decode}(\text{Encode}(X_{\text{batch}}, \Theta), \Theta)$
    Calculate reconstruction loss for each sample:
        $L_{\text{sample}} = \|X_{\text{batch}} - \hat{X}\|^2$
    if $L_{\text{sample}} > T$ **then**
        Mark as an attack
    else
        Mark as non-attack
    end if
end for
    return Detected attacks (binary labels)

4.3.4 Synthetic Water Distribution Systems Poisoned Data Generation

Unlike other DL-based attack detection approaches that require significant domain knowledge and passive awareness of the attacked model (Erba et al. [331]), GANs are proven to be effective in generating realistic attack samples (poisoned data) (Muñoz-González et al. [332]) with minimal information about either the domain or the DL model. A GAN architecture proposed by Goodfellow et al. [290] is used for synthetic data generation. GAN network learns feature statistics of a given dataset for generating a new set of synthetic data. A generator produces the synthetic samples in the GAN network, and a discriminator evaluates them. The generator learns by mapping latent feature space to a data distribution of particular interest. Discriminator maximizes its objective by learning how to distinguish original samples from the generated fake samples. The generator aims to minimize the discriminator's objective by fooling it into thinking otherwise (fake samples as real ones). For instance, while generating a synthetic data set, GAN keeps similar statistics to the generated data set from the training set; hence, those generated data distributions look superficially similar to human perception. During the training phase, both the generator and discriminator play a minimax game where a bi-level optimization (Equations 4.22) is performed to train the GAN network.

In this work, the generator learns the distribution of training samples X and maps data space as $G(z;\theta_g)$, where G is differentiable with respect to parameters θ_g. Then, the discriminator investigates whether the data comes from the training samples and not the generator itself. The discriminator is trained to correctly discriminate between original and generated samples from the generator. Both the discriminator and the generator participate in a minimax game, which is represented as a value function as $V(G,D)$, as Equations 4.22:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{4.22}$$

Here, the generator is trained to maximize $\log D(\boldsymbol{x})$ and minimize $\log(1 - D(G(\boldsymbol{z})))$ in a numerical and iterable fashion. This GAN approach generates synthetic poisoned data for the WDSs system. Later, these data are used to test if $HCAE$ is well generalized against data poisoning, in the experimental setup and execution.

4.4 Context-driven Short-Term Forecasting for WWTPs - cP$_2$O

In this section, I present the materials and methods for the proposed short-term forecasting model for WWTPs.

The objective of short-term forecasting in WWTPs is to predict key variables, such as inflow or water levels, over a short horizon, typically within 4–6 hours. The cP2O model performs forecasts based on past observations and external influencing factors. Specifically, I aim to forecast several hours into the future by predicting the sequence $yt + 1, y_{t+2}, \ldots, y_{t+H}$, using the historical observations $y_{t-M+1}, y_{t-M+2}, \ldots, y_t$ as input. Here, y_t denotes the system's state at time t, M is the length of the historical time series serving as the input data, and H represents the number of time steps in the forecast horizon.

4.4.1 Context Extraction and Forecasting Stages

As depicted in Figure 4.10, the proposed forecasting model comprises two interconnected stages: the **context extraction stage** and the **forecasting stage**. The context extraction

stage processes historical data from external sources—such as weather variables (rainfall, temperature), river data, and demographic or economic indicators—that provide additional context for the forecasting task. Additionally, I select a representative subset of WWTP data to incorporate context that captures historical patterns.

However, concatenating all context variables into a single high-dimensional input vector becomes computationally impractical. To address this challenge, my context model processes each context variable individually. For computational efficiency, multiple context variables are processed in parallel within a batch structure. At each time step, I flatten the outputs from the batch and generate a single context vector $\mathbf{r}_t$.

An optional modulation can be performed, yielding a general context vector $\mathbf{r}'_t$ for each time step; however, performance may decrease for high-dimensional datasets. The exogenous variables undergo preprocessing steps using a dynamic smoothing component, which includes normalization and deseasonalization.

The context extraction stage and the forecasting stage are synchronized in their time steps, ensuring that contextual information aligns with the internal data.

The forecasting stage processes the WWTP's internal sensor data, such as pump activity and inflow levels. Similar to the context stage, the internal data undergo dynamic smoothing for preprocessing. Before feeding the data into the dilated LSTM cells, the input is augmented by concatenating it with the context vector $\mathbf{r}_t$ from the context extraction stage. This integration enriches the forecasting model with external contextual information, enhancing its predictive capabilities.

When the number of exogenous series is relatively limited, I assign a weighting vector $\mathbf{h}$ to each series. This vector has the same length as the context vector $\mathbf{r}_t$ and is initially set to ones. The purpose of $\mathbf{h}$ is to adjust the general context information, customizing it to the

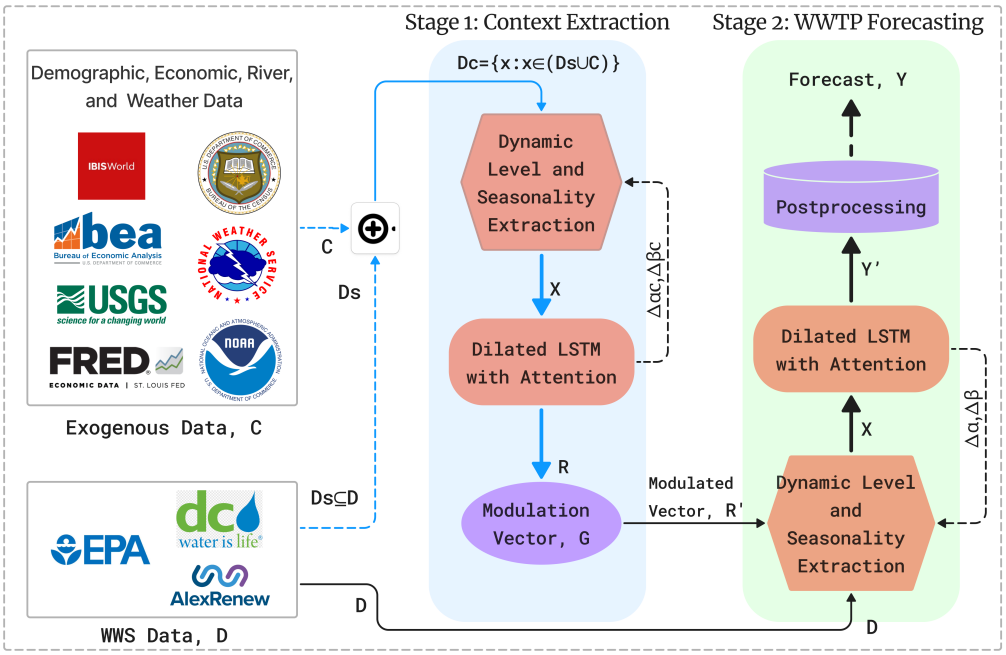


Figure 4.10: The diagram illustrates a two-stage forecasting framework where exogenous data and WWTP data are combined to enhance forecasting accuracy. In Stage 1 (Context Extraction Stage), relevant contextual information is extracted from external sources such as weather variables (e.g., rainfall, temperature), river data, and demographic or economic indicators. This stage involves preprocessing steps such as normalization and deseasonalization using a dynamic smoothing component to generate a dynamic context vector ($\mathbf{R}'$) for each time step. In Stage 2 (Forecasting Stage), this context vector is integrated with the WWTP's internal sensor data—such as pump activity and inflow levels—after similar preprocessing. The combined data is then fed into dilated LSTM cells for forecasting. Post-processing steps are applied to produce the final point forecasts and predictive intervals, providing both accurate predictions and uncertainty estimations.

specific needs of each series. Importantly, $\mathbf{h}$ remains constant and does not change across different time steps.

To minimize forecasting errors, I optimize all model parameters simultaneously without applying a separate loss function to the output of the context stage; instead, the entire model is trained in an end-to-end fashion. To manage computational resources and limitations in batch size, I use a batching mechanism that allows me to process many context series concurrently. However, practical constraints necessitate reducing the number of context

series. An effective approach is to leverage domain knowledge to select a representative subset of context series from the WWTP data, possibly by creating linear combinations of a few base series and incorporating relevant exogenous variables.

The final outcomes of the model are point forecasts and predictive intervals, providing accurate predictions along with uncertainty estimations essential for risk assessment and decision-making in WWTP operations. By leveraging external factors alongside internal WWTP data, and by efficiently integrating context information, the proposed model achieves greater forecasting accuracy compared to models that rely solely on internal data.

4.4.2 cP$_2$O Architecture

The architecture of the proposed forecasting solution, referred to as cP$_2$O, integrates data preprocessing and post-processing, dynamic pattern extraction, and dilated LSTM cells with an attention mechanism. Each time series variable—both from the WWTP and exogenous data sources—is decomposed by the dynamic smoothing component into its level (M_t) and seasonal (N_t) components.

As illustrated in Figure 4.10, time series data—including context variables C and utility data D—are processed to dynamically extract level, seasonality, and other patterns. This processing enables the forecasting stage to focus on more stable and normalized inputs. The prepared data undergoes normalization and the addition of calendar features (e.g., day of the week) to enhance forecasting accuracy.

The context vectors $\mathbf{r}_t$, computed in the context extraction stage, are concatenated with the plant data before being fed into the forecasting stage. The LSTM network, equipped with dilated cells and an internal attention mechanism, processes the combined data to capture temporal dependencies and leverage the learned context from the first stage.

In the post-processing step, the normalized forecast values are converted back to their original scale by applying inverse normalization and reintroducing the previously extracted patterns. The proposed model produces three key outputs:

1. Point forecasts of the target variable for multiple time steps ahead (e.g., 4–6 hours).
2. Prediction intervals (lower and upper bounds) for uncertainty estimation.
3. Adjustments to the smoothing parameters ($\Delta\gamma_t$ and $\Delta\delta_t$) to capture changes in level and seasonality over time.

Preprocessing and Input Pattern Generation

The preprocessing stage of my forecasting model serves two main purposes. First, it transforms the time series data into a format that is compatible with LSTM networks. Second, it generates input and output patterns that are compiled into training datasets for the model's training process.

Within both the context extraction and forecasting phases of the cP$_2$O architecture, I utilize dynamic smoothing based on the Holt-Winters method with multiplicative seasonality (Koehler et al. [333]). The essential equations for updating the level and seasonal components are:

$$M_t = \gamma_t \frac{X_t}{N_{t-m}} + (1-\gamma_t) M_{t-1} \tag{4.23}$$

$$N_t = \delta_t \frac{X_t}{M_t} + (1-\delta_t) N_{t-m} \tag{4.24}$$

Here, M_t denotes the level component, N_t represents the seasonal component, X_t is the

observed value at time t, P corresponds to the seasonal period (e.g., daily, weekly), and γ_t, $\delta_t \in [0, 1]$ are smoothing coefficients. These coefficients are dynamically modified by the LSTM network, which learns adjustments ($\Delta\gamma_t$ and $\Delta\delta_t$) during training to accommodate changes in the time series.

The preprocessing module reformats the time series for compatibility with the dilated LSTM. The main input for both context extraction and forecasting stages is the sequence immediately before the forecasted period. Let Ω_t^{in}, spanning 24 hours, represent the input window for the t-th sequence, and let Ω_t^{out}, spanning 4 hours, represent the output window. These windows are advanced by 4 hours to create subsequent input and output sequences. The input sequence undergoes deseasonalization, normalization, and a logarithmic transformation to mitigate the influence of outliers during the learning process.

The preprocessed input sequence is denoted by the vector $\mathbf{x}_t = [x_\tau]_{\tau \in \Omega_t^{\text{in}}} \in \mathbb{R}^{24}$. Deseasonalization removes weekly seasonal patterns, while normalization using the mean μ_t removes long-term trends within the input window. This scaling ensures that all preprocessed series are on a comparable scale, promoting effective cross-learning across multiple series.

To enrich the input data for the forecasting phase, I augment the input patterns with additional features, including the series' level and seasonality, calendar information, and the context vector from the context stage. The enhanced input vector is defined as in Equation (4.25):

$$\mathbf{x}'_t = [\mathbf{x}_t,\, \tilde{\mathbf{n}}_t,\, \log_{10}(\mu_t),\, \mathbf{c}_t^w,\, \mathbf{c}_t^m,\, \mathbf{c}_t^y,\, \mathbf{r}_t] \tag{4.25}$$

where:

- $\tilde{\mathbf{n}}_t \in \mathbb{R}^4$ is a vector of 4 seasonal components forecasted by the exponential smoothing (ES) model for $\tau \in \Omega_t^{\text{out}}$, adjusted for the forecast horizon.
- $\log_{10}(\mu_t)$ represents the local level of the series.
- $\mathbf{c}_t^w \in \{0,1\}^7$, $\mathbf{c}_t^m \in \{0,1\}^{31}$, and $\mathbf{c}_t^y \in \{0,1\}^{52}$ are one-hot encoded vectors indicating the day of the week, day of the month, and week of the year, respectively.
- $\mathbf{r}_t$ is the context vector derived from the context stage.

For the context extraction stage, an enriched input pattern follows a similar structure to $\mathbf{x}'_t$, but excludes the context vector $\mathbf{r}_t$.

The output pattern corresponds to the target sequence defined by Ω_t^{out}. I derive this pattern by normalizing the original sequence to ensure that the errors calculated by the loss function are consistent across different series. This normalization is shown in Equation (4.26):

$$y_\tau = \frac{X_\tau}{\mu_t}, \quad \text{where } \tau \in \Omega_t^{\text{out}} \tag{4.26}$$

Here, X_τ represents the original time series value at time τ. I train the model using patterns generated by shifting the input and output windows by 4 hours, which creates sequences for training. The LSTM in the main stage predicts a vector corresponding to the next 4-hour forecasts, as specified in Equation (4.27):

$$\hat{\mathbf{y}}_t^{\text{LSTM}} = [\hat{y}_\tau^{\text{LSTM}}]_{\tau \in \Omega_t^{\text{out}}} \in \mathbb{R}^4 \tag{4.27}$$

These predicted values are reverted to the original scale during post-processing, according

to Equation (4.28):

$$\hat{X}_\tau = \exp(\hat{y}_\tau^{\text{LSTM}})\, \mu_t\, \tilde{n}_{t,\tau}, \quad \text{where } \tau \in \Omega_t^{\text{out}} \tag{4.28}$$

The loss function utilizes the normalized predictions to maintain consistency, as shown in Equation (4.29):

$$\hat{y}_\tau = \frac{\hat{X}_\tau}{\mu_t} \tag{4.29}$$

Furthermore, the LSTM predicts two vectors representing the lower and upper bounds of the prediction intervals: $\underline{\hat{\mathbf{y}}}_t^{\text{LSTM}}$ and $\hat{\bar{\mathbf{y}}}_t^{\text{LSTM}}$. These are transformed into actual values using the same post-processing steps as the point forecasts.

The optimization algorithm uses the discrepancies between the predicted output patterns and the actual output patterns to adjust all model parameters, including those of the exponential smoothing components, the LSTM, and the adjustments $\Delta\gamma_t$ and $\Delta\delta_t$. Importantly, the context stage generates context vectors $\mathbf{r}_t$, which do not have target values; however, its parameters are updated in conjunction with those of the main stage to minimize the overall forecasting error.

Dilated LSTM with Attention Mechanism

In this subsection, I present the customized dilated LSTM cells designed to identify contextual events and seasonal patterns in time series data, including both exogenous and utility data. These cells, inspired by the concepts in Chang et al. [190] and Smyl et al. [334], are equipped with an internal attention mechanism to dynamically weigh input features.

Each dilated LSTM cell maintains two cell states (**c**-states) and two hidden states (**h**-states), all vectors in $\mathbb{R}^h$, where h is the dimension of the hidden state. Specifically:

1. Recent states: $\mathbf{c}^i_{t-1}$ and $\mathbf{h}^i_{t-1}$, which store information from the immediate past time step $t-1$, similar to a standard LSTM cell (Hochreiter and Schmidhuber [310]).

2. Delayed states: $\mathbf{c}^i_{t-d}$ and $\mathbf{h}^i_{t-d}$, which hold information from an earlier time step $t-d$ with $d > 1$. The dilation factor $d \in \mathbb{N}$ represents the number of time steps of delay and is crucial for capturing dependencies at different time scales. Incorporating delayed states effectively expands the receptive field and enhances the cell's ability to model long-term and seasonal patterns.

Gating Mechanisms: Inspired by both the LSTM and GRU architectures (Hochreiter and Schmidhuber [310], Cho et al. [335]), my cell incorporates two distinct gating mechanisms to manage the cell state $\mathbf{c}^i_t$:

1. Update Gate (u^i_t): Determines the extent to which the candidate cell state $\tilde{\mathbf{c}}^i_t$ contributes to the new cell state.

2. Forget Gate (f^i_t): Controls the influence of the recent cell state $\mathbf{c}^i_{t-1}$ on the new cell state.

The remaining influence is assigned to the delayed cell state $\mathbf{c}^i_{t-d}$, weighted by $\mathbf{1} - u^i_t - f^i_t$. This design leverages both recent and historical information, providing the cell with enhanced memory capabilities.

Cell Operations

The cell's operations at each time step t for layer i are defined as follows:

$$f_t^i = \sigma\left(\mathbf{W}_f^i \mathbf{x}_t^i + \mathbf{V}_f^i \mathbf{h}_{t-1}^i + \mathbf{U}_f^i \mathbf{h}_{t-d}^i + \mathbf{b}_f^i\right) \tag{4.30}$$

$$u_t^i = \sigma\left(\mathbf{W}_u^i \mathbf{x}_t^i + \mathbf{V}_u^i \mathbf{h}_{t-1}^i + \mathbf{U}_u^i \mathbf{h}_{t-d}^i + \mathbf{b}_u^i\right) \tag{4.31}$$

$$o_t^i = \sigma\left(\mathbf{W}_o^i \mathbf{x}_t^i + \mathbf{V}_o^i \mathbf{h}_{t-1}^i + \mathbf{U}_o^i \mathbf{h}_{t-d}^i + \mathbf{b}_o^i\right) \tag{4.32}$$

$$\tilde{\mathbf{c}}_t^i = \tanh\left(\mathbf{W}_c^i \mathbf{x}_t^i + \mathbf{V}_c^i \mathbf{h}_{t-1}^i + \mathbf{U}_c^i \mathbf{h}_{t-d}^i + \mathbf{b}_c^i\right) \tag{4.33}$$

Variable Definitions:

- $\mathbf{x}_t^i \in \mathbb{R}^n$: Input vector at time t for layer i.
- $\mathbf{h}_{t-1}^i, \mathbf{h}_{t-d}^i \in \mathbb{R}^h$: Hidden state vectors from recent and delayed time steps.
- $\mathbf{W}_*^i \in \mathbb{R}^{h \times n}$, $\mathbf{V}_*^i, \mathbf{U}_*^i \in \mathbb{R}^{h \times h}$: Weight matrices ($*$ denotes f, u, o, or c).
- $\mathbf{b}_*^i \in \mathbb{R}^h$: Bias vectors.
- $\sigma(\cdot)$: Sigmoid activation function, applied element-wise.
- $\tanh(\cdot)$: Hyperbolic tangent activation function, applied element-wise.
- $\otimes$: Element-wise multiplication.

Cell State Update: The cell state $\mathbf{c}_t^i \in \mathbb{R}^h$ is updated by combining the candidate cell state $\tilde{\mathbf{c}}_t^i$, the recent cell state $\mathbf{c}_{t-1}^i$, and the delayed cell state $\mathbf{c}_{t-d}^i$, weighted by the gating vectors:

$$\mathbf{c}_t^i = u_t^i \otimes \tilde{\mathbf{c}}_t^i + f_t^i \otimes \mathbf{c}_{t-1}^i + \left(\mathbf{1} - u_t^i - f_t^i\right) \otimes \mathbf{c}_{t-d}^i$$

- $\mathbf{1} \in \mathbb{R}^h$: Vector of ones.
- $u_t^i, f_t^i \in \mathbb{R}^h$: Gate vectors with elements in $[0, 1]$.

Constraints on Gates: To ensure proper weighting and stability:

$$0 \leq u_{t,k}^i, f_{t,k}^i \leq 1, \quad u_{t,k}^i + f_{t,k}^i \leq 1, \quad \forall k \in \{1, 2, \ldots, h\}$$

This ensures that the weights assigned to $\tilde{\mathbf{c}}_t^i$, $\mathbf{c}_{t-1}^i$, and $\mathbf{c}_{t-d}^i$ sum to at most 1 element-wise, and the remaining weight $(1 - u_{t,k}^i - f_{t,k}^i)$ is non-negative.

Hidden State Computation: The hidden state $\mathbf{h}_t^i$ is computed as:

$$\mathbf{h}_t^i = o_t^i \otimes \tanh(\mathbf{c}_t^i) \tag{4.34}$$

where $o_t^i \in \mathbb{R}^h$ is the output gate vector controlling the exposure of the cell state.

Attention Mechanism Integration: My model incorporates an internal attention mechanism to dynamically weigh input features. The input vectors for the two layers are defined as:

$$\mathbf{x}_t^1 = \mathbf{x}_t \tag{4.35}$$

$$\mathbf{x}_t^2 = \mathbf{x}_t \otimes \exp(\mathbf{m}_t) \tag{4.36}$$

- $\mathbf{x}_t \in \mathbb{R}^n$: Original input vector at time t.
- $\mathbf{m}_t \in \mathbb{R}^n$: Attention vector derived from the hidden state of the first layer.

Derivation of the Attention Vector: The attention vector $\mathbf{m}_t$ is obtained by partitioning the hidden state $\mathbf{h}_t^1$ of the first layer:

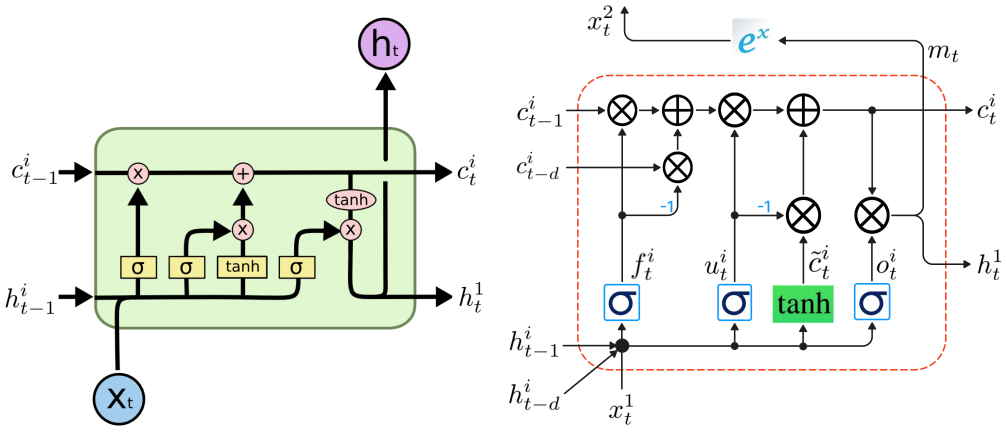
$$\mathbf{h}_t^1 = [\mathbf{h}_{t,\text{recurrent}}^1; \mathbf{m}_t] \tag{4.37}$$

- $\mathbf{h}_{t,\text{recurrent}}^1 \in \mathbb{R}^{s_h}$: Portion of the hidden state used for recurrent processing.
- $\mathbf{m}_t \in \mathbb{R}^n$: Attention vector used to modulate the input for the next layer.

After applying an exponential function to ensure positive weights, the attention vector modulates the inputs to the second layer. This mechanism allows the model to focus on the most relevant features at each time step.

Figure 4.11 illustrates the architecture of the dilated LSTM cell with the integrated attention mechanism.

Overall Network Architecture: Figure 4.12 depicts the overall architecture of the LSTM network, comprising three layers with dilation factors of 1, 2, and 4, respectively. By stacking



(a) Traditional LSTM Cell (b) Dilated LSTM Cell with attention

Figure 4.11: Cell architecture of cP$_2$O with dilated connections and attention mechanism

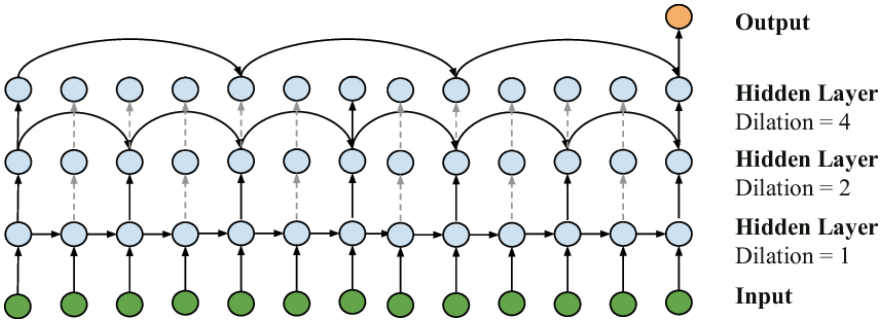


Figure 4.12: cP$_2$O architecture with dilated LSTM layers and attention mechanism (dashed line link is absent in the context stage)

layers with hierarchical dilations, the model captures features across multiple time scales, enhancing its ability to model seasonal and long-term patterns.

To prevent vanishing gradients when adding more layers, I integrate ResNet-style shortcut connections between layers (He et al. [336]). Additionally, the input vector $\mathbf{x}_t$ is supplied to all layers, enhancing the learning of complex patterns.

Embedding Layer: I employ a linear embedding layer to transform binary calendar vectors ($\mathbf{c}^w_t$, $\mathbf{c}^m_t$, and $\mathbf{c}^y_t$) into continuous vectors of dimension d, reducing dimensionality and capturing temporal features effectively. This embedding is learned during the training process.

Output Layer: The final linear output layer generates the model's outputs. In the main forecasting stage, these outputs include:

1. Point forecasts $\hat{\mathbf{y}}_t^{\text{LSTM}} \in \mathbb{R}^H$ for the forecast horizon H.
2. Lower and upper bounds for prediction intervals $\hat{\underline{\mathbf{y}}}_t^{\text{LSTM}}$ and $\hat{\bar{\mathbf{y}}}_t^{\text{LSTM}}$.
3. Adjustments to the smoothing coefficients $\Delta\gamma_t$ and $\Delta\delta_t$.

The complete output vector is:

$$\hat{\mathbf{y}}_t'^{\text{LSTM}} = \left[\hat{\mathbf{y}}_t^{\text{LSTM}}, \hat{\underline{\mathbf{y}}}_t^{\text{LSTM}}, \hat{\bar{\mathbf{y}}}_t^{\text{LSTM}}, \Delta\gamma_t, \Delta\delta_t\right] \tag{4.38}$$

In the context extraction stage, the output consists of: Context vector $\mathbf{r}_t^{(i)} \in \mathbb{R}^u$ for the i-th series and adjustments of $\Delta\gamma_t$ and $\Delta\delta_t$.

For a context batch containing K series, the context vectors are concatenated:

$$\mathbf{r}_t = \left[\mathbf{r}_t^{(1)}, \mathbf{r}_t^{(2)}, \ldots, \mathbf{r}_t^{(K)}\right] \in \mathbb{R}^{uK} \tag{4.39}$$

This combined context vector $\mathbf{r}_t$ is integrated into the input of the main forecasting stage, enriching it with contextual information.

4.4.3 Loss Function

Our model provides point forecasts for up to 4–6 hours ahead, along with the lower and upper bounds of the prediction intervals for each forecasted point. I apply the following loss function:

$$\mathcal{L}_\theta = \ell\left(y_\theta,\, \hat{y}_{p^*,\theta}\right) + \lambda\left[\ell\left(y_\theta,\, \hat{y}_{p_1,\theta}\right) + \ell\left(y_\theta,\, \hat{y}_{p_2,\theta}\right)\right]$$

where the pinball loss function $\ell(y,\, \hat{y}_p)$ is defined as:

$$\ell(y,\, \hat{y}_p) = \begin{cases} p(y - \hat{y}_p), & \text{if } y \geq \hat{y}_p \\ (p-1)(y - \hat{y}_p), & \text{if } y < \hat{y}_p \end{cases}$$

In these equations:

- y_θ represents the normalized observed value at time θ.
- $\hat{y}_{p,\theta}$ denotes the normalized predicted value at time θ for the p-th quantile.
- $p^* = 0.5$ corresponds to the median forecast (point forecast).
- p_1 and p_2 denote the lower and upper quantiles of the prediction interval, typically set to $p_1 = 0.05$ and $p_2 = 0.95$.
- $\lambda \geq 0$ is a weighting coefficient that determines the emphasis placed on the prediction interval components within the overall loss function.

The normalized observed value y_θ is obtained by:

$$y_\theta = \frac{X_\theta}{\mu_t}$$

where X_θ is the original time series value at time θ, and μ_t is the mean of the input window Ω_t^{in} as defined in the preprocessing stage.

The normalized predicted values $\hat{y}_{p,\theta}$ are derived from the LSTM outputs and adjusted during post-processing:

$$\hat{X}_{p,\theta} = \exp\left(\hat{y}_{p,\theta}^{\text{LSTM}}\right)\ \mu_t\, \tilde{n}_{t,\theta}$$

$$\hat{y}_{p,\theta} = \frac{\hat{X}_{p,\theta}}{\mu_t}$$

where, $\hat{y}_{p,\theta}^{\text{LSTM}}$ is the LSTM output for the p-th quantile at time θ, $\tilde{n}_{t,\theta}$ is the forecasted seasonal component from the exponential smoothing model, $\hat{X}_{p,\theta}$ is the predicted value in the original scale before normalization.

By operating on normalized values, the loss function ensures that errors have a consistent impact on the learning process across multiple time series with varying scales and error magnitudes.

This loss function consists of three components:

1. Point Forecast Loss: $\ell\left(y_\theta,\, \hat{y}_{p^*,\theta}\right)$- represents the loss associated with the point forecast. When $p^* = 0.5$, the pinball loss becomes symmetric and is equivalent to the mean absolute error (MAE).

2. Lower Prediction Interval Loss: $\ell\left(y_\theta,\, \hat{y}_{p_1,\theta}\right)$- represents the loss associated with the lower bound of the prediction interval. Encourages the lower quantile predictions to be below the observed values with a probability of p_1.

3. Upper Prediction Interval Loss: $\ell\left(y_\theta,\, \hat{y}_{p_2,\theta}\right)$- represents the loss associated with the upper bound of the prediction interval. Encourages the upper quantile predictions to be above the observed values with a probability of $1 - p_2$.

The pinball loss function is asymmetric, with the degree of asymmetry determined by the quantile levels p. This three-part structure allows for the simultaneous optimization of both point forecasts and prediction intervals, with the coefficient λ controlling the relative emphasis on each component. When $\lambda = 1$, all components are equally weighted; reducing λ places more focus on optimizing the point forecast.

Additionally, the pinball loss function helps mitigate forecast bias by penalizing positive and negative errors differently. By adjusting p^* to values less than or greater than 0.5, I can reduce tendencies toward positive or negative biases (Dudek et al. [337]). This approach can also be applied to adjust biases in the prediction intervals.

Chapter 5

Experimental Design

5.1 Model Agnostic Assurance Method

In this study, I design the experiments with both synthetic and real-world datasets. For ALSP, I use three datasets: Water distribution network, Pima Indian Diabetic, and Bank Loans. The water distribution network dataset is synthetic data Taormina and Galelli [338] that is generated using an emulator; it represents a sensor network within a hydraulic system. It also represents the Supervisory Control and Data Acquisition (SCADA) monitoring system (a commonplace dataset for simulations). I select this data to emphasize my study for assuring AI models in critical contexts. The remaining datasets are collected from Kaggle and University of California Irvine (UCI) Machine Learning Repository; Pima Indian Diabetic Dataset [1] and Bank Loan[2]. The Pima Indians Diabetes dataset comes from the National Institute of Diabetes and Digestive and Kidney Diseases. The dataset includes data from Pima Indian heritage female patients who are at least 21 years old and classifies if a patient is diabetic or not based on a specific diagnostic condition. I also collect Cellular Carrier data from the public website: Kaggle[3], where the data are provided by China Unicom (a mobile operator), the data contain 25 features and more than 1 Million instances.

[1] http://archive.ics.uci.edu/ml
[2] https://www.kaggle.com/zaurbegiev/my-dataset
[3] https://www.kaggle.com/pwang001/user-package-information-of-mobile-operators

5.1.1 Testing ALSP

In this experiment, I test if the algorithm provides accurate AIA scores for each sample of a given dataset using Weight Assessment. For Reverse Learning, I test if the actions of an AI model (GBDT) are logged for each epoch and illustrate how they provide XAI outcomes. Finally, for Secret Inversion, I test if the algorithm can detect adversarial inputs from the SCADA dataset.

Weight Assessment- Scoring AI System

For this experiment, I calculate Shapley values that help generate AIA scores for each sample of the dataset, namely: weights (W) of each feature contributing towards the final outcome of my AI model. These weights and the AIA in the diabetic dataset are multiplied to generate scores for each observation. I selected the Pima Indian Diabetic dataset (number of samples, N= 768 and number of features m = 8) for this study, where the label indicates if a patient is diabetic or not. Since there is a total of eight features in this dataset, a total of eight new AIACs are added that represent feature expectations. Feature expectations dictate the assurance goals for the AI system, therefore they must be designed based on the application requirements, I label binary values for each AIAC. Additionally, for this experiment, I select Extreme Gradient Boosting Decision Tree (XGBDT) for the AI model with the hyper-parameters set as follows: learning rate = 0.1, number of estimators = 1000, maximum depth = 5, minimum child weight = 1, gamma = 0, subsample = 0.8, colsample by tree = 0.8, objective = "binary:logistic", number of thread = 4, scale position weight = 1, and seed = 27.

The deployed AIACs represent TAI expectations. To test that, I injected bias and compared the scores with the unbiased dataset (Fig. 5.1). For injecting bias, I apply Gaussian noise

(mean = 0.3 and standard deviation = 0.1) and present the difference between the biased and unbiased datasets using Gaussian distribution. Fig. 5.2 represents the Gaussian score distribution difference for biased and unbiased datasets. It is evident from the Fig. 5.2 that intentional bias injection generates different AIA scores that help to explain relevant changes in the AI system.

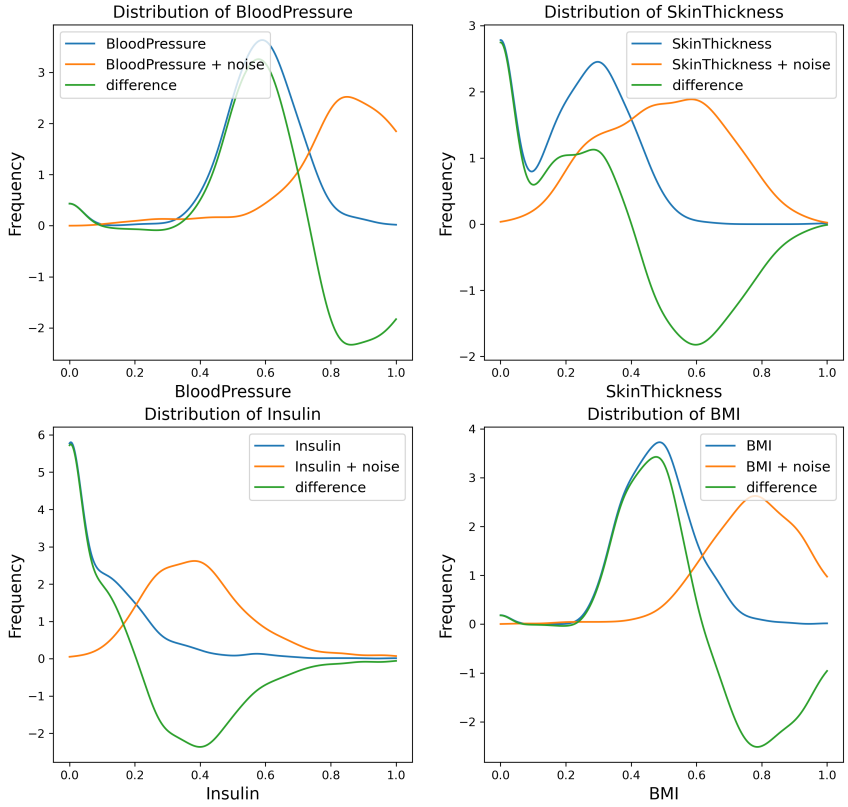


Figure 5.1: Distribution - normal and biased datasets

The scores are the expected outcome of the AI system; however, they do not necessarily mean anything until I deploy the AIACs properly in the dataset. I present that Weight Assessment generates AIA scores for every observation of the dataset. Additionally, after altering the dataset, I get a meaningful score representation of that alteration. It is evident from Fig. 5.2 that the score distribution changed when I injected *minimal* bias into the dataset, which was captured by the pipeline. Accordingly, this helps in indicating whether

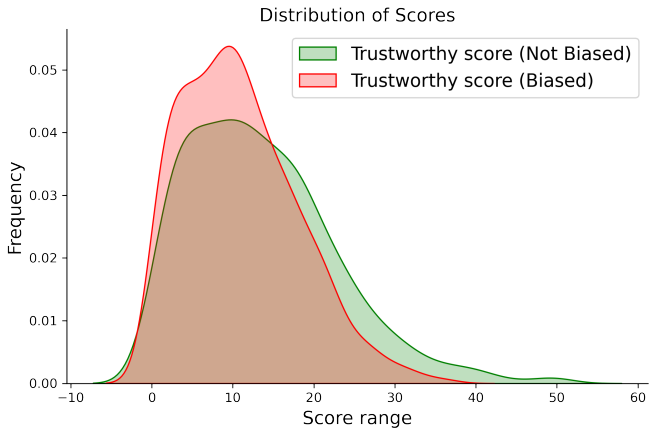


Figure 5.2: Distribution of TAI scores - normal and biased datasets

the outcomes are deemed more trustworthy or not.

Reverse Learning- Log and Optimize

In this experiment, I incorporate GBDT as my main AI algorithm. For GBDT, the primary target is to log each action while minimizing the loss function, therefore I design the AI algorithm as a white box (didnt use any off-the-shelf library). Reverse Learning doesn't provide any AIA score but dictates AIA goals such as FAI and EAI by visualizing and using exhaustive explanations (Explanations mean checking all the calculations and comparing them with the logs). This algorithm returns statistics on each epoch.

I select the Bank Loan dataset (number of samples, N= 614 and number of features $m = 12$) where the label indicates if a customer is likely to be accepted or not for a loan application. The features of this dataset are Gender, Married Dependents, Education, self-employed, Applicant Income, Co-applicant Income, Loan Amount, Loan Amount Term, Credit History, and Property Area. The GBDT model predicts the status of an applicant. Since my focus is minimizing loss function, I use default hyper-parameters including learning rate $l = 0.1$, max depth of the decision tree $d = 4$, and max-leaf nodes $nl = 7$. For the training stage,

Row	$p(t)$	$l(t)$	r	γ	$l(t+1)$	$p(t+1)$
1	0.74	1.06	-0.74	0.36	1.09	0.75
2	0.79	1.32	0.21	0.31	1.35	0.79
3	0.74	1.06	0.25	-1.58	0.90	0.71
4	0.71	0.91	0.28	0.63	0.97	0.72
5	0.74	1.04	0.26	0.63	1.10	0.75
6	0.74	1.06	0.25	-1.58	0.90	0.71
7	0.26	-1.02	-0.26	-0.84	-1.11	0.24
8	0.74	1.04	0.26	-1.58	0.88	0.70
9	0.64	0.59	-0.64	0.63	0.66	0.65
10	0.78	1.32	0.21	-1.58	1.16	0.76
11	0.74	1.04	0.26	-1.58	0.88	0.70
12	0.65	0.65	-0.65	-1.58	0.49	0.62
13	0.78	1.30	0.21	0.01	1.30	0.78
14	0.71	0.91	0.28	0.63	0.97	0.72
15	0.26	-1.02	-0.26	-0.84	-1.11	0.24

Table 5.1: Logs of GBDT algorithm learning cycle (Reverse Learning)

I perform 50 epochs for training the GBDT model. From Fig. 5.3, it's evident that the loss minimizes during the *13th epoch* (Fig. 5.4); the idea here is that all epochs post 13 are obsolete because the accuracy decreases afterward - something that wouldn't be traceable or explainable otherwise except by using an overly simple line chart. Table 5.1 presents the first 15 observations within the 13th epoch, where the rest of the observations (as well as all code and data used in this study) can be found in the MAA GitHub repository [4].

Secret Inversion- Detection of Adversarial Data Points

For the Secret Inversion algorithm, I design the experiment to present the successful detection of adversarial inputs in an AI system. Final outcomes are scores for AIA goals, including SAI and CAI (Table 5.2). For AE, detection performance varies with compression factor (cf) and the number of hidden layers (nl). Since I use labeled data, I select the best hyper-parameter (approximately I build 20 AE models) set for maximum accuracy. The outcome is

[4] https://github.com/AI-VTRC/MAA

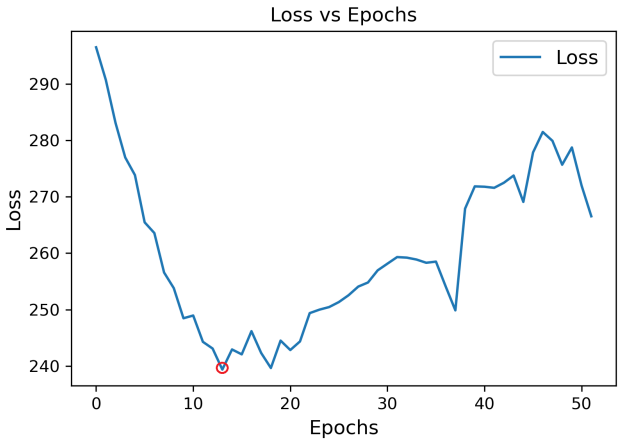


Figure 5.3: Loss function vs learning epochs

the status of the network, binary classification, which represents whether the overall system is under attack or not. Additionally, instead of *sigmoidal* function, I select rectified linear units due to their higher training performance in deep neural networks Goodfellow et al. [339]. Here, I use the SCADA dataset to test the secret inversion algorithm. Two-thirds of the training dataset is used for model development, and the other one-third is used for validation purposes.

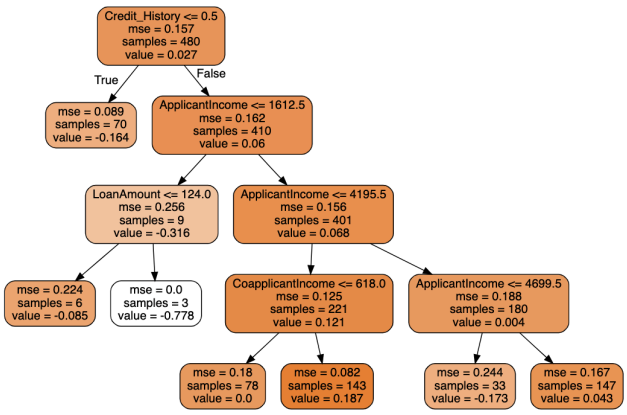


Figure 5.4: Decision tree 13 - minimum loss epoch

During validation, I am able to perform early stopping to prevent model overfitting scenarios. I use Adam algorithm Kingma and Ba [340] for my AE model, where I find the best trade-off between execution speed and convergence (number of training epochs $e = 200$ and mini-batch

Test Dataset	Accuracy	F1_Score	Precision	Recall
01	0.9466	0.7194	0.9438	0.5813
02	0.9128	0.7182	0.9707	0.5700

Table 5.2: Intrusion input detection performance using AE

size $b = 300$ samples). The mini-batch size is important because it updates the connection weights by propagating into the AE's network and computing the gradient. The average reconstruction errors, which are the mean squared errors between input and reconstructed patterns, are optimized using the SDG technique. Fig. 5.5 represents two different test datasets where scaled reconstruction errors are plotted on the 1st test dataset. By properly selecting threshold (θ), normal and adversarial samples for all 43 features have separated. For this experiment, I select the range between 99% and 100% percentile of the error distribution as adversarial inputs, where the rest of the ranges are considered normal samples. The detection accuracy of test datasets 1 and 2 are 94.66% and 91.28%, respectively. Table 5.2 presents the performance evaluation of the AE model.

