الجمهـوريـة الجـزائريـة الديـمقـراطيـة الشـعبيـة
PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
وزارة التعـليـم العـالـي و البـحث العـلـمي
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
جـامعـة عمّـار ثليـجـي بالأغـواط
AMAR TELIDJI UNIVERSITY OF LAGHOUAT

كـليـة العـلـوم
FACULTY OF SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Master's Thesis

Field: Mathematics and Computer Science

Specialty: Computer Science

Option: Data Science and Artificial Intelligence

By: Mohammed Lazhari KROBBA

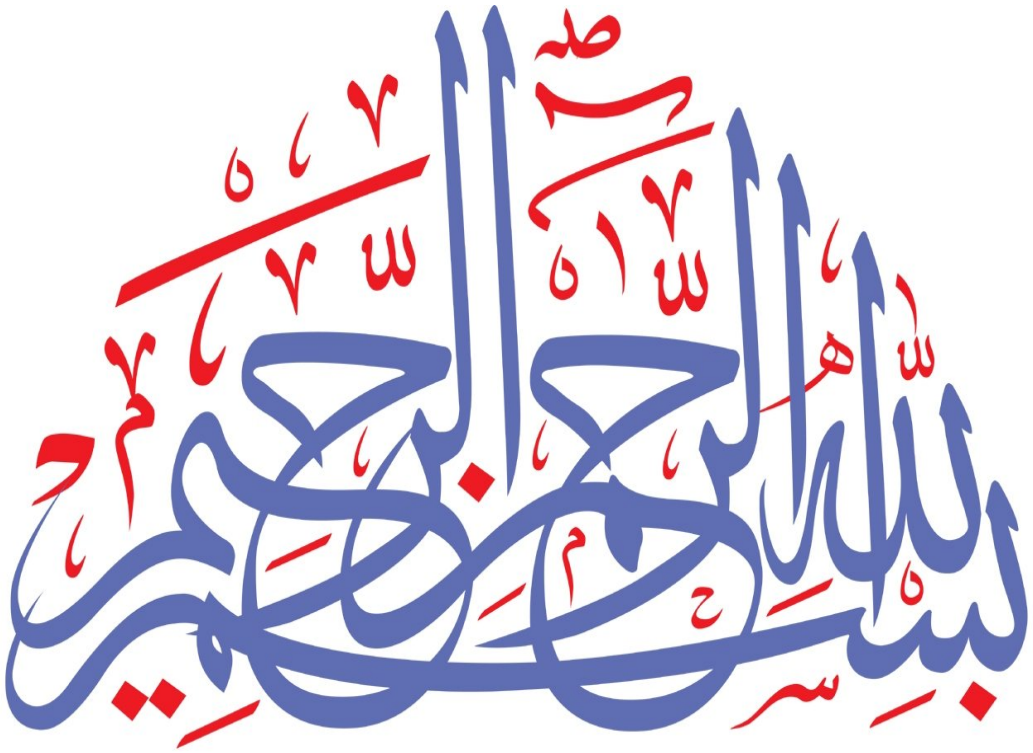
TOPIC

DEEP LEARNING DUAL-MODEL FOR PREDICTING AND VALIDATING SENSOR DATA IN UNDN

Defended publicly on June 22th, 2025, before a jury composed of:

Pr. Abdelhafidi Zohra	Professor	President
Dr. Lakhder Oulad DJedid	M.C.A	Examiner
Dr. Messaoud Babaghayou	M.A.B	Examiner
Dr. Abdelmadjid Benarfa	M.C.A	Supervisor
Dr. Tahar Bendouma	M.C.A	Co-Supervisor

Thesis No. _____ Academic Year 2024/2025


بسم الله الرحمن الرحيم

To my dear parents,

. . .

To my brothers and sisters,

. . .

To my friends,

. . .

To all my family,

. . .

To all my teachers,

. . .

I dedicate this modest work

Acknowledgments

I would like to express my deep appreciation to everyone who contributed to the completion of this work. I am especially thankful to my academic supervisor for their helpful guidance, continuous support, and constructive suggestions throughout the research process. Their expertise and encouragement played a key role in the progress and direction of this project.

I also extend my gratitude to my family and friends for their encouragement and patience during this journey. Their moral support provided the motivation I needed to stay focused and committed, especially during challenging moments of the research.