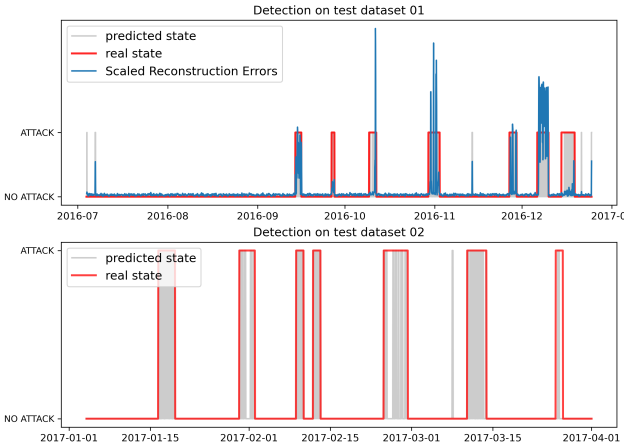


Figure 5.5: Intrusion detection using r on test datasets 1 and 2

5.2 DeepAg

All data are pre-processed before being fed into the baselines or LSTM models. To minimize bias, I employed data transformation techniques to normalize each of the input features using MinMaxScaler as represented in equation 5.1.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{5.1}$$

Equation 5.1 normalizes the values of the financial indices dataset, commodities dataset, and the outlier events dataset into a range of *0-1*. To prepare the datasets for anomaly detection, I then used *DoubleRollingAggregate* from the *ATDK Python* library to track the statistical behavior in a time series dataset. The DoubleRollingAggregate transformer rolls two sliding windows side-by-side along with a time series, aggregates using statistical mean, and tracks the difference of the aggregated metrics between the two windows. Figure 5.6 shows the changes (normalization) to the indices after applying *DoubleRollingAggregate* Transformer.

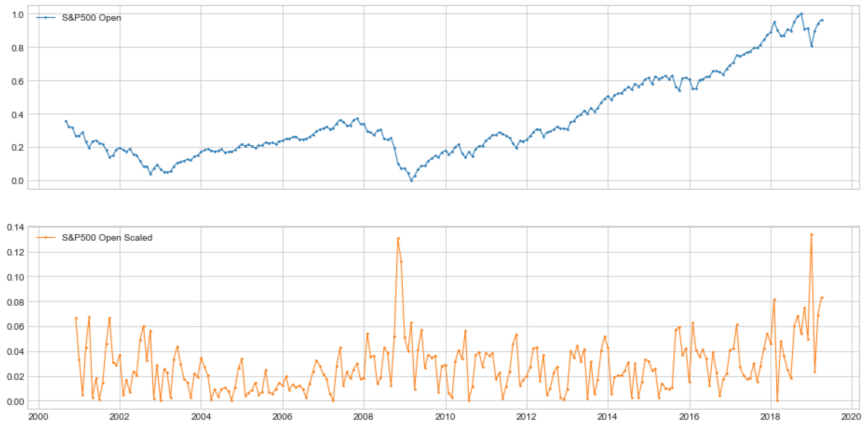


Figure 5.6: Original S&P 500 data from 2000-2019 (top) and scaled data (bottom)

5.3 DeepH$_2$O - Cyber Attack Detection in Water Systems

I pre-process the data before inputting them to the AE and the HCAE. The pre-processing step includes normalization and removing null samples from the data. For normalization, I perform standard minimum maximum scaling ranges from 0 to 1. Both AE (baseline) and $HCAE$ models are implemented using Scikit-learn API and are trained on a CPU with Intel core i5 10th gen. I use Adam Optimizer with learning rate = 0.0001, a decay factor of 0.5, and (β_1,β_2) = (0.9, 0.99). Additionally, 500 epochs are selected with a minibatch size of 32. The design of $HCAE$ differs from baseline AE in the design of the hidden layer definition. Early stopping is applied with patience = 3 for better regularization. Here, the patience parameter ensures convergence when the training loss and validation loss don't change for three consecutive epochs, and the training is marked complete. I am compressing input features using an under-complete autoencoder architecture, and both models' compression factor is selected as 2.5. Thus, I get the number of neurons in each layer as follows: encoder layers : [l_0, l_1, l_2] = [43, 34, 25]; bottleneck layer: [l_3] = [17]; and decoder layer as: [l_4, l_5, l_6] = [25, 34, 43].

Equations (4.17-4.20) represent AI assurance constraints, including Tide Weights, Orthogonal Weights, Uncorrelated Features, and Unit Norms constraints, which are applied to the AE. I pick a combination of these constraints and apply them to the hidden layers. My goal is to obtain a meaningful and uncorrelated latent representation, a prerequisite for dimensionality reduction. I empirically select optimal hyperparameters for the AE and the $HCAE$ models and maximize binary classification performance scores, including precision, recall, F1 score, accuracy, and specificity. Dataset 1 is used during model training, and Dataset 3 is used for model testing. Finally, I select threshold θ_{TH} by following an empirical approach.

I plot the F1 scores for baseline AE and *HCAE* models for Dataset 3 against a threshold range from 96% percentile to 100% percentile of their average reconstruction errors (Figure 5.7). I observe that both models reach a maximum F1 score at 98.5% percentile. Hence, I choose $\theta_{TH} = 0.985$ as the model's threshold.

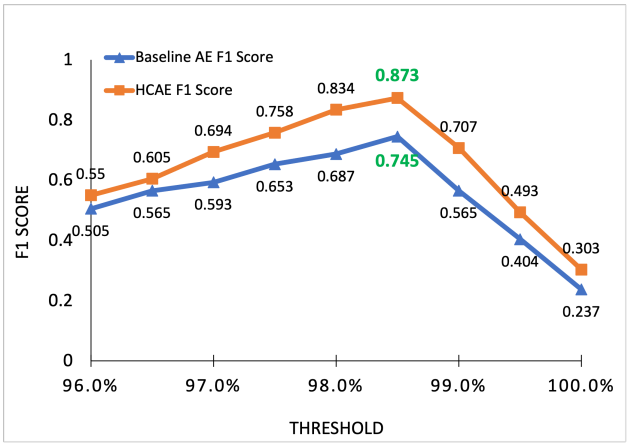


Figure 5.7: F1 Score obtained on Dataset 3 for different Thresholds θ

5.3.1 Model Performance Metrics

I use multiple performance metrics from the BATADAL competition to evaluate the model's ability to detect a threat in the shortest possible amount of time. In addition to this, I also use five additional metrics, namely accuracy, precision, recall, specificity, and F1-score, to measure the performance of a binary classifier.

Time-To-Detection Score: S_{TTD}

Time-to-detection is the difference between ground truth attack start time t_o (Equation 5.2) and algorithm detection start time t_d.

$$0 \leq \mathrm{S_{TTD}} = t_d - t_0 \leq \Delta t \tag{5.2}$$

The attack is indicated by Δt. A smaller TTD indicates that an algorithm has an improved detection performance during an ongoing attack. Additionally, the detection rate is associated with recall (%) or sensitivity which is otherwise referred as the True Positive Rate (TPR) and it is represented in Equation 5.3. Additionally, precision, what proportion of positive identifications was actually correct is represented in Equation 5.4.

$$\text{Sensitivity} = \text{Recall} = \text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5.3}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5.4}$$

Here, FN is the number of false negatives, and TP is the number of true positives. TPR is determined by the ratio of the correct attack classifications and the total number of attacks detected by the algorithm (including TP and FN). Additionally, I leverage True Negative Rate (TNR) or specificity metric to check false alarms by the models, and it is defined as (Equation 5.5),

$$\text{Specificity} = \text{TNR} = \frac{\text{TN}}{\text{FP} + \text{TN}} \tag{5.5}$$

TN is the number of True Negatives, and FP is the number of False Positives. TNR is determined by the ratio of the number of correct classifications for safe conditions (without attack) and the number of total classifications for safe conditions (including FP and TN).

Binary Classification Metric: F1-Score

Equations 5.3 and 5.4 are also known as recall and precision, respectively. In addition to accuracy and ranking, I calculate the F1-score using Equation (5.6) that accounts for both precision and recall,

$$F1\ Score = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.6}$$

Training a DL model with an imbalanced dataset and evaluating its performance using the accuracy metric can be misleading Branco et al. [341]. In such cases, an F1-score is preferred over accuracy as the F1-score represents a harmonic mean of precision and recall.

Classification Performance Score: S_{CLF}

To compare with the other state-of-the-art detection algorithms, Equations 5.3 and 5.5 are merged as classification performance score S_{CLF} (Equation 5.7), the mean of Equations 5.3 and 5.5.

$$S_{\mathrm{CLF}} = \frac{\mathrm{TPR} + \mathrm{TNR}}{2} \tag{5.7}$$

This score (S_{CLF}) represents detection as well as false-negative alarms. Additionally, this score is relevant to the F1 score, which is appropriate for problems with binary classification. The score can result in a 0 or 1 (where 1 indicates a perfect classification).

Ranking Score: S

Time-to-detection S_{TTD} and classification performance score S_{CLF} metrics can be merged further into a single ranking score as, Equation 5.8:

$$S = \gamma \cdot S_{\mathrm{TTD}} + (1 - \gamma) \cdot S_{\mathrm{CLF}} \tag{5.8}$$

According to the BATADAL competition, γ is set to 0.5 to ensure the weight of the early detection and the accuracy are equally adjusted Taormina et al. [8].

5.4 cP$_2$O Experimental Design

In this section, I assess the performance of our proposed model on Short-Term Water Level Forecasting tasks for WWTP. I present the dataset, the training and optimization methodologies, the baseline models used for comparison and our experimental results. The section concludes with an ablation study and a discussion of the findings.

5.4.1 WWTP Data

Blue Plains Advanced WWTP: DC Water

The Blue Plains Advanced WWTP at DC Water incorporates a sophisticated tunnel system designed to mitigate the overflow of stormwater and sewage during heavy rain events. Historically, the plant would reach its maximum capacity during such events, leading to untreated water being discharged directly into the river or causing widespread flooding in the city's sewer system. To address this issue, DC Water implemented an underground tunnel system that acts as a water retention mechanism. This tunnel captures excess stormwater and sewer overflows, temporarily storing them until the plant can process and treat the water post-rainfall.

The sewer system serving Washington, D.C., parts of Maryland, and Virginia connects to the tunnel system at multiple critical junctures. These connections are facilitated by Combined Sewer Overflow (CSO) structures (Botturi et al. [342]), which divert excess water from the sewer network into the tunnel system when sewer levels exceed certain thresholds. This preemptive diversion prevents overflow within the city's sewer infrastructure. Additionally, rain gauges and flow meters are strategically placed across the system to monitor water levels and trigger the diversion process when needed.

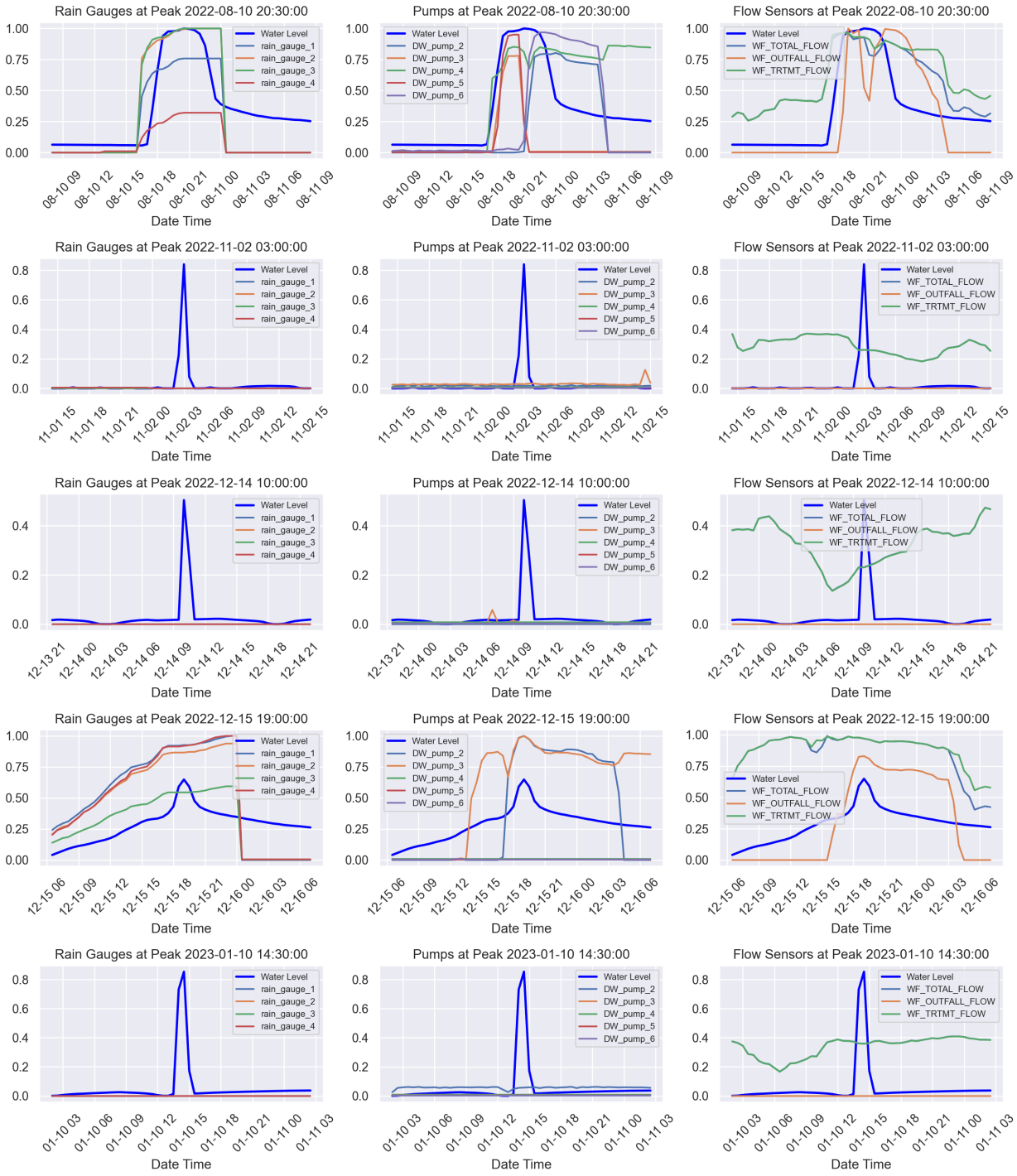


Figure 5.8: Identified peaks at DC Water in tunnel level with corresponding sensor data (rain Gauges, pumps, flow sensors) during critical events.

At the endpoint of the tunnel, a micro-treatment facility begins the treatment of the overflow water before it enters the larger WWTP. The plant's pumping system is key to managing inflows, with small pumps handling routine inflows and large pumps activated during peak events when the tunnel reaches capacity. Interestingly, energy efficiency plays a significant role in operational decisions. Small pumps, although slower, are more energy-efficient over prolonged use, while large pumps consume substantially more energy but can de-water the tunnel much faster. This trade-off requires utility operators to balance operational efficiency with energy costs, particularly in scenarios where heavy inflow can be predicted in advance.

To monitor tunnel water levels, DC Water employs various level indicators at multiple points along the tunnel. These indicators stage water levels and are used to guide operational decisions regarding pump activation. Depending on the event, operators may switch between different level indicators to ensure the most accurate measurements are used. Predictive models are employed to forecast water levels, providing operators with a 4-hour window to prepare pump operations. This predictive capability allows for the efficient scheduling of either small or large pumps, optimizing energy use and ensuring that the tunnel does not overflow.

Figure 5.8 shows a few identified peaks in tunnel water levels at DC Water during critical events, along with corresponding sensor data such as rain gauges, pump activity, and flow sensors. Several peaks in water levels were not associated with substantial rain gauge readings, pump activity, or flow sensor data. These peaks suggest the presence of events or anomalies not directly captured by DC Water's internal sensors, such as external factors like upstream river flow changes or unrecorded inflow sources. Analysis using the National Oceanic and Atmospheric Administration's (NOAA) (Center et al. [343]) storm events database indicates that these anomalies align with external events not directly captured by the plant's internal sensors. Following are the matched events from NOAA:

Table 5.3: Grouped tag names and descriptions for DC Water tunnel system and AlexRenew chemicals data

No.	Variable group	Data Description	Variable Count
		DC Water Tunnel System Data	
1	DIV_FLOW_CSO_X	Diversion Flow to tunnel from various CSO locations	7
2	DIV_FLOW_MPMP	Diversion Flow to tunnel from Main Pump Station	2
3	DIV_FLOW_JBAB	Diversion Flow to tunnel from JBAB	1
4	DIV_FLOW_POP	Diversion Flow to tunnel from Poplar Point	1
5	DIV_FLOW_MST	Diversion Flow to tunnel from M Street	1
6	TUNNEL_LEVEL	Tunnel Level measurements at various locations	7
7	OF_RVR_STCT	Overflow to River Structure	2
8	OF_BRL_CSO	Overflow to River Barrels (A, B, C) from various CSO locations	9
9	TIDE_GATE_LKG	Tide Gate Leakage (River level)	2
10	PLANT_FLOW	Plant influent and complete treatment flow	2
11	PLANT_OUTFALL	Plant outfall flow	1
12	TDP_FLOW	Flow at TDP (TDP-2 to TDP-6)	5
13	RAIN_GAUGE_DAY	Rain Gauge measurements at various pump stations	4
		AlexRenew Chemicals Data	
14	INFLUENT_EFFLUENT_FLOW	Total influent and effluent flows (MGD)	3
15	SOLID_TSS	Average dewatering central TSS (mg/L)	1
16	DISSOLVED_OXYGEN	Dissolved Oxygen (average, minimum, maximum) (mg/L)	3
17	AMMONIA_NH3	Ammonia (average, minimum, maximum) (mg/L)	3
18	NITRATE_NO3	Nitrate (average, minimum, maximum) (mg/L)	3
19	pH	pH levels (minimum, maximum)	2
20	TEMPERATURE	Water, air, and central (average, minimum, maximum) (F)	5
21	REACTOR_DECANT_FLOW	Reactor decant flow (GPD)	1
22	WAS_FLOW	Waste activated sludge flow (GPD)	1
23	PROCESS_AIR	Process air to CPT (scfm)	1
24	CARBON	Carbon transferred to CPT (gal)	1
25	PRECIPITATION	Precipitation (inches)	1

1. August 10, 2022 (See 1st-row plots of Figure 5.8): A flash flood event was triggered by a weak boundary overhead and anomalously high moisture levels, resulting in slow-moving thunderstorms. This caused significant water level rises in areas such as Rock Creek Parkway and Rhode Island Avenue NE. Although the plant's rain gauges recorded minimal rainfall, the flash flooding had a pronounced impact on water levels.

2. November 2, 2022 (See 2nd-row plots of Figure 5.8): Despite the absence of notable rainfall or pump activity, a water level peak was observed. This anomaly is likely attributable to upstream river inflows or unmonitored urban runoff entering the sewer system, which the plant sensors failed to detect.

3. December 15, 2022 (See 4th-row plots of Figure 5.8): A peak in water levels coincided with a heavy rainfall event that overwhelmed certain parts of the system. Although the plant sensors only partially captured the inflow, external factors such as rapid urban runoff likely played a significant role in the observed spike.

These events highlight the limitations of relying solely on internal plant sensors for forecasting. Incorporating contextual data—such as real-time weather data, upstream river flow rates, and external urban flood monitoring—can provide critical insights into such anomalies. By including this context, the cP$_2$O model can better identify whether these peaks are outliers or valid operational events driven by external conditions, enhancing decision-making and operational efficiency.

AlexRenew Chemicals Dataset

The AlexRenew WWTP, located in Alexandria, focuses on treating wastewater for its service areas. This real-world dataset provides a comprehensive range of parameters for wastewater treatment processes, which are highly relevant for short-term WWTP forecasting. The

dataset includes crucial water quality indicators and flow rates, such as total influent and effluent flow, dissolved oxygen (DO) levels, ammonia (NH_3) and nitrate (NO_3), pH levels, and temperatures across multiple reactors.

Table 5.3 summarizes the grouped tag names and descriptions for both the DC Water tunnel system and the AlexRenew chemicals data, providing an overview of the variables included in the datasets. In addition to these primary water quality attributes, external weather data has been incorporated into the AlexRenew dataset. This data, sourced from NOAA (Center et al. [343]) and the National Weather Service (NWS) (Weather [344]), includes parameters such as precipitation and atmospheric temperature. By merging these datasets, the study provides a more integrated view of how external conditions influence the water treatment process at AlexRenew.

The integration of the AlexRenew dataset with NOAA weather data provides a richer context for understanding the variability in wastewater treatment performance. By combining internal water quality parameters with external environmental factors, these datasets allow for better modeling of inflow levels during extreme weather events and provide deeper insights into how external conditions impact the treatment process.

The merged dataset offers a unique opportunity to apply predictive models for short-term wastewater inflow forecasting. The inclusion of weather-related data, such as rainfall and temperature, enhances the forecasting model's ability to capture the dynamic interactions between external conditions and WWTP performance. As a result, this holistic approach improves the accuracy of predictions, aiding in more efficient operational decisions for WWTPs.

5.4.2 Exogenous Contextual Data

Developing a comprehensive forecasting model for WWTPs necessitates the integration of multiple external data sources. These exogenous contextual variables provide valuable insights into factors that affect water quality and treatment processes. Below is a detailed breakdown of potential data sources and their relevant variables:

Weather Data

Sources: NOAA (Center et al. [343]), NWS (Weather [344]), Weather Underground (Galchen [345]).

Variables: Precipitation, temperature, humidity, wind speed, atmospheric pressure.

Weather conditions significantly influence both the volume and characteristics of wastewater inflow. For example, precipitation events can lead to increased inflow due to runoff and infiltration, while temperature and humidity affect evaporation rates and biological activity within the WWTP.

River Data (Water Quality and Flow)

Sources: United States Geological Survey (USGS) (Rabbitt [346]) and EPA (Bastian et al. [67])

Variables: River flow rates, water temperature, pH levels, dissolved oxygen, chemical contaminants, biological oxygen demand (BOD), nutrient levels (nitrogen, phosphorus).

Understanding river data is crucial for assessing natural water quality and evaluating the impact of effluents from the treatment plant on river ecosystems. Variables such as flow rates and water quality parameters help model the interactions between wastewater discharge and

the receiving water bodies.

Demographic Data

Sources: United States Census Bureau (Ratcliffe et al. [347]), IBIS World (Setar and MacFarland [348])
Variables: Population density, household size, urbanization rate.

Demographic factors affect both the volume and composition of wastewater generated. Higher population densities and urbanization rates typically result in increased wastewater production, while household size can influence per capita water patterns in the United States.

Economic Data

Sources: Bureau of Economic Analysis (BEA) (Budd and Radner [349]), Federal Reserve Economic Data (FRED) (McCracken and Ng [350])
Variables: Employment rates, industrial output, types of businesses.

Economic activity influences the types and quantities of industrial effluents entering the WWTP. Variables such as industrial output and business types help predict variations in wastewater characteristics due to industrial discharges.

5.4.3 Training, Optimization, and Evaluation Setup

The datasets from DC Water and AlexRenew encompass time series variables spanning from 2019 to 2023; however, many of these time series contain missing values within this period. For model development and hyperparameter tuning, I divide the data into training and

Algorithm 7 Training and Evaluating cP2O

Input: Training data $\mathcal{D}\text{train} \in \mathbb{R}^{n \times T}$, Testing data $\mathcal{D}\text{test} \in \mathbb{R}^{n \times T}$, Context data $\mathcal{D}\text{C} \in \mathbb{R}^{m \times T}$, Initial model parameters Θ, Batch size $\mathcal{B}$, Learning rate η, Max epochs $E_{\max}$
Output: Forecasts $\hat{\mathbf{u}} \in \mathbb{R}^{n \times T}$, Trained parameters Θ, Loss metrics $\mathcal{L}$
for $e = 1$ to $E_{\max}$ **do**
    Split training data into batches $\mathcal{B}_i \in \mathbb{R}^{b \times T}$
    for each batch $\mathcal{B}_i$ **do**
        Initialize LSTM_Θ and $\text{ContextLSTM}_\Theta$
        for $t = 1$ to T **do**
            Extract per series context including $\tilde{\mathbf{n}}_t$ and μ_t
            Forecast $\hat{\mathbf{u}}_t = f(\mathcal{B}_i, \mathcal{D}_{\text{C},i}, \Theta)$
        end for
        Compute loss $\mathcal{L} = ||\hat{\mathbf{u}} - \mathbf{u}||_2^2 + \lambda \cdot \text{reg}(\Theta)$
        Update parameters $\Theta \leftarrow \Theta - \eta \nabla_\Theta \mathcal{L}$
    end for
    if saveResults == True **then**
        Save parameters Θ and intermediate results
    end if
end for
return Final forecasts $\hat{\mathbf{u}}$, trained parameters Θ

validation sets, allocating 80% for training and 20% for validation. Hyperparameters were primarily selected to minimize the forecasting error on the validation set while ensuring near-zero forecast bias by adjusting q^*, as previously discussed in the loss function section. Each training epoch comprises n_o "sub-epochs," determined experimentally, with each sub-epoch covering a complete pass through all available data. Specifically, n_o was set to 10 for DC Water and 15 for AlexRenew.

I implement a training regimen that progressively increases batch sizes while simultaneously decreasing learning rates. Given the limited number of series, I begin with an initial batch size of 16, which is expanded to 64 starting at epoch 5. To further minimize the validation error, I employ a decaying learning rate schedule: 5×10^{-3} for epochs 1–5, 3×10^{-3} for epochs 6–7, 10^{-3} for epochs 8–9, and 10^{-4} from epoch 10 onward. I trained the model for a total of 50 epochs.

During each epoch, updates are conducted based on the average error accumulated over up to $T_b = 40$ forward steps, progressing one day at a time within each batch. The starting

point within each batch is selected randomly, which may result in fewer than 40 steps. Each batch contains a randomly chosen subset of b series.

The dimensions of the **c**-state and **h**-state were set to $s_z = 165$ and $s_h = 80$, respectively. These dimensions were determined through experimentation, beginning with $s_z = 100$ and $s_h = 50$, and incrementally increasing them to improve model performance. The output vector size O_v is calculated as the difference between the **c**-state and **h**-state sizes.

Our model architecture comprises three blocks, each containing one cell with dilation factors of 1, 2, and 4, as depicted in Figure 4.12. These dilation factors were selected experimentally to correspond with the seasonal patterns in the data. I apply the pinball loss function for quantile regression, using quantile values of $q^* = 0.62$, $q_1 = 0.039$, and $q_2 = 0.981$. The weighting coefficient in the loss function is set to $\lambda = 0.35$ to ensure that the average central loss during training is higher than the losses for the lower and upper intervals.

I employed embedding layers for time-related variables with dimensions set to 10, determined through experimentation. The context batch size C_b was configured to 20, encompassing 5 variables from each of four distinct context groups. I include all series without missing values to simplify the implementation and enable the processing of all context series within a single batch.

Finally, I apply an ensemble size of $E = 20$, however, as few as 5 ensemble members are sufficient. Ensemble forecasts were combined using a straightforward mean aggregation. The training was performed using the Adam optimizer. The model's weight and bias matrices were initialized with specific dimensions to accommodate the input and hidden state sizes. Details of these matrices are provided in Appendix B.

5.4.4 Baseline Models

To assess the performance of our proposed model, I compare it with a variety of baseline models spanning statistical methods, ML, and DL techniques. The Naive model predicts the water level profile for day i by replicating the profile from day $i - 7$. The ARIMA model (Box et al. [351]) employs an autoregressive integrated moving average methodology, while the **ES** model (Gardner [104]) utilizes exponential smoothing. Another baseline, Prophet (Taylor and Letham [352]), applies modular additive regression with seasonal components and nonlinear trends.

In the realm of ML and DL models, I include several approaches. The GRNN (Specht [353]) stands for General Regression Neural Network, and the SVM (Drucker et al. [107]) model uses Support Vector Machines for time series regression tasks. Among recurrent neural networks, I evaluate the LSTM (Hochreiter and Schmidhuber [310]), an LSTM network designed to capture temporal dependencies. The MTGNN (Wu et al. [354]) is a Graph Neural Network tailored for multivariate time series forecasting. I also examine the N-BEATS model (Oreshkin et al. [355]), a deep neural network with a hierarchical doubly residual architecture, and DeepAR (Salinas et al. [356]), an autoregressive recurrent neural network for probabilistic forecasting.

Additionally, I assess the WaveNet model (van den Oord et al. [357]), which employs dilated convolutions for autoregressive forecasting. The XGBoost model (Chen and Guestrin [106]) utilizes the eXtreme Gradient Boosting algorithm for regression tasks. Lastly, I include the $\mathbf{P_2O}$ model (Kulkarni et al. [3]), a recent multivariate multi-step attention-LSTM model used as a baseline in time series forecasting.

These baseline models were evaluated under the same experimental setup. Some models, such as ARIMA, Prophet, and XGBoost, struggle with handling the complex seasonality

and exogenous variables present in WWTP data. More advanced models such as LSTM, MTGNN, and N-BEATS perform better, but the inclusion of external context data in the proposed model offers significant advantages in predictive accuracy.

Chapter 6

Experimental Results

This chapter provides a detailed description of the experimental analyses and results for each project, organized by sections.

6.1 DeepAg Results

6.1.1 Baseline Models

Our experiment with the baseline models included default hyperparameters with no feature engineering. The results rank the overall performance of the models as follows: Linear Regression with Polynomial Features, Linear Regression, XGBoost Regressor, Random Forest Regressor, and Regression Tree. *Linear Regression with Polynomial Features* resulted in the most accurate and best-performing baseline model (as shown in Figure 6.1).

I use the R^2 score to determine the accuracy of the regression models as calculated by equation 6.1 and *RMSE* to evaluate the error as shown in equation 6.2 given our prediction, $\bar{y}$, and the actual value, y. The commodity production values typically range in the billions/year depending on the commodity.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y}_i)^2} \tag{6.1}$$

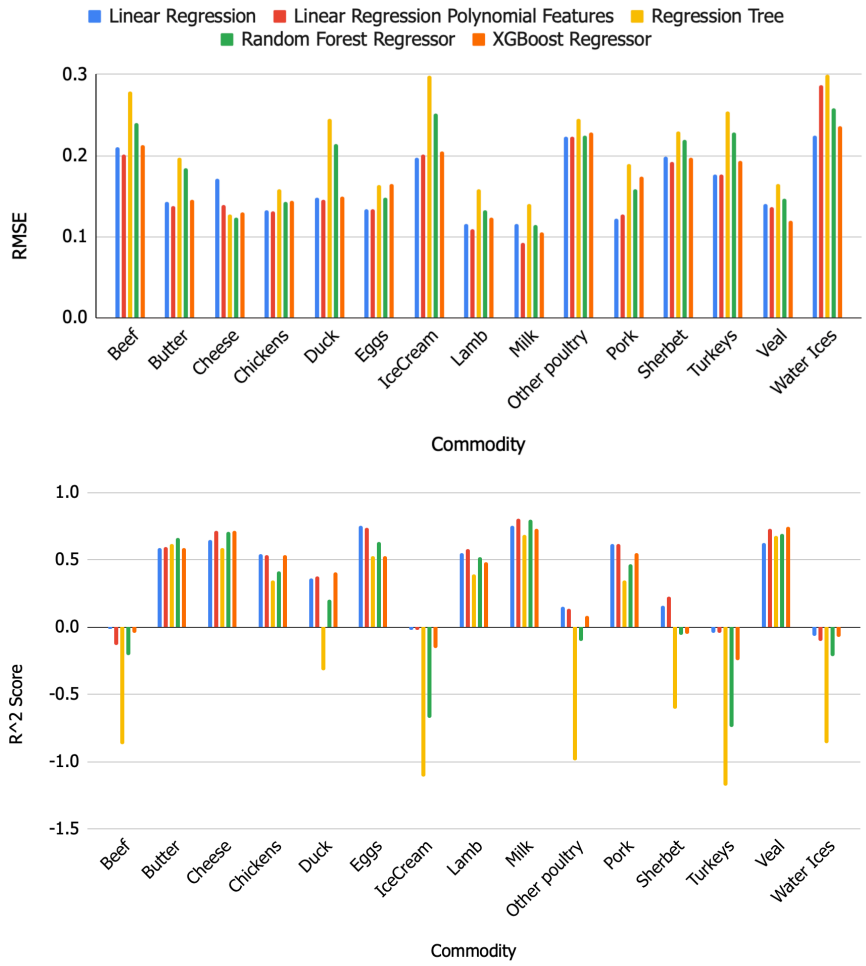


Figure 6.1: Baseline models Root Mean Squared Error ($RMSE$) and R^2 scores results

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \tag{6.2}$$

When comparing R^2 scores across models, the linear fit can be seen as unusually low for certain commodities such as Beef, Ice Cream, and Water Ices. This is most likely due to the variance in the data (i.e. variable production due to economic factors). In addition, USDA has more available historical production data for these commodities compared to others, and this may have also been a factor in affecting the fit.

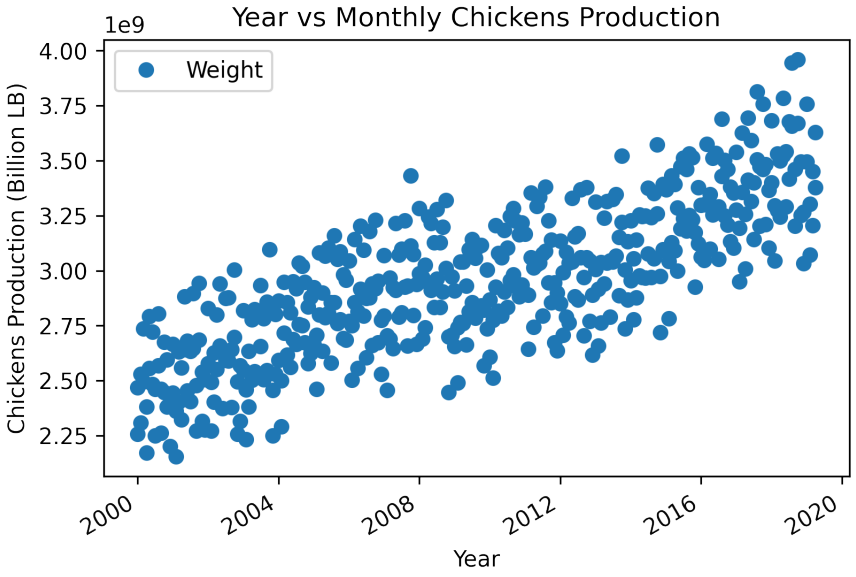


Figure 6.2: Historical chickens production

As shown in Figures 6.2 and 6.3, certain commodities, such as Beef, compared to Chickens, had significantly more variance in terms of their range. For instance, Chicken production has a clear upward trend with slight variance over time. In contrast, the Beef production range significantly varies with abrupt changes from year to year. Therefore, this is a factor that made a considerable difference in the linear fit R^2 score for the baselines. For the Linear Regression with the Polynomial Features model, Beef's R^2 score value was -0.1384 whereas Chicken was 0.5373, which had a much better linear fit for the data supporting our conclusion about data variance.

6.1.2 DeepAg Outcomes

Forecasting Commodities' Production

I employed the multi-step multivariate time series forecasting technique to accurately predict the commodity's production in the future. Multivariate time-series forecasting enables the

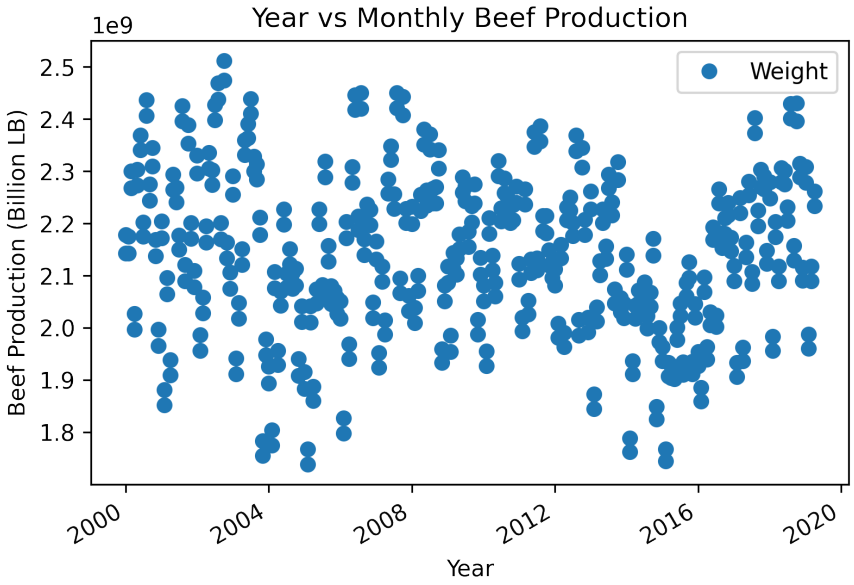


Figure 6.3: Historical beef production

prediction of a future dependent variable y, based on more than one independent variable x. Here, our dependent variable y is the commodity production, and independent variables x are the historical production data, highest causation, highest correlation index, and historical outliers. Lastly, multi-step forecasting tunes the model to predict a certain number of time-steps ahead instead of only predicting *one* future value. Our model had a look-back of *60* time-steps from the past to forecast approximately *30* multi-steps into the future. Based on the commodity, since every commodity has a different number of data points, our model was able to predict approximately *five years* ahead.

Table 6.1 indicates the results obtained using our DeepAg approach. For each commodity production, there is a prediction value and an RMSE score. These are generated *with* and *without* the outliers model for a total of 4 data points per commodity. The prediction value is the last data point from the commodity forecasting, and the *RMSE* score is mathematically calculated using Equation 6.2. Each prediction is paired with an *RMSE* value to measure the error rate and also to evaluate against the baselines. For each commodity, the most relevant

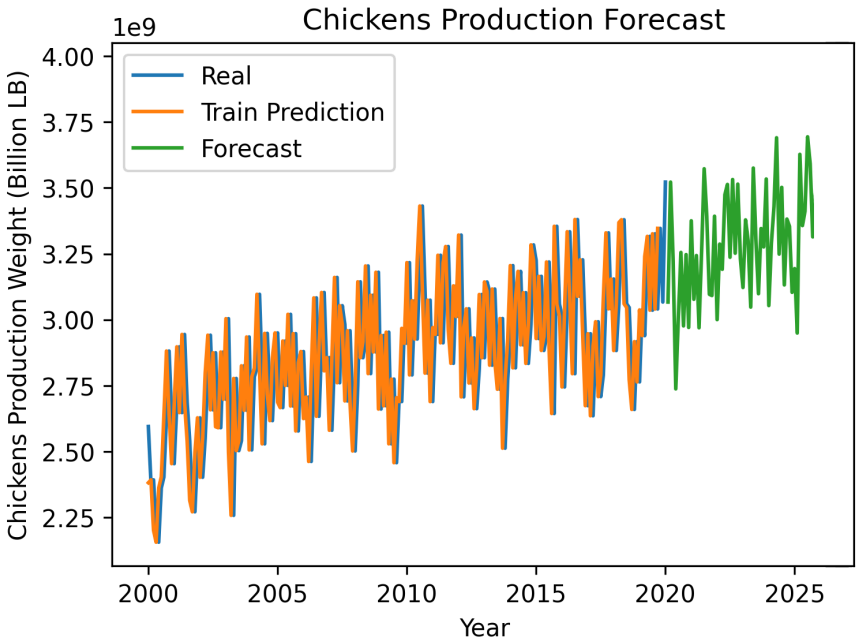


Figure 6.4: Chickens production forecast 2020-2025

indices are noted as *(Highest Causation, Highest Correlation)* and *(Highest Causation)* if they are the same. It follows as Beef *(DOW)*, Butter *(Gold)*, Cheese *(Gold)*, Chickens *(DOW)*, Duck *(S&P 500)*, Eggs *(DOW)*, IceCream *(S&P 500, DOW)*, Lamb and mutton *(Gold, DOW)*, Milk *(S&P 500, Gold)*, Other poultry *(Oil, VIX)*, Pork *(DOW)*, Sherbet *(S&P 500, DOW)*, Turkeys *(DOW)*, Veal *(DOW)*, Water Ices *(VIX, DOW)*.

6.1.3 The Effect of Outlier Detection

The Figures 6.4 and 6.5 illustrate the overall prediction fit and forecast for two of the commodities, Chickens and Beef. It's evident that the model had a close train prediction compared to the real values and was able to accurately fit the data in addition to the forecasting. From Table 6.1, the RMSE values demonstrate the high prediction accuracy of our DeepAg approach compared to the baselines. The lower the RMSE value, the better our approach is able to fit the production data and predict accurately. For instance, the RMSE

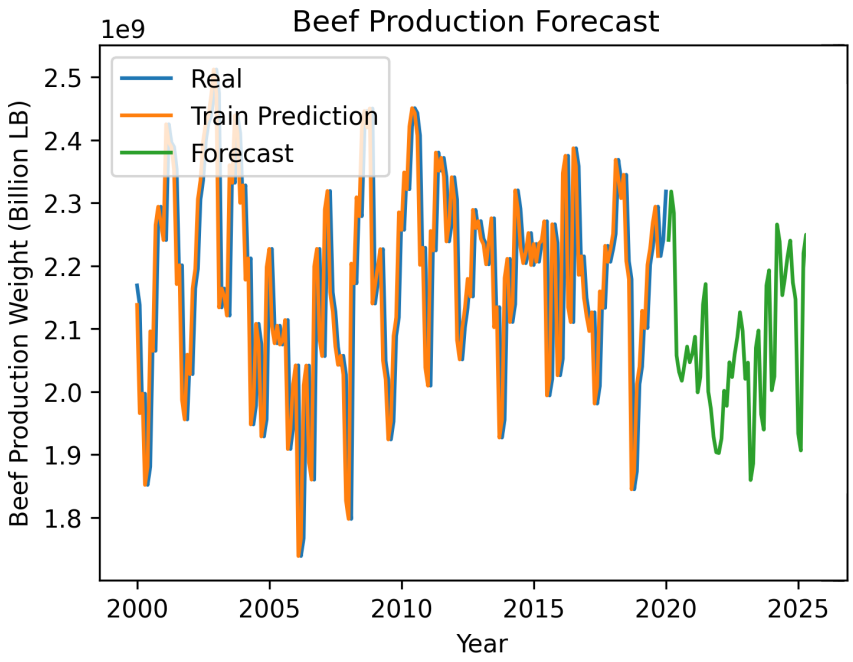


Figure 6.5: Beef production forecast 2020-2025

for Beef production was 0.164, 0.120, and 0.201 using DeepAg with outliers, DeepAg without outliers, and with Linear Regression with Polynomial Features baseline model, respectively, indicating that DeepAg with isolation forest outperforms DeepAg without outliers and the baselines. Out of the 15 commodities, 12 commodities had a lower RMSE with DeepAg - presented in Table 6.2. For the remaining three commodities, baseline RMSE was lower, and this may be because those are consistent commodities (Chickens, Eggs, Milk) and tend to have a clear upward trend that's better predicted by linear regression models.

Table 6.2 supports that outlier events are useful and contribute to higher accuracy when predicting commodities' production. Outlier events are the cause of sudden upward or downward spikes in economics, and in our work, I have captured that ambiguity through a more formal process. The use of outliers as one of the input features allows the LSTM model to learn and optimize for these extreme events.

Commodity	RMSE With Outliers	Prediction With Outliers	RMSE W/O Outliers	Prediction W/O Outliers
Beef	0.164	2060199936	0.12	1960800000
Butter	0.001	164524992	0.009	164524992
Cheese	0.004	991345984	0.058	991345984
Chickens	0.258	3531831040	0.197	3302286080
Duck	0.001	6348000	0.001	11891000
Eggs	0.708	8599600128	0.501	8599600128
IceCream	0	12010000	0.004	82232000
Lamband mutton	0.001	12000000	0.001	10100000
Milk	1	18872999936	1	16984999936
Otherpoultry	0	142000	0	100000
Pork	0.158	2022099968	0.118	2157299968
Sherbet	0	3541000	0	3541000
Turkeys	0.024	523214016	0.028	458169984
Veal	0.001	6700000	0	6000000
Water Ices	0	4647000	0	4647000

Table 6.1: DeepAg LSTM model results

6.2 P_2O Experimental Results

This section presents the results of the prediction implemented in P_2O. The prediction module uses preprocessed data for the experiments, with 42 columns i.e., sensors and 367,943 rows. The results of the prediction module are presented in this section, which focuses on comparisons and selecting an AI model for wastewater prediction.

6.2.1 Results: Prediction Module

This section presents the prediction module's results by providing details on summary statistics, visualizations, hyperparameter tuning, and model performance.

Commodity	Linear Regression with Polynomial Features RMSE	DeepAg With Isolation Forest RMSE	DeepAg Without Isolation Forest RMSE
Beef	0.201	0.164	**0.120**
Butter	0.138	**0.001**	0.009
Cheese	0.139	**0.004**	0.058
Chickens	**0.131**	0.258	0.197
Duck	0.146	**0.001**	**0.001**
Eggs	**0.134**	0.708	0.501
IceCream	0.201	**0.000**	0.004
Lamband mutton	0.109	**0.001**	**0.001**
Milk	**0.093**	1.000	1.000
Otherpoultry	0.224	**0.000**	**0.000**
Pork	0.127	0.158	**0.118**
Sherbet	0.192	**0.000**	**0.000**
Turkeys	0.177	**0.024**	0.028
Veal	0.137	0.001	**0.000**
Water Ices	0.287	**0.000**	**0.000**

Table 6.2: Comparison of best baseline model and DeepAg

Summary Statistics and Visual Inspection

The summary statistics: minimum, maximum, median, interquartile range, mean, and standard deviation are calculated for all the sensors. For the tunnel water level depth sensor (output), the observation ranges from -121.21 (0 is sea level; negative values indicate below sea level) to 15.76, while the mean and standard deviation values are -114.125 and 10.413, respectively. Further, sensor observations are also visualized to check the patterns, as shown in Figure 6.6. Based on the visual inspection, it is easy to identify that most values are negative (367,058) while very few are positive (851). Based on the information from one of the WWTP's Process Engineers, it was noted that the overflow from the tunnel occurs when the wastewater level observation reaches 3. Thus, based on the EDA, it can be seen that in the last four years (2018 - 2022), there have been 94 incidences at the WWTP when the wastewater overflowed from the tunnel.

Next, Pearson's correlation coefficients are calculated to investigate the relationship between the wastewater level depth sensor and other sensors. The coefficients indicated that the

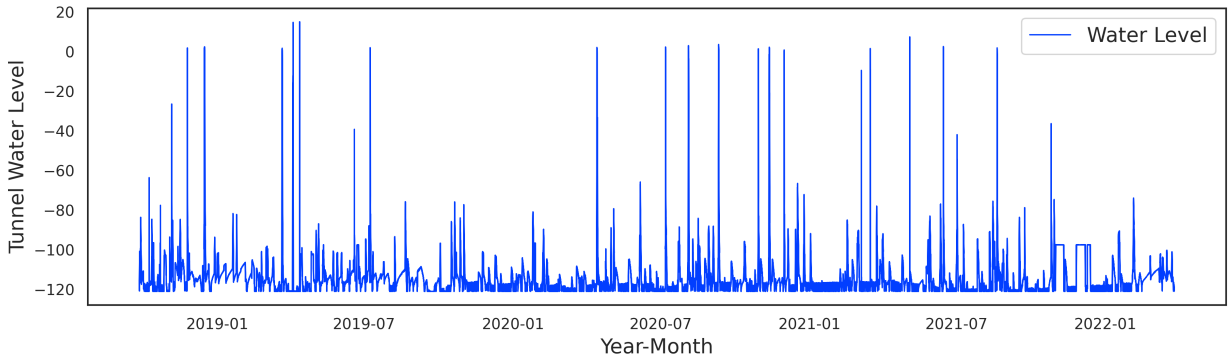


Figure 6.6: Wastewater level sensor observations from 2018 to 2022.

outflow sensor showed the highest (0.57), and rain gauges showed the second highest (0.56) significant positive correlations with the wastewater level depth sensor. Overall, most of the variables (24) showed weak positive correlations (between 0 and 3), while some (15) variables indicated a moderate positive correlation (between 0.3 and 0.7). The summary statistics show that the minimum and maximum values for the rain gauge and wastewater outflow sensor were 0, 5.01, and 0, 221.95, respectively. The time-series plots of these three variables - wastewater level, rain gauge, and wastewater outflow – are shown in Figure 6.7. Based on the correlations, the VIF analysis indicated the multi-collinearity between all the rain gauges and some sensors measuring the other critical main plant flows. Thus, three sensors from the rain gauge and four sensors from the other critical main plant flows are removed to eliminate multicollinearity. After performing VIF analysis, the final version of the data included 35 sensors (output sensor and time axis), and the same seven sensors were also removed from the data derived from downsampling.

Further, PCA is performed on the pre-processed data with 41 sensors (without the dependent variable), and a Scree plot is used to visualize the explained variance ratio for every Principal Component (PC). It was observed that the first PC contributes the most (about 20%), and then there is a gradual increase in the explained variance after PC 10. Thus, a threshold of 70% of cumulative explained variance was set after visual inspection, and 15 PCs were

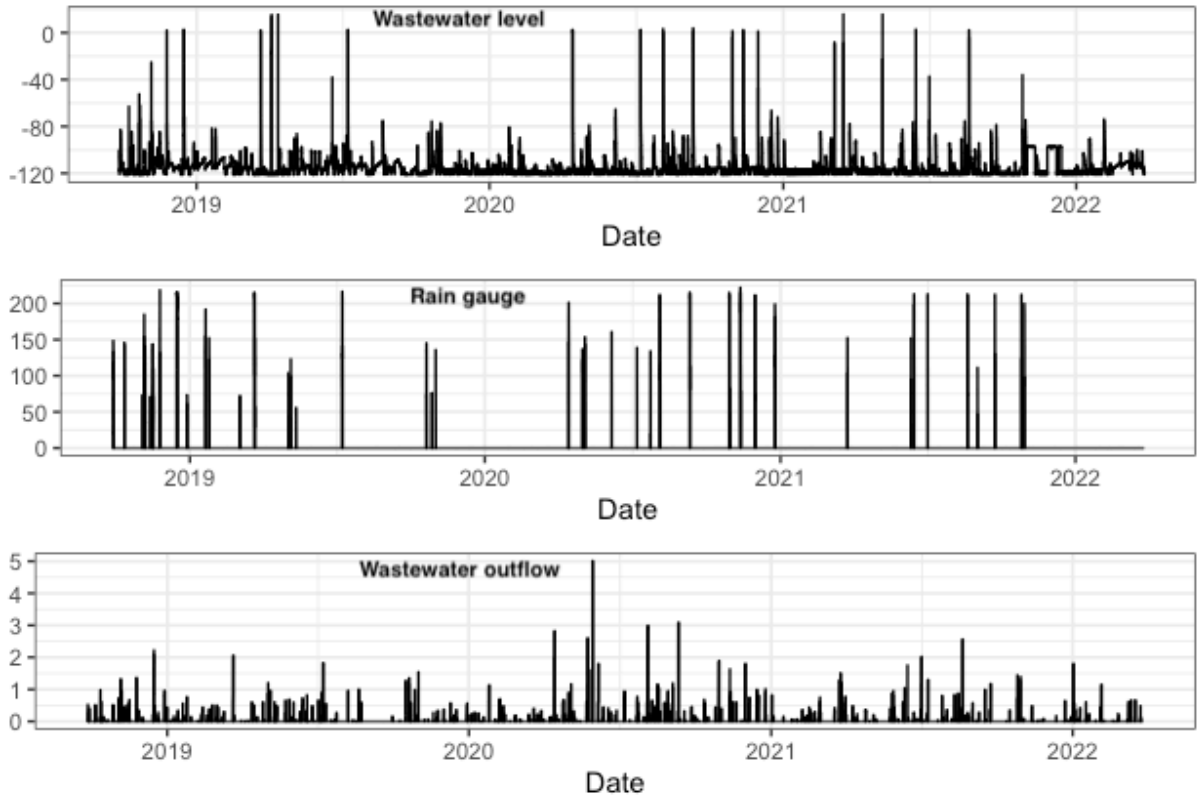


Figure 6.7: Visual patterns of wastewater level, rain gauge, and wastewater outflow.

selected for the model development.

Hyperparameter Tuning for ML and DL Models

The hyperparameter selection for RF, XGBoost, and LightGBM is performed for three versions of the preprocessed data. The details of the selected hypermeters for each version of the data are provided in Table 6.3. This hyper-parameter tuning procedure resulted in 10800, 17280, and 86400 model fits to find the optimal combinations for the RF, XGBoost, and LightGBM models, respectively.

For the FF-ANN, a random search algorithm was executed for five trials, and five different model configurations were tested in each trial. Thus, a total of 100 FF-ANN models are

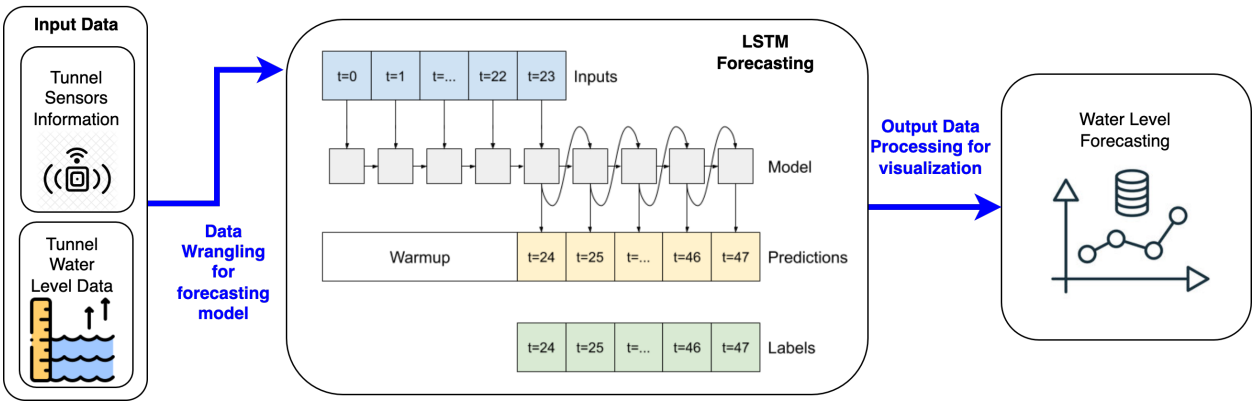


Figure 6.8: The architecture of LSTM used for wastewater level predictions.

developed and evaluated based on Mean Absolute Error (MAE) to find the optimal configuration of hyperparameters. During the training process, the batch size was set to 500; epochs were set to 200, and 20% of the data from the training set was used as a validation set. Further, the loss was set to Mean Absolute Error (MAE) during the training phase. The optimal hyperparameters obtained from the experiments are provided in Table 6.4. A random search algorithm was executed for tuning hyperparameters in the development of a multivariate LSTM model. The LSTM model is trained for 500 epochs with a batch size of 8. Further, a cubic loss function is used as an objective function for minimization in this process. The results indicated that the LSTM model with a configuration of 512 neurons with one hidden layer and a learning rate of 0.001 performed the best. The architecture of LSTM used for predicting wastewater level is shown in Figure 6.8.

Model Comparison based on RMSE, RSR, NSE, and R^2

The RF, LightGBM, XGBoost, and FF-ANN results are presented in Table 6.5. It can be observed that the RF model performs better with the downsampled data compared to other versions. The same pattern can also be observed for the LightGBM model, but XGBoost and FF-ANN models perform better with all the columns. For the LSTM models, three

Table 6.3: Tuned hyperparameters for RF, XGBoost, and LightGBM.

Hyperparamter	All data	Downsampled data	PCA processed data
RF Model			
n_estimator	200	300	100
max_depth	10	10	5
max_features	sqrt	sqrt	auto
min_samples_split	7	2	3
min_samples_leaf	1	1	5
XGBoost Model			
eta	0.05	0.2	0.3
n_estimator	100	50	20
max_depth	4	2	2
colsample_bytree	0.5	0.5	1
alpha	1	1	0.5
LightGBM Model			
learning_rate	0.1	0.1	0.2
num_leaves	50	50	100
num_iterations	200	300	10
max_depth	2	2	8
bagging_fraction	0.5	0.5	0.5
lambda_l1	0.5	0.5	1

Table 6.4: Details on the optimal hyperparameters obtained using the random search algorithm for the FF-ANN model.

Data	# of hidden layers	# of neurons	Activation functions	Learning rate	Execution time
All (96,801)	2	480 160	linear tanh	0.0001	4:23:31
Downsampled (68,417)	4	32 160 320 32	relu tanh tanh relu	0.001	3:56:30
PCA (56,033)	4	352 128 32 32	relu linear relu relu	0.0001	7:10:23

Table 6.5: Comparison of RF, LightGBM, XGBoost, and FF-ANN using RMSE, RSR, NSE, and R^2.

Model	Data	RMSE	RSR	NSE	R^2
Random Forest	All	7.628	0.784	0.385	0.41
	Downsampled	7.577	0.774	0.395	0.43
	PCA	7.857	0.807	0.347	0.36
LightGBM	All	7.548	0.775	0.398	0.42
	Downsampled	7.515	0.771	0.405	0.42
	PCA	7.824	0.771	0.405	0.01
XGBoost	All	7.450	0.765	0.413	0.40
	Downsampled	7.615	0.781	0.389	0.39
	PCA	7.984	0.820	0.326	0.37
FF-ANN	All	8.195	0.842	0.290	0.33
	Downsampled	8.228	0.844	0.287	0.29

input sequences – 12, 24, 30 – and four output sequences – 2, 4, 6, 8 – are evaluated. The results for these configurations are shown in Figure 6.9. The important results noted from the experiments are as follows:

- After comparing four models, the least RMSE (7.515) and RSR (0.771) values are noted for the LightGBM model with downsampled data.
- The LightGBM model with downsampled data indicates the highest NSE (0.413) compared to other models.
- For the test set, for the 30-hour input sequence, the NSE values are negative for all the output sequence hours.
- The LSTM model with a 12-hour input sequence and 2-hour output sequence indicates the lowest RMSE (0.036), RSR (0.276), and highest NSE (0.723) values.
- The 24-hour input sequence and 2-hour output sequence indicate the lowest RMSE (0.036), RSR (0.260), and highest NSE (0.739) values for this configuration.

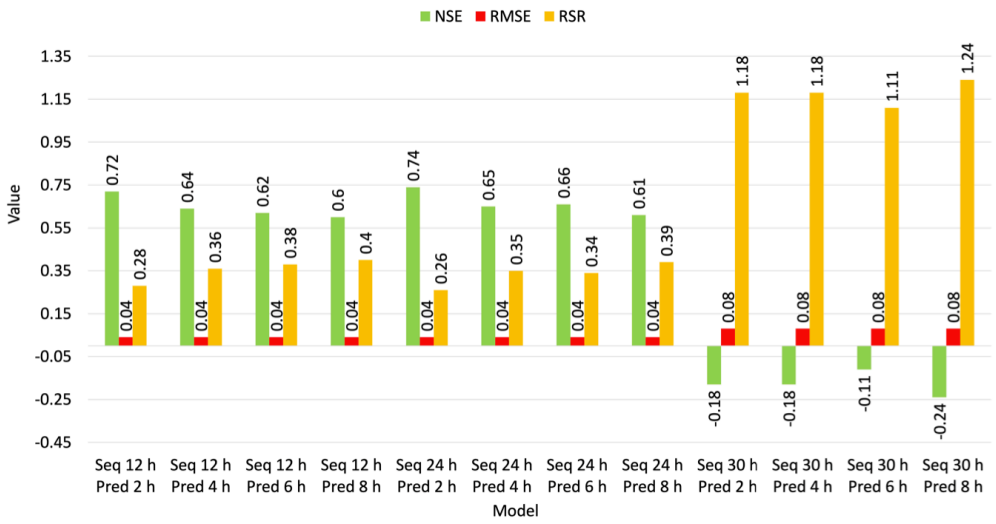


Figure 6.9: The LSTM model with a 24-hour input sequence and a 2-hour output sequence shows the best performance on the test dataset.

- Overall, it can be noted that the LSTM model with a 24-hour input sequence and a 2-hour output sequence manifests the best performance.

In the WWTP, the sea surface level (equal to 0) is used as a reference to measure the wastewater level. The stored wastewater is collected in the underground tunnels below the sea surface level (less than 0, which makes it negative). Considering this, a threshold is selected to provide soft warning predictions to check the model's performance. For a soft warning, a threshold of $-50m$ (50 meter down the sea level; total tunnel depth is $120m$ below sea level) is selected for the potential effluent overflow. Based on this threshold, the selected LSTM model correctly predicted 85% incidence of overflow in the test dataset. The results for the overflow threshold are visualized and shown in Figure 6.10.

6.3 DeepH$_2$O Cyber Attacks Detection Results

This section presents the results of the three research questions presented prior (RQ1 - RQ3).

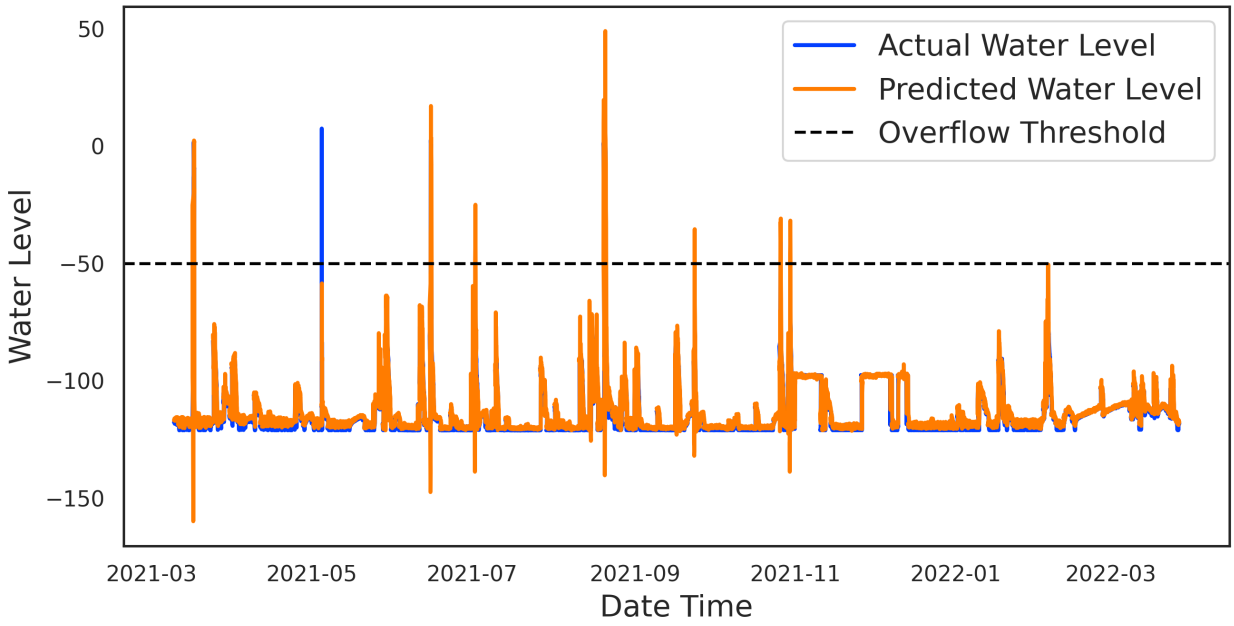


Figure 6.10: The LSTM model (24 hours input sequence and 2 hours output) prediction on test dataset with $-50m$ (85% accuracy at $50m$ below sea level) as the peak threshold.

6.3.1 RQ1: AI Assurance

Supervised Detection Results

Table 6.6 presents the attack detection performance of both supervised and unsupervised models. Among the two supervised models, results suggest that the TGCN with attention model performs better in attack detection in WDS. With the introduction of Attention and RMD assurance methods, the TGCN with attention model results in a significant improvement in recall and F1 score performance metrics. Out of the five metrics presented in Table 6.6, I observe an improvement in precision, recall, F1 score, and accuracy by 7.6%, 22.1%, 15.5%, and 4.7%, respectively. Precision, recall, and specificity metrics improve from baseline TGCN to TGCN with attention.

In terms of detecting attacks in WDS, results indicate that both the TGCN model (baseline)

Performance Metrics	**Supervised Model**			**Unsupervised Model**		
(Dataset 3)	Tsiami	TGCN	TGCN with Attention	Taormina	AE	*HCAE*
Precision	0.843	0.645	**0.721**	0.881	0.882	**0.972**
Recall	0.906	0.553	**0.774**	0.602	0.604	**0.865**
F1 Score	0.873	0.591	**0.746**	0.715	0.745	**0.873**
Accuracy	N/A	0.850	**0.897**	N/A	0.919	**0.951**
Specificity	N/A	0.922	**0.927**	N/A	0.972	**0.983**

Table 6.6: Attack detection performance comparison between baseline and improved models on BATADAL Dataset 3

and TGCN with attention model successfully detect all seven attacks. Nevertheless, I notice (Figures 6.11a, 6.11b, 6.11c, 6.11d) that the baseline model (TGCN) has a higher number of false positives compared to TGCN with attention model. I believe that the introduction of assurance methods (Attention and RMD) improves the TGCN with the attention model and minimizes the number of false positives. This is also reflected in the model's performance with an improved F1 and precision score compared to its baseline. Overall, our results suggest that TGCN with attention model performs better than TGCN (baseline).

Unsupervised Detection Results

To study the impact of the assurance constraints (Equations: 4.17 - 4.20) on HCAE, I present the model performance with and without assurance constraints in Table 6.6. Not all four constraints bring optimal performance, but a combination of these constraints achieves better classification and dimensionality reduction performance than the baseline AE model. From Table 6.6, I observe that sensitivity, specificity, accuracy, and F1 score improve significantly with assurance constraints applied to the AE. Also, I observe that precision is increased because *HCAE* learns the imbalanced data (BATADAL dataset is unbalanced) better than

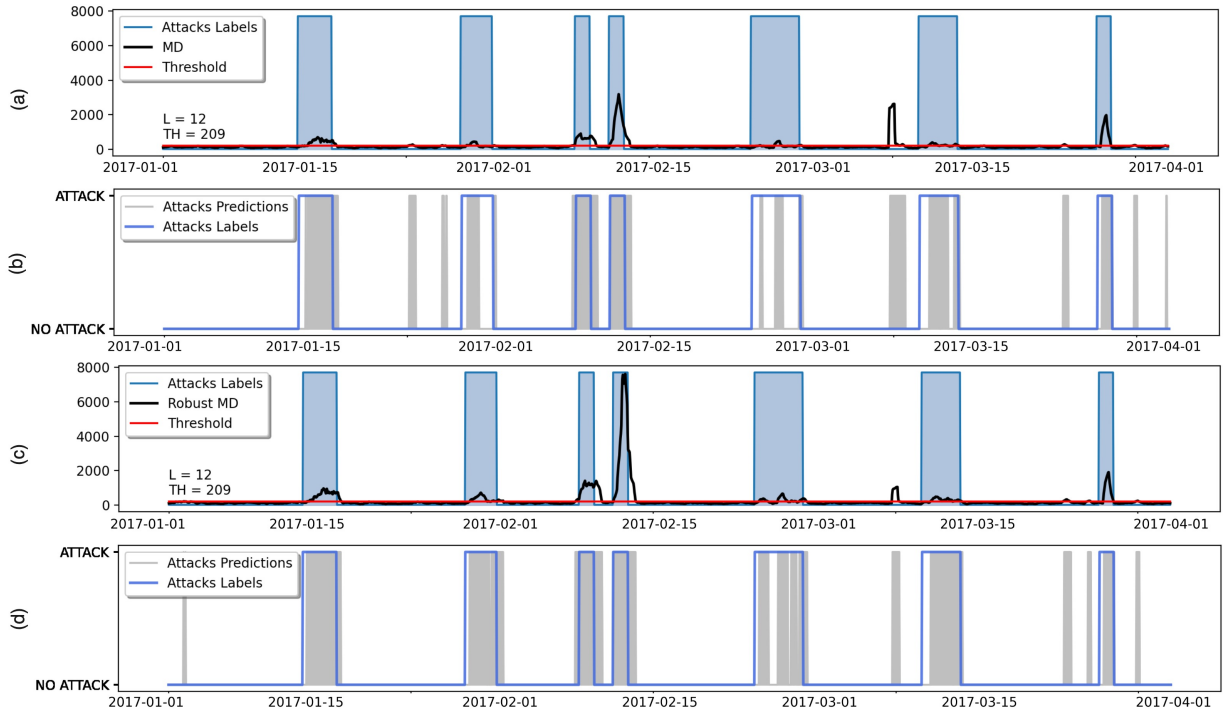


Figure 6.11: (a) Apply threshold on TGCN (b) TGCN detection results on the test dataset (c) Apply threshold on TGCN with attention (d) TGCN with attention detection results on test dataset

AE.

Figure 6.12 presents the attack detection performance of the unsupervised models. The test dataset (Dataset 3) consists of seven different attacks. All seven attacks are classified as "ATTACK" by both AE and *HCAE* models. Figures 6.12a and 6.12b present all seven attacks detected by the AE model and the *HCAE* model, respectively. As Figure 6.12a illustrates, in addition to detecting all SEVEN attacks, the AE model results in 21 sets of false alarms. On the contrary, the *HCAE* model results in a single false alarm (Figure 6.12b). The result suggests that *HCAE* learns the complex interdependencies between the features during concealed attacks, hence performing better than AE in detecting the attacks.

Table 6.9 presents the performance metrics, including the ranking score (S). Although both *HCAE* and AE time-to-detection performance is identical (94.7%), the classification per-

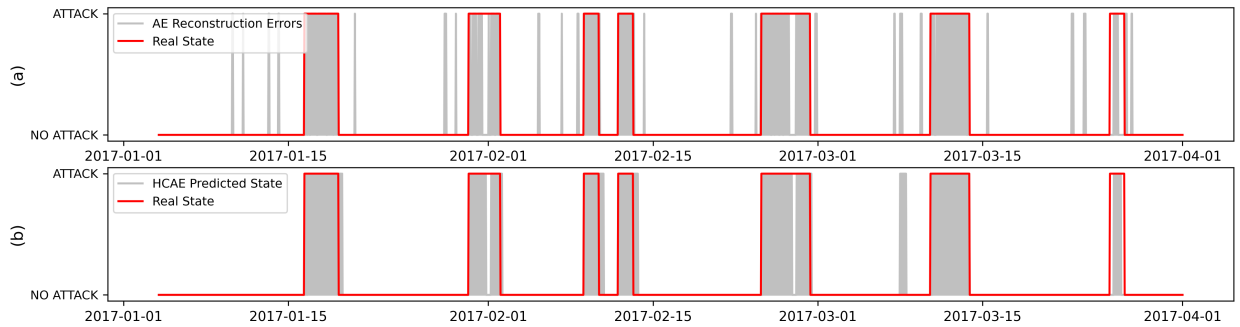


Figure 6.12: (a) AE detection results on test dataset (b) *HCAE* detection results on test dataset

formance (Ranking Score S) of the *HCAE* has significantly improved than the baseline AE. From the table, I observe that classification performance has been improved from 80% to 92%. Similarly, TPR also improved from 60.4% to 86.5%, a significant increase. This performance improvement is expected as *HCAE* learns the complex relationships between features in a deterministic scheme, whereas AE learns them in a non-deterministic approach.

6.3.2 RQ2: Data Poisoning

In this sub-section, I present the performance of both supervised and unsupervised models on poisoned data. Table 6.7 presents the attack detection performance on synthetic poisoned data generated using GAN.

Results (Figures 6.13a, 6.13b, 6.13c, 6.13d) suggest the supervised models perform poorly with poisoned data. More specifically for TGCN, I observe, by comparing Tables 6.6 and 6.7, that the attack detection performance of the model is decreased by more than 50% across all five metrics. Similar to the baseline model, the TGCN with attention model behaves poorly; results suggest, on average, a 65% reduction in performance across all five metrics. The poor performance of supervised models on the poisoned data can be explained as follows. The TGCN and TGCN with attention models learn the behavior of a WDS by

embedding the spatio-temporal structure of the WDS. In other words, they learn to detect attacks based on the sequential information inferred from the dataset during the training process. As the attacks are randomly distributed across the poisoned dataset (GAN data), both the supervised models fail to detect the attacks, resulting in poor performance.

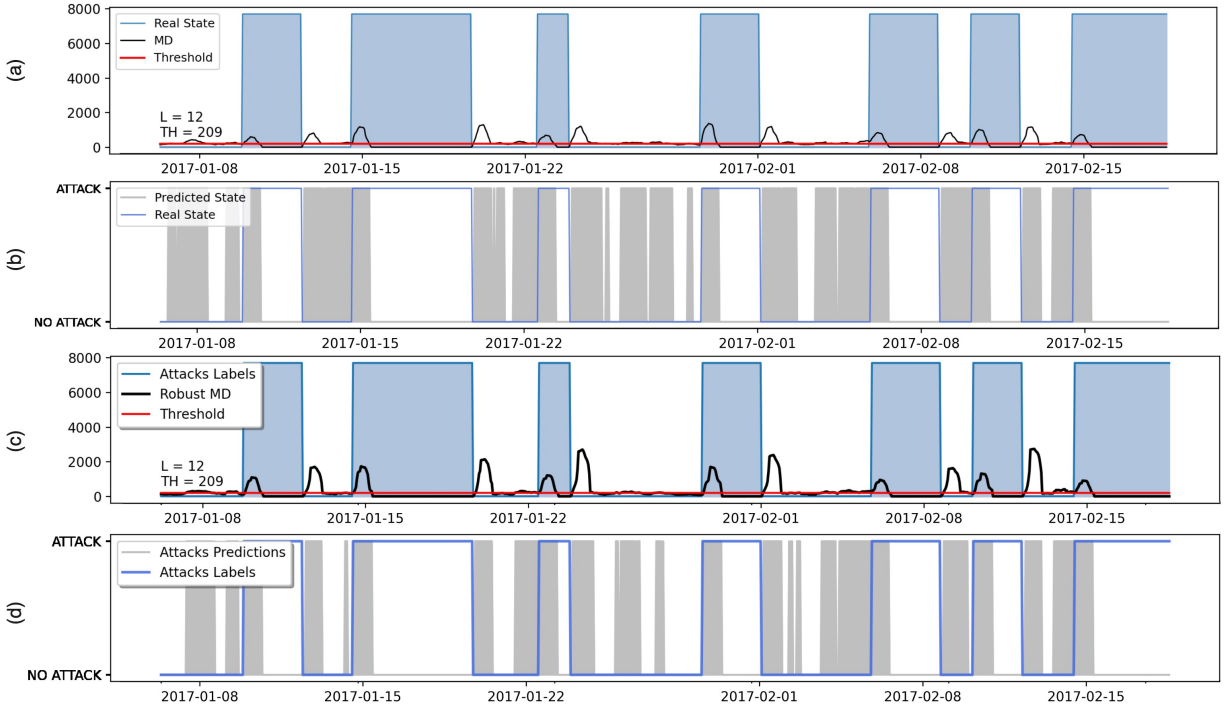


Figure 6.13: (a) Apply threshold on TGCN; (b) TGCN detection results on the poisoned dataset (c) Apply threshold on TGCN with attention; (d) TGCN with attention detection results on poisoned dataset

On the contrary to supervised models, I observe that the AE and *HCAE* (unsupervised models) perform well (Figures 6.14a, 6.14b, 6.14c, 6.14d) on the poisoned data. In some cases, results suggest the performance of both unsupervised models is better on poisoned data (Table 6.7) than their performance on the test dataset (Table 6.6). This improved performance can be attributed to the fact that AE and *HCAE* models treat the training samples as non-sequential data, and the data are randomly placed with poisoned samples. Hence, they can detect the randomly distributed attacks from the poisoned dataset effectively.

Performance Metrics	Supervised Model		Unsupervised Model	
(GAN Samples)	TGCN	TGCN with Attention	AE	*HCAE*
Precision	**0.310**	0.239	0.974	**0.984**
Recall	**0.264**	0.252	1	**1**
F1 Score	**0.285**	0.245	0.986	**0.991**
Accuracy	**0.365**	0.257	0.987	**0.992**
Specificity	**0.458**	0.262	0.976	**0.985**

Table 6.7: Attack detection performance comparison between baseline and improved models on GAN generated samples

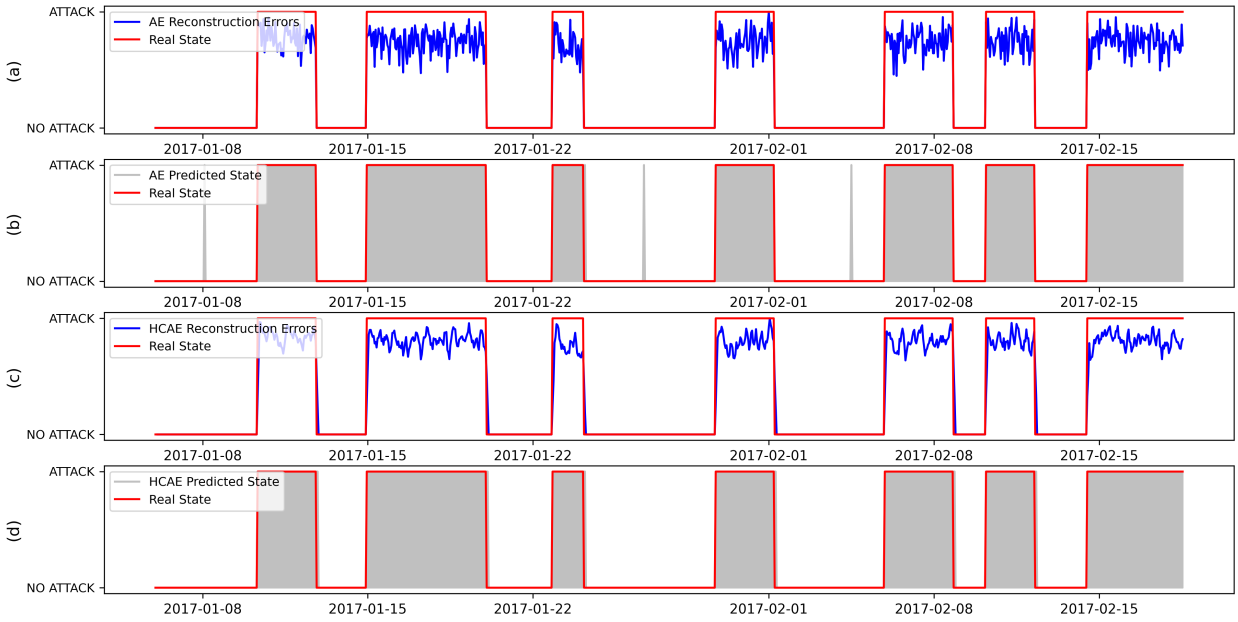


Figure 6.14: (a) AE reconstruction errors on the poisoned dataset, (b) AE detection results on the poisoned dataset, (a) *HCAE* reconstruction errors on the poisoned dataset, (b) *HCAE* detection results on poisoned dataset

6.3.3 RQ3: Feature Localization

In this sub-section, I present the model's ability to identify the features impacted by an attack (feature localization). I perform feature localization by estimating the deviation of the features from the "NO ATTACK" dataset distribution.

Attack Labels	Real Attacks Description	Predicted Feature Localization			
		TGCN with Attention	FP	HCAE	FP
Attack 8	Alteration of L_T3 thresholds leading to underflow	P_J256, **L_T3**, P_J289, L_T2	3	P_J256, **L_T3**, L_T6, P_J280, **F_PU4**, F_PU7	**4**
Attack 9	Alteration of L_T2	P_J289, P_J422, P_J300, L_T7	4	P_J422, P_J289, P_J280, P_J300, L_T6, F_PU1, L_T5, **F_V2**, L_T7, L_T4	**9**
Attack 10	Activation of PU3	**F_PU3**, P_J280, L_T7, L_T4, P_J269, F_PU1, F_PU9	6	**F_PU3**, S_PU10, F_PU10, L_T1, P_J269, P_J307, P_J14, P_J317	**7**
Attack 11	Activation of PU3	**F_PU3**, P_J280, L_T7, F_PU1, L_T4, L_T6, P_J307, P_J415, F_PU6, P_J289	**9**	L_T1, **F_PU3**, F_PU10, P_J269, P_J14, F_PU1, F_PU2	6
Attack 12	Alteration of L_T2 readings leading to overflow	P_J289, P_J300, **L_T2**	2	P_J300, P_J289, L_T1, P_J280, F_PU7, L_T5, L_T6, **L_T2**, P_J422	**8**
Attack 13	Change the L_T7 thresholds	L_T6	1	P_J307, P_J302, F_PU8, **F_PU10**, **L_T7**, L_T6, P_J306	**4**
Attack 14	Alteration of L_T4 signal	**L_T4**, L_T7, P_J415, **L_T6**	2	P_J415, F_PU7, L_T1, **L_T6**, P_J307, F_PU10, P_J14	**6**

Table 6.8: Feature localization results of TGCN with attention and *HCAE* on Dataset 3

The results presented so far suggest that the model customized with AI assurance methods and constraints performs better than their respective baseline model. Hence, for feature localization, I limit our evaluation to two models: TGCN with attention and HCAE. Table 6.8 presents the localization results for supervised and unsupervised models. The localized feature that matches the ground truth is highlighted in bold.

For TGCN with attention model, to localize the impacted features during an attack, I compare the mean squared error of the network from the testing set with its corresponding maximum error from the validation set (25% of Dataset 1). The supervised model can successfully localize five attacked nodes among the seven attacks while failing to localize attacked nodes for Attack 9 and Attack 13.

Next, I present the feature localization performance of the unsupervised model. To localize the impacted features, I select the features with the highest number of deviations from the threshold (θ_{th}) by estimating their mean squared error. During a predicted attack, I pick the top features for which reconstruction errors deviate most from the threshold (θ_{th}).

Overall, the results indicate that the modified models can successfully localize various attacks, including alteration of thresholds, signals, and meter readings. Furthermore, I observe that TGCN with attention localizes the attacked features with a minimal number of False Positives (FP) among the two models. False positives are estimated for each attack category by subtracting the set of total nodes detected by the model in that category from the set of ground truths. For example, Consider Attack 12, in which the readings of L_T2 are altered. While both the models successfully localize the feature (L_T2), the TGCN with attention model, in addition to identifying L_T2, also identifies two additional features as potentially attacked. In contrast, $HCAE$ models identify an additional eight nodes within the proximity as potentially attacked features. This is because, the neighbor nodes show similar behavior during normal operations, and therefore, during an attack, the model predicts those neighbor

nodes are highly likely to be attacked.

6.3.4 DeepH$_2$O Model Sensitivity Analysis

In this section, I evaluate the attack detection outcomes of both supervised and unsupervised models using Shapley values (Shapley values are the outcome of a game theoretic approach that explains the output of any ML model). In literature, variance-based sensitivity analysis is a popular approach that explains black box models; primarily, Sobol-based methods are gaining traction Bagherzadeh and Shafighfard [358]. However, this approach has one major limitation: it cannot explain localized observations. In this work, I am more concerned with local observation explanations than global ones since the models detect attacks from different nodes and time points in a WDS. Therefore, I elected to use Deep Explainer[1], an enhanced approach from SHapley Additive exPlanations (SHAP) library similar to Kernel SHAP. It approximates the conditional expectations of SHAP values using a selection of background samples.

In this analysis, I provide all seven categories of attack samples separately, generate SHAP values, and plot the scaled SHAP values ranging from 0 to 100 in Figure 6.15. This figure represents the importance of local features during the attack detection by the models. From the figure, I can observe that feature localization (Table 6.8) and Deep Explainer provide similar insights about the model's outcome. By observing Figure 6.15, it becomes evident that the model gives less attention to deactivated nodes from the training set (Dataset 1), including PU3, PU5, PU6, PU9, and PU11 while giving much more importance to the flow of pumps that worked during the training set. Additionally, I observed that all tanks were given attention during all seven categories because they were always active in the training set. One shortcoming of both models is that they could not learn the relationships among

[1] github.com/slundberg/shap

junctions because of the imbalance of the training dataset; most influential junctions got prioritized by the model over less participating ones. Therefore, model attack localization was incorrect during attacks 9 and 13; L_T2 and L_T7 weren't detected because other junctions got more "attention", including P_J280, P_J289, P_J300, when compared to ground truth nodes V2 and PU10/PU11.

6.3.5 Comparison with BATADAL Models

Next, I compare the attack classification performance of our models with the top-performing models from the BATADAL competition. To maintain consistency in our evaluation, all the models are evaluated using the test dataset (Dataset 3). Table 6.9 presents the comparison results, where our models are highlighted in bold. The results indicate both the unsupervised and supervised model exhibits better performance. $HCAE$ (unsupervised model), with a ranking score of 0.933, is ranked 3, whereas TGCN, with attention, achieves 0.845 and ranks eighth amongst the models from the BATADAL competition. Although our models do not achieve the highest ranking, they are superior compared to the top two models for the following reasons: 1). The top-ranked model is physics-based, and hence it is not relevant to compare with our model, an AI-based model. 2). The second-ranked model, although an AI-based model, might not perform well (detecting attacks) on previously unseen data. On the contrary, our models are scalable and demonstrate a better attack detection performance on unseen data.

Results indicate that adding the AI assurance methods to the TGCN model improves its overall performance. Compared to the baseline model ($S = 0.754$), TGCN with attention model achieves a better score ($S = 0.845$). Additionally, I observe that the time-to-detection (S_{TTD}) has improved significantly; the baseline model achieves 0.735, whereas the TGCN

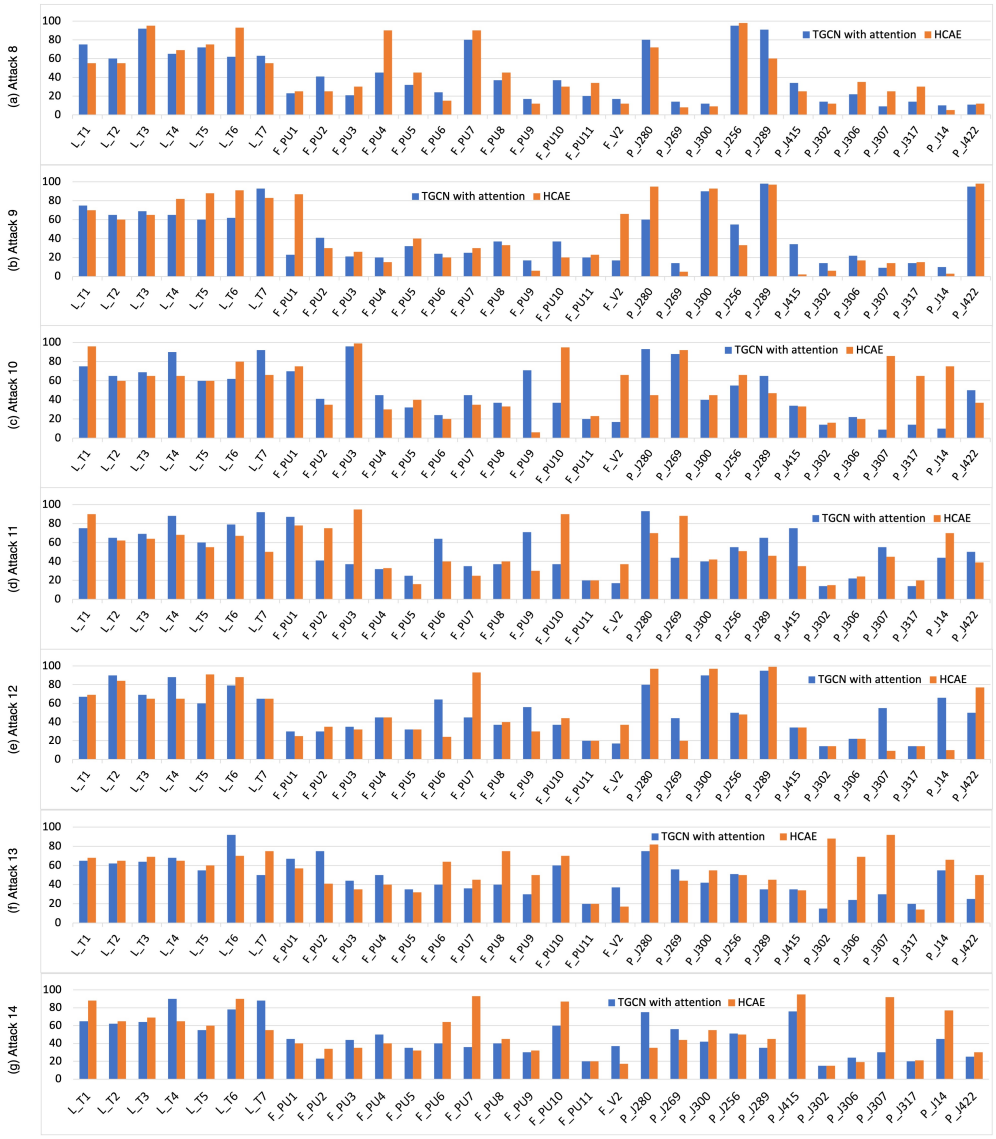


Figure 6.15: DeepH_2O attack detection local explanations using Shapley values. Ground truths are as follows: (a) Attack 8: Alteration of L_T3 thresholds leading to underflow, (b) Attack 9: Alteration of L_T2, (c) Attack 10: Activation of PU3, (d) Attack 11: Activation of PU3, (e) Attack 12: Alteration of L_T2 readings leading to overflow, (f) Attack 13: Change the L_T7 thresholds, (g) Attack 14: Alteration of L_T4 signal

with attention model achieves 0.839. A higher S_{TTD} is significant in the context of WDS; an improved S_{TTD} score indicates that the TGCN with attention model can swiftly identify an attack at the earliest compared to its baseline model.