ABSTRACT

This thesis presents a deep learning dual-model approach aimed at improving the reliability of sensor data in Underwater Named Data Networks (UNDNs)a novel paradigm that integrates Named Data Networking (NDN) into Underwater Wireless Sensor Networks (UWSNs). UWSNs are critical for applications such as environmental monitoring, seismic activity detection, and underwater exploration, yet they suffer from severe challenges including high latency, limited bandwidth, signal attenuation, and sensor faults. NDN enhances underwater communication by enabling content-centric data retrieval with in-network caching and built-in data-centric security. However, NDN alone cannot handle issues like sensor malfunctions, missing readings, or poor data quality. To address these limitations, we propose a dual deep learning framework consisting of two inde- pendent models: a time-series prediction model used to estimate future sensor values and recover missing data, and a classification model designed to validate the correctness of sensor readings. Several architectures were evaluated, including LSTM, GRU, CNN-1D, Transformer, and Temporal Convolutional Networks (TCN). While most models demonstrated promising performance during testing, only the TCN maintained strong generalization on unseen data. For the classification task, a Multi-Layer Perceptron (MLP) model was employed and showed robust accuracy in distinguishing valid from faulty readings. Both models performed well despite being trained on a limited dataset. Importantly, all models were implemented from scratch without relying on external frameworks. This study contributes toward building intelligent, adaptive, and resilient underwater monitoring systems capable of maintaining data integrity in challenging environ- ments

Keywords:Underwater Named Data Network (UNDN), Underwater Wireless Sensor Networks (UWSN), Named Data Networking (NDN), Deep Learning, Time-Series Prediction, Sen- sor Valida-

tion, Sensor Data Reliability

RÉSUMÉ

Ce mémoire présente une approche à double modèle dapprentissage profond visant à améliorer la fiabilité des données capteurs dans les Réseaux de Données Nommées Sous-Marins (UNDNs). Ce paradigme novateur intègre le Réseautage de Données Nommées (NDN) aux Réseaux de Capteurs Sans Fil Sous-Marins (UWSNs). Les UWSNs jouent un rôle clé dans des domaines tels que la surveillance environnementale, la détection de lactivité sismique et lexploration sous-marine. Cependant, ils rencontrent de sérieuses limitations, notamment une latence élevée, une bande passante restreinte, une atténuation importante du signal et des défaillances fréquentes des capteurs. Le NDN améliore la communication sous-marine en permettant une récupération des données centrée sur le contenu, grâce à une mise en cache dans le réseau et à une sécurité intégrée. Toutefois, le NDN seul ne peut résoudre les problèmes de lectures manquantes, de dysfonctionnements des capteurs ou de mauvaise qualité des données. Pour remédier à cela, nous proposons une architecture à double modèle basée sur lapprentissage profond : un modèle de prédiction de séries temporelles pour estimer les valeurs futures et compléter les données manquantes, et un modèle de classification pour vérifier la validité des lectures. Plusieurs architectures ont été évaluées, telles que LSTM, GRU, CNN-1D, Transformer et TCN. Parmi elles, seul le TCN a montré une bonne capacité de généralisation sur des données inédites. Pour la classification, un Perceptron Multi-Couche (MLP) a été utilisé avec des performances solides. Tous les modèles ont été implémentés sans frameworks externes. Cette étude contribue à la conception de systèmes de surveillance sous-marins intelligents, adaptatifs et fiables.

Mots clés:Réseau de Données Nommées Sous-Marines (UNDN), Réseaux de Capteurs Sans Fil Sous-Marins (UWSN), Réseautage de Données Nommées (NDN), Apprentissage Profond, Prévision de Séries Temporelles, Validation des Capteurs, Fiabilité des Données des Capteurs

ملخص

يقدم هذا البحث نموذجًا مزدوجًا يعتمد على التعلم العميق، يهدف إلى تحسين موثوقية بيانات أجهزة الاستشعار في شبكات بيانات مسماة تحت الماء ،(UNDNs) وهو مفهوم جديد يدمج شبكات البيانات المسماة (NDN) ضمن شبكات الاستشعار اللاسلكية تحت الماء .(UWSNs) تعتبر هذه الشبكات ضرورية لتطبيقات مثل مراقبة البيئة، واكتشاف النشاط الزلزالي، والاستكشاف تحت الماء، لكنها تعاني من مشاكل مثل التأخر العالي، وانخفاض عرض النطاق، وتوهين الإشارة، وأعطال المستشعرات.

يعزز NDN الاتصال تحت الماء من خلال تمكين استرجاع البيانات المتمحور حول المحتوى مع التخزين المؤقت في الشبكة وأمان مدمج. ومع ذلك، فإن NDN وحده لا يمكنه معالجة أعطال المستشعرات أو القراءات المفقودة أو ضعف جودة البيانات. لمعالجة هذه المشكلات، نقترح إطارًا مزدوجًا يتكون من نموذجين مستقلين: نموذج تنبؤ زمني لتقدير القيم المستقبلية واستعادة البيانات المفقودة، ونموذج تصنيف للتحقق من صحة قراءات المستشعر.