Authors/Models	No. of Attacks Detected	Ranking Score (S)	time-to-detection (S_{TTD})	Classification Score (S_{CLF})	TPR	TNR
Housh and Ohar	7	0.97	0.965	0.975	0.953	0.997
Abokifa et al.	7	0.949	0.958	0.944	0.921	0.959
HCAE	**7**	**0.933**	**0.947**	**0.919**	**0.865**	**0.983**
Tsiami et al.	7	0.931	0.934	0.928	0.885	0.971
Giacomoni et al.	7	0.927	0.936	0.917	0.838	0.997
Brentan et al.	6	0.894	0.857	0.931	0.889	0.973
AE	**7**	**0.873**	**0.947**	**0.800**	**0.604**	**0.972**
TGCN with attention	**7**	**0.845**	**0.839**	**0.851**	**0.774**	**0.927**
Chandy et al.	7	0.802	0.835	0.768	0.857	0.678
Pasha et al.	7	0.773	0.885	0.66	0.329	0.992
TGCN	**7**	**0.754**	**0.735**	**0.773**	**0.553**	**0.922**
Aghashahi et al.	3	0.534	0.429	0.64	0.396	0.884

Table 6.9: Comparison of AI Assured models with BATADAL competition models

For unsupervised models, both $HCAE$ and AE (baseline) achieve an identical score for time-to-detection ($S_{TTD} = 0.947$). However, I observe that the $HCAE$ model has an improved TPR score (0.865) compared to its baseline (0.604). This results in the $HCAE$ model achieving a ranking score ($S = 0.933$) substantially higher than its baseline ($S = 0.873$). Furthermore, a higher TPR indicates that the model detects most attack samples. Thus, improving the trustworthiness of the model during deployment.

Both TGCN with attention and $HCAE$ models achieve a better ranking score compared to their respective baseline models.

6.4 cP$_2$O Forecasting Performance Evaluation

The forecasting performance metrics, summarized in Table 6.10, include RMSE, MAPE, Interquartile Range of Absolute Percentage Error (iqrAPE), Standard Deviation of Percentage Error (StDPE), Peak Detection Rate (PDR), and Mean Percentage Error (MPE), as defined in Equations (6.3) to (6.9). Our proposed model is evaluated in two variations: cP$_2$O and

its ensemble version, cP_2Oe. For comparison, I also include the performance of our previous model, P_2O, as reported in prior work (Kulkarni et al. [3]).

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{6.3}$$

$$\text{MdAPE} = \text{median}\left(\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\right) \tag{6.4}$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \tag{6.5}$$

$$\text{iqrAPE} = Q_3\left(\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\right) - Q_1\left(\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\right) \tag{6.6}$$

$$\text{MPE} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i} \times 100\right) \tag{6.7}$$

$$\text{StDPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i} \times 100 - \text{MPE}\right)^2} \tag{6.8}$$

$$\text{PDR} = \frac{\text{Number of Correctly Detected Peaks}}{\text{Total Number of True Peaks}} \tag{6.9}$$

It is important to note that separate models were trained for each dataset—DC Water and AlexRenew—using the same architecture and methods but different training data specific to each WWTP. This approach allows the models to capture the unique characteristics and patterns inherent in each dataset while leveraging the strengths of the proposed architecture.

As shown in Table 6.10, our proposed hybrid models outperform state-of-the-art methods across most metrics on both datasets. Particularly, the ensemble model cP_2Oe demonstrates superior accuracy, achieving the lowest MAPE values of 2.10% for the DC Water model and 1.90% for the AlexRenew model.

Table 6.10: Performance metrics comparison across models for tunnel water level forecasting and nitrate level predictions

	Tunnel water level forecast (DC Water)								NO_3 level prediction (AlexRenew)						
Model	RMSE	MAPE	MPE	iqrAPE	StDPE	MdAPE	PDR	Score	RMSE	MAPE	MPE	iqrAPE	StDPE	MdAPE	Score
Naive	6.98	5.15	0.25	3.35	7.82	4.94	50.25	1.02	3.45	5.20	-0.20	3.40	7.60	5.00	5.10
ARIMA	4.71	3.32	-0.02	3.05	5.20	3.08	75.85	17.89	2.90	3.15	**0.02**	3.10	5.35	3.20	15.25
ES	4.34	3.18	0.01	2.74	5.10	2.92	77.45	26.25	2.75	3.05	0.05	2.70	5.25	2.90	20.30
SVM	3.86	2.89	-0.15	2.55	4.54	2.65	85.55	34.70	2.30	2.55	0.08	2.45	4.55	2.35	40.15
LGBM	3.75	2.82	0.02	2.60	4.41	2.63	83.63	37.30	2.50	2.70	-0.10	2.50	4.70	2.55	35.60
XGB	3.62	2.71	**0.00**	2.44	4.30	2.48	84.24	47.15	2.35	2.60	0.04	2.40	4.60	2.45	43.25
Prophet	4.09	3.60	-0.12	2.85	4.76	3.45	60.55	9.77	3.20	3.80	0.15	3.20	6.90	4.40	8.10
N-WE	3.26	2.55	-0.14	2.32	4.12	2.35	86.45	57.33	2.25	2.50	0.10	2.30	4.35	2.30	45.50
GRNN	3.25	2.53	-0.10	2.30	4.12	2.32	86.63	60.40	2.20	2.45	-0.05	2.25	4.30	2.35	48.75
MLP	3.86	2.97	0.05	2.80	4.67	2.73	80.85	33.68	2.55	2.85	0.12	2.75	5.15	2.65	30.20
LSTM	3.75	2.82	0.01	2.60	4.41	2.63	83.63	37.30	2.50	2.75	-0.07	2.55	4.80	2.60	32.85
ANFIS	4.98	3.70	-0.12	3.69	6.25	3.22	65.36	12.89	3.10	3.65	0.18	3.60	6.55	3.20	12.75
MTGNN	3.69	2.95	-0.02	2.62	4.49	2.70	82.28	32.80	2.40	2.90	0.08	2.60	4.90	2.65	33.50
DeepAR	4.84	3.50	-0.50	3.00	4.95	3.31	70.78	16.99	3.00	3.40	-0.45	2.95	5.25	3.25	18.65
WaveNet	4.15	3.12	-0.80	2.77	4.58	2.90	78.06	28.15	2.70	3.00	-0.75	2.70	4.95	2.90	25.80
N-BEATS	3.57	2.62	0.04	2.41	4.21	2.40	85.05	48.95	2.30	2.55	-0.03	2.35	4.50	2.40	46.85
P_2O	3.35	2.71	0.18	2.25	3.90	2.48	86.51	84.15	2.20	2.35	0.03	2.40	4.60	2.50	60.50
cP_2O	2.96	2.15	-0.18	1.75	3.22	1.55	93.54	90.85	2.00	1.95	0.05	1.90	3.25	1.70	89.32
cP_2Oe	**2.94**	**2.10**	-0.18	**1.71**	**3.19**	**1.50**	**93.54**	**94.85**	**1.95**	**1.90**	0.05	**1.85**	**3.20**	**1.65**	**90.35**

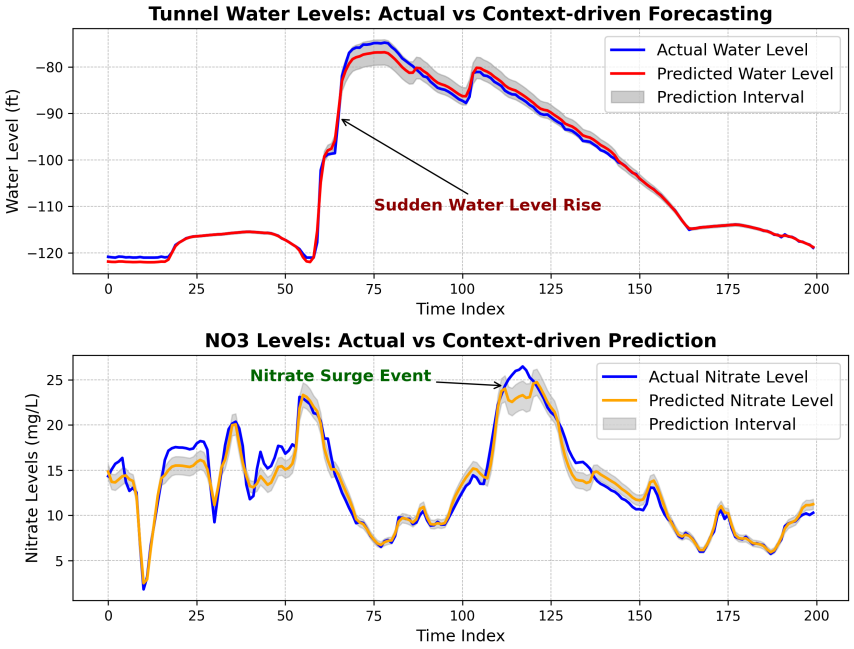


Figure 6.16: cP$_2$Oe forecasting results on WWTP data (actual values in black, forecasts in red, predictive intervals in light gray shades).

Figure 6.16 illustrates the predictions and dynamic predictive intervals for a test period. Observations fall within the predictive intervals approximately 90.51% ± 3.21% of the time, while 5.86% ± 1.85% fall below and 3.63% ± 1.47% above. The narrow iqrAPE and StDPE further highlight each cP$_2$Oe model's ability to maintain consistent predictions for its respective dataset.

The proposed cP$_2$Oe models significantly improve accuracy compared to the earlier version, P$_2$O, when trained on their respective datasets. For the DC Water dataset, MAPE is reduced from 2.71% to 2.10%, a reduction of approximately 22%, and RMSE is reduced by about 12% (from 3.35 to 2.94). For the AlexRenew dataset, MAPE is reduced from 2.35% to 1.90%, a reduction of approximately 19%, solidifying the cP$_2$Oe models' positions as the top-performing models for each dataset.

The lowest StDPE and iqrAPE values in Table 6.10 demonstrate that cP$_2$Oe models deliver

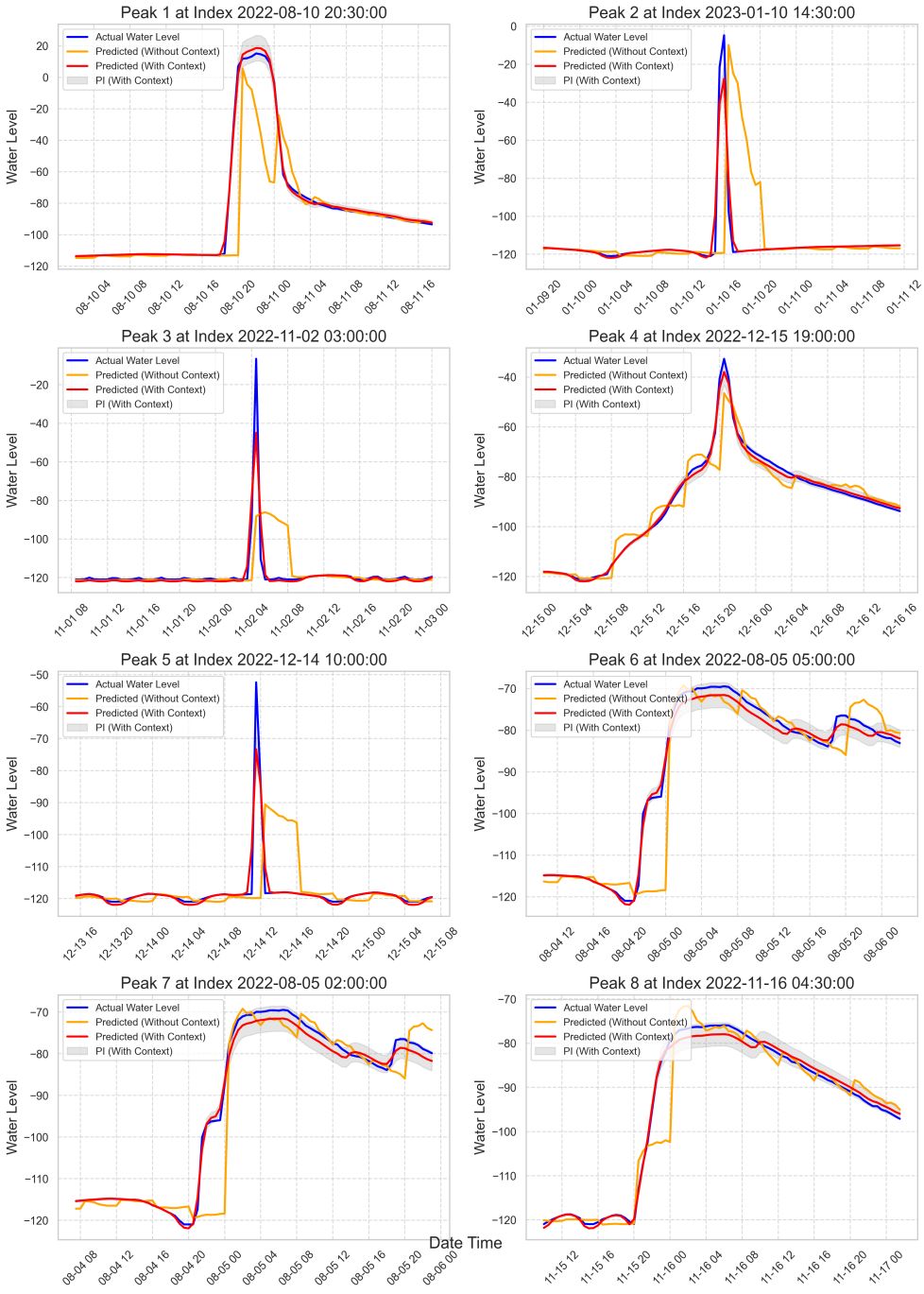


Figure 6.17: Actual tunnel water level vs predicted values by cP$_2$O (red line) and P$_2$O (orange line) for total eight different extreme events

more consistent and less variable predictions compared to baseline models such as Prophet and DeepAR on both datasets. For instance, it achieves the lowest iqrAPE (1.71% for DC Water and 1.85% for AlexRenew) and StDPE (3.19% for DC Water and 3.20% for AlexRenew), highlighting their robustness in controlling the spread of percentage errors specific to each dataset.

Although the DC Water model exhibits a slightly negative MPE value (indicating a minor tendency to under-predict), the MPE is balanced in the AlexRenew model, suggesting that the predictions are well-calibrated for both datasets. Additionally, my models are more robust to outliers, as observed from the MdAPE values; cP_2Oe attains the lowest MdAPE values, such as 1.50% for DC Water and 1.65% for AlexRenew. The cP_2Oe model for DC Water also excels in PDR, identifying 93.54% of peaks, indicating its robustness in detecting extreme events such as floods.

These results validate my hypothesis for Research Question **RQ2** (See the section 3.2.3):, demonstrating that the cP_2O model is both adaptable and effective across different WWTPs and forecasting tasks. By training separate models using the same architecture and methods on diverse datasets—specifically for tunnel water level forecasting at DC Water and nitrate level prediction at AlexRenew, the model consistently achieve high accuracy tailored to each dataset's unique characteristics and forecasting objectives.

To evaluate the statistical significance of my forecasting performance, I perform a pairwise Giacomini-White test for conditional predictive ability. The test scores in Table 6.10 were calculated by comparing the forecast errors, including MAPE, RMSE, MPE, and PDR, of the models to determine if one model's predictive performance was statistically superior to another's at the $\alpha = 0.05$ significance level. For the DC Water experiment, I include PDR in the comparison; for AlexRenew, I apply the rest of the key metrics. The results, labeled as "*Score*" in Table 6.10, reveal that each cP_2Oe model achieved significantly lower forecasting

errors than other models in over 90% of pairwise comparisons for their respective datasets. This confirms the models' superiority, particularly the cP$_2$Oe configuration, which benefits from exogenous input, validating the design choice of dynamic, context-driven adjustments.

Figure 6.17 showcases dynamic adjustments during peak events for the DC Water dataset. Notably, the proposed model cP$_2$O aligns well with extreme water level increases compared to my previous model P$_2$O, a key requirement for critical event forecasting. This plot and the results from Table 6.10 support the hypothesis of Research Question **RQ4** (See section 3.2.3):: using a quantile loss function employed in cP$_2$O effectively reduces forecast bias during peak events or extreme conditions and improves uncertainty estimation in multi-step-ahead forecasting.

6.4.1 Ablation Study

To assess the contributions of key components in the cP$_2$O model, I conducted an ablation study by systematically removing the context stage, the attention mechanism, and the dilated LSTM layers. This study was performed on both the DC Water and AlexRenew datasets, with results presented in Table 6.11.

Table 6.11: Impact of key components of cP$_2$O on forecasting performance

Model Variant	**Tunnel water level forecast (DC Water)**				**NO$_3$ level prediction (AlexRenew)**		
	MAPE (%)	**RMSE**	**PDR** (%)	**StDPE** (%)	**MAPE** (%)	**RMSE**	**StDPE** (%)
cP$_2$O (Full Model)	**2.10**	**2.94**	**93.54**	**3.19**	**1.90**	**1.90**	**3.20**
Without Context Stage	2.48	3.32	85.50	3.50	2.25	2.20	3.50
Without Attention Mechanism	2.60	3.50	83.00	3.80	2.05	2.10	3.50
Without Dilated LSTM Layers	2.85	3.80	80.50	4.00	2.20	2.30	3.80

1. **Without Context Stage:** Removing the context stage led to performance declines across both datasets. For DC Water, MAPE increased from 2.10% to 2.48%, RMSE from 2.94 to 3.32, and PDR decreased from 93.54% to 85.50%. For AlexRenew, MAPE rose from 1.90% to 2.25% and RMSE from 1.90 to 2.20. This underscores the critical role of external contextual data (e.g., weather, influent characteristics) in enhancing prediction accuracy and robustness. The consistent impact supports my hypothesis for Research Question **RQ1** (See section 3.2.3): incorporating contextual data significantly improves the accuracy of short-term predictions in WWTPs.

2. **Without Attention Mechanism:** Excluding the attention mechanism increased errors on both datasets. In DC Water, RMSE rose to 3.50, MAPE to 2.60%, and StDPE to 3.80%; in AlexRenew, RMSE increased to 2.10, MAPE to 2.05%, and StDPE to 3.50%. This highlights the models' reduced capacity to dynamically adjust to varying input importance over time, leading to less accurate and more variable predictions. These results confirm my hypothesis for Research Question **RQ3** (See section 3.2.3):: integrating an attention mechanism in cP_2O enhances the models' ability to weigh input features effectively, thereby improving forecasting accuracy.

3. **Without Dilated LSTM Layers:** Removing the dilated LSTM layers impaired the models' ability to capture temporal dependencies over multiple time scales. For DC Water, RMSE increased to 3.80, MAPE to 2.85%, and PDR dropped to 80.50%; for AlexRenew, RMSE rose to 2.30 and MAPE to 2.20%. This emphasizes the importance of dilated LSTM layers in learning both short-term fluctuations and long-term trends, contributing to the models' adaptability and precision.

Overall, the ablation study confirms that each component significantly enhances the cP_2O models' performance across both datasets. Including external context data improves adapt-

ability to changing conditions, the attention mechanism allows dynamic weighting of input features, and dilated LSTM layers enable capturing temporal dependencies over multiple scales. The consistent performance degradation when components are removed validates my design choices and supports my research hypotheses. These findings demonstrate the effectiveness of the full cP$_2$O architecture for short-term forecasting in wastewater treatment plants across different settings.

Chapter 7

Assessing the Fidelity and Utility of Water Systems Data Using Generative Adversarial Networks: A Technical Review

Limited data access in Water Distribution Systems (WDSs) is a longstanding barrier to data-driven research and development. This limited access is compounded by the hesitation of agencies and facilitates to integrate and share data, driven by the absence of standard mandates, resource constraints, privacy and security concerns, and legal challenges. This review paper addresses this limitation by utilizing Generative Adversarial Networks (GANs) to generate realistic synthetic datasets, overcoming data scarcity and privacy concerns in WDSs. Seven state-of-the-art GAN models are trained and evaluated using three multivariate time-series datasets. The core contribution of this work lies in its comprehensive technical review of the GANs, evaluating their ability to replicate temporal dynamics and maintain spatio-temporal dependencies within WDSs. Techniques like t-distributed Stochastic Neighbor Embedding (t-SNE) and Principal Component Analysis (PCA) are used to quantify the diversity of the generated synthetic data.

Key findings indicate that specific GAN models, such as Cramer GAN and CTGAN, are

effective in generating data for predictive modeling, replacing the need for original WDS datasets. Additionally, DoppelGANger and TimeGAN exhibit strong capabilities in preserving essential spatio-temporal relationships, which are critical for applications like environmental impact estimation. The results also highlight the potential of GAN-generated synthetic data in enhancing the management and security of WDSs, particularly in scenarios where data are scarce or sensitive. This research contributes to Artificial Intelligence (AI) in water resource management and guides the selection of appropriate GAN models for specific tasks, demonstrating their practical implications in real-world scenarios.

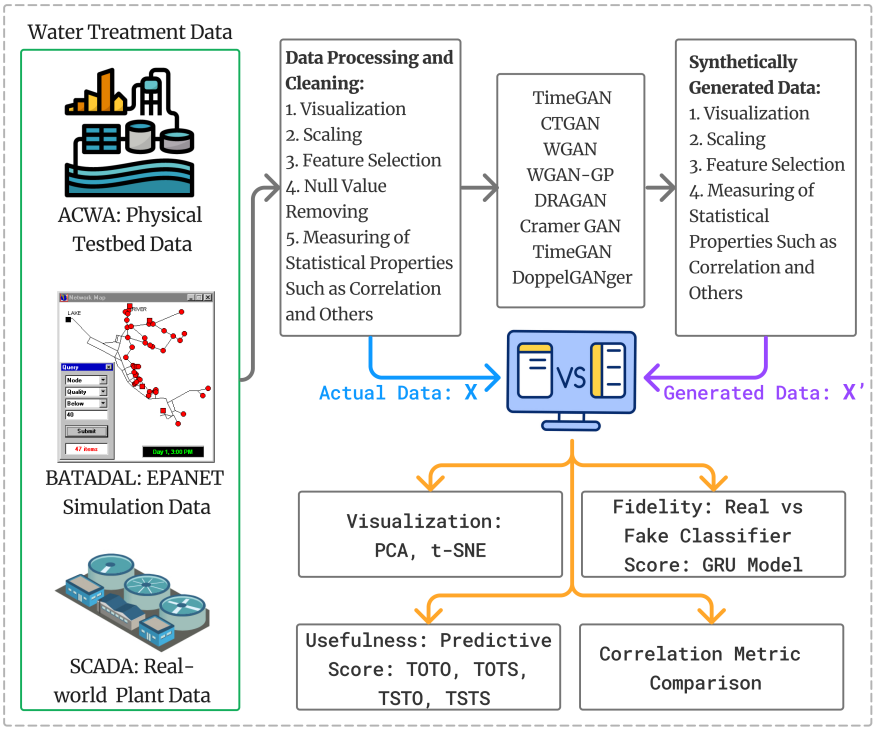


Figure 7.1: Pipeline for Synthetic Data Generation and Evaluation. Three Datasets- (1) a Physical Water Testbed (ACWA), (2) Simulated Data (via EPANET), and (3) Real-world Water Treatment Plant Data (via Supervisory Control and Data Acquisition) are Used to Generate Synthetic Data by Applying Seven Different GAN models and Assessed via Quantifiable Measures to Test Data Fidelity and Utility.

7.1 Introduction

In recent days, data-driven methodologies (Halevy et al. [359]) are essential in exploring empirically-driven design decisions and management strategies (Batarseh and Kulkarni [21], Baltrunas et al. [360], Bischof et al. [361], Chen et al. [362], Grandl et al. [363], Jiang et al. [364], Liu et al. [365], Mao et al. [366], Montazeri et al. [367], Sundaresan et al. [368]). The unprecedented advancements in AI, particularly in DL, have prompted an urgent need for extensive datasets necessary for training, testing, and validating DL models (Tuptuk et al. [123]). This demand is especially pronounced in WDSs, where data are critical for understanding and managing their complex dynamics. However, data availability often remains a significant challenge, typically confined to entities already possessing such data. Despite the potential for mutual advantages, concerns over disclosing confidential business information and violating privacy standards deter data sharing among stakeholders (McGregor et al. [369]). To address this AI challenge, generating and distributing synthetic datasets derived from authentic data sources (Antonatos et al. [370], Denneulin et al. [371], Di et al. [372], Ganapathi et al. [373], Juan et al. [374], Li and Liu [375]) has become a practical solution. The outcome of realistic synthetic data has therefore achieved prominence, with DL methodologies emerging as critical contributors in data generation steps (Frid-Adar et al. [376], Zhang et al. [377], Xu et al. [378], Bowles et al. [379], Assefa et al. [380]). Compared to traditional ML techniques, DL provides a more nuanced understanding and management of the inherent complexities in WDSs. GANs (Goodfellow et al. [189]) demonstrate this state-of-the-art capability, as they excel in producing accurate representations of complex, multidimensional data relationships, particularly in scenarios where it is challenging to obtain original data due to scarcity, sensitivity, or other factors (Lin et al. [381]). The significance of such synthetic datasets is highlighted in environments where data accessibility is a significant challenge (ICS-CERT [54], Walton [56], Cava [57], Rubin [58]), proving significant

development of AI models that can effectively administer and protect WDS infrastructures.

The synthesis of these datasets via DL, primarily through GANs, signifies a considerable advancement in applying AI to WDS management. These developments have significantly enhanced my understanding of complex WDSs and contributed meaningfully to these essential infrastructures' efficient and secure operation.

7.1.1 Motivation: The Need for Water Data

One of the primary motivations for synthetic data generation for WDSs is improving cybersecurity. In recent years, WDSs have increasingly relied on automated control systems that introduce significant cyber-physical security vulnerabilities (Sikder et al. [5], Batarseh and Freeman [202], Sikder and Batarseh [382]), as a rising wave of adversarial cyber activities targeting these systems. The incident on February 5th, 2021, at the Oldsmar water treatment plant in Florida[1], where an attacker altered the chemical levels, illustrates this vulnerability. To counter such threats, initiatives such as the BATtle of the Attack Detection ALgorithms (BATADAL), which employs EPANET[2] (Taormina et al. [8], Erba et al. [383], Cheung et al. [384]), have been established. However, the need for original WDS datasets often restricts these tasks, emphasizing the significance of synthetic data.

Furthermore, synthetic data also plays an indispensable role in addressing environmental and operational challenges (El Emam et al. [385]). It supports modeling the impacts of environmental factors, such as droughts or floods, on water supply and distribution networks, thereby facilitating the development of adequate contingency plans. Synthetic data simulates infrastructure aging and maintenance needs, promoting proactive management and planning.

[1] https://www.wired.com/story/oldsmar-florida-water-utility-hack/

[2] EPANET is a public-domain software package for WDS modeling developed by the United States Environmental Protection Agency (EPA)'s Water Supply and Water Resources Division

Highlighting its broader application, a study by Lin et al. [386] demonstrates AI techniques, including clustering and neural networks, to develop a comprehensive flood susceptibility index known as NeuralFlood. This index evaluates multiple factors, aiding decision-makers in allocating resources efficiently and identifying high-risk areas for effective flood mitigation.

Additionally, technological innovation in water management benefits significantly from synthetic data. For example, developing soft sensing (Wang et al. [387]) or innovative metering technologies (Rahim et al. [388]) using synthetic datasets reduces the need for costly and time-consuming real-world trials. Moreover, AI is essential in creating decision support systems in WDSs, enabling more accurate and efficient modeling and forecasting (Kulkarni et al. [3], Batarseh et al. [389]). Synthetic data can aid in reducing operational costs and optimizing potential chemical and electricity consumption due to system failures or environmental hazards. This efficient allocation and utilization of resources contribute to cost savings and the sustainable management of water resources.

The contributions of physical water testbeds such as AI & Cyber for Water & Agriculture (ACWA) (Batarseh et al. [4]), and (Batarseh et al. [390])[3], Secure Water Treatment (SWaT) (Mathur and Tippenhauer [391]), and Water Distribution (WADI) (Ahmed et al. [392]) are vital for water systems research. However, these datasets alone are insufficient to cover the potential scenarios WDSs may encounter, underlining the importance of synthetic data for comprehensive coverage and preparedness.

7.1.2 Research Contributions

This section outlines my contribution and presents my research questions. My primary goal is to produce realistic synthetic water data and validate the quality of generated data by

[3] https://github.com/AI-VTRC/ACWA-Data

assessing its fidelity and utility. I leverage seven GAN models (TimeGAN Yoon et al. [393], CTGAN Xu et al. [378], WGAN Ring et al. [394], WGAN-GP Desai et al. [395], DRAGAN Kodali et al. [396], Cramer GAN Bellemare et al. [397], and DoppelGANger Lin et al. [381]) in experiments on three multivariate time-series datasets. Each model offers a unique approach to data generation. For example, TimeGAN leverages supervised and unsupervised learning to generate datasets mirroring real-world dynamics, potentially a better-suited model for my time series datasets. WGAN and DRAGAN are notable for their stability and convergence, while Cramer GAN and DoppelGANger allow for diverse data generation approaches. My experiments test whether GANs can accurately replicate the temporal dynamics of water systems, ensuring that the synthetic data sequences reflect the characteristics of original data sequences.

I select three distinct multivariate time-series datasets: a physical testbed- ACWA (Batarseh et al. [4]), EPANET-based data BATADAL (Erba et al. [383]), and data from a real-world water treatment plant (name withheld for confidentiality). The ACWA dataset represents an operational testbed generated by my team[4], mirroring a modern, large-scale water supply facility. The EPANET dataset provides insights into water flow dynamics and conceals attacks on physical layer components (Erba et al. [383]). The third dataset, from a water treatment plant, offers a real-world perspective on operational challenges in water treatment.

My evaluation metrics include t-distributed Stochastic Neighbor Embedding (t-SNE) (Belkina et al. [398]) and Principal Component Analysis (PCA)(Bro and Smilde [399], Van der Maaten and Hinton [400]) to compare between synthetic and original datasets. I also used a post-hoc classifier (GRU) to distinguish between generated and original data and applied the "train on synthetic, test on original (TSTO)" framework (Esteban et al. [401]) for sequence prediction.

[4] https://ai.bse.vt.edu/ACWA_Lab.html

All of this study's ACWA-generated datasets are available in a public repository [5].

My central research question in this technical review is as follows:

1. In the context of generating realistic WDS data, how do different GAN methods (e.g., TimeGAN, CTGAN) compare in terms of data fidelity (accuracy in mimicking real data) and utility (usefulness for specific applications or tasks)?

This question breaks down into two key aspects:

a. Given a GAN G, can I generate a WDS multivariate time sequential dataset D_{synth} such that the accuracy $A(D_{\text{synth}})$ is comparable to the accuracy $A(D_{\text{original}})$ of an original dataset D_{original}?

b. Can I evaluate synthetic time-series data generation D_{synth} in a 3-fold manner for WDSs?

   - Quantitatively, using statistical measures $S(D_{\text{synth}})$,
   - Qualitatively, with expert assessment $Q(D_{\text{synth}})$,
   - Visually, with graphs $G(D_{\text{synth}})$.

This paper introduces a comprehensive technical review integrating seven distinct GANs to explore my research question across three multivariate time-series datasets. Figure 7.1 presents a high-level workflow, illustrating the key stages of my experimental processes. I have employed multiple testing and evaluation methods, including diversity, fidelity, and usefulness, to estimate the quality and utility of the synthetically generated datasets and documented all experimental results. Section 2 reviews related literature, while section 3 covers data description and GAN models. Section 4 delves into my methodologies, section 5

[5] https://github.com/AI-VTRC/ACWA-Data/tree/main/GANs

elaborates on experimental results and their discussion, section 6 discusses the implications of synthetic data in water policy, and section 7 summarizes and concludes the paper.

7.2 Related Works

Data generation is vital in water systems, particularly when balancing two key objectives: privacy preservation and maintaining data distribution and availability. This trade-off is challenging; prioritizing privacy preservation can reduce data utility due to limited availability. My work emphasizes capturing distribution relevancy across time points and understanding complex variable interdependence over time. For instance, for multivariate sequential data $x_{1:T} = (x_1, ..., x_T)$, I aim to accurately model the conditional distribution of temporal transitions $p(x_t|x_{1:t-1})$.

Privacy concerns in essential infrastructure, such as water utilities, have escalated, highlighted by the 2019 ransomware attack on the Riviera Beach Water Utility (RBWU), which paralyzed the computer systems controlling pumping stations, water quality testing, and payment operations. The government authorities paid 65 bitcoins - approximately $600,000 – to the attacker in a few days, but still, after two weeks, water pump stations and water quality testing systems were partially available (Hassanzadeh et al. [402]). This incident led to the U.S. Environmental Protection Agency (EPA) proposing, then withdrawing, a rule to evaluate cybersecurity in public water utilities due to legal pushback[6](Dwork et al. [403]).

Synthetic data generation is proposed as one of the solutions to utilize data for research and development without compromising sensitive real-world data (Patki et al. [404]). Generating synthetic datasets can mitigate overfitting and enhance model generalization by introducing

[6] https://www.theregister.com/2023/10/13/epa_rescinds_water_cybersecurity_rule/#~:text=,attack

unseen data, especially where real-world data are scarce (Sikder et al. [5], Sarkar et al. [405]). Sikder et al. 2023, demonstrated that adversarial testing through synthetic data generation yields more generalizable models. Critical system research data are classified into original, synthetic, and testbed types, each with its own significance (Buczak and Guven [406]). For example, PGGAN (Gautam et al. [407]) has generated high-resolution river images and aided with various hydrological studies. Synthetic time series data has also been used to improve models in predicting the burst failure risk of corroded pipelines (Mazumder et al. [408]) and in combined sewer flow predictions (Bakhshipour et al. [409]).

Goodfellow et al.'s introduction of GANs (Goodfellow et al. [189]) revolutionized data generation, with architectures like WGAN (Arjovsky et al. [410]) and WGAN-GP Gulrajani et al. [411] improving training stability. TimeGAN (Sauber-Cole and Khoshgoftaar [412]) and CGAN Mirza and Osindero [413] are effective for time series data, capturing temporal dependencies. DRAGAN (Kodali et al. [396]) and Cramer GAN (Bellemare et al. [397]) address training stability and accurate temporal dependency representation. CTGAN (Xu et al. [378]) is notable for handling discrete and continuous data and missing data problems. TimeGAN is less sensitive to parameter changes during training, suitable for data with static and sequential features (Yoon et al. [393]). DoppelGANger (Lin et al. [381]) excels in preserving privacy and managing time series correlations.

7.2.1 Synthetic Data Generation on Multivariate Time-Series

Traditional time series data generation approaches are limited by data type distributions and computational challenges, affecting synthetic data reliability (Aviñó et al. [414], Cormode et al. [415], Sun et al. [416]). GAN-based methods offer more flexibility and performance enhancement (Xu et al. [378], Lin et al. [381], Park et al. [417]). However, many GAN

experiments focus on static dependencies, overlooking temporal aspects crucial in real-world data (Xu et al. [378], Ring et al. [418]). Recent attempts partially incorporate temporal dependence in GANs, but limitations still remain (Lin et al. [381], Jan et al. [419]).

Table 7.1: Comparisons of GANs for Synthetic Data Generation

Methods	**Capture Attribute Dependence**	**Capture Temporal Dependence**	**Multivariate Generation (Continuous)**	**Categorical Variable**
WGAN Ring et al. [394]	Yes	No	No	Yes
CTGAN Xu et al. [378]	Yes	Partially	Yes	Yes
DRAGAN Kodali et al. [396]	Yes	No	Yes	Yes
Cramer GAN Bellemare et al. [397]	Yes	No	Yes	Yes
TimeGAN Yoon et al. [393]	Yes	Yes	Yes	No
WGAN-GP Desai et al. [395]	Yes	No	Yes	Yes
DoppelGANger Lin et al. [381]	Yes	Partially	Yes	Yes

In WDS research, Zhou et al. [291] tackled the scarcity of industrial control dataset attacks using GANs, claiming significant attack detections (Zhou et al. [291]). However, their framework, while innovative, is computationally intensive. My approach with various GANs aims to bridge the gap in generating diverse and similar synthetic WDS data. Table 7.1 summarizes the GANs used in my experiments, highlighting their strengths and applications for synthetic data generation.

7.3 Data Descriptions and Methodologies

This section describes the datasets used in this work and briefly discusses all GANs used in the experiment.

7.3.1 Datasets Collection

This section describes three datasets: the ACWA testbed dataset, the BATADAL (EPANET) dataset, and a real-world water treatment plant dataset. Collectively, these datasets are

integral for comprehending and examining water systems. They encompass the diverse data collection methods applicable to water systems, offering a comprehensive view of data acquisition and management variations. The datasets mentioned are further detailed as follows:

AI & Cyber for Water & Agriculture: ACWA

My study actively employs the ACWA testbed, a dynamic and versatile platform, for data collection for real-time water quality monitoring and supply management. The ACWA infrastructure includes three distinct topologies - Line, Bus, and Star - each tailored for collecting a broad range of data pertinent to water quality metrics. During the operation of these topologies, I record key parameters such as pH, temperature, Dissolved Oxygen (DO), turbidity, nitrate levels, Electrical Conductivity (EC), soil moisture, water level, pressure, and flow rate. I systematically store the data in a MongoDB database, ensuring efficient retrieval for advanced modeling and AI-based analyses.

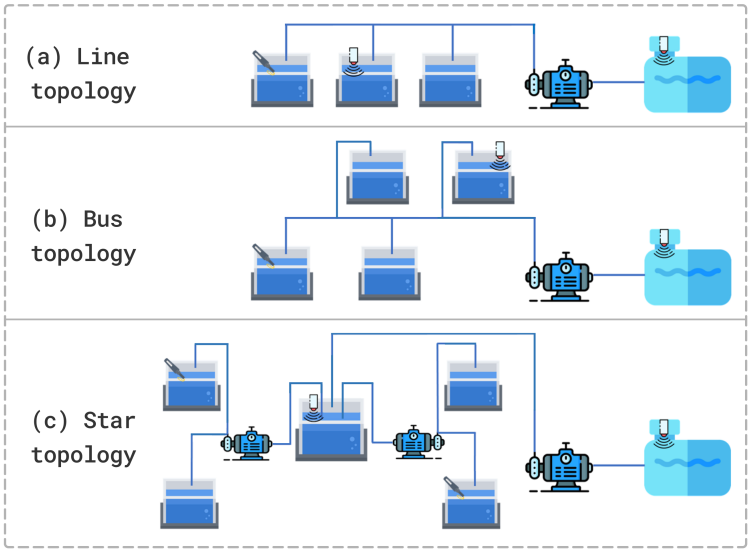


Figure 7.2: Schematic Representations of the (a) Line Topology, (b) Bus Topology, and (c) Star Topology as in the ACWA Testbed Batarseh et al. [4]

ACWA Testbed Topologies ACWA testbed mirrors the core Water Supply System (WSS) structures such as Grid-Iron, Ring, Radial, and Dead-end, conceptually similar to computer network topologies. My analysis utilizes explicitly Line, Star, and Bus topologies to simulate various WSS scenarios. These topologies, characterized by industry-recommended water tanks, pipes, pumps, and reservoir configurations, offer diverse data sets for my experiment. Although I haven't selected every variable for the experiment, only those with high variability in continuous time series are selected since I focus on collecting time series variables. Each topology contributes unique data points, enhancing the complexity and realism of the generated synthetic data. They are briefly discussed as follows:

1. **Line Topology:** This topology (Figure 7.2a) features point-to-point connections between tanks, enabling the study of linear water flow systems. Equipped with sensors for real-time data collection on water level, nitrate, pH, and temperature, the Line topology provides a foundational dataset on linear water distribution patterns.

2. **Bus Topology:** The Bus topology (Figure 7.2b), with a central pipe distributing water to multiple tanks, simulates branched water distribution networks. This setup produces complex, multi-directional water flow scenarios.

3. **Star Topology:** The Star topology emulates radial water supply systems (Figure 7.2c) and offers data on centralized distribution networks. The diversity in tank sizes and connections in this topology enriches the dataset.

EPANET Simulation: BATADAL

My research utilizes a simulated dataset, called BATADAL[7], designed using EPANET (Taormina et al. [8]), which features a C-Town virtual city's WDS. This simulated environment, de-

[7] https://www.batadal.net/data.html

picted in Figure (as depicted in Figure 7.3a), is characterized by its intricate infrastructure consisting of 429 pipes, 388 junctions, 7 storage tanks, 11 pumps, 5 valves, and a reservoir. This dataset provides a rich ground for testing and enhancing my synthetic data generation and evaluation methodologies.

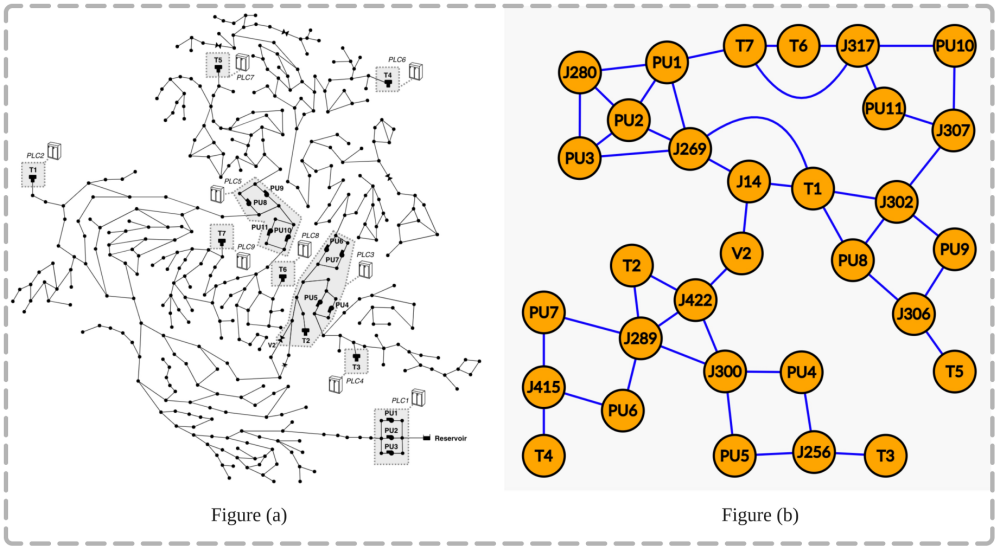


Figure 7.3: WDS Nodes Representation Sikder et al. [5] - (a): Nodes Layout of a Virtual Town Distribution Network; (b): Reduced Nodes (31 Nodes)

The virtual town "C-Town" leverages a sophisticated Supervisory Control and Data Acquisition (SCADA) system for data collection and monitoring via the EPANET tool. This setup is pivotal in capturing time-series data reflecting the system's performance under various operational scenarios, including labeled physical anomalies. The SCADA system's detailed data on hydraulic components and their operations is essential for my study, providing a baseline for generating synthetic scenarios.

The primary functionality of the C-Town WDS is its seven tanks (T1-T7) and five pumping stations (S1-S5). The stations are central to the water distribution and storage processes, each comprising a valve and eleven pumps. Additionally, the system incorporates nine Programmable Logic Controllers (PLCs) located near control components, which relay oper-

ational data to the SCADA system. The interplay between these elements, including water levels, flow rates, and pump operations, forms a comprehensive dataset for my synthetic data generation and analysis.

Focusing on the first dataset of the BATADAL series, my study examines 12 months of operation without intrusion events. This dataset, critical for understanding the normal operational baseline of the WDS, includes 44 features across 8,762 data samples. The comprehensive nature of this dataset provides a robust foundation for developing and validating my GAN-based approaches to synthetic data generation and evaluation.

Real-world Water Plant SCADA Dataset

My research employs a third and final dataset from a real-world Wastewater Treatment Plant (WWTP). Due to confidentiality constraints, the specific identity of the WWTP remains undisclosed. This dataset represents the plant's daily processing capacity, handling massive

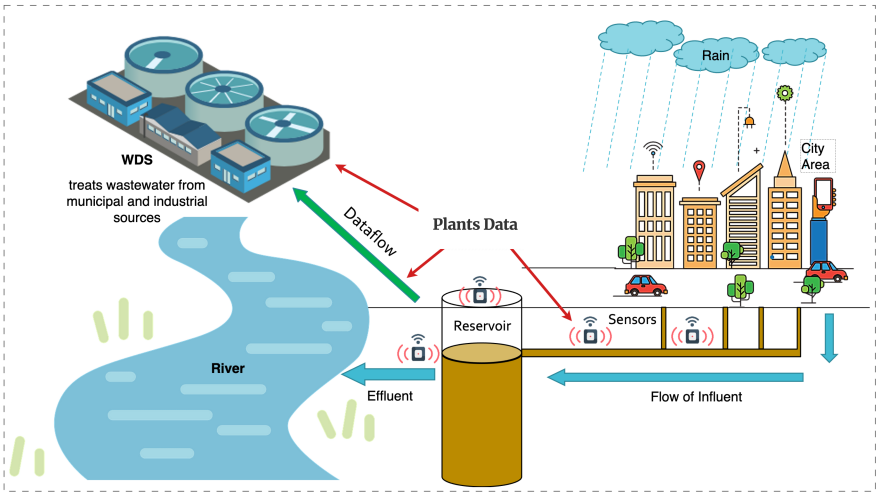


Figure 7.4: A Real-world Waste Water Treatment Plant Process Sikder et al. [5]

amounts of wastewater. The data spans from March 1st, 2018, to March 26th, 2022, offering a detailed and extensive view of the plant's operations, recorded at five-minute intervals.

The WWTP dataset contains a total dimension of 1458 columns and 2,569,464 rows. This extensive dataset is categorized into six distinct operational aspects:

1. Principal inflows to the tunnel system.
2. Overflow incidents from the tunnel to the river.
3. Readings from level sensors within the tunnel.
4. Rainfall measurements.
5. Data from flow meters linked to the tunnel's dewatering pumps.
6. Other critical flows within the main plant.

This rich dataset is instrumental for my research, offering an extensive range of operational parameters. However, approximately 95% of the data consists of 'NA' values, underscoring the need for comprehensive data preprocessing to extract meaningful insights. Specific data subsets, such as pump usage, tunnel overflow incidents, and water mass measurements, are emphasized in my experiments. This subset yields an essential understanding of the WWTP's efficiency and the complexities of its operations, forming an integral part of my study's multivariate time series data.

7.3.2 Generative Adversarial Networks

This section briefly discusses seven GANs, including TimeGAN, CTGAN, WGAN, WGAN-GP, DRAGAN, Cramer GAN, and DoppleGANger, and their high-level architecture.

TimeGAN

TimeGAN generates sequential data while preserving temporal dynamics. It comprises an embedding network, a recovery network, a generator, and a discriminator. The embedding network learns to represent time-series data in a latent space. The generator produces realistic synthetic time-series data, while the discriminator distinguishes between original and synthetic data. A key feature of TimeGAN is its use of a supervised loss to ensure that the generated sequences follow the temporal dynamics of the original data.

$$\mathcal{L} = \mathcal{L}_{\text{unsupervised}} + \lambda \times \mathcal{L}_{\text{supervised}} \tag{7.1}$$

$$\mathcal{L}_{\text{unsupervised}} = \mathbb{E}_{\mathbf{X} \sim p_{\text{data}}}[\log D(\mathbf{X})] + \mathbb{E}_{\mathbf{Z} \sim p_{\mathbf{Z}}}[\log(1 - D(G(\mathbf{Z})))] \tag{7.2}$$

$$\mathcal{L}_{\text{supervised}} = \mathbb{E}_{(\mathbf{X},\mathbf{Y}) \sim p_{\text{data}}}[\|\mathbf{Y} - E(G(\mathbf{X}))\|_2] \tag{7.3}$$

Here, λ is a hyperparameter that balances the unsupervised and supervised losses, E represents the embedding network, G the generator, and D the discriminator.

CTGAN (Conditional Tabular GAN)

CTGAN generates synthetic tabular data with a focus on handling discrete, continuous, and mixed-type data. It uses conditional generators and a novel training procedure to handle class imbalance and mode collapse issues. CTGAN introduces a conditional vector that allows the model to generate data conditioned on specific attributes, helping in generating diverse and representative samples.

$$\mathcal{L}G = -\mathbb{E}\mathbf{z} \sim p_{\mathbf{z}}, \mathbf{c} \sim p_{\mathbf{c}}[\log D(G(\mathbf{z}, \mathbf{c}))] \tag{7.4}$$

$$\mathcal{L}D = -\mathbb{E}\mathbf{x} \sim p_{\text{data}}[\log D(\mathbf{x})] - \mathbb{E}\mathbf{z} \sim p\mathbf{z}, \mathbf{c} \sim p_{\mathbf{c}}[\log(1 - D(G(\mathbf{z}, \mathbf{c})))] \tag{7.5}$$

Here, G is the generator, D is the discriminator, $\mathbf{z}$ is the noise vector, and $\mathbf{c}$ is the conditional vector.

WGAN (Wasserstein GAN)

WGAN introduces the Wasserstein distance as a loss function to address the mode collapse and training instability issues in GANs. This approach modifies the traditional GAN's discriminator to become a critic that estimates the Wasserstein distance between the original and generated distributions. The critic is trained to maximize this distance, while the generator aims to minimize it.

$$\mathcal{L} = \min_{G} \max_{D \in \mathcal{D}} \mathbb{E}\mathbf{x} \sim p\text{data}[D(\mathbf{x})] - \mathbb{E}\mathbf{z} \sim p\mathbf{z}[D(G(\mathbf{z}))] \tag{7.6}$$

Here, $\mathcal{D}$ denotes the set of 1-Lipschitz functions, G is the generator, D is the discriminator (or critic), and $\mathbf{z}$ is the noise vector.

WGAN-GP (Wasserstein GAN with Gradient Penalty)

WGAN-GP is an improvement over WGAN that uses a gradient penalty term to enforce the Lipschitz constraint, which is crucial for the Wasserstein distance calculation. This modification stabilizes training and improves the quality of generated samples.

$$\mathcal{L} = \min_{G} \max_{D} \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}}[D(\mathbf{x})] - \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}}[D(G(\mathbf{z}))] + \lambda \mathbb{E}_{\hat{\mathbf{x}} \sim p_{\hat{\mathbf{x}}}}[(\|\nabla_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}})\|_2 - 1)^2] \tag{7.7}$$

Here, $\hat{\mathbf{x}}$ is sampled uniformly along straight lines between pairs of points sampled from the

data distribution p_{data} and the generator distribution p_g, and λ is the penalty coefficient.

5. DRAGAN (Deep Regret Analytic GAN): DRAGAN aims to improve training stability by regularizing the gradient norm of the discriminator's output with respect to its input. This is particularly effective in preventing mode collapse, ensuring a more diverse generation.

$$\begin{aligned}\mathcal{L}_D = &- \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})] - \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\log(1 - D(G(\mathbf{z})))] \\ &+ \lambda \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[(\|\nabla_{\mathbf{x}} D(\mathbf{x})\|_2 - 1)^2]\end{aligned} \tag{7.8}$$

Here, λ is a regularization coefficient.

Cramer GAN

Cramer GAN uses the Cramer distance as a loss function, offering a more robust metric for distribution comparison. This approach helps better capture the diversity of the data distribution and stabilize the training process.

$$\mathcal{L} = \min_G \max_D \mathbb{E}_{\mathbf{x},\mathbf{x}'\sim p_{\text{data}}}[\|D(\mathbf{x}) - D(\mathbf{x}')\|] - \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{z}\sim p_{\mathbf{z}}}[\|D(\mathbf{x}) - D(G(\mathbf{z}))\|] \tag{7.9}$$

Here, D is the discriminator, G is the generator, and $\mathbf{z}$ is the noise vector.

DoppelGANger

DoppelGANger generates high-dimensional, mixed-type sequential data. It uses two generators: one for generating feature vectors and another for generating time sequences. This architecture allows it to capture complex relationships and dependencies in the data.

$$\mathcal{L} = \mathcal{L}_{\text{feature}} + \mathcal{L}_{\text{time}} \tag{7.10}$$

$$\begin{aligned}\mathcal{L}_{\text{feature}} = \min_{G_{\text{feature}}} \max_{D_{\text{feature}}} \Big(& \mathbb{E}_{\mathbf{x}_{\text{feature}} \sim p_{\text{data}}}[D_{\text{feature}}(\mathbf{x}_{\text{feature}})] \\ & - \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}}[D_{\text{feature}}(G_{\text{feature}}(\mathbf{z}))] \Big)\end{aligned} \tag{7.11}$$

$$\mathcal{L}_{\text{time}} = \min_{G_{\text{time}}} \max_{D_{\text{time}}} \mathbb{E}_{\mathbf{x}_{\text{time}} \sim p_{\text{data}}}[D_{\text{time}}(\mathbf{x}_{\text{time}})] - \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}}[D_{\text{time}}(G_{\text{time}}(\mathbf{z}))] \tag{7.12}$$

Here, G_{feature} and G_{time} are the feature

7.4 Experimental Design

This section explores the methods used to qualitatively and quantitatively evaluate the utility of GAN-generated synthetic data from three datasets. Recognizing the complexity and multidimensionality of water systems data, I analyze four key metrics: Diversity Assessment, Fidelity Evaluation, Usefulness Analysis, and Correlation Analysis. I have carefully selected these metrics to thoroughly investigate how well the synthetic data from my suite of GAN models—TimeGAN, CTGAN, WGAN, WGAN-GP, DRAGAN, Cramer GAN, and DoppelGANger—replicate the characteristics and dynamics of water systems data. My experimental design, which combines quantitative and qualitative methods, aims to comprehensively understand how well these models perform and their applicability in replicating and utilizing complex water systems data.

7.4.1 Diversity Assessment

Diversity assessment includes visual and quantitative techniques to evaluate the distributional similarity of synthetic samples to original data. I use PCA (Tipping and Bishop [420]) and t-SNE (Van der Maaten and Hinton [400]) visualizations to compare the overlap of two distinctly colored clusters—each representing the original and synthetic data. Though distinct in operational mechanisms, PCA and t-SNE are dimension-reduction techniques that collectively offer a multi-faceted view of the data's topological structure. PCA preserves the variance within the data, highlighting the principal components that account for significant variances (exceeding 70%). In contrast, t-SNE focuses on maintaining the relationships between data points in a reduced dimensional space, an attribute that makes it particularly adept at visualizing high-dimensional datasets.

Evaluation Metrics

Quantitatively, I calculate the Centroid Distance (CD) and Nearest Neighbor Distance (NND) among the principal components for both PCA and t-SNE. This step is important in quantifying the spatial distributional characteristics of the water data. Additionally, I employ a k-means clustering approach and compare Cluster Entropy (CE) between the original and synthetic datasets enables me to estimate the diversity and representation of data.

Mathematically, CD (CD) is calculated as follows:

$$CD = \frac{1}{N} \sum_{i=1}^{N} \|x_i - c_i\| \tag{7.13}$$

where N is the number of data points in the cluster, x_i is the data point, and c_i is the centroid of the cluster.

The CD is essential in evaluating the compactness and separation of clusters. It measures the average distance between a cluster's data points and its centroid. A smaller CD indicates a higher density and better-defined cluster, suggesting that synthetic data closely aligns with original data regarding cluster formation. NND complements this by measuring the distance between each data point and its closest neighbor in a different cluster. This metric estimates how well-separated different clusters are, with a larger distance indicating dispersion between clusters.

NND (NND) is calculated as:

$$NND = \frac{1}{N}\sum_{i=1}^{N}\min_{j\neq i}\|x_i - x_j\| \tag{7.14}$$

where N is the number of data points, x_i is the ith data point, and x_j is its nearest neighbor in a different cluster.

I also apply the Interquartile Range (IQR) of distances to provide insights into the clusters' variability. A smaller IQR suggests that most data points are closely packed, indicating uniformity in the synthetic data's distribution relative to the original data.
Mathematically, IQR is calculated as the difference between the third quartile ($Q3$) and the first quartile ($Q1$):

$$IQR = Q3 - Q1 \tag{7.15}$$

The rationale behind employing a k-means clustering (Arthur and Vassilvitskii [421]) is its efficiency and effectiveness in partitioning the data into distinct clusters. By comparing CE – a measure of the randomness or unpredictability in the cluster assignments – between the

original and synthetic datasets, I aim to determine how well synthetic data preserves the inherent groupings and structures present in the original dataset. Higher similarity in CE indicates that synthetic data has successfully captured the complex, underlying patterns of the original data, affirming its utility and fidelity in representing real-world scenarios.

Mathematically, CE (CE) is calculated as:

$$CE = -\sum_{i=1}^{k} p_i \cdot \log(p_i) \tag{7.16}$$

where k is the number of clusters and p_i is the proportion of data points in the ith cluster.

7.4.2 Fidelity Estimation

I evaluate fidelity by determining if generated time series data could be differentiated from the original data. I design an Original vs. Synthetic classification model pipeline, in which each data batch is labeled as either 'original' or 'synthetic'. The data are partitioned for training and validating purposes, with 80% allocated for training and the remaining 20% for testing. Subsequently, I have a GRU classifier (Cho et al. [335]), a variant of recurrent neural networks renowned for their efficiency in classifying sequence data. Unlike traditional recurrent neural networks, GRUs are equipped with 'gates' that regulate the flow of information. These gates effectively manage the model's ability to retain or discard information across different time steps, making GRUs adept at capturing temporal dependencies and patterns in sequential data.

Evaluation Metrics

The performance of the synthetic data is inversely related to the classifier's accuracy in this test; a lower accuracy rate indicates higher fidelity in the synthetic data, meaning the GRU classifier has difficulties distinguishing it from the original data. Given the GRU algorithm's advanced capabilities in handling time series data, the model learns and classifies complex patterns over a series of epochs. Therefore, I quantify the model's learning efficacy and speed by monitoring the number of epochs required for the validation accuracy to reach specific thresholds: 80%, 90%, and 100%, where applicable. This approach not only evaluates the immediate performance of the GRU model but also provides deeper insights into the temporal dynamics and intricacies captured within the data. It is a powerful measure to understand how synthetic data mirrors original data, emphasizing the GRU model's pivotal role in my classification task.

7.4.3 Usefulness Analysis

This technique determines whether the synthetic data could parallel the utility of original data in predictive tasks. I compare the performance of a sequence prediction model under four scenarios: Train on Original, Test on Original (TOTO); Train on Original, Test on Synthetic (TOTS); Train on Synthetic, Test on Original (TSTO); and Train on Synthetic, Test on Synthetic (TSTS). Each of these scenarios serves a specific purpose in my analysis. The TOTO test is designed to establish a baseline for the efficiency of my classifier, which is the GRU model, as previously discussed. This setup compares the model's performance under conventional conditions with original data. In contrast, the TOTS test evaluates the classifier's ability to discern original data when tested against synthetic data, determining whether the synthetic data can be mistaken for original data. The TSTO scenario shifts the

focus to training, examining the viability of substituting original training data with synthetic data and its impact on model performance when tested on original data. Lastly, the TSTS test extends this concept to training and testing, probing the feasibility of using synthetic data as a complete replacement for original datasets. Those four tests combined provide key insight into understanding the practicality and adaptability of synthetic data in real-world scenarios. It assesses the immediate utility of the synthetic data and its potential to serve as a viable alternative or complement to original data in various applications.

Evaluation Metrics

To facilitate a systematic comparison of the test results derived from the four scenarios across different GAN models, I devised a meticulous approach to presenting my findings. I construct four distinct plots in one grid for each synthetic data generated by the various GANs. These plots depict the progression of the Mean Absolute Error (MAE) (Equation 7.17) during both the training and validation phases. This visual representation enables an immediate and clear understanding of how the MAE decreases over time, highlighting the learning efficiency and accuracy of the models under each prediction condition. Furthermore, I record the minimum MAE (Equation 7.18) achieved in each task, allowing me to compare the performance of different GAN-generated datasets quantifiably.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{7.17}$$

$$MAE_{\text{min}} = \min\left(\text{MAE}_{\text{training}}, \text{MAE}_{\text{validation}}\right) \tag{7.18}$$

7.4.4 Correlation Matrix Investigation

I analyze the synthetic data's ability to preserve the original dataset's spatio-temporal dependencies by comparing the correlation matrices within selected features of both the original and synthetic datasets. Such a comparison is important in evaluating the strength and consistency of the interrelationships among these features, thereby providing contextual insights into the extent to which the synthetic data sustains the intrinsic properties of the original dataset. I display the correlation matrix using heatmaps annotated by correlation coefficients. This method offers an intuitive understanding of the correlations, facilitating an easy comparison between the original and synthetic datasets.

Evaluation Metrics

I adopt the Mean Squared Error (MSE) between the correlation matrices to quantitatively measure the deviation between the original and synthetic data's correlation structures. Mathematically, the MSE between two correlation matrices C_{original} and $C_{\text{synthetic}}$ is defined as:

$$MSE = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(C_{\text{original}}(i,j) - C_{\text{synthetic}}(i,j) \right)^2 \tag{7.19}$$

where n is the size of the correlation matrices. A lower MSE value indicates a higher similarity in the synthetic data, signifying a more accurate replication of the complex interrelationships present in the original dataset.

Building on my rigorous evaluation of data generation quality, I introduce another comparison where Table 7.2 compares seven GAN models' training times across three datasets. Tabular GANs such as WGAN, CTGAN, DRAGAN, Cramer GAN, and WGAN-GP demon-

strate much faster training times, with WGAN-GP being notably the fastest. Conversely, TimeGAN incurs over 1000 minutes of training time for each dataset, underscoring its substantial computational demands for time series data. Meanwhile, DoppelGANger's efficiency is on par with tabular GANs despite the complexity of the data. The training durations underscore the variability and efficiency of each GAN model, with tabular models generally offering time-saving advantages.

Table 7.2: Model Training Time Difference for the 3 Datasets and 7 GANs

GAN Model	Type	Dataset		
		ACWA (minutes)	BATADAL (minutes)	Real-world (minutes)
WGAN	Tabular	20	77	78
CTGAN		2.7	12	13.7
DRAGAN		2.85	12.4	15
Cramer GAN		3.4	15.35	18
WGAN-GP		0.23	1.3	1.5
TimeGAN	Time Series	1046	1320	1400
DoppelGANger		10.2	13.2	12.6

7.5 Experimental Results and Analysis

This section presents the experimental results for the synthetic multivariate time-series data, as per the evaluation metrics outlined in the preceding section. Please refer to Appendix A-Tables B.1, B.2, and B.3 for detailed training parameters for all seven GANs.

7.5.1 Diversity Assessment

To measure diversity, I aim to align the distribution of synthetically generated samples as closely as possible with the original data in PCA and t-SNE visualizations. In Figure 7.5, I illustrate this comparison using the TimeGAN models trained on both ACWA testbed and BATADAL datasets. Additionally, I have measured metrics such as CD, NND, IQR, and CE to quantify diversity in all three datasets, as presented in Tables 7.3, 7.4, and 7.5.

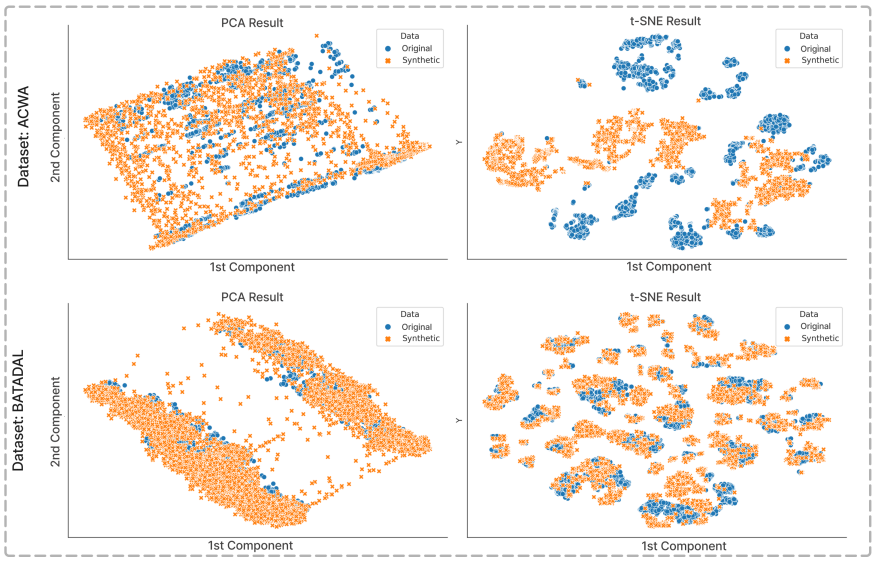


Figure 7.5: Visualization of PCA and t-SNE on ACWA and BATADAL Datasets after Applying TimeGAN

The CD metric, applicable to both PCA and t-SNE, gauges the proximity of generated data to the original distribution. Lower values in GANs, particularly CTGAN, and DoppelGANger, suggest more realistic data generation and accurate cluster formation. The NND metric, evaluating cluster compactness, also shows CTGAN and DoppelGANger excelling in t-SNE with tighter clusters indicated by lower values. Additionally, the IQR of Distances in PCA highlights uniform data generation, with DRAGAN, Cramer GAN, and TimeGAN displaying lower values for consistent distribution. Complementing these metrics, the CE metric quantifies clustering randomness, with similar entropy levels in synthetic and original data denoting comparable characteristics. TimeGAN, in particular, shows minimal entropy differences, closely mirroring the original data.

For the ACWA dataset, in Table 7.3, WGAN performs best among all seven GANs, closely mimicking the original data distribution, as indicated by the lowest CD in PCA. DoppelGANger also performs well, especially regarding the t-SNE CD metric, demonstrating its effectiveness in capturing the original data distribution in a different dimensional space.

DRAGAN shows good consistency in data generation, as indicated by its low IQR. On the other hand, WGAN-GP, TimeGAN, and DoppelGANger tie for the best performance among all other GANs in terms of PCA CE metric, presenting realistic data generation. Overall, DoppelGANger might be slightly favored for the ACWA due to its excellent CD and CE metrics performance.

Table 7.3: Diversity Test on the Physical Testbed-ACWA Data

Metrics for Diversity Assessment	**CTGAN**	**WGAN**	**DRAGAN**	**WGAN-GP**	**Cramer GAN**	**TimeGAN**	**DoppelGANger**
CD (PCA)	0.120	**0.016**	0.089	0.105	0.130	0.157	0.084
NND (PCA)	**0.025**	0.050	0.028	0.034	0.036	0.035	0.033
IQR of Distances (PCA)	0.026	0.055	**0.025**	0.032	0.031	0.039	0.034
CE (Original, PCA)	0.691	0.691	0.691	0.691	0.691	0.691	0.691
CE (Synthetic, PCA)	0.657	0.693	0.689	**0.690**	0.693	**0.692**	**0.692**
CD (t-SNE)	16.410	51.953	47.654	46.378	28.320	17.387	**10.370**
NND (t-SNE)	**3.900**	34.001	31.387	23.128	30.110	14.402	4.156

For the BATADAL dataset, in Table 7.4, DoppelGANger excels with the lowest CD in PCA, indicating its effective mimicry of the original data distribution. TimeGAN shows exceptional results with the lowest NND and IQR in PCA, demonstrating its ability to preserve data diversity and consistency. In t-SNE analysis, CTGAN and TimeGAN lead with the lowest CD and NND, respectively, highlighting their strong performance in different dimensional reductions. Overall, TimeGAN demonstrates remarkable effectiveness in generating diverse and realistic synthetic samples using simulated data.

Table 7.4: Diversity Test on BATADAL- EPANET Data

Metrics for Diversity Assessment	**CTGAN**	**WGAN**	**DRAGAN**	**WGAN-GP**	**Cramer GAN**	**TimeGAN**	**DoppelGANger**
CD (PCA)	0.074	0.473	0.559	0.194	0.510	0.042	**0.041**
NND (PCA)	0.493	0.800	0.819	0.549	0.792	**0.128**	0.361
IQR of Distances (PCA)	0.252	0.109	0.099	0.203	0.097	**0.062**	0.268
CE (Original, PCA)	1.565	1.565	1.565	1.565	1.565	1.565	1.565
CE (Synthetic, PCA)	1.594	1.577	1.603	1.598	1.579	**1.576**	1.584
CD (t-SNE)	**1.898**	34.268	26.012	16.972	30.288	3.135	1.975
NND (t-SNE)	4.863	38.528	32.963	28.435	37.056	**1.871**	7.126

For the real-world plant dataset, in Table 7.5, TimeGAN stands out with the lowest CD and NND in PCA, indicating its superior capability in mimicking the original data distribution and preserving data diversity. In the t-SNE analysis, WGAN shows the lowest CD, while DoppelGANger leads in NND. TimeGAN's exceptional performance is further underscored in PCA's CE, where it closely matches the original data, signifying realistic data generation. These results suggest that TimeGAN is particularly adept at handling the complexities of real-world plant data compared to the other GANs.

Table 7.5: Diversity Test on Real-world Plant Data

Metrics for Diversity Assessment	**CTGAN**	**WGAN**	**DRAGAN**	**WGAN-GP**	**Cramer GAN**	**TimeGAN**	**DoppelGANger**
CD (PCA)	0.365	1.336	1.250	1.021	1.140	**0.062**	0.333
NND (PCA)	0.149	0.327	0.250	0.309	0.322	**0.038**	0.102
IQR of Distances (PCA)	0.109	0.271	0.170	0.081	0.189	**0.031**	0.074
CE (Original, PCA)	0.826	0.826	0.826	0.826	0.826	0.826	0.826
CE (Synthetic, PCA)	0.961	1.062	1.094	1.050	1.077	**0.901**	0.956
CD (t-SNE)	75.068	**68.059**	71.781	73.836	75.539	78.118	84.287

Continued on the next page

Table 7.5 – Continued from previous page

Metrics for Diversity Assessment	CTGAN	WGAN	DRAGAN	WGAN-GP	Cramer GAN	TimeGAN	DoppelGANger
NND (t-SNE)	40.058	46.004	41.286	44.203	56.012	44.527	**38.166**

7.5.2 Fidelity Assessment

In assessing infidelity, the goal is to demonstrate that synthetic data is indistinguishable from the original dataset. I use a GRU classifier to classify original or synthetic data to achieve this. Ideally, I want to see if the RNN struggles to classify correctly, suggesting that the synthetic data closely resembles the original data.

In Figure 7.6, I compare the AUC scores and ROC curves for the GRU model on the ACWA and BATADAL datasets. Moreover, fidelity is quantified across three distinct datasets, as detailed in Tables 7.6, 7.7, and 7.8. A visual examination of Figure 7.6 reveals the GRU's inferior performance on the test set, suggesting the synthetic datasets effectively deceive the classifier. This indicates a high level of similarity between the synthetic and original datasets.

In the fidelity assessment of ACWA data, as shown in Table 7.6, different GANs exhibit varying speeds in reaching accuracy thresholds.

Table 7.6: Fidelity Assessment on Physical Testbed Data (ACWA)

Metric: Epoch Where Accuracy First Reaches	CTGAN	WGAN	DRAGAN	WGAN-GP	Cramer GAN	TimeGAN	DoppelGANger
80%	186	78	77	93	88	153	194
90%	258	85	85	105	108	166	209
100%	N/A	131	119	137	126	227	215

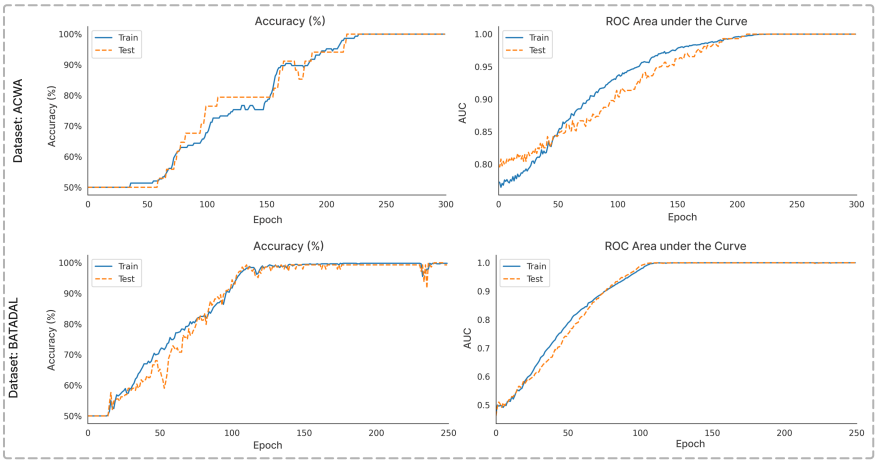


Figure 7.6: Accuracy and AUC Scores on ACWA and BATADAL Dataset after Applying TimeGAN

DoppelGANger, for instance, is slower, achieving 80% accuracy at epoch 194 and 90% at epoch 209. However, TimeGAN took the longest to reach 100% accuracy, achieving it at epoch 227. Contrarily, CTGAN did not reach 100% accuracy within the observed epochs. Overall, CTGAN, TimeGAN, and DoppleGANger take longer to reach full accuracy, demonstrating the similarity between the synthetic and original datasets.

For the BATADAL Data, as illustrated in Table 7.7, TimeGAN takes the longest to achieve 80% and 90% accuracy, at epochs 92 and 101 respectively, and does not reach 100% accuracy, suggesting its synthetic data closely mimics the original. In contrast, WGAN-GP and DRAGAN, which reach accuracy thresholds relatively quickly, may produce data that is easier for the classifier to distinguish from the original, indicating less fidelity.

Table 7.7: Fidelity Assessment on BATADAL EPANET Data

Metric: Epoch Where Accuracy First Reaches	CTGAN	WGAN	DRAGAN	WGAN-GP	Cramer GAN	TimeGAN	DoppelGANger
80%	29	23	23	19	26	92	45
90%	37	24	26	27	28	101	57
100%	64	40	33	56	47	N/A	91

For Real-world Plant data, as detailed in Table 7.8, the performance of TimeGAN is notably distinct across different accuracy thresholds. When measuring the time required to reach 80% accuracy, TimeGAN takes the longest, achieving this milestone at epoch 46. This trend of TimeGAN being the slowest continues at the 90% accuracy level, reaching epoch 54. The pattern is consistent even when the benchmark is elevated to 100% accuracy, indicating high fidelity.

Table 7.8: Fidelity Assessment on Real-world Plant Data

Metric: Epoch Where Accuracy First Reaches	CTGAN	WGAN	DRAGAN	WGAN-GP	Cramer GAN	TimeGAN	DoppelGANger
80%	25	29	28	17	28	46	31
90%	29	29	28	18	28	54	35
100%	40	30	33	32	45	111	57

Overall, in this assessment, TimeGAN presents high fidelity in data generation. It consistently records the highest epoch values at all three accuracy levels—80%, 90%, and 100%, demonstrating its suitability and effectiveness across all selected categories of datasets.

7.5.3 Usefulness Estimation

In this evaluation, I assess whether synthetic data are sufficiently useful to replace original data for AI model training and testing. Among the four tests, I primarily focus on TOTS and

TSTO, as these scenarios effectively demonstrate the ability of synthetic data to substitute original data in training and testing AI models. Figure 7.7 presents the loss convergence for all four scenarios on ACWA datasets, comparing original and synthetic datasets. Upon visual inspection, I observed that the testing accuracies closely match the training accuracies, indicating that the synthetic dataset generated by TimeGAN using ACWA can effectively replace the original one.

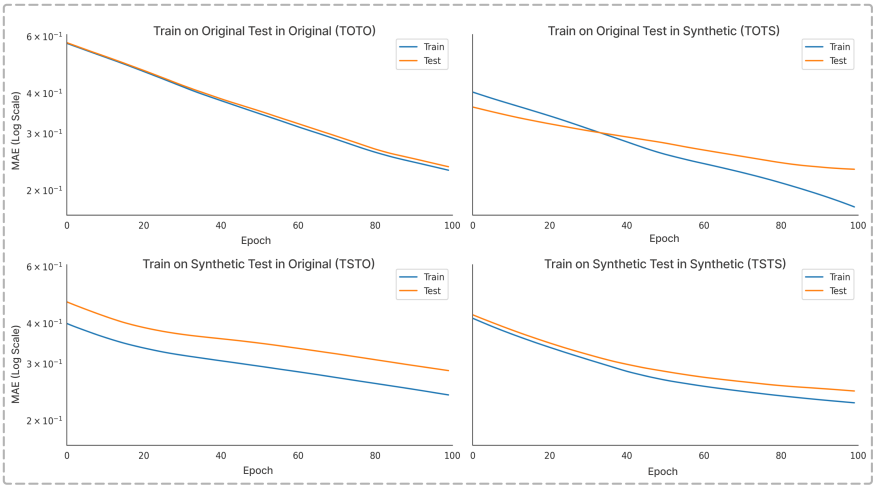


Figure 7.7: Train and Test Loss for TOTO, TOTS, TSTO, TSTS on ACWA Data After Applying TimeGAN

For ACWA Data, in the table (Table 7.9), all models have an identical lowest validation loss for the TOTO scenario. To determine the better GAN, I look at the performance across the remaining three scenarios, TOTS, TSTO, and TSTS. For TOTS, CTGAN has the lowest validation loss, indicating that it can generate synthetic data that closely resembles the distribution of the original test data when the model is trained on original data. In the TSTO scenario, which tests the model's ability to generalize from synthetic to original data, CTGAN outperforms all remaining models. For TSTS, Cramer GAN exhibits the best performance with the lowest validation loss, suggesting that it is particularly adept at generating consistent synthetic data that is useful for both training and testing.

Overall, for physical testbed data, when considering the usefulness of synthetic data for training and testing purposes, Cramer GAN stands out in the TSTS scenario, which is a strong indicator of the quality of the synthetic data it generates. This could imply that Cramer GAN's data are remarkably coherent and may contain patterns that benefit the model in learning and performing well when the test data are also synthetic. However, CTGAN appears to be the most versatile, performing best in the TOTS scenario and second-best in the TSTS scenario, indicating good performance in generating synthetic data for testing and the complete cycle of training and testing. Choosing the better GAN will depend on the specific use case. If the priority is on using synthetic data for model validation (TOTS), CTGAN would be preferable. However, if the focus is on the entire process of training and testing models on synthetic data (TSTS), Cramer GAN would be the choice.

Table 7.9: Usefulness Evaluation on Physical Testbed Data (ACWA)

Metric: Lowest Validation Loss	**CTGAN**	**WGAN**	**DRAGAN**	**WGAN-GP**	**Cramer GAN**	**TimeGAN**	**DoppelGANger**
TOTO	0.237	0.237	0.237	0.237	0.237	0.237	0.237
TOTS	**0.163**	0.223	0.201	0.231	0.183	0.233	0.188
TSTO	**0.263**	0.369	0.337	0.349	0.326	0.286	0.264
TSTS	0.179	0.164	0.133	0.124	**0.100**	0.247	0.191

For BATADAL Data, evaluating the performance of various GANs using Table 7.10, I see a nuanced picture of strengths and weaknesses across different scenarios. The CTGAN shows moderate uniform performance, not excelling in any particular category but not falling behind drastically in any. This suggests consistency in its output, but it lacks a clear advantage. The WGAN stands out in two scenarios— TOTS and TSTS. This indicates WGAN's robust ability to generate highly useful synthetic data that can serve well both as a substitute for original data in testing scenarios and as a source for training models that

perform competently on unseen data. WGAN-GP excels distinctly in the scenario where original data are used for TOTS, showcasing a particular strength in creating synthetic data that behaves similarly to original data under a testing environment. This desirable trait suggests that WGAN-GP's synthetic data can effectively represent real-world conditions in test cases. In contrast, TimeGAN shows its prowess when synthetic data are used for TSTO. This indicates TimeGAN's synthetic data quality, demonstrating an excellent generalization to original data, an essential characteristic if the end goal is to apply the trained model to real-world situations.

Table 7.10: Usefulness Evaluation on BATADAL EEPANET Data

Metric: Lowest Validation Loss	**CTGAN**	**WGAN**	**DRAGAN**	**WGAN-GP**	**Cramer GAN**	**TimeGAN**	**DoppelGANger**
TOTO	0.223	0.223	0.222	0.223	0.223	0.223	0.223
TOTS	0.218	0.186	0.187	**0.144**	0.176	0.233	0.222
TSTO	0.227	0.279	0.281	0.250	0.277	**0.222**	0.234
TSTS	0.213	0.103	0.103	0.103	0.111	0.243	0.202

Considering simulated data, if the priority is to have a GAN that generates data capable of training models that perform well on original data, TimeGAN would be the ideal choice. However, if the goal is to use synthetic data extensively for training and testing, the WGAN presents the most efficient option, given its superior performance in those scenarios. For applications where the synthetic data are primarily used for testing against models trained on original data that is, WGAN-GP might be the GAN of choice, given its exceptional performance in that specific scenario.

For Real-world Plant data, in table 7.11, CTGAN exhibits relatively low validation loss in the TOTO scenario, a standard benchmark since it represents training and testing on original data. However, its performance in the other scenarios could be more competitive. The WGAN shows moderate performance in the TOTO and TSTS scenarios but has significantly higher validation losses in the TOTS metric. This suggests less effectiveness in generating

synthetic data for testing against original data. DRAGAN achieves a competitive validation loss in the TSTS scenario. However, like WGAN, it does not perform well in the TOTS scenario, indicating it may not be superior at creating test-ready synthetic data. WGAN-GP, while performing well in the TSTS scenario, indicating good quality synthetic data for both training and testing, shows a higher validation loss in the TOTS scenario. Cramer GAN does not lead in any of the scenarios, indicating that it might not be the optimal choice among the models considered. TimeGAN shows an impressive performance, particularly in the TOTS and TSTS scenarios, suggesting that it is very effective in generating synthetic data useful for training and testing purposes, thus indicating a high degree of usefulness in synthetic data generation. DoppelGANger also has low validation losses in the TSTS scenario and performs reasonably well in the TOTS and TSTO scenarios.

Table 7.11: Usefulness Evaluation on Real-world Plant Data

Metric: Lowest Validation Loss	**CTGAN**	**WGAN**	**DRAGAN**	**WGAN-GP**	**Cramer GAN**	**TimeGAN**	**DoppelGANger**
TOTO	0.026	0.026	0.026	0.026	0.026	0.026	0.026
TOTS	0.208	0.424	0.412	0.285	0.366	**0.076**	0.123
TSTO	0.095	0.092	0.092	0.095	0.097	**0.059**	0.096
TSTS	0.127	0.119	0.102	0.069	0.125	**0.044**	0.049

Considering all the scenarios, TimeGAN stands out as the most suitable model due to its low validation losses when synthetic data are used, especially in TSTS scenario. It demonstrates the ability to generate synthetic data that closely mimics original data and can be used effectively for training and testing classifiers.

7.5.4 Correlation Check

I also analyze whether the synthetic multivariate time series can keep the spatio-dependency of the original one. From both Figure 7.8a and 7.8b, I observe that the synthetic dataset can reasonably preserve spatio-dependency on the ACWA dataset after applying TimeGAN.

In evaluating the performance of various GANs across different datasets, I focus on the MSE between correlations as another essential performance metric (Table 7.12). The lower the MSE value, the better the performance. My analysis reveals the following:

1. On the ACWA Data, DoppelGANger emerged as the most effective GAN, with the lowest MSE value of 0.0051. This suggests that DoppelGANger is the best at capturing and replicating the statistical properties of the dataset compared to the other GANs.

2. On the BATADAL Data, DoppelGANger again presents superior performance with the lowest MSE value of 0.0054. This indicates its consistency and effectiveness in dealing with different types of datasets.

3. On the real-world plant data, DoppelGANger also outperformed other models, with the lowest MSE value of 0.0677. This highlights DoppelGANger's capability in effectively modeling the spatial characteristics specific to the real-world plant dataset.

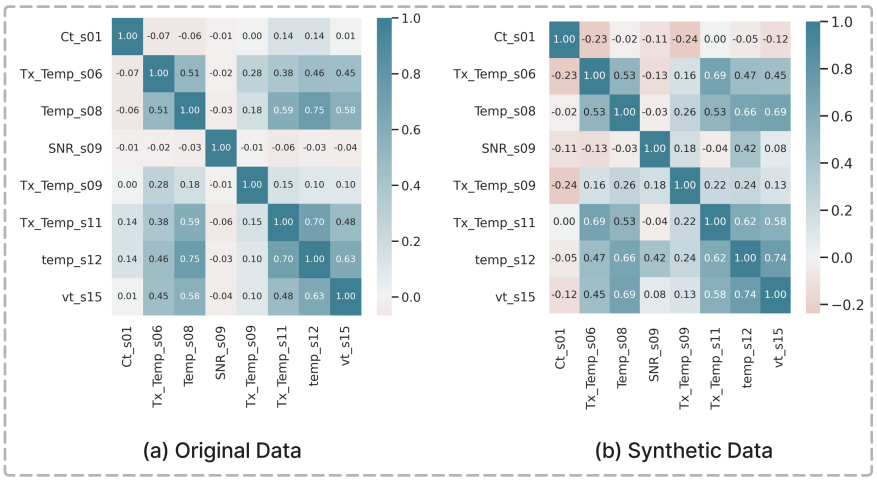


Figure 7.8: Correlation of Multivariate Time-series of Original and Synthetic ACWA dataset after Applying TimeGAN

Table 7.12: MSE between correlation matrices for All Datasets and GANs

Metrics: MSE	**CTGAN**	**WGAN**	**DRAGAN**	**WGAN-GP**	**Cramer GAN**	**TimeGAN**	**DoppelGANger**
ACWA Data	0.105	0.914	0.133	0.034	0.428	0.019	**0.005**
BATADAL Data	0.022	0.469	0.049	0.012	0.175	0.008	**0.005**
DC Water Data	0.132	0.941	0.120	0.145	0.686	0.068	**0.043**

These results underscore the varying effectiveness of different GAN models across distinct datasets, highlighting the importance of model selection based on the specific characteristics and requirements of the data being analyzed.

7.6 Summary and Conclusions

Consider a network of sensors in a lake measuring water pH and temperature; using these GAN models, I generate synthetic data that closely mimics the spatial distribution of water pH and temperatures. Then, I analyze this data using PCA and t-SNE to understand the spatial relationships and to predict how a temperature change in one node might affect nearby nodes, a preeminent aspect of environmental monitoring. My study utilizes PCA and t-SNE to visualize the diversity in synthetic data, with CTGAN and DoppelGANger demonstrating promising results.

My work also assesses the fidelity of synthetic data using a GRU classifier. For instance, TimeGAN demonstrates slower progression to high accuracy, indicating better mimicry and accurate temporal representation of the original data. This model can generate synthetic datasets that closely resemble pollution levels for water quality management, such as in a treatment plant, allowing for the development of more accurate predictive models to ensure

water quality, especially when real-world pollution data are scarce. I find that synthetic data, like that from Cramer GAN and CTGAN, can replace original data in training predictive models. In the context of an urban water distribution network, these GAN models generate data representing various pressure and flow scenarios. I use this synthetic data for emergency response simulations, such as predicting the effects of a main pipe burst or the need for its preventive maintenance, aiding in efficient crisis management and resource allocation.