تم تقييم عدة معماريات، بما في ذلك LSTM وGRU و1D-CNN وTransformer وTCN. أظهر TCN أفضل قدرة على التعميم عند التعامل مع بيانات غير مرئية. في مهمة التصنيف، تم استخدام نموذج Perceptron متعدد الطبقات (MLP) وحقق دقة قوية في التمييز بين القراءات الصحيحة والخاطئة. تم تنفيذ جميع النماذج من الصفر دون الاعتماد على أطر عمل خارجية. تساهم هذه الدراسة في تطوير أنظمة مراقبة ذكية وقابلة للتكيف وموثوقة للحفاظ على سلامة البيانات في البيئات الصعبة.

الكلمات المفتاحية: شبكات بيانات مسماة تحت الماء، شبكات الاستشعار اللاسلكية تحت الماء، شبكات البيانات المسماة، التعلم العميق، التنبؤ بالسلاسل الزمنية، التحقق من صحة المستشعرات، موثوقية بيانات المستشعر.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

GENERAL INTRODUCTION

With the growing demand for reliable data collection in underwater environments, efficient communication and data management have become essential. Underwater Wireless Sensor Networks (UWSNs) address this need by deploying sensor nodes to monitor parameters such as temperature, pressure, pollution, and movement. Despite their usefulness, UWSNs face significant challenges including high latency, limited energy, signal degradation, and harsh underwater conditions [1]. To enhance the performance and resilience of these networks, we propose integrating Named Data Networking (NDN) with Machine Learning (ML) techniques. NDN is a new way of designing networks that focuses on what data is needed instead of where it is stored. Instead of sending data to a specific address, NDN allows users to request data by its name. This makes it easier to share and access information, even in networks that are unstable or have delays. One big advantage of NDN is in network caching, which allows routers and nodes to temporarily store data. This means if someone else requests the same data, it can be delivered faster without contacting the original source again. NDN also has built-in security that protects the data itself, rather than the connection[4].

Machine Learning adds an extra layer of intelligence to the system. It allows the network to learn from past behavior and data patterns, which helps it make better decisions. For example, ML can help choose the best path to send data, predict network failures, reduce energy use, and even detect unusual events like equipment damage or changes in the environment[5]. With the help of ML, UWSNs can become more adaptive and self-managing, reacting to changing conditions without human control. By combining the smart decision-making of ML with the efficient, flexible communication model of NDN, we can build underwater networks that are faster, more reliable, and more energy-efficient. This helps improve underwater exploration, ocean research, disaster detection, and many other marine monitoring tasks. The work that we present in this thesis is focus on the applications of the ML in

this domain of research and her benefit and to be more clear we use Deep Learning DL models. and one of the most problems in UWSN and UNDN is the how to manipulate sensors in different cases like losing data or the sensors give us a fake and poor data, for these cases and many more situations researchers integrate the Artificial intelligence to get a good solutions then the classical methods with minimum cost, and in our work we focus on some DL models to predict the failure of sensors before it happened and compensate the messing values or correct any errors, and all of these points gives us strong control over these sensors and over the network in general.

1.1 Problem Statement

In Underwater Named Data Networks (UNDN), the primary challenge lies in the **management and quality of sensor data.** UWDNs rely heavily on sensor nodes deployed underwater to collect critical environmental data such as temperature, pressure, and pollution levels. However, sensor data in such networks is prone to various issues, including **sensor failures, data corruption, and missing readings**, often due to the harsh underwater conditions. These issues are exacerbated by the unreliable communication channels and energy limitations of underwater sensor networks. A major issue is the presence of **missing or incomplete data**, which typically occurs when sensors malfunction or when environmental interference prevents proper data transmission. Additionally, some sensors may generate **inaccurate data** due to factors such as drift, calibration errors, or unexpected environmental influences, leading to potentially misleading insights. Another significant problem is **data noise**, as underwater sensors are often affected by disturbances like water currents or electromagnetic interference, which can distort the readings collected. Traditionally, these challenges have been addressed using simple methods such as interpolation or filtering. However, these approaches often fall short as they do not effectively capture complex patterns of sensor failures or anomalies. Furthermore, they are reactive in nature and lack predictive capabilities, making it difficult to preempt sensor issues before they impact the data quality.

1.2 Motivation

To overcome the limitations of conventional data handling techniques in UNDNs, the use of Deep Learning (DL) presents a promising direction. DL models can be trained on historical sensor data to learn complex temporal and contextual patterns, enabling the prediction of sensor behavior and intelligent handling of anomalies. Specifically, models such as Recurrent Neural Networks (RNNs), Long

Short-Term Memory (LSTM) networks, and autoencoders are particularly effective for processing time-series data generated by underwater sensors.

These models can be leveraged to predict sensor failures by recognizing early signs of malfunction, enabling timely maintenance or recalibration. They can also be used to fill in missing data by reconstructing it based on learned patterns from existing readings, thereby improving data completeness. Furthermore, DL techniques are capable of detecting anomalies by identifying outliers that may indicate faulty sensors or unusual environmental changes.