The correlation analysis in my study highlights the ability of models like DoppelGANger and TimeGAN to preserve spatio-temporal dependencies. Applying this to environmental impact assessments near a river, these models simulate how a new industrial project might affect water quality and/or flow. Synthetic data can assist in predicting environmental impacts, aiding in regulatory compliance and sustainable development planning. The nuanced capabilities of various GAN models identified in my study, such as capturing dataset diversity, fidelity, and usefulness for predictive modeling, directly apply to water resource management. For instance, in regions facing water scarcity, choosing the suitable GAN model based on these insights leads to effective modeling of water usage scenarios, assisting in strategic planning and conservation efforts.

Overall, the findings from my study on GAN models offer valuable insights into the selection and application of these models in water utilities. From temperature monitoring in lakes to predictive modeling in water treatment and distribution and even environmental impact estimation (such as for water-related public policies), choosing a GAN model plays a vital role. I can strategically leverage each model's strengths in fidelity, data mimicry, and spatio-temporal correlation preservation to address specific challenges in water resource management and environmental monitoring on the national and global levels.

Chapter 8

Real World Deployments - Forecasting Model at DC Water

In this Chapter, I detail the deployment process of the cP_2O forecasting model and present the results obtained during real-time operation in a WWTP setting. The deployment aimed to evaluate the model's practical performance and its ability to assist operators in decision-making processes.

8.1 Deployment Steps at DC Water

The forecasting model was deployed within the operational environment of the DC Water treatment facility. The deployment architecture included the following components:

- Data Acquisition System: Real-time data streams from sensors and external sources (e.g., weather stations, river flow gauges) were collected via a SCADA system.
- Processing Server: A dedicated Amazon Web Services (AWS) instance equipped with high-performance computational CPUs hosted the forecasting model and managed data processing tasks.
- Model Integration: cP_2O model was implemented in Python 3.8 using the PyTorch

1.8 DL framework. It was seamlessly integrated into the processing pipeline to receive real-time data inputs and generate forecasts.

- User Interface: An AWS-hosted web application provided a user-friendly interface for monitoring forecasts and interacting with the system.

8.2 Real-Time Forecasting Process

The deployed system operated on a rolling basis, generating forecasts every hour with a 4-hour ahead horizon. The process involved:

1. Data Ingestion: The latest data from internal sensors ($\mathbf{D}_t$) and context variables ($\mathbf{C}_t$) were ingested into the system.
2. Preprocessing: Data were cleaned to handle missing values, and features were scaled based on the training data parameters.
3. Forecast Generation: The cP$_2$O model processed the input data to generate forecasts for the next 4 hours, including prediction intervals.
4. Visualization and Alerts: Forecasts were visualized on the dashboard, and alerts were triggered if predicted water levels exceeded predefined thresholds.

8.3 Deployment Results

The model's performance was monitored over a period of two months during varying operational conditions, including dry weather and heavy rainfall events. The key results are summarized below.

8.3.1 Overall Performance Metrics

The model maintained high accuracy during deployment, with performance metrics consistent with those observed during validation. Table 8.1 presents the aggregated metrics over the deployment period.

Table 8.1: Performance metrics during deployment

Metric	Overall Value	Dry Weather	Rainfall Events
MAPE (%)	2.05	1.80	2.60
RMSE	3.35	3.10	3.90
PDR (%)	95.5	N/A	95.5
Predictive Interval (%)	92.5 ± 1.50	94.0 ± 1.20	90.0 ± 1.80

The model demonstrated slightly higher errors during rainfall events due to increased variability in inflow rates. However, the prediction intervals effectively captured the uncertainty, maintaining a conditional probability close to the desired 90%.

8.3.2 Case Study: Heavy Rainfall and Coastal Flood Events

During two significant flooding events on January 9th and 10th, 2024, at DC Water, the model's ability to forecast inflow surges was critically evaluated. Figure 8.1 displays the actual and forecasted water levels, prediction intervals, and key event annotations. Figure 8.2 presents a dry run day at DC Water.

The model effectively captured the sharp increase in water levels during the events, accurately predicting both the coastal flood peak (6.19' MLLW) and the Rock Creek flood crest (8.04 ft). These predictions provided operators with a 4-hour advance warning, enabling proactive management actions such as adjusting pump operations and diverting flows to storage tunnels. The integration of context variables, particularly during rainfall events,

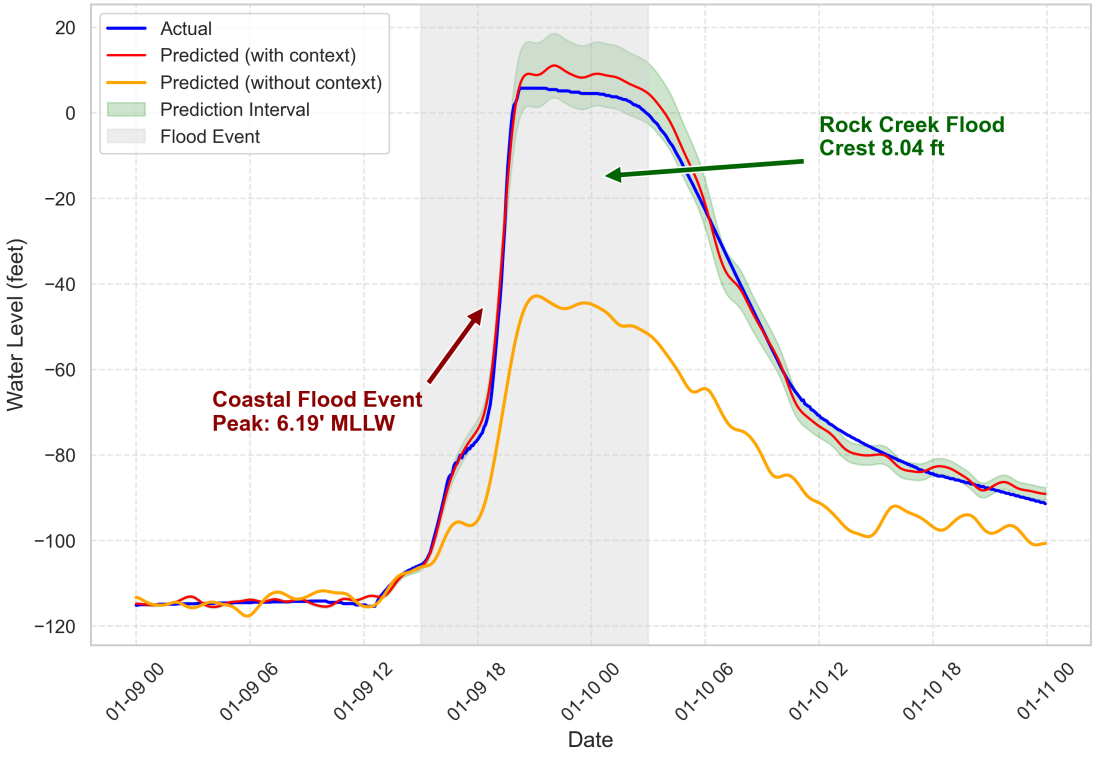


Figure 8.1: Forecasts during a heavy rainfall and coastal flood event. The actual water levels are shown in blue, forecasts with context in red, forecasts without context in orange, and prediction intervals in light green. Shaded areas with gray color indicate flood events.

proved crucial in enhancing prediction accuracy and supporting critical decision-making under extreme weather conditions.

8.4 Challenges and Mitigations

During deployment, several challenges were encountered:

- Data Quality Issues: Occasional sensor malfunctions led to missing or erroneous data. This was mitigated by implementing real-time data validation checks and fallback strategies using historical averages.

Figure 8.2: Deployed model evaluation during a dry day at DC Water

- Model Retraining: To maintain accuracy, the model was retrained weekly using the latest data. Automated retraining pipelines were set up to facilitate this process.
- System Integration: Integrating the model into the existing SCADA system required careful coordination to ensure compatibility and data security.

8.5 Future Improvements

Based on the deployment experience, the following improvements are planned:

- Incorporation of Additional Context Variables: Including more granular weather forecasts and upstream flow data to enhance prediction accuracy during extreme events.
- Enhanced Anomaly Detection: Integrating anomaly detection mechanisms to identify and handle outliers or unexpected patterns in the data.
- Scalability Enhancements: Optimizing the model for deployment across multiple facilities with varying configurations and data sources.

8.6 Conclusion of Deployment

The deployment of the cP_2O model demonstrated its practical applicability and effectiveness in a real-world WWTP environment. The model provided accurate short-term forecasts, aiding operators in making informed decisions and improving operational efficiency. The positive outcomes reinforce the value and need of integrating AI-driven forecasting models into wastewater management systems across the country.

Chapter 9

Discussions and Conclusions

This chapter provides a comprehensive discussion of the key experimental findings, highlighting the strengths and limitations of the research conducted for each project. It also presents the overall conclusions drawn from the studies.

9.1 Model Agnostic Assurance - MAA

In this manuscript, I provide two MAA pipelines for achieving quantifiable assurance goals, including XAI, FAI, TAI, EAI, CAI, and SAI. Although the algorithms are model-agnostic in nature, the use cases are model-specific (SCADA and Telco). ALSP is a model-driven approach that generates quantifiable assurance scores. It leverages game theory, AE, and logging to provide AIA goals; RFSP is a user-driven approach, where user input their expected AIA weights as a form of an equilibrium, the desired optimum set points dictate the final outcomes of assurance. Many works in AI systems management exist Kulkarni et al. [422], but due to the unavailability of benchmark assurance standards, I am unable to compare the results with existing algorithms; nonetheless, in this manuscript, I present multiple empirical outcomes that are deemed successful for that goal. The two use cases presented are for (1) a critical infrastructure: SCADA system, where I explain attack localization as a form of explainability using reconstruction errors from the AE and show that the Secret Inversion algorithm is capable of detecting adversarial inputs; and for a (2) Telco dataset,

I test it by injecting intentional bias and testing if the pipeline detects it and reflects that in AIA scores. The algorithms had different success rates, albeit they all improved on the assurance of the AI systems at hand. Furthermore, potential areas of application include but are not limited to water distribution systems, smart grids, and telecommunication systems. As part of future work, I plan to test other AI models using my framework and aim to create benchmarks for water treatment plants' usage of AI, with the long-term goal of securing complex and critical water distribution networks across the country.

9.2 AI for Agriculture - DeepAg

Merging outlier events with production forecasts also reveals more accurate insights. A global supply shock for commodities, weather events, or an important international affair can affect production and can cause sudden spikes. Typically, these outlier events are sudden and cannot be planned for. During these times, producers are left with little insight into how their production will be affected. For instance, the COVID-19 pandemic resulted in a global supply shock for many essential commodities. The event created a large spike in the demand for products such as Beef and Chickens, amongst other commodities, and caused the price of these commodities to rise. This caused a shortage of essential commodities and caused the price of these commodities to rise even higher, creating a cycle until the production supply of these commodities exceeded market demand. My approach could potentially help producers plan for such events. Moreover, spikes in demand often contribute to global food waste, where producers/retailers/consumers typically purchase more than they need and often end up wasting the excess produce. The supply chain can be better equipped with future production trends with my approach and strategically release products in appropriate batches to mitigate such issues. I envision that these methods are also useful in cases of

detecting cyberbiosecurity attacks on national infrastructure, such as water management systems and supply chains.

Predicting agricultural production is essential for feeding the world in the years ahead. The amount of agricultural production has a direct effect on supply, demand, and trade. While I demonstrate my approach at the aggregate level, they can also be used at micro scales such as at a farm or county. The aggregate analysis can particularly aid in shaping policies since USDA, the Food and Agriculture Organization (FAO), and many other nations rely on country-specific as well as global forecasts to set policy parameters. Seasonal pattern data can also be used as a source of ground truth to determine trends and anticipate future demand. For example, note the prediction of chicken and beef production to 2025, which accounts for potential outlier events in the future; they are sharply different from existing straight-line forecasts and provide a bounded pathway for policy and other decisions. My forecasting approach can help producers determine how low the production should drop and help them take preventative measures to continue operating even when production demand is low. During the winter, the producers may reduce the number of workers to save costs, minimize distribution, and reduce production volume. In summary, DeepAg can positively affect agriculture through on-time outcomes and can increase overall farm performance using DL.

9.3 P_2O Conclusion and Future Work

The framework presented in this paper explores AI's role in preventing wastewater overflow and in detecting security threats. To achieve these objectives, P_2O is proposed and developed. Three decision-tree-based (RF, LightGBM, and XGBoost) and two NN-based (FF-ANN and LSTM) models were developed to constitute a prediction module in P_2O.

The results showed that the LSTM model predicts tunnel water levels better than the other AI models used in the experiments. The LSTM model with a 24-hour input sequence with a 2-hour output sequence is selected as the best model for the protection module based on RMSE (0.036), RSR (0.260), and NSE (0.739) evaluation metrics. SHAP analysis is also performed, and it revealed that the top five important variables that affect the prediction the most are water level sensor data, overflow indicator sensor, total water flow sensor, pump five, and wastewater treatment flow sensor. Further, SMOD dataset is used to develop P_2O's protection module for detecting security threats at WWTPs. For this purpose, two experiments focused on intention classification and attack situation detection are performed. These experiments are executed using LSTM and GRU models. For the intention classification, the LSTM model showed 94% accuracy, while the GRU model showed 96% accuracy in identifying intentional attacks. Further, the LSTM model misclassifies about 4% of intentional attacks as outlier events, but the misclassification rate for the GRU model is only about 0.5%. The LSTM model misclassified three attack scenarios for attack situation detection as normal operations, while the GRU model misclassified only two attacks as normal operations. These results revealed that the LSTM model showed higher misclassification than the GRU model. These experiments conclude that the GRU model is the best suitable for detecting security threats considering the accuracy and severity of not detecting an attack at WWTP. Finally, the simulation results of the optimization module indicate a reduction in the amount of influent directed to the wet-weather treatment plant by 23% while preventing overflow incidents under extremely wet weather conditions based on five years of data.

In the future, I would like to focus on three objectives for improving the framework: context, AI assurance, and Attention-based modeling. In the first objective, I would like to understand the effect of the utilization of weather variables (snow, air temperature, humidity) and demographic data on the models, as a "context" for improved water level predictions. In

the second objective, I would like to evaluate the AI models further against implicit bias and cyber attacks with minimum perturbations such as adversarial networks, especially via threat detection solutions. Finally, in the third objective, I would like to use an attention-based model to understand the effect of existing and new variables on water level predictions, especially for predictions during wet seasons.

9.4 Cyber Physical Attacks Detection for Water Systems - DeepH$_2$O

9.4.1 Water Laws and Public Policy

Environmental and water laws govern our nation's water, air, waste, and other natural components. Most of the time, and due to the public's lack of awareness or attention, voters are usually drawn to water and environmental issues after wide-scale incidents of environmental damage, such as the Flint Water crisis[1] and its effects on safe drinking water in the state. The Clean Water Act (CWA) establishes the basic rules and benchmarks for regulating quality standards and discharging pollutants into the waters of the United States. The work presented in this manuscript aims to provide preventive measures for the health of water treatment plants against the rising dangers of cyber attacks. DeepH_2O is instrumental in governing cyber components of a water facility, providing recommendations to WDS operators on when and where the attack occurs, and validating against water policies and Environmental Protection Agency (EPA) regulations. This project continues as a collaboration with WDSs in Northern Virginia and the District of Columbia (DC) to deploy DeepH_2O at local facilities and aim to expand it to other WDSs as well. Conclusions and future work

[1] https://www.michigan.gov/mdhhs/inside-mdhhs/legal/flint-water-settlement

items are presented next.

9.4.2 Conclusions and Future Work

This manuscript presents DeepH_2O, a novel cyber attack detection framework for WDSs. DeepH_2O applies AI assurance to two DL architectures, TGCN with attention and HCAE, and compares their performance improvement over their baseline models. For TGCN with the attention model (supervised model), it has been observed that applying AI assurance, including attention and RMD with TGCN, improves the model's attack detection accuracy. Similarly, for $HCAE$ (unsupervised model), applying AI assurance, including tide weights, orthogonality constraints, and other constraints, improves detection accuracy and F1-score of the $HCAE$ model compared to AE.

The performance of both supervised and unsupervised models on poisoned data has been evaluated. For the supervised model, compared to its performance on the test dataset, it has been observed that most of the metrics decrease significantly. The supervised model struggles to perform (i.e., to detect an attack) if there is randomness in the dataset. Unlike the supervised model that performs poorly on poisoned data, my result indicates that the predictive performance of the unsupervised model (HCAE) is similar for the test data and the poisoned GAN data. No significant drop in the model's performance has been observed. To explain this phenomenon, the unsupervised model learns uncorrelated feature representation in the latent dimension and does not learn the sequential attributes. Hence the model can identify randomness in the poisoned data.

The result suggests that the $HCAE$ model has better generalizability. Among the two models, the unsupervised model (HCAE) performs better in terms of ranking score and time-to-detection score. Also, $HCAE$ is well generalized and regularized while detecting

attacked samples on the BATADAL test set. This improved classification performance and recall values make $HCAE$ a better choice for deployment in the WDS.

The study uses multiple performance metrics, including time-to-detection score, classification score, ranking score, precision, recall, accuracy, and F1 score, to measure the model's performance. The F1 score improvement is focused on the various metrics because of the heavily imbalanced BATADAL dataset. Therefore, this particular case, the F1 score becomes an important metric that considers model attack prediction errors and accounts for the type of errors by taking the harmonic mean of precision and recall. That is, only if both precision and recall values are high the F1 score gets higher; in this study, a higher F1 score indicates higher "$ATTACK$" and "$NO\ ATTACK$" harmonic class detection. Additionally, the unsupervised model outperforms the supervised model for WDS, including a better F1 score. The unsupervised model is a one-class classification method that generalizes well regardless of the water systems' spatio-temporal structure, making the model simpler than TGCN with attention. Additionally, the unsupervised model does not require labeling, an expensive and time-consuming activity in the model development process.

The ability of both supervised and unsupervised models in feature localization has been evaluated. Localizing a feature is tedious for both models during a concealed attack. Although the results are not highly accurate, they are promising and vital for WDS. For instance, both models can identify attacked node(s) or neighboring nodes during an "$ATTACK$". Further refining the model hyper-parameters by applying a grid search technique can improve the performance and result in better feature localization results, which is a potential future work. The sensitivity analysis of the two models showed that less important or sensitive variables were inactive in the training set, while active components were the most influential during a cyber-attack. However, some common junctions had high sensitivity or importance flags due to imbalanced training data.

Additionally, the extension of this work can be the following: 1) The GAN used in these experiments to generate synthetic data fails to replicate the time-series information from the original dataset. The attack samples are generated merely using GAN. Consequently, the next plan is to use TimeGAN Yoon et al. [393], a variant of GAN, to generate sequential (time-series) synthetic data consisting of both attack and non-attack samples and test the performance of DL models on the time-series synthetic data. 2) A large metropolitan city can have multiple WDSs across various locations within the metroplex. A bad actor can start a concealed attack on one of the WDS and continue to spread the attack across all locations. To swiftly detect and prevent such attacks, Federated Learning (FL) techniques Yang et al. [423] can be adapted to learn from the initial concealed attack and leverage that information to prevent future attacks (of a similar nature) across other WDSs. Furthermore, using the real-time data collected from the WDSs to retrain the DL model can significantly improve the detection performance of the model. However, given the geographically distributed nature of WDSs, it is essential to preserve the privacy of the real-time data (collected from the WDSs). Therefore, the plan is to use FL techniques to guarantee data and model privacy. 3) Training and deploying a DL model across different WDSs is challenging as the threshold might vary across different WDSs locations. This is further complicated by a set of different operations across WDSs. Another interesting idea is to explore Context learning [72] to enable DL models to be context-aware (such as population and weather) and efficiently detect attacks that vary based on different thresholds. Furthermore, training and evaluation of the Deep$H_2$0 framework using real-world WDS datasets[2] such as: Water Distribution (WADI) dataset and Secure Water Treatment(SWaT) is a future task. Lastly, a plan to develop approaches that explain the model's outcomes to water plant operators could be a great study, which would result in higher adoption rates and increased trustworthiness Batarseh et al. [317] of such frameworks at water facilities in the United States.

[2]https://itrust.sutd.edu.sg/

9.5 Context to AI for Water Systems

In this work, I introduced cP_2O, a hybrid DL model designed for short-term forecasting in Wastewater Treatment Plants (WWTPs). The model integrates contextual data through dynamic smoothing and Long Short-Term Memory (LSTM) architectures to improve predictive accuracy. By leveraging both internal sensor data and exogenous variables—such as weather conditions—cP_2O effectively captures temporal dependencies and external influences on WWTP operations. It outperforms several baseline approaches, including ARIMA, Exponential Smoothing, and other contemporary ML techniques, demonstrating notable gains in both accuracy and robustness.

The incorporation of context variables significantly enhanced the model's reliability. By employing a two-stage framework to process both internal and external data sources, cP_2O adapts to complex temporal patterns, including multiple seasonalities and abrupt changes induced by demographic shifts or local events. Such predictive power is crucial in WWTP management, where accurate short-term forecasts enable more informed decision-making, optimized resource allocation, and proactive measures to prevent system overload during peak demand periods.

An ablation study confirmed that each component—dynamic smoothing, context integration, and the use of dilated LSTM cells—contributes meaningfully to the model's improved performance. Contextual information, in particular, proved integral to enhancing forecasting accuracy. The adoption of a multi-step ahead forecasting approach further equips operators with the foresight needed to implement timely interventions and maintain operational stability.

Despite these positive outcomes, cP_2O has limitations. Its scalability can be constrained by the number of time series variables and the associated parameterization. While highly

efficient and effective for smaller datasets, future research will need to address strategies for scaling the model to handle larger data volumes without compromising accuracy or interpretability.

1. Real-World Deployment and Scalability: Ongoing efforts will involve collaborating with additional facilities in different regions and contexts. By evaluating the model's performance in diverse operational settings, I aim to enhance its scalability, ensuring it can manage larger, more heterogeneous datasets while maintaining robust predictive capabilities.

2. Richer External Variables and Expanded Contextual Data: Beyond meteorological and demographic data, incorporating additional exogenous factors—such as economic indicators, policy changes, or sensor data from upstream agricultural activities—may further improve forecasting accuracy. This expanded contextualization will help the model better adapt to complex, evolving environmental and infrastructural conditions.

3. Chemistry-Based AI Understanding: An intriguing direction for future work lies in integrating chemistry-based knowledge into the AI modeling process. For instance, incorporating chemical composition data, reaction kinetics, or nutrient-removal dynamics into the context stage could offer deeper insights into the underlying processes of wastewater treatment. By aligning AI-driven forecasting with chemical and biochemical principles, the model may better capture the root causes of fluctuations, enhancing both interpretability and decision support for operators who rely on chemical treatments, aeration strategies, or nutrient dosing to maintain compliance and efficiency.

Through these developments, the ultimate goal remains to refine and broaden the applicability of cP_2O and its successors. By continually expanding the model's contextual knowledge

and integrating insights from chemistry and other scientific domains, I aim to create more resilient, trustworthy, and actionable predictive tools for WWTP operations and potentially other critical infrastructures.

Appendices

Appendix A

cP$_2$O Model Supplemental Materials

A.1 Context Definition for WWTPs

In this appendix, I provide a detailed mathematical definition of the context variables used in the cP$_2$O model and explain how they are integrated into the forecasting framework.

A.1.1 Context Variables Representation

Let $\mathbf{D}_t \in \mathbb{R}^N$ represent the vector of WWTP internal variables at time t, where N is the number of internal variables (e.g., influent flow rate, water levels). The context variables, denoted by $\mathbf{C}_t \in \mathbb{R}^M$, capture external factors influencing the WWTP, where M is the number of context variables (e.g., weather data, river flow rates, demographic data).

The combined input vector at time t is defined as:

$$\mathbf{x}_t = \begin{bmatrix} \mathbf{D}_t \\ \mathbf{C}_t \end{bmatrix} \in \mathbb{R}^{N+M}$$

The context variables $\mathbf{C}_t$ can include, but are not limited to:

- *Weather Data*: Precipitation (P_t), temperature (T_t), humidity (H_t), wind speed (W_t).

- *River Data*: River flow rates (R_t), water levels (L_t).
- *Demographic Data*: Population density (ρ_t), urbanization rate (U_t).
- *Economic Data*: Industrial output (I_t), employment rates (E_t).

These variables provide external information that influences WWTP operations and are crucial for improving forecasting accuracy.

A.1.2 Integration into the Model

The cP$_2$O model incorporates context variables through an enriched input vector and a context extraction stage. The model consists of two main components:

1. *Context Extraction Stage*: Processes context variables to generate a context vector $\mathbf{r}_t$.
2. *Forecasting Stage*: Utilizes both internal variables and the context vector to make predictions.

Context Extraction Stage

The context extraction stage employs a dilated LSTM network to process context variables over a sequence of past time steps. Let $\Omega_t^{\text{ctx}} = \{t - T_c + 1, \ldots, t\}$ represent the context input window of length T_c. The context LSTM processes the sequence $\{\mathbf{C}_\tau\}_{\tau=t-T_c+1}^{t}$ to generate the context vector $\mathbf{r}_t$:

$$\mathbf{r}_t = \text{LSTM}_{\text{ctx}}\left(\{\mathbf{C}_\tau\}_{\tau=t-T_c+1}^{t}; \boldsymbol{\theta}_{\text{ctx}}\right) \in \mathbb{R}^u$$

where $\boldsymbol{\theta}_{\text{ctx}}$ are the parameters of the context LSTM, and u is the dimension of the context vector.

Forecasting Stage Input Enhancement

The context vector $\mathbf{r}_t$ is concatenated with the internal variables to form the enhanced input vector for the forecasting stage:

$$\mathbf{x}'_t = \begin{bmatrix} \mathbf{D}_t \\ \mathbf{r}_t \end{bmatrix} \in \mathbb{R}^{N+u}$$

The forecasting LSTM processes the sequence $\{\mathbf{x}'_\tau\}_{\tau=t-T_f+1}^{t}$, where $\Omega_t^{\text{in}} = \{t - T_f + 1, \ldots, t\}$ is the input window of length T_f, to generate the forecasted output $\hat{\mathbf{y}}_{t+1}$:

$$\hat{\mathbf{y}}_{t+1} = \text{LSTM}_{\text{fcast}}\left(\{\mathbf{x}'_\tau\}_{\tau=t-T_f+1}^{t};\, \boldsymbol{\theta}_{\text{fcast}}\right)$$

where $\boldsymbol{\theta}_{\text{fcast}}$ are the parameters of the forecasting LSTM.

A.1.3 Attention Mechanism in Context Integration

The forecasting LSTM incorporates an attention mechanism to dynamically weigh the contributions of the context vector and internal variables. At each time step t, attention weights $\boldsymbol{\alpha}_t \in \mathbb{R}^{N+u}$ are computed:

$$\boldsymbol{\alpha}_t = \text{softmax}\left(\mathbf{W}_a \mathbf{h}_{t-1} + \mathbf{b}_a\right)$$

where $\mathbf{W}_a \in \mathbb{R}^{(N+u)\times s_h}$ and $\mathbf{b}_a \in \mathbb{R}^{N+u}$ are learnable parameters, s_h is the dimension of the hidden state, and $\mathbf{h}_{t-1}$ is the hidden state from the previous time step.

The enhanced input vector is then modulated by the attention weights:

$$\tilde{\mathbf{x}}_t = \boldsymbol{\alpha}_t \odot \mathbf{x}'_t$$

where $\odot$ denotes element-wise multiplication.

The forecasting LSTM processes the modulated input $\tilde{\mathbf{x}}_t$:

$$\mathbf{h}_t,\, \mathbf{c}_t = \mathrm{LSTM}_{\mathrm{fcast}}\left(\tilde{\mathbf{x}}_t,\, \mathbf{h}_{t-1},\, \mathbf{c}_{t-1};\, \boldsymbol{\theta}_{\mathrm{fcast}}\right)$$

where $\mathbf{h}_t$ and $\mathbf{c}_t$ are the hidden and cell states at time t.

A.1.4 Output Generation

The final forecast is generated by applying a linear transformation to the hidden state $\mathbf{h}_t$:

$$\hat{\mathbf{y}}_{t+1} = \mathbf{W}_o \mathbf{h}_t + \mathbf{b}_o$$

where $\mathbf{W}_o \in \mathbb{R}^{s_y \times s_h}$ and $\mathbf{b}_o \in \mathbb{R}^{s_y}$ are the output weight matrix and bias vector, and s_y is the dimension of the output vector.

A.1.5 Summary of Notation

- $\mathbf{D}_t \in \mathbb{R}^N$: Internal WWTP variables at time t.

- $\mathbf{C}_t \in \mathbb{R}^M$: Context variables at time t.
- $\mathbf{x}_t \in \mathbb{R}^{N+M}$: Combined input vector.
- $\mathbf{r}_t \in \mathbb{R}^u$: Context vector extracted by the context LSTM.
- $\mathbf{x}'_t \in \mathbb{R}^{N+u}$: Enhanced input vector for forecasting.
- $\boldsymbol{\alpha}_t \in \mathbb{R}^{N+u}$: Attention weights.
- $\tilde{\mathbf{x}}_t \in \mathbb{R}^{N+u}$: Modulated input vector after applying attention.
- $\mathbf{h}_t$, $\mathbf{c}_t$: Hidden and cell states of the forecasting LSTM.
- $\hat{\mathbf{y}}_{t+1}$: Forecasted output at time $t+1$.
- $\boldsymbol{\theta}_{\text{ctx}}$, $\boldsymbol{\theta}_{\text{fcast}}$: Parameters of the context and forecasting LSTMs, respectively.

A.1.6 Mathematical Formulation of the Forecasting Function

Combining the components, the forecasting function can be summarized as:

$$\hat{\mathbf{y}}_{t+1} = f\left(\mathbf{D}_t, \mathbf{C}_t; \boldsymbol{\theta}\right) = \mathbf{W}_o \mathbf{h}_t + \mathbf{b}_o$$

where $\mathbf{h}_t$ is obtained through the following steps:

$$\mathbf{r}_t = \text{LSTM}_{\text{ctx}}\left(\{\mathbf{C}_\tau\}_{\tau=t-T_c+1}^{t};\, \boldsymbol{\theta}_{\text{ctx}}\right) \tag{A.1}$$

$$\mathbf{x}'_t = \begin{bmatrix} \mathbf{D}_t \\ \mathbf{r}_t \end{bmatrix} \tag{A.2}$$

$$\boldsymbol{\alpha}_t = \text{softmax}\left(\mathbf{W}_a \mathbf{h}_{t-1} + \mathbf{b}_a\right) \tag{A.3}$$

$$\tilde{\mathbf{x}}_t = \boldsymbol{\alpha}_t \odot \mathbf{x}'_t \tag{A.4}$$

$$\mathbf{h}_t,\, \mathbf{c}_t = \text{LSTM}_{\text{fcast}}\left(\tilde{\mathbf{x}}_t,\, \mathbf{h}_{t-1},\, \mathbf{c}_{t-1};\, \boldsymbol{\theta}_{\text{fcast}}\right) \tag{A.5}$$

Equation (A.1) computes the context vector $\mathbf{r}_t$ by processing the sequence of context variables through the context LSTM. This captures temporal patterns in the external factors influencing the WWTP. Equation (A.2) forms the enhanced input vector by concatenating the internal variables $\mathbf{D}_t$ with the context vector $\mathbf{r}_t$. Equation (A.3) calculates the attention weights $\boldsymbol{\alpha}_t$ based on the previous hidden state $\mathbf{h}_{t-1}$, allowing the model to dynamically focus on the most relevant features at each time step. Equation (A.4) modulates the enhanced input vector with the attention weights, effectively weighting each feature according to its importance. Equation (A.5) processes the modulated input through the forecasting LSTM to update the hidden and cell states, capturing the temporal dependencies in the data. The final forecast $\hat{\mathbf{y}}_{t+1}$ is generated by applying a linear transformation to the updated hidden state.

A.1.7 Dimensions Clarification

For clarity, I specify the dimensions of the key variables:

- $\mathbf{D}_t \in \mathbb{R}^N$: Column vector of internal variables.

- $\mathbf{C}_t \in \mathbb{R}^M$: Column vector of context variables.
- $\mathbf{r}_t \in \mathbb{R}^u$: Column vector from context LSTM.
- $\mathbf{x}'_t \in \mathbb{R}^{N+u}$: Column vector (concatenation of $\mathbf{D}_t$ and $\mathbf{r}_t$).
- $\boldsymbol{\alpha}_t \in \mathbb{R}^{N+u}$: Column vector of attention weights.
- $\tilde{\mathbf{x}}_t \in \mathbb{R}^{N+u}$: Column vector of modulated input.
- $\mathbf{h}_t \in \mathbb{R}^{s_h}$: Hidden state vector.
- $\hat{\mathbf{y}}_{t+1} \in \mathbb{R}^{s_y}$: Output vector.

A.1.8 Context Variables Examples

As previously mentioned, the context variables $\mathbf{C}_t$ can include various external factors:

- *Weather Data* (M_{weather} variables):
- Precipitation (P_t)
- Temperature (T_t)
- Humidity (H_t)
- Wind speed (W_t)
- *River Data* (M_{river} variables):
- River flow rates (R_t)
- Water levels (L_t)
- *Demographic Data* (M_{demo} variables):

- Population density (ρ_t)
- Urbanization rate (U_t)
- *Economic Data* (M_{econ} variables):
- Industrial output (I_t)
- Employment rates (E_t)

Total context variables: $M = M_{\text{weather}} + M_{\text{river}} + M_{\text{demo}} + M_{\text{econ}}$.

A.1.9 Learning Objective

The model parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta}_{\text{ctx}}, \boldsymbol{\theta}_{\text{fcast}}, \mathbf{W}_a, \mathbf{b}_a, \mathbf{W}_o, \mathbf{b}_o\}$ are learned by minimizing the loss function defined in the main text, typically involving the pinball loss for quantile regression:

$$\mathcal{L} = \sum_t \ell\left(y_t, \hat{y}_t\right)$$

where y_t is the observed value, and $\hat{y}_t$ is the predicted value.

A.2 Hyperparameters

In this appendix, I provide a detailed parameter choice for the cP$_2$O and other baseline models.

A.2.1 cP$_2$O Hyperparameter Choice

- Batch size (B): Initially set to 16 and raised to 64 after the fourth epoch, based on experimental results. Further increases were restricted due to the dataset size.
- Initial seasonality adjustments: Computed as ratios between the first input window's values and the mean values of that window.
- Initial smoothing factors: $S_\alpha = -4$ and $S_\beta = 0.45$, chosen based on the average smoothing coefficients dynamic behavior.
- Learning rate schedule: Initially assigned to 5×10^{-3} for the first five epochs, then reduced progressively to 10^{-4} by epoch 9.
- Dilation rates: Experimentally set to 1, 2, and 4, following the rule of increasing dilations, ideally in an exponential fashion.
- Embedding dimensions: Set to 10, determined through experimentation for time-related variables.
- Embedding layer weight and bias matrices: Shared across paths, defined by $W \in \mathbb{R}^{90\times4}$ and $b \in \mathbb{R}^4$.
- Training steps per batch (T_b): Set to 40, based on trial and error.
- Total epochs: Set to 50, during which both batch size increases and learning rate decreases.
- Sub-epoch calculation (S_e): Set to 10 for DC Water and 15 for AlexRenew
- Updates per epoch (U_e): Set to 1500 iterations to ensure significant accuracy improvement per epoch.

- Optimizer: The Adam optimization algorithm, chosen based on experimentation.
- Loss function parameter (λ): Set to 0.35, ensuring that the average central loss during training is higher than the losses for the lower and upper intervals.
- Pinball loss quantiles: Experimentally adjusted to $q^* = 0.62$, $q = 0.039$, and $q = 0.981$.
- Context batch size (C_b): Set to 20, indicating the number of exogenous variables.
- Hidden state size (H_s): Set to 165 for the c-state and 80 for the h-state, based on experimentation.
- Output vector size (O_v): Calculated as the difference between c-state and h-state sizes.
- Forecasting stage weight and bias matrices: Comprising 12 sets of matrices (four per layer across three layers): $W \in \mathbb{R}^{n\times 150}$, $V \in \mathbb{R}^{70\times 150}$, $U \in \mathbb{R}^{70\times 150}$, and $b \in \mathbb{R}^{150}$. For the first layer, $n = 193$; for subsequent layers, $n = 273$.
- Forecasting stage output layer weights and biases: Defined by $W \in \mathbb{R}^{80\times 74}$ and $b \in \mathbb{R}^{74}$.
- Context stage weight and bias matrices: Also comprising 12 sets of matrices, with $W \in \mathbb{R}^{m\times 150}$, $V \in \mathbb{R}^{70\times 150}$, $U \in \mathbb{R}^{70\times 150}$, and $b \in \mathbb{R}^{150}$. In the first layer, $m = 247$; in other layers, $m = 327$.
- Context path output layer weights and biases: Defined by $W \in \mathbb{R}^{80\times 5}$ and $b \in \mathbb{R}^{5}$.
- Ensemble method: Simple averaging across the ensemble members.
- Ensemble size (E): Set to 20, depending on the scenario for each experiment.

A.2.2 Baseline Models Hyperparameter Choice

- ES: Divided into 24 hourly time series, predicted with ‘ets‘ in R.

- Naive: Using the previous step's value as the prediction.
- ARIMA: Splitting into 24 time series (one per hour), with forecasts generated using 'auto.arima' in R.
- LGBM: Utilizes 'LGBMRegressor' with 'max_depth' set to 10, 500 iterations, and a learning rate of 0.01.
- XGB: Uses 'XGBRegressor' with 'max_depth' of 10 and learning rate 0.05, with additional features.
- SVM: Predictions made using 'fitrsvm' in Matlab. Key hyperparameters are fine-tuned.
- N-WE: Implemented in MATLAB, with a fixed pattern length of 24.
- GRNN: Similar to N-WE, implemented in Matlab with cross-validated smoothing parameters.
- MLP: Uses 'feedforwardnet' in Matlab, with a single hidden layer and trained with Bayesian regularization.
- LSTM: Implemented using 'lstm' in Matlab, with fixed 24 neurons.
- ANFIS: Forecasts with 'anfis' in Matlab, using a Sugeno fuzzy inference system.
- MTGNN: Implemented with default settings from the repository at https://github.com/nnzhan/MTGNN.
- DeepAR: Uses GluonTS with 'context_length' set to seven times the 'prediction_length'.
- Prophet: Forecasts generated using the 'prophet' package in R with default settings.
- WaveNet: Uses GluonTS with default hyperparameters.

- N-BEATS: Implemented via GluonTS with ‘context_length‘ set to seven times the ‘prediction_length‘.

Appendix B

GAN Model Parameters

B.1 GAN Model Parameters

Table B.1: GAN Models Parameters on ACWA Dataset

Model	**Parameters**
CTGAN	batch_size = 150, epochs = 101, learning_rate = 5e-5, beta_1 = 0.5, beta_2 = 0.9
WGAN	noise_dim = 32, dim = 64, batch_size = 64, epochs = 101, learning_rate = 5e-5, beta_1 = 0.5, beta_2 = 0.9
DRAGAN	noise_dim = 64, dim = 64, batch_size = 150, epochs = 101, learning_rate = 2e-6, beta_1 = 0.5, beta_2 = 0.9
WGAN-GP	noise_dim = 64, dim = 64, batch_size = 150, epochs = 101, learning_rate = [5e-5, 1e-3], beta_1 = 0.5, beta_2 = 0.9
Cramer GAN	noise_dim = 32, dim = 64, batch_size = 64, epochs = 101, learning_rate = 1e-5, beta_1 = 0.5, beta_2 = 0.9
TimeGAN (Default)	seq_len=24, n_seq = 6, hidden_dim=24, gamma=1, noise_dim = 32, dim = 128, batch_size = 128, learning_rate = 5e-4
TimeGAN (After Tuning)	seq_len=24, n_seq = 8, hidden_dim=24, gamma=1, noise_dim = 32, dim = 128, batch_size = 32, log_step = 100, learning_rate = 5e-4, Train_steps = 10000
DoppelGANger (Default)	batch_size=100, lr=0.001, betas=(0.2, 0.9), latent_dim=20, gp_lambda=2, pac=1, epochs=400, sequence_length=56
DoppelGANger (After Tuning)	batch_size=32, lr=0.001, betas=(0.2, 0.9), latent_dim=24, gp_lambda=2, pac=1, epochs=1000, sequence_length=24, sample_length=6, rounds=1

Table B.2: GAN Models Parameters on Real-world Plant Dataset

Model	**Parameters**
CTGAN	batch_size = 250, epochs = 101, learning_rate = 5e-5, beta_1 = 0.5, beta_2 = 0.9
WGAN	noise_dim = 32, dim = 128, batch_size = 128, epochs = 101, learning_rate = 5e-5, beta_1 = 0.5, beta_2 = 0.9
DRAGAN	noise_dim = 128, dim = 128, batch_size = 250, epochs = 101, learning_rate = 2e-6, beta_1 = 0.5, beta_2 = 0.9
WGAN-GP	noise_dim = 128, dim = 128, batch_size = 250, epochs = 101, learning_rate = [5e-5, 1e-3], beta_1 = 0.5, beta_2 = 0.9
Cramer GAN	noise_dim = 32, dim = 128, batch_size = 128, epochs = 101, learning_rate = 1e-5, beta_1 = 0.5, beta_2 = 0.9
TimeGAN (Default)	seq_len=24, n_seq = 6, hidden_dim=24, gamma=1, noise_dim = 32, dim = 128, batch_size = 128, learning_rate = 5e-4
TimeGAN (After Tuning)	seq_len=24, n_seq = 13, hidden_dim=24, gamma=1, noise_dim = 32, dim = 128, batch_size = 200, log_step = 100, learning_rate = 5e-4, Train_steps = 10000
DoppelGANger (Default)	batch_size=100, lr=0.001, betas=(0.2, 0.9), latent_dim=20, gp_lambda=2, pac=1, epochs=400, sequence_length=56
DoppelGANger (After Tuning)	batch_size=200, lr=0.001, betas=(0.2, 0.9), latent_dim=24, gp_lambda=2, pac=1, epochs=1000, sequence_length=24, sample_length=6, rounds=1

Table B.3: GAN Models Parameters for BATADAL Dataset

Model	**Parameters**
CTGAN (Default)	batch_size = 500, epochs = 501, learning_rate = 2e-4, beta_1 = 0.5, beta_2 = 0.9 critic_loss and generator_loss observations
CTGAN (Tuned)	batch_size = 250, epochs = 101, learning_rate = 5e-5, beta_1 = 0.5, beta_2 = 0.9
WGAN (Default)	noise_dim = 32, dim = 128, batch_size = 128, log_step = 100, epochs = 501, learning_rate = 5e-4, beta_1 = 0.5, beta_2 = 0.9, generator and discriminator loss observations
WGAN (Tuned)	noise_dim = 32, dim = 128, batch_size = 128, epochs = 101, learning_rate = 5e-5, beta_1 = 0.5, beta_2 = 0.9

Continued on next page

Table B.3 – continued from previous page

Model	**Parameters**
DRAGAN (Default)	noise_dim = 128, dim = 128, batch_size = 500, epochs = 501, learning_rate = 1e-5, beta_1 = 0.5, beta_2 = 0.9, loss observations
DRAGAN (Tuned)	noise_dim = 128, dim = 128, batch_size = 250, epochs = 101, learning_rate = 2e-6, beta_1 = 0.5, beta_2 = 0.9
WGAN-GP (Default)	noise_dim = 128, dim = 128, batch_size = 500, epochs = 501, learning_rate = [5e-4, 3e-3], beta_1 = 0.5, beta_2 = 0.9
WGAN-GP (Tuned)	noise_dim = 128, dim = 128, batch_size = 250, epochs = 101, learning_rate = [5e-5, 1e-3], beta_1 = 0.5, beta_2 = 0.9
Cramer GAN (Default)	noise_dim = 32, dim = 128, batch_size = 128, epochs = 501, learning_rate = 5e-4, beta_1 = 0.5, beta_2 = 0.9, loss observations at epoch 17
Cramer GAN (Tuned)	noise_dim = 32, dim = 128, batch_size = 128, epochs = 40, learning_rate = 1e-5, beta_1 = 0.5, beta_2 = 0.9
TimeGAN (Default)	seq_len=24, n_seq = 6, hidden_dim=24, gamma=1, noise_dim = 32, dim = 128, batch_size = 128, learning_rate = 5e-4
TimeGAN (Tuned)	seq_len=24, n_seq = 26, hidden_dim=24, gamma=1, noise_dim = 32, dim = 128, batch_size = 200, log_step = 100, learning_rate = 5e-4, train_steps = 10000
DoppelGANger (Default)	batch_size=100, lr=0.001, betas=(0.2, 0.9), latent_dim=20, gp_lambda=2, pac=1, epochs=400, sequence_length=56
DoppelGANger (Tuned)	batch_size=200, lr=0.001, betas=(0.2, 0.9), latent_dim=24, gp_lambda=2, pac=1, epochs=1000, sequence_length=24, sample_length=6, rounds=1

Appendix C

DeepH$_2$O Model Parameters and Supplemental Materials

C.1 First Seven Attacks Set Descriptions (C-Town Dataset 2)

ID	Start Time (D/M/Y H)	End Time (D/M/Y H)	Duration (H)	Attack Description	SCADA Concealment	Label (h)
1	13/09/2016 23	16/09/2016 00	50	Alters SCADA transmission to PLC9, changes L_T7 thresholds for PU10/PU11 operation. Low T7 levels.	Replay attack on L_T7	42
2	26/09/2016 11	27/09/2016 10	24	Similar to Attack #1.	Extended replay on PU10/PU11 flow and status.	0
3	09/10/2016 09	11/10/2016 20	60	Alters L_T1 readings to maintain low levels, keeps PU1/PU2 running. Overflow in T1.	Offset polyline for L_T1 rise.	60
4	29/10/2016 19	02/11/2016 16	94	Similar to Attack #3.	Replay on L_T1, PU1/PU2 flow/status, and P_J269.	37
5	26/11/2016 17	29/11/2016 04	60	Reduces PU7 speed to 90%. Lower T4 levels.	None	7
6	06/12/2016 07	10/12/2016 04	94	Similar to Attack #5, but speed reduced to 70%.	Replay on L_T4.	73
7	14/12/2016 15	19/12/2016 04	110	Similar to Attack #6.	Replay on L_T4, PU6/PU7 flow, and status.	0

Table C.1: Description of the first seven attacks in Dataset 2 Taormina et al. [8].

C.2 Remaining Seven Attacks Set Descriptions (C-Town Dataset 3)

ID	Start Time (D/M/Y H)	End Time (D/M/Y H)	Duration (H)	Attack Description	SCADA Concealment
8	16/01/2017 09	19/01/2017 06	70	Controls PLC3, alters L_T3 thresholds for PU4/PU5 operation. Low T3 levels.	Replay on L_T3, PU4/PU5 flow, and status.
9	30/01/2017 08	02/02/2017 00	65	Alters L_T2 readings to PLC3, showing low levels, keeps V2 open. T2 overflows.	Offset polyline for L_T2 rise.
10	09/02/2017 03	10/02/2017 09	31	Malicious activation of PU3.	None
11	12/02/2017 01	13/02/2017 07	31	Similar to Attack #10.	None
12	24/02/2017 05	28/02/2017 08	100	Similar to Attack #9.	Replay on L_T2, V2 flow/status, and pressure readings (P_J14, P_J422).
13	10/03/2017 14	13/03/2017 21	80	Controls PLC5, alters L_T7 thresholds, forces PU10/PU11 cycling.	Replay on L_T7, PU10/PU11 flow/status, and pressure readings.
14	25/03/2017 20	27/03/2017 01	30	Alters T4 signal to PLC6. Overflow in T6.	None

Table C.2: Description of the remaining seven attacks in Dataset 3 Taormina et al. [8].

C.3 Hyperparameters Selection for DeepH$_2$O

Hyperparameter	**Baseline AE**	**HCAE**	**Baseline TGCN**	**TGCN with Attention**
Adam Optimizer Learning Rate	**0.0001**, 0.001, 0.01, 0.1	**0.0001**, 0.001, 0.01, 0.1	0.0001, 0.001, 0.005, **0.01**, 0.1	0.0001, 0.001, **0.005**, 0.01, 0.1
Batch Size	8, 16, **32**, 64, 128, 256	8, 16, **32**, 64, 128, 256	8, **16**, 32, 64, 128, 256	8, 16, 32, 64, **128**, 256
Sequence Length	N/A	N/A	4, **8**, 16, 24, 32 hours	4, **8**, 16, 24, 32 hours
Number of Epochs	**500**, 1000, 2500, 5000	**500**, 1000, 2500, 5000	500, **1000**, 2500, 5000	500, **1000**, 2500, 5000
Number of Hidden Layers	3, **5**, 7, 9, 11	3, **5**, 7, 9, 11	N/A	N/A
Hidden Dimensions	N/A	N/A	8, 16, 32, **64**, 100, 128	8, **16**, 32, 64, 100, 128

Table C.3: Hyperparameter selection using random search (bold values indicate the final selections).

Bibliography

[1] Yousra Regaya, Fodil Fadli, and Abbes Amira. Point-denoise: Unsupervised outlier detection for 3d point clouds enhancement. *Multimedia Tools and Applications*, 80(18): 28161–28177, 2021.

[2] Sai Gurrapu, Feras A Batarseh, Pei Wang, MD Nazmul Kabir Sikder, Nitish Gorentala, and Munisamy Gopinath. Deepag: Deep learning approach for measuring the effects of outlier events on agricultural production and policy. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE, 2021.

[3] Ajay Kulkarni, Mehmet Yardimci, Md Nazmul Kabir Sikder, and Feras A Batarseh. P2o: Ai-driven framework for managing and securing wastewater treatment plants. *Journal of Environmental Engineering*, 149(9):04023045, 2023.

[4] Feras Batarseh, Ajay Kulkarni, Chhayly Sreng, Justice Lin, and Siam Maksud. Acwa: an ai-driven cyber-physical testbed for intelligent water systems. *Water Practice & Technology*, 2023. doi: 10.2166/wpt.2023.197.

[5] Md Nazmul Kabir Sikder, Minh BT Nguyen, E Donald Elliott, and Feras A Batarseh. Deep h2o: Cyber attacks detection in water distribution systems using deep learning. *Journal of Water Process Engineering*, 52:103568, 2023.

[6] CISA. Aa21-287a: Conti ransomware, 2021. URL https://www.cisa.gov/news-events/cybersecurity-advisories/aa21-287a. Accessed on: [Insert Access Date].

[7] Amin Hassanzadeh, Amin Rasekh, Stefano Galelli, Mohsen Aghashahi, Riccardo

Taormina, Avi Ostfeld, and M Katherine Banks. A review of cybersecurity incidents in the water sector. *Journal of Environmental Engineering*, 146(5):03120003, 2020.

[8] Riccardo Taormina, Stefano Galelli, Nils Ole Tippenhauer, Elad Salomons, Avi Ostfeld, Demetrios G Eliades, Mohsen Aghashahi, Raanju Sundararajan, Mohsen Pourahmadi, M Katherine Banks, et al. Battle of the attack detection algorithms: Disclosing cyber attacks on water distribution networks. *Journal of Water Resources Planning and Management*, 144(8):04018048, 2018.

[9] Zhaohui Wang, Houbing Song, David W Watkins, Keat Ghee Ong, Pengfei Xue, Qing Yang, and Xianming Shi. Cyber-physical systems for water sustainability: challenges and opportunities. *IEEE Communications Magazine*, 53(5):216–222, 2015.

[10] DHS. Homeland security presidential directive 7: Critical infrastructure identification, prioritization, and protection, 2003.

[11] Romany F Mansour. Artificial intelligence based optimization with deep learning model for blockchain enabled intrusion detection in cps environment. *Scientific Reports*, 12(1):12937, 2022.

[12] Xiulong Liu, Jiannong Cao, Yanni Yang, and Shan Jiang. Cps-based smart warehouse for industry 4.0: A survey of the underlying technologies. *Computers*, 7(1):13, 2018.

[13] Nicholas Jeffrey, Qing Tan, and José R Villar. A review of anomaly detection strategies to detect threats to cyber-physical systems. *Electronics*, 12(15):3283, 2023.

[14] Moeen Hassanalieragh, Alex Page, Tolga Soyata, Gaurav Sharma, Mehmet Aktas, Gonzalo Mateos, Burak Kantarci, and Silvana Andreescu. Health monitoring and management using internet-of-things (iot) sensing with cloud-based processing: Oppor-

tunities and challenges. In *2015 IEEE international conference on services computing*, pages 285–292. IEEE, 2015.

[15] Karl de Fine Licht and Jenny de Fine Licht. Artificial intelligence, transparency, and public decision-making: Why explanations are key when trying to produce perceived legitimacy. *AI & society*, 35:917–926, 2020.

[16] François Kammerer. Self-building technologies. *AI & SOCIETY*, 35(4):901–915, 2020.

[17] Lanfang Sun, Xin Jiang, Huixia Ren, and Yi Guo. Edge-cloud computing and artificial intelligence in internet of medical things: architecture, technology and application. *IEEE Access*, 8:101079–101092, 2020.

[18] Alan M Turing. *Computing machinery and intelligence*. Springer, 2009.

[19] J McCarthy. Programs with common sense in wayback machine. *Mechanization of thought processes*, 1, 1959.

[20] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[21] Feras A Batarseh and Ajay Kulkarni. Ai for water. *Computer*, 56(03):109–113, 2023.

[22] V Dharmaraj and C Vijayanand. Artificial intelligence (ai) in agriculture. *International Journal of Current Microbiology and Applied Sciences*, 7(12):2122–2128, 2018.

[23] Sai Gurrapu, Nazmul Sikder, Pei Wang, Nitish Gorentala, Madison Williams, and Feras A Batarseh. Applications of machine learning for precision agriculture and smart farming. In *The International FLAIRS Conference Proceedings*, volume 34, 2021.

[24] Sai Gurrapu, Nazmul Sikder, Pei Wang, Nitish Gorentala, Madison Williams, and Feras Batarseh. Applications of machine learning for precision agriculture and smart

farming. *The International FLAIRS Conference Proceedings*, 34, Apr. 2021. doi: 10.32473/flairs.v34i1.128497. URL https://journals.flvc.org/FLAIRS/article/view/128497.

[25] SM Zahraee, M Khalaji Assadi, and R Saidur. Application of artificial intelligence methods for hybrid energy system optimization. *Renewable and sustainable energy reviews*, 66:617–630, 2016.

[26] Mounia Achouch, Mariya Dimitrova, Khaled Ziane, Sasan Sattarpanah Karganroudi, Rizck Dhouib, Hussein Ibrahim, and Mehdi Adda. On predictive maintenance in industry 4.0: Overview, models, and challenges. *Applied Sciences*, 12(16):8081, 2022.

[27] Yassine Himeur, Bhagawat Rimal, Abhishek Tiwary, and Abbes Amira. Using artificial intelligence and data fusion for environmental monitoring: A review and future perspectives. *Information Fusion*, 86:44–75, 2022.

[28] Mazin Alshamrani. Iot and artificial intelligence implementations for remote healthcare monitoring systems: A survey. *Journal of King Saud University-Computer and Information Sciences*, 34(8):4687–4701, 2022.

[29] Hammad Khawar, Tariq Rahim Soomro, and Muhammad Ayoub Kamal. Machine learning for internet of things-based smart transportation networks. In *Machine Learning for Societal Improvement, Modernization, and Progress*, pages 112–134. IGI Global, 2022.

[30] Olufemi A Omitaomu and Haoran Niu. Artificial intelligence techniques in smart grid: A survey. *Smart Cities*, 4(2):548–568, 2021.

[31] Muhammad Usama Usman, Ashraful Haque, Md Nazmul Kabir Sikder, Mengmeng Cai, Saifur Rahman Bradley, Shikhar Pandey, Chris Kliros, and Liuxi Zhang. Quan-

tification of peak demand reduction potential in commercial buildings due to hvac set point and brightness adjustment. In *2021 IEEE Power & Energy Society General Meeting (PESGM)*, pages 1–6. IEEE, 2021.

[32] Mehrdokht Pournader, Hadi Ghaderi, Amir Hassanzadegan, and Behnam Fahimnia. Artificial intelligence applications in supply chain management. *International Journal of Production Economics*, 241:108250, 2021.

[33] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.

[34] Charu C Aggarwal. An introduction to outlier analysis. In *Outlier analysis*, pages 1–34. Springer, 2017.

[35] Charu C Aggarwal, Yuchen Zhao, and S Yu Philip. Outlier detection in graph streams. In *2011 IEEE 27th international conference on data engineering*, pages 399–409. IEEE, 2011.

[36] MD Nazmul Kabir Sikder and Feras A. Batarseh. Outlier detection using ai: a survey. In *AI Assurance*, pages 231–291. Elsevier, 2022.

[37] Utkarsh Porwal and Smruthi Mukund. Credit card fraud detection in e-commerce: An outlier detection approach. *arXiv preprint arXiv:1811.02196*, 2018.

[38] Gebeyehu Belay Gebremeskel, Chai Yi, Zhongshi He, and Dawit Haile. Combined data mining techniques based patient data outlier detection for healthcare safety. *International Journal of Intelligent Computing and Cybernetics*, 9(1):42–68, 2016.

[39] Jeffrey T Bordogna, Donald E Brown, and James H Conklin. Design and implementation of an automated anomaly detection system for crime. In *2007 IEEE Systems and Information Engineering Design Symposium*, pages 1–6. IEEE, 2007.

[40] Ömer Aslan, Semih Serkant Aktuğ, Merve Ozkan-Okay, Abdullah Asim Yilmaz, and Erdal Akin. A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. *Electronics*, 12(6):1333, 2023.

[41] Premkumar Chithaluru, AL-Turjman Fadi, Manoj Kumar, and Thompson Stephan. Computational intelligence inspired adaptive opportunistic clustering approach for industrial iot networks. *IEEE Internet of Things Journal*, 2023.

[42] Yoshio Umezawa, Kayoko Umezawa, and Hitoshi Sato. Selectivity coefficients for ion-selective electrodes: Recommended methods for reporting ka, bpot values (technical report). *Pure and applied chemistry*, 67(3):507–518, 1995.

[43] Nadine Wirkuttis and Hadas Klein. Artificial intelligence in cybersecurity. *Cyber, Intelligence, and Security*, 1(1):103–119, 2017.

[44] Alshaibi Ahmed Jamal, Al-Ani Mustafa Majid, Anton Konev, Tatiana Kosachenko, and Alexander Shelupanov. A review on security analysis of cyber physical systems using machine learning. *Materials Today: Proceedings*, 80:2302–2306, 2023.

[45] Muna Al-Hawawreh, Nour Moustafa, Sahil Garg, and M Shamim Hossain. Deep learning-enabled threat intelligence scheme in the internet of things networks. *IEEE Transactions on Network Science and Engineering*, 8(4):2968–2981, 2020.

[46] Ankit Thakkar and Ritika Lohiya. A survey on intrusion detection system: feature selection, model, performance measures, application perspective, challenges, and future research directions. *Artificial Intelligence Review*, 55(1):453–563, 2022.

[47] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Mądry, Bo Li, and Tom Goldstein. Dataset security for machine

learning: Data poisoning, backdoor attacks, and defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1563–1580, 2022.

[48] Edward A Lee. Cyber physical systems: Design challenges. In *2008 11th IEEE international symposium on object and component-oriented real-time distributed computing (ISORC)*, pages 363–369. IEEE, 2008.

[49] Alaeddin Bobat, Tolga Gezgin, and Hüseyin Aslan. The scada system applications in management of yuvacik dam and reservoir. *Desalination and Water Treatment*, 54(8): 2108–2119, 2015.

[50] Elyahu Alperovits and Uri Shamir. Design of optimal water distribution systems. *Water resources research*, 13(6):885–900, 1977.

[51] Jill Slay and Michael Miller. Lessons learned from the maroochy water breach. In *International conference on critical infrastructure protection*, pages 73–82. Springer, 2007.

[52] DHS. Presidential policy directive—critical infrastructure security and resilience. *Press Release, February*, 12, 2013.

[53] WH. Presidential executive order on strengthening the cybersecurity of federal networks and critical infrastructure. Technical report, The White House, 2017. Available at: https://www.whitehouse.gov/presidential-actions.

[54] DHS ICS-CERT. Us department of homeland security–industrial control systems-cyber emergency response team, year in review. 2015.

[55] USEPA. Information about public water systems. Washington, DC: USEPA, 2019.

[56] B Walton. Water sector prepares for cyberattacks. *Circle of Blue*, 9, 2016.

[57] MD Cava. Uber to pay $148 million over undisclosed data breach that ex-ceo paid hackers to keep quiet. *USA Today*, 2018.

[58] GT Rubin. Many company hacks go undisclosed to sec despite regulator efforts. *The Wall Street Journal*, 2019.

[59] Sanika Ratnaparkhi, Suvaid Khan, Chandrakala Arya, Shailesh Khapre, Prabhishek Singh, Manoj Diwakar, and Achyut Shankar. Withdrawn: Smart agriculture sensors in iot: A review, 2020.

[60] Lucio Colizzi, Danilo Caivano, Carmelo Ardito, Giuseppe Desolda, Annamaria Castrignanò, Maristella Matera, Raj Khosla, Dimitrios Moshou, Kun-Mean Hou, François Pinet, et al. Introduction to agricultural iot. In *Agricultural Internet of Things and Decision Support for Precision Smart Farming*, pages 1–33. Elsevier, 2020.

[61] Fiona C McKenzie and John Williams. Sustainable food production: constraints, challenges and choices by 2050. *Food Security*, 7:221–233, 2015.

[62] Virgil Chichernea. The use of decision support systems (dss) in smart city planning and management. *Journal of Information Systems & Operations Management*, 8(2), 2014.

[63] Nikesh Gondchawar, RS Kawitkar, et al. Iot based smart agriculture. *International Journal of advanced research in Computer and Communication Engineering*, 5(6):838–842, 2016.

[64] Konstantinos Demestichas, Nikolaos Peppes, and Theodoros Alexakis. Survey on security threats in agricultural iot and smart farming. *Sensors*, 20(22):6458, 2020.

[65] Alireza Gheisi, Mark Forsyth, and Gh Naser. Water distribution systems reliability: A

review of research literature. *Journal of Water Resources Planning and Management*, 142(11):04016047, 2016.

[66] Arti Malviya and Dipika Jaspal. Artificial intelligence as an upcoming technology in wastewater treatment: a comprehensive review. *Environmental Technology Reviews*, 10(1):177–187, 2021.

[67] Robert K Bastian, Peter E Shanaghan, and Brian P Thompson. Use of wetlands for municipal wastewater treatment and disposal–regulatory issues and epa policies. *Constructed Wetlands for Wastewater Treatment*, pages 265–278, 2020.

[68] Fi-John Chang, Li-Chiu Chang, and Jui-Fa Chen. Artificial intelligence techniques in hydrology and water resources management, 2023.

[69] Anthony Njuguna Matheri, Belaid Mohamed, Freeman Ntuli, Esther Nabadda, and Jane Catherine Ngila. Sustainable circularity and intelligent data-driven operations and control of the wastewater treatment plant. *Physics and Chemistry of the Earth, Parts A/B/C*, 126:103152, 2022.

[70] Yiqi Liu, Pedram Ramin, Xavier Flores-Alsina, and Krist V Gernaey. Transforming data into actionable knowledge for fault detection, diagnosis and prognosis in urban wastewater systems with ai techniques: A mini-review. *Process Safety and Environmental Protection*, 172:501–512, 2023.

[71] Chhayly Sreng. *Trustworthy Soft Sensing in Water Supply Systems using Deep Learning*. PhD thesis, Virginia Tech, 2024.

[72] Gulzar Alam, Ihsanullah Ihsanullah, Mu Naushad, and Mika Sillanpää. Applications of artificial intelligence in water treatment for optimization and automation of adsorption

processes: Recent advances and prospects. *Chemical Engineering Journal*, 427:130011, 2022.

[73] Soma Safeer, Ravi P Pandey, Bushra Rehman, Tuba Safdar, Iftikhar Ahmad, Shadi W Hasan, and Asmat Ullah. A review of artificial intelligence in water purification and wastewater treatment: Recent advancements. *Journal of Water Process Engineering*, 49:102974, 2022.

[74] J Flores, B Arcay, and J Arias. An intelligent system for distributed control of an anaerobic wastewater treatment process. *Engineering Applications of Artificial Intelligence*, 13(4):485–494, 2000.

[75] Rejoan Kobir Nishan, Shapla Akter, Rayhanul Islam Sony, Md Mozammal Hoque, Meratun Junnut Anee, and Amzad Hossain. Development of an iot-based multi-level system for real-time water quality monitoring in industrial wastewater. *Discover Water*, 4(1):43, 2024.

[76] Anita Mohanty, Subrat Kumar Mohanty, and Ambarish G Mohapatra. Real-time monitoring and fault detection in ai-enhanced wastewater treatment systems. In *The AI Cleanse: Transforming Wastewater Treatment Through Artificial Intelligence: Harnessing Data-Driven Solutions*, pages 165–199. Springer, 2024.

[77] Jonathan M Aitken, Mathew H Evans, Rob Worley, Sarah Edwards, Rui Zhang, Tony Dodd, Lyudmila Mihaylova, and Sean R Anderson. Simultaneous localization and mapping for inspection robots in water and sewer pipe networks: A review. *IEEE access*, 9:140173–140198, 2021.

[78] Vitor Sousa, José P Matos, and Natércia Matias. Evaluation of artificial intelligence tool performance and uncertainty for predicting sewer structural condition. *Automation in Construction*, 44:84–91, 2014.

[79] Rakiba Rayhana, Yutong Jiao, Amirhossein Zaji, and Zheng Liu. Automated vision systems for condition assessment of sewer and water pipelines. *IEEE Transactions on Automation Science and Engineering*, 18(4):1861–1878, 2020.

[80] Frank R Spellman. *Water & wastewater infrastructure: Energy efficiency and sustainability*. Crc Press, 2013.

[81] Avi Ostfeld, James G Uber, Elad Salomons, Jonathan W Berry, William E Hart, Cindy A Phillips, Jean-Paul Watson, Gianluca Dorini, Philip Jonkergouw, Zoran Kapelan, et al. The battle of the water sensor networks (bwsn): A design challenge for engineers and algorithms. *Journal of water resources planning and management*, 134(6):556–568, 2008.

[82] William E Hart and Regan Murray. Review of sensor placement strategies for contamination warning systems in drinking water distribution systems. *Journal of Water Resources Planning and Management*, 136(6):611–619, 2010.

[83] Weijiao Gong, Mahima Agumbe Suresh, Lidia Smith, Avi Ostfeld, Radu Stoleru, Amin Rasekh, and M Katherine Banks. Mobile sensor networks for optimal leak and backflow detection and localization in municipal water networks. *Environmental modelling & software*, 80:306–321, 2016.

[84] Alessandro Cominola, Matteo Giuliani, Dario Piga, Andrea Castelletti, and Andrea Emilio Rizzoli. Benefits and challenges of using smart meters for advancing residential water demand modeling and management: A review. *Environmental Modelling & Software*, 72:198–214, 2015.

[85] Amin Rasekh, Amin Hassanzadeh, Shaan Mulchandani, Shimon Modi, and M Katherine Banks. Smart water networks and cyber security, 2016.

[86] James Andrew Lewis. *Assessing the risks of cyber terrorism, cyber war and other cyber threats*. Center for Strategic & International Studies Washington, DC, 2002.

[87] Richard Horta. The city of boca raton: A case study in water utility cybersecurity. *Journal-American Water Works Association*, 99(3):48–50, 2007.

[88] Jack Moyer, Rick Dakin, Richard Hewman, and Daniel Groves. The case for cyber security in the water sector. *Journal-American Water Works Association*, 101(12): 30–32, 2009.

[89] Marine Board, National Research Council, et al. *Macondo well deepwater horizon blowout: Lessons for improving offshore drilling safety*. National Academies Press, 2012.

[90] Riccardo Taormina, Stefano Galelli, Nils Ole Tippenhauer, Elad Salomons, and Avi Ostfeld. Characterizing cyber-physical attacks on water distribution systems. *Journal of Water Resources Planning and Management*, 143(5):04017009, 2017.

[91] Luis Garcia, Ferdinand Brasser, Mehmet Hazar Cintuglu, Ahmad-Reza Sadeghi, Osama A Mohammed, and Saman A Zonouz. Hey, my malware knows physics! attacking plcs with physical model aware rootkit. In *NDSS*, 2017.

[92] Yilin Mo and Bruno Sinopoli. Secure control against replay attacks. In *2009 47th annual Allerton conference on communication, control, and computing (Allerton)*, pages 911–918. IEEE, 2009.

[93] André Teixeira, Iman Shames, Henrik Sandberg, and Karl H Johansson. Revealing stealthy attacks in control systems. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1806–1813. IEEE, 2012.

[94] Riccardo Taormina and Stefano Galelli. Deep-learning approach to the detection and localization of cyber-physical attacks on water distribution systems. *Journal of Water Resources Planning and Management*, 144(10):04018065, 2018.

[95] Ll Corominas, Manel Garrido-Baserba, Kris Villez, Gustaf Olsson, Ulises Cortés, and Manel Poch. Transforming data into knowledge for improved wastewater treatment operation: A critical review of techniques. *Environmental modelling & software*, 106: 89–103, 2018.

[96] Titilayo Abimbola Owolabi, Saeed Reza Mohandes, and Tarek Zayed. Investigating the impact of sewer overflow on the environment: A comprehensive literature review paper. *Journal of Environmental Management*, 301:113810, 2022.

[97] Manfred Schütze, Alberto Campisano, Hubert Colas, Wolfgang Schilling, and Peter A Vanrolleghem. Real-time control of urban wastewater systems-where do we stand today? In *Global Solutions for Urban Drainage*, pages 1–17. 2002.

[98] Public Comment Start Date, Public Comment Expiration Date, and Martin Merz. Us environmental protection agency (epa) proposes to reissue a national pollutant discharge elimination system (npdes) general permit to discharge pollutants pursuant to the provisions of the clean water act (cwa) to. 2022.

[99] Adebayo Olatunbosun Sojobi and Tarek Zayed. Impact of sewer overflow on public health: A comprehensive scientometric analysis and systematic review. *Environmental research*, 203:111609, 2022.

[100] Kelly T Sanders and Michael E Webber. Evaluating the energy consumed for water use in the united states. *Environmental Research Letters*, 7(3):034034, 2012.

[101] Guangtao Fu, Yiwen Jin, Siao Sun, Zhiguo Yuan, and David Butler. The role of deep

learning in urban water management: A critical review. *Water Research*, page 118973, 2022.

[102] Liping Yang, Joshua Driscol, Sarigai Sarigai, Qiusheng Wu, Christopher D Lippitt, and Melinda Morgan. Towards synoptic water monitoring systems: a review of ai methods for automating water body detection and water quality monitoring using remote sensing. *Sensors*, 22(6):2416, 2022.

[103] K Sathya, K Nagarajan, G Carlin Geor Malar, S Rajalakshmi, and P Raja Lakshmi. A comprehensive review on comparison among effluent treatment methods and modern methods of treatment of industrial wastewater effluent from different sources. *Applied Water Science*, 12(4):70, 2022.

[104] Everette S. Gardner. Exponential smoothing: The state of the art. *Journal of Forecasting*, 4(1):1–28, 1985.

[105] Siddharth Arora and James W. Taylor. Rule-based autoregressive moving average models for forecasting load on special days: A case study for france. *European Journal of Operational Research*, 266(1):259–268, 2018.

[106] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[107] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. *Advances in neural information processing systems*, 9, 1997.

[108] Vaia I Kontopoulou, Athanasios D Panagopoulos, Ioannis Kakkos, and George K Mat-

sopoulos. A review of arima vs. machine learning approaches for time series forecasting in data driven networks. *Future Internet*, 15(8):255, 2023.

[109] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104, 2018.

[110] Jie Yan, Yongqian Liu, Shuang Han, Yimei Wang, and Shuanglei Feng. Reviews on uncertainty analysis of wind power forecasting. *Renewable and Sustainable Energy Reviews*, 52:1322–1330, 2015.

[111] Oussama Boussif, Ghait Boukachab, Dan Assouline, Stefano Massaroli, Tianle Yuan, Loubna Benabbou, and Yoshua Bengio. Improving* day-ahead* solar irradiance time series forecasting by leveraging spatio-temporal context. *Advances in Neural Information Processing Systems*, 36, 2024.

[112] Balamurali Murugesan, Kaushik Sarveswaran, Sharath M Shankaranarayana, Keerthi Ram, Mohanasankar Sivaprakasam, et al. A context based deep learning approach for unbalanced medical image segmentation. In *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, pages 1949–1953. IEEE, 2020.

[113] Adir Solomon, Mor Kertis, Bracha Shapira, and Lior Rokach. A deep learning framework for predicting burglaries based on multiple contextual factors. *Expert Systems with Applications*, 199:117042, 2022.

[114] Tom Boyle and Andrew Ravenscroft. Context and deep learning design. *Computers & Education*, 59(4):1224–1233, 2012.

[115] Hao Miao, Yan Fei, Senzhang Wang, Fang Wang, and Danyan Wen. Deep learning

based origin-destination prediction via contextual information fusion. *Multimedia Tools and Applications*, pages 1–17, 2022.

[116] Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The eleventh international conference on learning representations*, 2023.

[117] Gary Stein and Avelino J Gonzalez. Learning in context: enhancing machine learning with context-based reasoning. *Applied intelligence*, 41:709–724, 2014.

[118] Moshe Unger, Alexander Tuzhilin, and Amit Livne. Context-aware recommendations based on deep learning frameworks. *ACM Transactions on Management Information Systems (TMIS)*, 11(2):1–15, 2020.

[119] Tim N Palmer and David L T Anderson. The prospects for seasonal forecasting—a review paper. *Quarterly Journal of the Royal Meteorological Society*, 120(518):755–793, 1994.

[120] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[121] Md Nazmul Kabir Sikder and Feras A Batarseh. Outlier detection using ai: A survey. *arXiv preprint arXiv:2112.00588*, 2021.

[122] Saurabh Amin, Alvaro A Cárdenas, and S Shankar Sastry. Safe and secure networked control systems under denial-of-service attacks. In *Hybrid Systems: Computation and Control: 12th International Conference, HSCC 2009, San Francisco, CA, USA, April 13-15, 2009. Proceedings 12*, pages 31–45. Springer, 2009.

[123] Nilufer Tuptuk, Peter Hazell, Jeremy Watson, and Stephen Hailes. A systematic review of the state of cyber-security in water systems. *Water*, 13(1):81, 2021.

[124] Nilufer Tuptuk, Peter Hazell, Jeremy Watson, and Stephen Hailes. A systematic review of the state of cyber-security in water systems. *Water*, 13(1):81, 2021.

[125] Petar Radanliev, David De Roure, Max Van Kleek, Omar Santos, and Uchenna Ani. Artificial intelligence in cyber physical systems. *AI & society*, 36(3):783–796, 2021.

[126] Andreas Kamilaris and Francesc X. Prenafeta-Boldú. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90, 2018. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2018.02.016. URL https://www.sciencedirect.com/science/article/pii/S0168169917308803.

[127] Konstantinos G. Liakos, Patrizia Busato, Dimitrios Moshou, Simon Pearson, and Dionysis Bochtis. Machine learning in agriculture: A review. *Sensors*, 18(8), 2018. ISSN 1424-8220. doi: 10.3390/s18082674. URL https://www.mdpi.com/1424-8220/18/8/2674.

[128] Robin Gebbers and Viacheslav I. Adamchuk. Precision agriculture and food security. *Science*, 327(5967):828–831, 2010. ISSN 0036-8075. doi: 10.1126/science.1183899. URL https://science.sciencemag.org/content/327/5967/828.

[129] Munisamy Gopinath, Feras A Batarseh, and Jayson Beckman. Machine learning in gravity models: An application to agricultural trade. Working Paper 27151, National Bureau of Economic Research, May 2020. URL http://www.nber.org/papers/w27151.

[130] Imlak Shaikh. On the relationship between economic policy uncertainty and the implied volatility index. *Sustainability*, 11(6):1628, 2019.

[131] Catherine M Brown, Johanna Vostok, Hillary Johnson, Meagan Burns, Radhika Gharpure, Samira Sami, Rebecca T Sabo, Noemi Hall, Anne Foreman, Petra L Schubert, et al. Outbreak of sars-cov-2 infections, including covid-19 vaccine breakthrough infections, associated with large public gatherings—barnstable county, massachusetts, july 2021. *Morbidity and Mortality Weekly Report*, 70(31):1059, 2021.

[132] Hugo Storm, Kathy Baylis, and Thomas Heckelei. Machine learning in agricultural and applied economics. *European Review of Agricultural Economics*, 47(3):849–892, 08 2019. ISSN 0165-1587. doi: 10.1093/erae/jbz033. URL https://doi.org/10.1093/erae/jbz033.

[133] Sjaak Wolfert, Lan Ge, Cor Verdouw, and Marc-Jeroen Bogaardt. Big data in smart farming – a review. *Agricultural Systems*, 153:69–80, 2017. ISSN 0308-521X. doi: https://doi.org/10.1016/j.agsy.2017.01.023. URL https://www.sciencedirect.com/science/article/pii/S0308521X16303754.

[134] Feras A Batarseh, Laura Freeman, and Chih-Hao Huang. A survey on artificial intelligence assurance. *Journal of Big Data*, 8(1):60, 2021. ISSN 2196-1115. doi: 10.1186/s40537-021-00445-7. URL https://doi.org/10.1186/s40537-021-00445-7.

[135] Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. Hatexplain: A benchmark dataset for explainable hate speech detection. In *AAAI*, 2021.

[136] MD Nazmul Kabir Sikder, Feras A. Batarseh, Pei Wang, and Nitish Gorentala. Model-agnostic scoring methods for artificial intelligence assurance. In *2022 IEEE 29th Annual Software Technology Conference (STC)*, pages 9–18, 2022. doi: 10.1109/STC55697.2022.00011.

[137] Fahd A Alhaidari and Ezaz Mohammed Al-Dahasi. New approach to determine ddos attack patterns on scada system using machine learning. In *2019 International conference on computer and information sciences (ICCIS)*, pages 1–6. IEEE, 2019.

[138] Chaminda Hewage. Opportunities, challenges and strategies for integrating cyber security and safety in engineering practice. 2021.

[139] Aymen Abid, Abdennaceur Kachouri, and Adel Mahfoudhi. Outlier detection for wireless sensor networks using density-based clustering approach. *IET Wireless Sensor Systems*, 7(4):83–90, 2017.

[140] Hailin Feng, Lun Liang, and Hua Lei. Distributed outlier detection algorithm based on credibility feedback in wireless sensor networks. *IET Communications*, 11(8):1291–1296, 2017.

[141] Haibin Zhang, Jiajia Liu, and Cheng Zhao. Distance based method for outlier detection of body sensor networks. *EAI Endorsed Transactions on Wireless Spectrum*, 2(7), 2016.

[142] Nauman Shahid, Ijaz Haider Naqvi, and Saad Bin Qaisar. Characteristics and classification of outlier detection techniques for wireless sensor networks in harsh environments: a survey. *Artificial Intelligence Review*, 43:193–228, 2015.

[143] Goverdhan Singh, Florent Masseglia, Céline Fiot, Alice Marascu, and Pascal Poncelet. Mining common outliers for intrusion detection. *Advances in Knowledge Discovery and Management*, pages 217–234, 2010.

[144] Jinita Tamboli and Madhu Shukla. A survey of outlier detection algorithms for data streams. In *2016 3rd international conference on computing for sustainable global development (INDIACom)*, pages 3535–3540. IEEE, 2016.

[145] Madhu Shukla, YP Kosta, and Prashant Chauhan. Analysis and evaluation of outlier detection algorithms in data streams. In *2015 International Conference on Computer, Communication and Control (IC4)*, pages 1–8. IEEE, 2015.

[146] Luan Tran, Liyue Fan, and Cyrus Shahabi. Distance-based outlier detection in data streams. *Proceedings of the VLDB Endowment*, 9(12):1089–1100, 2016.

[147] Manish Gupta, Jing Gao, Charu C Aggarwal, and Jiawei Han. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and data Engineering*, 26 (9):2250–2267, 2013.

[148] Silvia Cateni, Valentina Colla, Marco Vannucci, Jesus Aramburo, and Antonio Ramirez Trevino. Outlier detection methods for industrial applications. *Advances in Robotics, Automation and Control*, pages 265–282, 2008.

[149] Tan Xiao, Chao Zhang, and Hongbin Zha. Learning to detect anomalies in surveillance video. *IEEE Signal Processing Letters*, 22(9):1477–1481, 2015.

[150] Saeed Ghanbari, Ali B Hashemi, and Cristiana Amza. Stage-aware anomaly detection through tracking log points. In *Proceedings of the 15th international middleware conference*, pages 253–264, 2014.

[151] Ciro D'Urso. Experience: glitches in databases, how to ensure data quality by outlier detection techniques. *Journal of Data and Information Quality (JDIQ)*, 7(3):1–22, 2016.

[152] Kamal Chenaoua, Fatih Kurugollu, and Ahmed Bouridane. Data cleaning and outlier removal: Application in human skin detection. In *2014 5th European Workshop on Visual Information Processing (EUVIP)*, pages 1–6. IEEE, 2014.

[153] Stephen Ranshous, Shitian Shen, Danai Koutra, Steve Harenberg, Christos Faloutsos, and Nagiza F Samatova. Anomaly detection in dynamic networks: a survey. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(3):223–247, 2015.

[154] Mohammad Braei and Sebastian Wagner. Anomaly detection in univariate time-series: A survey on the state-of-the-art. *arXiv preprint arXiv:2004.00433*, 2020.

[155] Kwei-Herng Lai, Daochen Zha, Guanchu Wang, Junjie Xu, Yue Zhao, Devesh Kumar, Yile Chen, Purav Zumkhawaka, Minyang Wan, Diego Martinez, et al. Tods: An automated time series outlier detection system. In *Proceedings of the aaai conference on artificial intelligence*, volume 35, pages 16060–16062, 2021.

[156] Ali S Hadi, AHM Rahmatullah Imon, and Mark Werner. Detection of outliers. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(1):57–70, 2009.

[157] Di Yang, Elke A Rundensteiner, and Matthew O Ward. Neighbor-based pattern detection for windows over streaming data. In *Proceedings of the 12th international conference on extending database technology: advances in database technology*, pages 529–540, 2009.

[158] Shohei Hido, Yuta Tsuboi, Hisashi Kashima, Masashi Sugiyama, and Takafumi Kanamori. Statistical outlier detection using direct density ratio estimation. *Knowledge and information systems*, 26:309–336, 2011.

[159] Markus Goldstein and Andreas Dengel. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: poster and demo track*, 1:59–63, 2012.

[160] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: iden-

tifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[161] Jian Tang, Zhixiang Chen, Ada Wai-Chee Fu, and David W Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *Advances in Knowledge Discovery and Data Mining: 6th Pacific-Asia Conference, PAKDD 2002 Taipei, Taiwan, May 6–8, 2002 Proceedings 6*, pages 535–548. Springer, 2002.

[162] Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. Loop: local outlier probabilities. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1649–1652, 2009.

[163] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[164] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[165] George Karypis, Eui-Hong Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *computer*, 32(8):68–75, 1999.

[166] Ke Zhang, Marcus Hutter, and Huidong Jin. A new local distance-based outlier detection approach for scattered real-world data. In *Advances in Knowledge Discovery and Data Mining: 13th Pacific-Asia Conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 Proceedings 13*, pages 813–822. Springer, 2009.

[167] Huaming Huang, Kishan Mehrotra, and Chilukuri K Mohan. Rank-based outlier detection. *Journal of Statistical Computation and Simulation*, 83(3):518–531, 2013.

[168] Guilherme O Campos, Arthur Zimek, and Wagner Meira. An unsupervised boosting strategy for outlier detection ensembles. In *Advances in Knowledge Discovery and Data Mining: 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part I 22*, pages 564–576. Springer, 2018.

[169] Shebuti Rayana and Leman Akoglu. Less is more: Building selective anomaly ensembles. *Acm transactions on knowledge discovery from data (tkdd)*, 10(4):1–33, 2016.

[170] Jayanta K Dutta, Bonny Banerjee, and Chandan K Reddy. Rods: Rarity based outlier detection in a sparse coding framework. *IEEE Transactions on Knowledge and Data Engineering*, 28(2):483–495, 2015.

[171] Charu C Aggarwal and Philip S Yu. An effective and efficient algorithm for high-dimensional outlier detection. *The VLDB journal*, 14:211–221, 2005.

[172] Gheorghe Sebestyen, Anca Hangan, and Zoltan Czako. Anomaly detection in water supply infrastructure systems. In *2021 23rd International Conference on Control Systems and Computer Science (CSCS)*, pages 349–355. IEEE, 2021.

[173] Steve Wise, Joseph Braden, Danielle Ghalayini, Joseph Grant, Chris Kloss, Edward MacMullan, Stephanie Morse, Franco Montalto, Dan Nees, David Nowak, et al. Integrating valuation methods to recognize green infrastructure's multiple benefits. In *Low impact development 2010: Redefining water in the city*, pages 1123–1143. 2010.

[174] Stefano Longo, Benedetto Mirko d'Antoni, Michael Bongards, Antonio Chaparro, Andreas Cronrath, Francesco Fatone, Juan M Lema, Miguel Mauricio-Iglesias, Ana Soares, and Almudena Hospido. Monitoring and diagnosis of energy consumption in wastewater treatment plants. a state of the art and proposals for improvement. *Applied energy*, 179:1251–1268, 2016.

[175] Nawab Khan, Ram L Ray, Ghulam Raza Sargani, Muhammad Ihtisham, Muhammad Khayyam, and Sohaib Ismail. Current progress and future prospects of agriculture technology: Gateway to sustainable agriculture. *Sustainability*, 13(9):4883, 2021.

[176] Feras A. Batarseh, Mehmet Oguz Yardimci, Ryu Suzuki, Md Nazmul Kabir Sikder, Zhiwu Wang, and WanYi Mao. Realtime management of wastewater treatment plants using ai. Technical report, Virginia Tech & DC Water, 2022. Available at: https://www.waterrf.org/news/2022-intelligent-water-systems-challenge.

[177] D Trump. Executive order on strengthening the cybersecurity of federal networks and critical infrastructure. *The White House, Office of the Press Secretary*, 2017.

[178] Sridhar Adepu and Aditya Mathur. Introducing cyber security at the design stage of public infrastructures: A procedure and case study. In *Complex systems design & management asia*, pages 75–94. Springer, 2016.

[179] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146. PMLR, 2018.

[180] Frances Robles and Nicole Perlroth. Dangerous stuff': Hackers tried to poison water supply of florida town. *The New York Times*, 2021.

[181] Kevin Collier. 50,000 security disasters waiting to happen: The problem of america's water supplies. *Available from NBC News: https://www. nbcnews. com/tech/security/hacker-tried-poison-calif-water-supply-was-easyentering-password-rcna1206*, 2021.

[182] Savita Mohurle and Manisha Patil. A brief study of wannacry threat: Ransomware

attack 2017. *International journal of advanced research in computer science*, 8(5): 1938–1940, 2017.

[183] Qiao Yan, F Richard Yu, Qingxiang Gong, and Jianqiang Li. Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges. *IEEE communications surveys & tutorials*, 18(1):602–622, 2015.

[184] Jing Dong, Wei Wang, and Tieniu Tan. Casia image tampering detection evaluation database. In *2013 IEEE China summit and international conference on signal and information processing*, pages 422–426. IEEE, 2013.

[185] David Brumley and Dan Boneh. Remote timing attacks are practical. *Computer Networks*, 48(5):701–716, 2005.

[186] Christian W Probst, Jeffrey Hunker, Matt Bishop, and Dieter Gollmann. *Insider threats in cyber security*, volume 49. Springer Science & Business Media, 2010.

[187] Sridhar Adepu and Aditya Mathur. Distributed attack detection in a water treatment plant: Method and case study. *IEEE Transactions on Dependable and Secure Computing*, 18(1):86–99, 2018.

[188] Feras A Batarseh, Mehmet Oguz Yardimci, Ryu Suzuki, Md Nazmul Kabir Sikder, Zhiwu Wang, and Wan-Yi Mao. Realtime management of wastewater treatment plants using ai.

[189] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[190] Shiyu Chang, Yang Zhang, Wei Han, Mo Yu, Xiaoxiao Guo, Wei Tan, Xiaodong Cui, Michael Witbrock, Mark A Hasegawa-Johnson, and Thomas S Huang. Dilated recurrent neural networks. *Advances in neural information processing systems*, 30, 2017.

[191] Sandra Wachter, Brent D. Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *CoRR*, abs/1711.00399, 2017.

[192] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. *CoRR*, abs/1602.04938, 2016.

[193] Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. Interpretml: A unified framework for machine learning interpretability. *CoRR*, abs/1909.09223, 2019.

[194] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *ArXiv*, abs/1705.07874, 2017.

[195] Jerome H Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. doi: 10.1214/aos/1013203451. URL https://doi.org/10.1214/aos/1013203451.

[196] Francesca Rossi and Nicholas Mattei. Building ethically bounded ai. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9785–9789, 2019.

[197] Sina Aghaei, Mohammad Javad Azizi, and Phebe Vayanos. Learning optimal and fair decision trees for non-discriminative decision-making. *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances*

in Artificial Intelligence, EAAI 2019, pages 1418–1426, 2019. ISSN 2159-5399. doi: 10.1609/aaai.v33i01.33011418.

[198] Tony Loeser and Yumi Iwasaki. Safety Verification Proofs for Physical Systems.

[199] Ugur Kuter and Jennifer Golbeck. SUNNY: A new algorithm for trust inference in social networks using probabilistic confidence models. *Proceedings of the National Conference on Artificial Intelligence*, 2:1377–1382, 2007.

[200] Lloyd S Shapley. *A Value for N-Person Games*. Princeton University Press, Santa Monica, CA, 1953. doi: 10.1515/9781400881970-018.

[201] Sheikh Rabiul Islam, William Eberle, and Sheikh K. Ghafoor. Towards quantification of explainability in explainable artificial intelligence methods. *CoRR*, abs/1911.10104, 2019.

[202] Feras A Batarseh and Laura Freeman. *AI Assurance: Towards Trustworthy, Explainable, Safe, and Ethical AI*. Academic Press, 2022.

[203] Dragoljub Pokrajac, Aleksandar Lazarevic, and Longin Jan Latecki. Incremental local outlier detection for data streams. In *2007 IEEE symposium on computational intelligence and data mining*, pages 504–515. IEEE, 2007.

[204] Jun Gao, Weiming Hu, Zhongfei Zhang, Xiaoqin Zhang, and Ou Wu. Rkof: robust kernel-based local outlier detection. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 270–283. Springer, 2011.

[205] Arnold P Boedihardjo, Chang-Tien Lu, and Feng Chen. Fast adaptive kernel density estimator for data streams. *Knowledge and Information Systems*, 42:285–317, 2015.

[206] Xiu-ming Tang, Rong-xiang Yuan, and Jun Chen. Outlier detection in energy disaggregation using subspace learning and gaussian mixture model. *International Journal of Control and Automation*, 8(8):161–170, 2015.

[207] Paul Inuwa Dalatu, Anwar Fitrianto, and Aida Mustapha. A comparative study of linear and nonlinear regression models for outlier detection. In *Recent Advances on Soft Computing and Data Mining: The Second International Conference on Soft Computing and Data Mining (SCDM-2016), Bandung, Indonesia, August 18-20, 2016 Proceedings Second*, pages 316–326. Springer, 2017.

[208] Haizhou Du et al. Robust local outlier detection. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 116–123. IEEE, 2015.

[209] Ricardo JGB Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 10(1):1–51, 2015.

[210] Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. Copod: copula-based outlier detection. In *2020 IEEE international conference on data mining (ICDM)*, pages 1118–1123. IEEE, 2020.

[211] Xiao Qin, Lei Cao, Elke A Rundensteiner, and Samuel Madden. Scalable kernel density estimation-based local outlier detection over large data streams. In *Proceedings of the 22nd International Conference on Extending Database Technology (EDBT)*, 2019.

[212] Kai Ming Ting, Bi-Cun Xu, Takashi Washio, and Zhi-Hua Zhou. Isolation distributional kernel: A new tool for kernel based anomaly detection. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 198–206, 2020.

[213] Wen Jin, Anthony KH Tung, Jiawei Han, and Wei Wang. Ranking outliers using symmetric neighborhood relationship. In *Advances in Knowledge Discovery and Data Mining: 10th Pacific-Asia Conference, PAKDD 2006, Singapore, April 9-12, 2006. Proceedings 10*, pages 577–593. Springer, 2006.

[214] Rana Momtaz, Nesma Mohssen, and Mohammad A Gowayyed. Dwof: A robust density-based outlier detection approach. In *Pattern Recognition and Image Analysis: 6th Iberian Conference, IbPRIA 2013, Funchal, Madeira, Portugal, June 5-7, 2013. Proceedings 6*, pages 517–525. Springer, 2013.

[215] Spiros Papadimitriou, Hiroyuki Kitagawa, Phillip B Gibbons, and Christos Faloutsos. Loci: Fast outlier detection using the local correlation integral. In *Proceedings 19th international conference on data engineering (Cat. No. 03CH37405)*, pages 315–326. IEEE, 2003.

[216] Dongmei Ren, Imad Rahal, William Perrizo, and Kirk Scott. A vertical distance-based outlier detection method with local pruning. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 279–284, 2004.

[217] Fabian Keller, Emmanuel Muller, and Klemens Bohm. Hics: High contrast subspaces for density-based outlier ranking. In *2012 IEEE 28th international conference on data engineering*, pages 1037–1048. IEEE, 2012.

[218] Ke Wu, Kun Zhang, Wei Fan, Andrea Edwards, and S Yu Philip. Rs-forest: A rapid density estimator for streaming anomaly detection. In *2014 IEEE international conference on data mining*, pages 600–609. IEEE, 2014.

[219] Ji Zhang, Wynne Hsu, and Mong Li Lee. Clustering in dynamic spatial databases. *Journal of intelligent information systems*, 24:5–27, 2005.

[220] Charu C Aggarwal, Jiawei Han, Jianyong Wang, and Philip S Yu. A framework for projected clustering of high dimensional data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 852–863, 2004.

[221] Feng Cao, Martin Estert, Weining Qian, and Aoying Zhou. Density-based clustering over an evolving data stream with noise. In *Proceedings of the 2006 SIAM international conference on data mining*, pages 328–339. SIAM, 2006.

[222] Yixin Chen and Li Tu. Density-based clustering for real-time stream data. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, 2007.

[223] Ira Assent, Philipp Kranen, Corinna Baldauf, and Thomas Seidl. Anyout: Anytime outlier detection on streaming data. In *Database Systems for Advanced Applications: 17th International Conference, DASFAA 2012, Busan, South Korea, April 15-19, 2012, Proceedings, Part I 17*, pages 228–242. Springer, 2012.

[224] Tao Liu, Yan Zhou, Zhifeng Hu, and Zhijie Wang. A new clustering algorithm based on artificial immune system. In *2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, volume 2, pages 347–351. IEEE, 2008.

[225] Hossein Moradi Koupaie, Suhaimi Ibrahim, and Javad Hosseinkhani. Outlier detection in stream data by clustering method. *International Journal of Advanced Computer Science and Information Technology*, 2(3):25–34, 2013.

[226] Safal V Bhosale. A survey: outlier detection in streaming data using clustering approached. *IJCSIT) International Journal of Computer Science and Information Technologies*, 5:6050–6053, 2014.

[227] Masud Moshtaghi, James C Bezdek, Timothy C Havens, Christopher Leckie, Shanika

Karunasekera, Sutharshan Rajasegarar, and Marimuthu Palaniswami. Streaming analysis in wireless sensor networks. *Wireless Communications and Mobile Computing*, 14 (9):905–921, 2014.

[228] Masud Moshtaghi, James C Bezdek, Christopher Leckie, Shanika Karunasekera, and Marimuthu Palaniswami. Evolving fuzzy rules for anomaly detection in data streams. *IEEE Transactions on Fuzzy Systems*, 23(3):688–700, 2014.

[229] Mahsa Salehi, Christopher A Leckie, Masud Moshtaghi, and Tharshan Vaithianathan. A relevance weighted ensemble model for anomaly detection in switching data streams. In *Advances in Knowledge Discovery and Data Mining: 18th Pacific-Asia Conference, PAKDD 2014, Tainan, Taiwan, May 13-16, 2014. Proceedings, Part II 18*, pages 461–473. Springer, 2014.

[230] Milad Chenaghlou, Masud Moshtaghi, Christopher Leckie, and Mahsa Salehi. An efficient method for anomaly detection in non-stationary data streams. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–6. IEEE, 2017.

[231] Jianhua Yin and Jianyong Wang. A model-based approach for text clustering with outlier detection. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 625–636. IEEE, 2016.

[232] Vikash Sehwag, Mung Chiang, and Prateek Mittal. Ssd: A unified framework for self-supervised outlier detection. *arXiv preprint arXiv:2103.12051*, 2021.

[233] Edwin M Knox and Raymond T Ng. Algorithms for mining distancebased outliers in large datasets. In *Proceedings of the international conference on very large data bases*, pages 392–403. Citeseer, 1998.

[234] Amol Ghoting, Srinivasan Parthasarathy, and Matthew Eric Otey. Fast mining of

distance-based outliers in high-dimensional datasets. *Data Mining and Knowledge Discovery*, 16:349–364, 2008.

[235] Gautam Bhattacharya, Koushik Ghosh, and Ananda S Chowdhury. Outlier detection using neighborhood rank difference. *Pattern Recognition Letters*, 60:24–31, 2015.

[236] Taurus T Dang, Henry YT Ngan, and Wei Liu. Distance-based k-nearest neighbors outlier detection method in large-scale traffic data. In *2015 IEEE International Conference on Digital Signal Processing (DSP)*, pages 507–510. IEEE, 2015.

[237] Xiaochun Wang, Xia Li Wang, Yongqiang Ma, and D Mitchell Wilkes. A fast mst-inspired knn-based outlier detection method. *Information Systems*, 48:89–112, 2015.

[238] Jinlong Huang, Qingsheng Zhu, Lijun Yang, and Ji Feng. A non-parameter outlier detection algorithm based on natural neighbor. *Knowledge-Based Systems*, 92:71–77, 2016.

[239] Jihyun Ha, Seulgi Seok, and Jong-Seok Lee. A precise ranking method for outlier detection. *Information Sciences*, 324:88–107, 2015.

[240] Bo Tang and Haibo He. A local density-based approach for outlier detection. *Neurocomputing*, 241:171–180, 2017.

[241] Stephen D Bay and Mark Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 29–38, 2003.

[242] Fabrizio Angiulli and Fabio Fassetti. Detecting distance-based outliers in streams of data. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 811–820, 2007.

[243] Fabrizio Angiulli and Fabio Fassetti. Distance-based outlier queries in data streams: the novel task and algorithms. *Data Mining and Knowledge Discovery*, 20:290–324, 2010.

[244] Maria Kontaki, Anastasios Gounaris, Apostolos N Papadopoulos, Kostas Tsichlas, and Yannis Manolopoulos. Continuous monitoring of distance-based outliers over data streams. In *2011 IEEE 27th International Conference on Data Engineering*, pages 135–146. IEEE, 2011.

[245] Keyan Cao, Lingxu Shi, Guoren Wang, Donghong Han, and Mei Bai. Density-based local outlier detection on uncertain data. In *International Conference on Web-Age Information Management*, pages 67–71. Springer, 2014.

[246] Aleksandar Lazarevic and Vipin Kumar. Feature bagging for outlier detection. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 157–166, 2005.

[247] Yue Zhao and Maciej K Hryniewicki. Dcso: dynamic combination of detector scores for outlier ensembles. *arXiv preprint arXiv:1911.10418*, 2019.

[248] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[249] Hoang Vu Nguyen, Hock Hee Ang, and Vivekanand Gopalkrishnan. Mining outliers with ensemble of heterogeneous detectors on random subspaces. In *Database Systems for Advanced Applications: 15th International Conference, DASFAA 2010, Tsukuba, Japan, April 1-4, 2010, Proceedings, Part I 15*, pages 368–383. Springer, 2010.

[250] Arthur Zimek, Ricardo JGB Campello, and Jörg Sander. Data perturbation for outlier

detection ensembles. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management*, pages 1–12, 2014.

[251] José Ramon Pasillas-Díaz and Sylvie Ratté. Bagged subspaces for unsupervised outlier detection. *Computational intelligence*, 33(3):507–523, 2017.

[252] Yue Zhao, Zain Nasrullah, Maciej K Hryniewicki, and Zheng Li. Lscp: Locally selective combination in parallel outlier ensembles. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 585–593. SIAM, 2019.

[253] Arthur Zimek, Matthew Gaudet, Ricardo JGB Campello, and Jörg Sander. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 428–436, 2013.

[254] Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C Suh, Ikkyun Kim, and Kuinam J Kim. A survey of deep learning-based network anomaly detection. *Cluster Computing*, 22:949–961, 2019.

[255] Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*, 2019.

[256] Jifu Zhang, Yiyong Jiang, Kai H Chang, Sulan Zhang, Jianghui Cai, and Lihua Hu. A concept lattice based outlier mining method in low-dimensional subspaces. *Pattern Recognition Letters*, 30(15):1434–1439, 2009.

[257] Emmanuel Müller, Matthias Schiffer, and Thomas Seidl. Statistical selection of relevant subspace projections for outlier ranking. In *2011 IEEE 27th international conference on data engineering*, pages 434–445. IEEE, 2011.

[258] Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. Outlier detection in axis-parallel subspaces of high dimensional data. In *Advances in Knowledge Discovery and Data Mining: 13th Pacific-Asia Conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 Proceedings 13*, pages 831–838. Springer, 2009.

[259] Bas Van Stein, Matthijs Van Leeuwen, and Thomas Bäck. Local subspace-based outlier detection using global neighbourhoods. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1136–1142. IEEE, 2016.

[260] Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. Toward supervised anomaly detection. *Journal of Artificial Intelligence Research*, 46:235–262, 2013.

[261] Shubhomoy Das, Md Rakibul Islam, Nitthilan Kannappan Jayakodi, and Janardhan Rao Doppa. Active anomaly detection via ensembles: Insights, algorithms, and interpretability. *arXiv preprint arXiv:1901.08930*, 2019.

[262] Daochen Zha, Kwei-Herng Lai, Mingyang Wan, and Xia Hu. Meta-aad: Active anomaly detection with deep reinforcement learning. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 771–780. IEEE, 2020.

[263] HDK Moonesinghe and Pang-Ning Tan. Outrank: a graph-based outlier detection framework using random walk. *International Journal on Artificial Intelligence Tools*, 17(01):19–36, 2008.

[264] Chao Wang, Hui Gao, Zhen Liu, and Yan Fu. Outlier detection using diverse neighborhood graphs. In *2018 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pages 58–62. IEEE, 2018.

[265] Jinghui Chen, Saket Sathe, Charu Aggarwal, and Deepak Turaga. Outlier detection

with autoencoder ensembles. In *Proceedings of the 2017 SIAM international conference on data mining*, pages 90–98. SIAM, 2017.

[266] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 1285–1298, 2017.

[267] Daniel Sobien, Mehmet O Yardimci, Minh BT Nguyen, Wan-Yi Mao, Vinita Fordham, Abdul Rahman, Susan Duncan, and Feras A Batarseh. Ai for cyberbiosecurity in water systems—a survey. In *Cyberbiosecurity*, pages 217–263. Springer, 2023.

[268] Jian-hua Li. Cyber security meets artificial intelligence: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(12):1462–1474, 2018.

[269] RSRS Jayakrishnan, R Srinivasan, C Santhi, and JG Arnold. Advances in the application of the swat model for water resources management. *Hydrological Processes: An International Journal*, 19(3):749–762, 2005.

[270] Hanan Hindy, David Brosset, Ethan Bayne, Amar Seeam, and Xavier Bellekens. Improving siem for critical scada water infrastructures using machine learning. In *Computer Security: ESORICS 2018 International Workshops, CyberICPS 2018 and SECPRE 2018, Barcelona, Spain, September 6–7, 2018, Revised Selected Papers 2*, pages 3–19. Springer, 2019.

[271] Marwan A Albahar, Ruaa A Al-Falluji, and Muhammad Binsawad. An empirical comparison on malicious activity detection using different neural network-based models. *IEEE Access*, 8:61549–61564, 2020.

[272] Azin Moradbeikie, Kamal Jamshidi, Ali Bohlooli, Jordi Garcia, and Xavi Masip-Bruin.

An iiot based ics to improve safety through fast and accurate hazard detection and differentiation. *IEEE access*, 8:206942–206957, 2020.

[273] Abhijeet Sahu, Zeyu Mao, Patrick Wlazlo, Hao Huang, Katherine Davis, Ana Goulart, and Saman Zonouz. Multi-source multi-domain data fusion for cyberattack detection in power systems. *IEEE Access*, 9:119118–119138, 2021.

[274] Luca Faramondi, Francesco Flammini, Simone Guarino, and Roberto Setola. A hardware-in-the-loop water distribution testbed dataset for cyber-physical security testing. *IEEE Access*, 9:122385–122396, 2021.

[275] Paola Perrone, Francesco Flammini, and Roberto Setola. Machine learning for threat recognition in critical cyber-physical systems. In *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 298–303. IEEE, 2021.

[276] Iván Ramírez Morales, Daniel Rivero Cebrián, Enrique Fernández Blanco, and Alejandro Pazos Sierra. Early warning in egg production curves from commercial hens: A svm approach. *Computers and Electronics in Agriculture*, 121:169–179, 2016. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2015.12.009. URL https://www.sciencedirect.com/science/article/pii/S0168169915003919.

[277] Jaime Alonso, Alfonso Villa, and Antonio Bahamonde. Improved estimation of bovine weight trajectories using support vector machine classification. *Computers and Electronics in Agriculture*, 110:36–41, 2015. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2014.10.001. URL https://www.sciencedirect.com/science/article/pii/S0168169914002488.

[278] X.E. Pantazi, D. Moshou, T. Alexandridis, R.L. Whetton, and A.M. Mouazen. Wheat yield prediction using machine learning and advanced sensing techniques.

Computers and Electronics in Agriculture, 121:57–65, 2016. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2015.11.018. URL https://www.sciencedirect.com/science/article/pii/S0168169915003671.

[279] Munisamy Gopinath, Feras A. Batarseh, Jayson Beckman, Ajay Kulkarni, and Sei Jeong. International agricultural trade forecasting using machine learning. *Data and Policy*, 3:e1, 2021. doi: 10.1017/dap.2020.22.

[280] Anderson Monken, Flora Haberkorn, Munisamy Gopinath, Laura Freeman, and Feras Batarseh. Graph neural networks for modeling causality in international trade. *The International FLAIRS Conference Proceedings*, 34, Apr. 2021. doi: 10.32473/flairs.v34i1.128485. URL https://journals.flvc.org/FLAIRS/article/view/128485.

[281] Feras A. Batarseh, Munisamy Gopinath, Anderson Monken, and Zhengrong Gu. Public policymaking for international agricultural trade using association rules and ensemble machine learning. *Machine Learning with Applications*, 5:100046, 2021. ISSN 2666-8270. doi: https://doi.org/10.1016/j.mlwa.2021.100046. URL https://www.sciencedirect.com/science/article/pii/S2666827021000232.

[282] Fei Tony Liu, Kai Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery From Data - TKDD*, 6:1–39, 03 2012. doi: 10.1145/2133360.2133363.

[283] Madison J Williams, Md Nazmul Kabir Sikder, Pei Wang, Nitish Gorentala, Sai Gurrapu, and Feras A Batarseh. The application of artificial intelligence assurance in precision farming and agricultural economics. In *AI Assurance*, pages 501–529. Elsevier, 2023.

[284] Petar Radanliev, David De Roure, Max Van Kleek, Omar Santos, and Uchenna Ani. Artificial intelligence in cyber physical systems. *AI & society*, 36(3):783–796, 2021.

[285] Mohammad Sultan Mahmud, Joshua Zhexue Huang, and Xianghua Fu. Variational autoencoder-based dimensionality reduction for high-dimensional small-sample data classification. *International Journal of Computational Intelligence and Applications*, 19(01):2050002, 2020.

[286] Chong Zhou and Randy C Paffenroth. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 665–674, 2017.

[287] Jiayu Sun, Xinzhou Wang, Naixue Xiong, and Jie Shao. Learning sparse representation with variational auto-encoder for anomaly detection. *IEEE Access*, 6:33353–33361, 2018.

[288] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJJLHbb0-.

[289] Kunfeng Wang, Chao Gou, Yanjie Duan, Yilun Lin, Xinhu Zheng, and Fei-Yue Wang. Generative adversarial networks: introduction and outlook. *IEEE/CAA Journal of Automatica Sinica*, 4(4):588–598, 2017.

[290] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[291] Wen Zhou, Xiang-min Kong, Kai-li Li, Xiao-ming Li, Lin-lin Ren, Yong Yan, Yun Sha, Xue-ying Cao, and Xue-jun Liu. Attack sample generation algorithm based on data association group by gan in industrial control dataset. *Computer Communications*, 173:206–213, 2021.

[292] Md Hasan Shahriar, Nur Imtiazul Haque, Mohammad Ashiqur Rahman, and Miguel Alonso. G-ids: Generative adversarial networks assisted intrusion detection system. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 376–385. IEEE, 2020.

[293] Noy Kadosh, Alex Frid, and Mashor Housh. Detecting cyber-physical attacks in water distribution systems: One-class classifier approach. *Journal of Water Resources Planning and Management*, 146(8):04020060, 2020.

[294] Kyoung Won Min, Young Hwan Choi, Abobakr Khalil Al-Shamiri, and Joong Hoon Kim. Application of artificial neural network for cyber-attack detection in water distribution systems as cyber physical systems. In *International Conference on Harmony Search Algorithm*, pages 82–88. Springer, 2019.

[295] Xiang-Yun Zou, Yi-Li Lin, Bin Xu, Zi-Bo Guo, Sheng-Ji Xia, Tian-Yang Zhang, An-Qi Wang, and Nai-Yun Gao. A novel event detection model for water distribution systems based on data-driven estimation and support vector machine classification. *Water Resources Management*, 33(13):4569–4581, 2019.

[296] Faramarz Bagherzadeh, Mohamad-Javad Mehrani, Milad Basirifard, and Javad Roostaei. Comparative study on total nitrogen prediction in wastewater treatment plant and effect of various feature selection methods on machine learning algorithms performance. *Journal of Water Process Engineering*, 41:102033, 2021.

[297] Faramarz Bagherzadeh, Amirreza Shojaei Nouri, Mohamad-Javad Mehrani, and Suresh Thennadil. Prediction of energy consumption and evaluation of affecting factors in a full-scale wwtp using a machine learning approach. *Process Safety and Environmental Protection*, 154:458–466, 2021.

[298] Mohamad-Javad Mehrani, Faramarz Bagherzadeh, Min Zheng, Przemyslaw Kowal, Dominika Sobotka, and Jacek Mąkinia. Application of a hybrid mechanistic/machine learning model for prediction of nitrous oxide (n2o) production in a nitrifying sequencing batch reactor. *Process Safety and Environmental Protection*, 162:1015–1024, 2022.

[299] Wan-Yi Mao, Mehmet Yardimci, Minh Nguyen, Dan Sobien, Laura Freeman, Feras A Batarseh, Abdul Rahman, and Vinita Fordham. Trustworthy ai solutions for cyberbiosecurity challenges in water supply systems. In *The International FLAIRS Conference Proceedings*, volume 35, 2022.

[300] Cheah Huei Yoong and Jonathan Heng. Framework for continuous system security protection in swat. In *Proceedings of the 2019 3rd International Symposium on Computer Science and Intelligent Control*, pages 1–6, 2019.

[301] Sridhar Adepu and Aditya Mathur. Distributed detection of single-stage multipoint cyber attacks in a water treatment plant. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pages 449–460, 2016.

[302] Mayra Macas and Chunming Wu. An unsupervised framework for anomaly detection in a water treatment system. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1298–1305. IEEE, 2019.

[303] Giulio Zizzo, Chris Hankin, Sergio Maffeis, and Kevin Jones. Adversarial attacks on time-series intrusion detection for industrial control systems. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 899–910. IEEE, 2020.

[304] Eirini Anthi, Lowri Williams, Matilda Rhode, Pete Burnap, and Adam Wedgbury. Adversarial attacks on machine learning cybersecurity defences in industrial control systems. *Journal of Information Security and Applications*, 58:102717, 2021.

[305] Andrew C. Harvey. *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[306] Wenya Li, Qing Shi, Muhammad Sibtain, Daoliang Li, and Delene E. Mbanze. A hybrid forecasting model for short-term power load based on sample entropy, two-phase decomposition, and whale algorithm optimized support vector regression. *IEEE Access*, 8:166907–166921, 2020.

[307] Grzegorz Dudek. Neural networks for pattern-based short-term load forecasting: A comparative study. *Neurocomputing*, 205:64–74, 2016.

[308] Kyunghyun Cho et al. Learning phrase representations using rnn encoder–decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[309] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[310] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[311] Ruina Gao, Liang Du, Ponnuthurai Nagaratnam Suganthan, Qing Zhou, and Kwok-Fai Yuen. Random vector functional link neural network-based ensemble deep learning for short-term load forecasting. *Expert Systems with Applications*, 206:117784, 2022.

[312] Jongwoo Kim, Jaeyoon Moon, Euisung Hwang, and Piljae Kang. Recurrent inception convolution neural network for multi short-term load forecasting. *Energy and Buildings*, 194:328–341, 2019.

[313] Manel Massaoudi et al. A novel stacked generalization ensemble-based hybrid lgbm-xgb-mlp model for short-term load forecasting. *Energy*, 214:118874, 2021.

[314] Piotr Piotrowski, Dariusz Baczyński, Mateusz Kopyt, and Tomasz Gulczyński. Advanced ensemble methods using machine learning and deep learning for one-day-ahead forecasts of electric energy production in wind farms. *Energies*, 15(4):1252, 2022.

[315] Sarin E Chandy, Amin Rasekh, Zachary A Barker, and M Ehsan Shafiee. Cyberattack detection using deep generative models with variational inference. *Journal of Water Resources Planning and Management*, 145(2):04018093, 2019.

[316] Mashor Housh, Noy Kadosh, and Jack Haddad. Detecting and localizing cyber-physical attacks in water distribution systems without records of labeled attacks. *Sensors*, 22 (16):6035, 2022.

[317] Feras A Batarseh, Laura Freeman, and Chih-Hao Huang. A survey on artificial intelligence assurance. *Journal of Big Data*, 8(1):1–30, 2021.

[318] Pamela Wilcox, Tamara D Madensen, and Marie Skubak Tillyer. Guardianship in context: Implications for burglary victimization risk and prevention. *Criminology*, 45 (4):771–803, 2007.

[319] Matt R Nobles, Jeffrey T Ward, and Rob Tillyer. The impact of neighborhood context on spatiotemporal patterns of burglary. *Journal of Research in Crime and Delinquency*, 53(5):711–740, 2016.

[320] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[321] Junhai Zhai, Sufang Zhang, Junfen Chen, and Qiang He. Autoencoder and its various variants. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 415–419, 2018. doi: 10.1109/SMC.2018.00080.

[322] Zhaomin Chen, Chai Kiat Yeo, Bu Sung Lee, and Chiew Tong Lau. Autoencoder-based network anomaly detection. In *2018 Wireless Telecommunications Symposium (WTS)*, pages 1–5. IEEE, 2018.

[323] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[324] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[325] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 54(2): 1–38, 2021.

[326] Chun Chet Tan and Chikkannan Eswaran. Performance comparison of three types of autoencoder neural networks. In *2008 Second Asia International Conference on Modelling & Simulation (AMS)*, pages 213–218. IEEE, 2008.

[327] Droniou Alain and Sigaud Olivier. Gated autoencoders with tied input weights. In *International Conference on Machine Learning*, pages 154–162. PMLR, 2013.

[328] Lei Huang, Xianglong Liu, Bo Lang, Adams Wei Yu, Yongliang Wang, and Bo Li. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[329] Minseong Kim and Hyun-Chul Choi. Uncorrelated feature encoding for faster image style transfer. *Neural Networks*, 140:148–157, 2021.

[330] Scott C Douglas, Shun-ichi Amari, and S-Y Kung. On gradient adaptation with unit-norm constraints. *IEEE Transactions on Signal processing*, 48(6):1843–1847, 2000.

[331] Alessandro Erba, Riccardo Taormina, Stefano Galelli, Marcello Pogliani, Michele Carminati, Stefano Zanero, and Nils Ole Tippenhauer. Constrained concealment attacks against reconstruction-based anomaly detectors in industrial control systems. In *Annual Computer Security Applications Conference*, pages 480–495, 2020.

[332] Luis Muñoz-González, Bjarne Pfitzner, Matteo Russo, Javier Carnerero-Cano, and Emil C Lupu. Poisoning attacks with generative adversarial nets. *arXiv preprint arXiv:1906.07773*, 2019.

[333] Anne B Koehler, Ralph D Snyder, and J Keith Ord. Forecasting models and prediction intervals for the multiplicative holt–winters method. *International Journal of Forecasting*, 17(2):269–286, 2001.

[334] Slawek Smyl, Grzegorz Dudek, and Pawel Pelka. Es-drnn with dynamic attention for short-term load forecasting. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.

[335] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[336] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[337] Grzegorz Dudek, Paweł Pełka, and Slawek Smyl. A hybrid residual dilated lstm and exponential smoothing model for midterm electric load forecasting. *IEEE Transactions on Neural Networks and Learning Systems*, 33(7):2879–2891, 2021.

[338] R Taormina and S Galelli. Real-time detection of cyber-physical attacks on water distribution systems using deep learning. In *World Environmental and Water Resources Congress 2017*, pages 469–479, 2017.

[339] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. Available at:http://www.deeplearningbook.org.

[340] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[341] Paula Branco, Luís Torgo, and Rita P Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, 49(2):1–50, 2016.

[342] Alice Botturi, E Gozde Ozbayram, Katharina Tondera, Nathalie I Gilbert, Pascale Rouault, Nicolas Caradot, Oriol Gutierrez, Saba Daneshgar, Nicola Frison, Çağrı Akyol, et al. Combined sewer overflows: A critical review on best practice and innovative solutions to mitigate impacts on environment and human health. *Critical Reviews in Environmental Science and Technology*, 51(15):1585–1618, 2021.

[343] Western Regional Climate Center, Hydrometeorological Prediction Center, and Environmental Modeling Center. National oceanic and atmospheric administration (noaa). *NOAA*, 2003.

[344] Past Weather. National weather service. *NWS*, 2009.

[345] Rivka Galchen. Weather underground. *New Yorker*, 13:34–40, 2015.

[346] Mary C Rabbitt. *The United States Geological Survey, 1879-1989*, volume 1050. US Government Printing Office, 1989.

[347] Michael Ratcliffe, Charlynn Burd, Kelly Holder, and Alison Fields. Defining rural at the us census bureau. *American community survey and geography brief*, 1(8):1–8, 2016.

[348] Lauren Setar and Matthew MacFarland. Top 10 fastest-growing industries. *Special Report IbisWorld*, 2012.

[349] Edward C Budd and Daniel B Radner. The bureau of economic analysis and current population survey size distributions: Some comparisons for 1964. In *The Personal Distribution of Income and Wealth*, pages 449–559. NBER, 1975.

[350] Michael W McCracken and Serena Ng. Fred-md: A monthly database for macroeconomic research. *Journal of Business & Economic Statistics*, 34(4):574–589, 2016.

[351] George EP Box, Gwilym M Jenkins, and Gregory C Reinsel. *Time series analysis: forecasting and control*. Holden-Day San Francisco, 1970.

[352] Sean J Taylor and Benjamin Letham. Forecasting at scale. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1515–1524, 2018.

[353] Donald F Specht. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[354] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xinghua Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 753–763, 2020.

[355] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2019.

[356] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar:

Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

[357] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *9th ISCA Speech Synthesis Workshop*, page 125, 2016.

[358] Faramarz Bagherzadeh and Torkan Shafighfard. Ensemble machine learning approach for evaluating the material characterization of carbon nanotube-reinforced cementitious composites. 2022.

[359] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE intelligent systems*, 24(2):8–12, 2009.

[360] Dziugas Baltrunas, Ahmed Elmokashfi, and Amund Kvalbein. Measuring the reliability of mobile broadband networks. In *Proceedings of the 2014 conference on internet measurement conference*, pages 45–58, 2014.

[361] Zachary S Bischof, Fabian E Bustamante, and Nick Feamster. Characterizing and improving the reliability of broadband internet access. *arXiv preprint arXiv:1709.09349*, 2017.

[362] Li Chen, Justinas Lingys, Kai Chen, and Feng Liu. Auto: Scaling deep reinforcement learning for datacenter-scale automatic traffic optimization. In *Proceedings of the 2018 conference of the ACM special interest group on data communication*, pages 191–205, 2018.

[363] Robert Grandl, Ganesh Ananthanarayanan, Srikanth Kandula, Sriram Rao, and

Aditya Akella. Multi-resource packing for cluster schedulers. *ACM SIGCOMM Computer Communication Review*, 44(4):455–466, 2014.

[364] Junchen Jiang, Rajdeep Das, Ganesh Ananthanarayanan, Philip A Chou, Venkata Padmanabhan, Vyas Sekar, Esbjorn Dominique, Marcin Goliszewski, Dalibor Kukoleca, Renat Vafin, et al. Via: Improving internet telephony call quality using predictive relay selection. In *Proceedings of the 2016 ACM SIGCOMM Conference*, pages 286–299, 2016.

[365] Ning Liu, Zhe Li, Jielong Xu, Zhiyuan Xu, Sheng Lin, Qinru Qiu, Jian Tang, and Yanzhi Wang. A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*, pages 372–382. IEEE, 2017.

[366] Hongzi Mao, Mohammad Alizadeh, Ishai Menache, and Srikanth Kandula. Resource management with deep reinforcement learning. In *Proceedings of the 15th ACM workshop on hot topics in networks*, pages 50–56, 2016.

[367] Behnam Montazeri, Yilong Li, Mohammad Alizadeh, and John Ousterhout. Homa: A receiver-driven low-latency transport protocol using network priorities. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 221–235, 2018.

[368] Srikanth Sundaresan, Xiaohong Deng, Yun Feng, Danny Lee, and Amogh Dhamdhere. Challenges in inferring internet congestion using throughput measurements. In *Proceedings of the 2017 Internet Measurement Conference*, pages 43–56, 2017.

[369] Tony McGregor, Shane Alcock, and Daniel Karrenberg. The ripe ncc internet measurement data repository. In *International Conference on Passive and Active Network Measurement*, pages 111–120. Springer, 2010.

[370] Spyros Antonatos, Kostas G Anagnostakis, and Evangelos P Markatos. Generating realistic workloads for network intrusion detection systems. In *Proceedings of the 4th International Workshop on Software and Performance*, pages 207–215, 2004.

[371] Yves Denneulin, Emmanuel Romagnoli, and Denis Trystram. A synthetic workload generator for cluster computing. In *18th International Parallel and Distributed Processing Symposium, 2004. Proceedings.*, page 243. IEEE, 2004.

[372] Sheng Di, Derrick Kondo, and Franck Cappello. Characterizing and modeling cloud applications/jobs on a google data center. *The Journal of Supercomputing*, 69(1): 139–160, 2014.

[373] Archana Ganapathi, Yanpei Chen, Armando Fox, Randy Katz, and David Patterson. Statistics-driven workload modeling for the cloud. In *2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW 2010)*, pages 87–92. IEEE, 2010.

[374] Da-Cheng Juan, Lei Li, Huan-Kai Peng, Diana Marculescu, and Christos Faloutsos. Beyond poisson: Modeling inter-arrival time of requests in a datacenter. In *Advances in Knowledge Discovery and Data Mining: 18th Pacific-Asia Conference, PAKDD 2014, Tainan, Taiwan, May 13-16, 2014. Proceedings, Part II 18*, pages 198–209. Springer, 2014.

[375] Ting Li and Jason Liu. Cluster-based spatiotemporal background traffic generation for network simulation. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 25(1):1–25, 2014.

[376] Maayan Frid-Adar, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Synthetic data augmentation using gan for improved liver lesion classification. In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 289–293. IEEE, 2018.

[377] Chi Zhang, Sanmukh R Kuppannagari, Rajgopal Kannan, and Viktor K Prasanna. Generative adversarial network for synthetic time series data generation in smart grids. In *2018 IEEE international conference on communications, control, and computing technologies for smart grids (SmartGridComm)*, pages 1–6. IEEE, 2018.

[378] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32, 2019.

[379] Christopher Bowles, Liang Chen, Ricardo Guerrero, Paul Bentley, Roger Gunn, Alexander Hammers, David Alexander Dickie, Maria Valdés Hernández, Joanna Wardlaw, and Daniel Rueckert. Gan augmentation: Augmenting training data using generative adversarial networks. *arXiv preprint arXiv:1810.10863*, 2018.

[380] Samuel A Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E Tillman, Prashant Reddy, and Manuela Veloso. Generating synthetic data in finance: opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*, pages 1–8, 2020.

[381] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. Using gans for sharing networked time series data: Challenges, initial promise, and open questions. In *Proceedings of the ACM Internet Measurement Conference*, pages 464–483, 2020.

[382] Md Nazmul Kabir Sikder and Feras A Batarseh. Outlier detection using ai: a survey. *AI Assurance*, pages 231–291, 2023.

[383] Alessandro Erba, Riccardo Taormina, Stefano Galelli, Marcello Pogliani, Michele Carminati, Stefano Zanero, and Nils Ole Tippenhauer. Constrained concealment attacks against reconstruction-based anomaly detectors in industrial control systems.

In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2020. doi: 10.1145/3427228.3427660.

[384] Peter Batista Cheung, Jakobus E Van Zyl, and Luisa Fernanda Ribeiro Reis. Extension of epanet for pressure driven demand modeling in water distribution system. *Computing and Control for the Water Industry*, 1(1):311–16, 2005.

[385] Khaled El Emam, Lucy Mosquera, and Richard Hoptroff. *Practical synthetic data generation: balancing privacy and the broad availability of data*. O'Reilly Media, 2020.

[386] Justice Lin, Chhayly Sreng, Emma Oare, and Feras A Batarseh. Neuralflood: an ai-driven flood susceptibility index. *Frontiers in Water*, 5:1291305, 2023.

[387] Gongming Wang, Qing-Shan Jia, MengChu Zhou, Jing Bi, Junfei Qiao, and Abdullah Abusorrah. Artificial neural networks for water quality soft-sensing in wastewater treatment: a review. *Artificial Intelligence Review*, 55(1):565–587, 2022.

[388] Md Shamsur Rahim, Khoi Anh Nguyen, Rodney Anthony Stewart, Damien Giurco, and Michael Blumenstein. Machine learning and data analytic techniques in digital water metering: A review. *Water*, 12(1):294, 2020.

[389] Feras A Batarseh, Mehmet Oguz Yardimci, Ryu Suzuki, Md Nazmul Kabir Sikder, Zhiwu Wang, and W Mao. Realtime management of wastewater treatment plants using ai, 2022.

[390] F. Batarseh, A. Kulkarni, C. Sreng, Lin. J., and S. Maksud. Acwa data, 2023. URL https://github.com/AI-VTRC/ACWA-Data.

[391] Aditya P Mathur and Nils Ole Tippenhauer. Swat: A water treatment testbed for research and training on ics security. In *2016 international workshop on cyber-physical systems for smart water networks (CySWater)*, pages 31–36. IEEE, 2016.

[392] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P Mathur. Wadi: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*, pages 25–28, 2017.

[393] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

[394] Markus Ring, Daniel Schlör, Dieter Landes, and Andreas Hotho. Flow-based network traffic generation using generative adversarial networks. *Comput. Secur.*, 82:156–172, 2019.

[395] Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. Timevae: A variational auto-encoder for multivariate time series generation. *ArXiv*, abs/2111.08095, 2021.

[396] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

[397] Marc G Bellemare, Ivo Danihelka, Will Dabney, Shakir Mohamed, Balaji Lakshminarayanan, Stephan Hoyer, and Rémi Munos. The cramer distance as a solution to biased wasserstein gradients. *arXiv preprint arXiv:1705.10743*, 2017.

[398] Anna C Belkina, Christopher O Ciccolella, Rina Anno, Richard Halpert, Josef Spidlen, and Jennifer E Snyder-Cappione. Automated optimized parameters for t-distributed stochastic neighbor embedding improve visualization and analysis of large datasets. *Nature communications*, 10(1):5415, 2019.

[399] Rasmus Bro and Age K Smilde. Principal component analysis. *Analytical methods*, 6 (9):2812–2831, 2014.

[400] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[401] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[402] Amin Hassanzadeh, Amin Rasekh, Stefano Galelli, Mohsen Aghashahi, Riccardo Taormina, Avi Ostfeld, and Katherine Banks. A review of cybersecurity incidents in the water sector. *arXiv preprint arXiv:2001.11144*, 2020.

[403] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[404] Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. The synthetic data vault. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410. IEEE, 2016.

[405] Tirthajyoti Sarkar et al. Synthetic data generation—a must-have skill for new data scientists. *Towards Data Science, Dec*, 19, 2018.

[406] Anna L Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications surveys & tutorials*, 18(2):1153–1176, 2015.

[407] Akshat Gautam, Muhammed Sit, and Ibrahim Demir. Realistic river image synthesis using deep generative adversarial networks. *Frontiers in water*, 4:784441, 2022.

[408] Ram Krishna Mazumder, Gourav Modanwal, and Yue Li. Synthetic data generation using generative adversarial network (gan) for burst failure risk analysis of oil and gas

pipelines. *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg*, pages 1–21, 2023.

[409] Amin E Bakhshipour, Alireza Koochali, Ulrich Dittmer, Ali Haghighi, Sheraz Ahmad, and Andreas Dengel. A bayesian generative adversarial network (gan) to generate synthetic time-series data, application in combined sewer flow prediction. *arXiv preprint arXiv:2301.13733*, 2023.

[410] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[411] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[412] Rick Sauber-Cole and Taghi M Khoshgoftaar. The use of generative adversarial networks to alleviate class imbalance in tabular data: a survey. *Journal of Big Data*, 9 (1):98, 2022.

[413] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[414] Laura Aviñó, Matteo Ruffini, and Ricard Gavaldà. Generating synthetic but plausible healthcare record datasets. *arXiv preprint arXiv:1807.01514*, 2018.

[415] Graham Cormode, Cecilia Procopiuc, Divesh Srivastava, Entong Shen, and Ting Yu. Differentially private spatial decompositions. In *2012 IEEE 28th International Conference on Data Engineering*, pages 20–31. IEEE, 2012.

[416] Yi Sun, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Learning vine copula models for synthetic data generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5049–5057, 2019.

[417] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *arXiv preprint arXiv:1806.03384*, 2018.

[418] Markus Ring, Daniel Schlör, Dieter Landes, and Andreas Hotho. Flow-based network traffic generation using generative adversarial networks. *Computers & Security*, 82: 156–172, 2019.

[419] Steve TK Jan, Qingying Hao, Tianrui Hu, Jiameng Pu, Sonal Oswal, Gang Wang, and Bimal Viswanath. Throwing darts in the dark? detecting bots with limited data using neural data augmentation. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1190–1206. IEEE, 2020.

[420] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 61 (3):611–622, 1999.

[421] David Arthur and Sergei Vassilvitskii. K-means++ the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035, 2007.

[422] Ajay Kulkarni, Deri Chong, and Feras A Batarseh. Foundations of data imbalance and solutions for a data democracy. In *data democracy*, pages 83–106. Elsevier, 2020.

[423] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning:

Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.