By incorporating DL methods, it becomes possible to automate the correction of data errors and enhance the overall reliability of information collected through underwater networks. This, in turn, improves decision-making in critical applications such as marine research, environmental monitoring, and early warning systems for natural disasters.

The motivation for this research is to investigate how deep learning models can be effectively applied to address data quality challenges in UNDNs. The goal is to enhance sensor data integrity, reduce the frequency and impact of failures, and ultimately strengthen the performance and robustness of underwater communication and monitoring systems.

1.3 Organization

The remainder of this thesis is organized as follows. Chapter 1 presents a general introduction along with the motivation behind this work. In Chapter 2, we discuss Underwater Wireless Sensor Networks (UWSNs) and explore how applying Named Data Networking (NDN) can help reduce some of the key challenges associated with this technology. Chapter 3 is dedicated to reviewing related work, providing context and background for our research. Chapter 4 presents the core of this thesis: a Deep Learning dual-model framework for predicting and validating sensor data in Underwater Named Data Networks (UNDN), including all experiments, results, and detailed discussion. Finally, Chapter 5 concludes the thesis with a summary and outlines possible future perspectives for extending this research.

CHAPTER 2

APPLYING NDN IN UWSNS

Contents

This chapter is divided into three main sections. The first section introduces Underwater Wireless Sensor Networks (UWSNs), outlining their applications and major challenges such as high latency, limited energy, and communication unreliability. The second section presents Named Data Networking (NDN), emphasizing its content-centric approach, in-network caching, and security features. The third section analyzes how NDN mitigates several UWSN limitations by enhancing efficiency, data delivery, and robustness, while also acknowledging persistent issues like sensor failures and environmental interference.

2.1 Underwater Wireless Sensor Networks (UWSNs)

2.1.1 Introduction

The ocean, covering over 70% of Earth's surface and supporting nearly half the global population near coasts, plays a crucial role in nourishment, transportation, resource extraction, defense, and exploration. Despite its importance, less than 10% of the ocean has been explored. Traditional underwater monitoring methods face challenges due to the ocean's vastness and harsh conditions, making human presence unfeasible. As a result, interest in Underwater Wireless Sensor Networks (UWSNs) has grown, particularly among researchers familiar with terrestrial sensor networks.[6]

2.1.2 UWSNs Overview

The UWSN is a network used to perform monitoring of tasks over a specific region; it is equipped with smart sensors and vehicles that are adapted to communicate cooperatively through wireless connections [6]. The surface sink retrieves the data from sensor nodes. The sink node has a transceiver that can control acoustic signals received from underwater nodes. The transceiver also can transmit and receive long-range radio frequency signals for communication with the onshore station. The collected data are used locally or connected to another network for a particular purpose [7]. Figure 2.1 illustrates an overview of the UWSN environment. The network architecture incorporates traditional underwater wireless sensor networks designed by [6] and real-time underwater wireless sensor network architecture in the form of Internet of Underwater Things proposed by [8]. Underwater Wireless

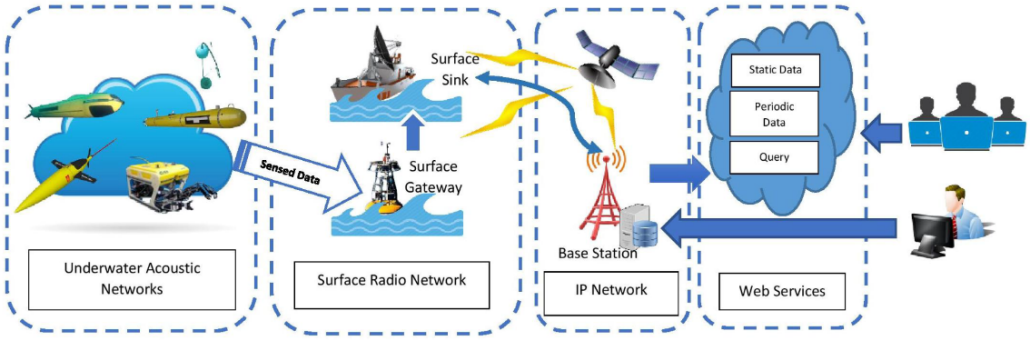


Figure 2.1: Overview of Underwater Wireless Sensor Networks.[1]

Sensor Networks (UWSNs) comprise nodes that can be deployed both on the surface and underwater. All nodes must communicate and exchange information with other nodes in the same network and with a base station [9]. Communication systems in these networks transmit data using acoustic, electromagnetic, or optical wave media. Among these, acoustic communication is the most popular and

widely used method due to its favorable propagation characteristics in water. This low transmission is influenced by the absorption and conversion of energy into heat in water. Acoustic signals operate at low frequencies, which allows them to be transmitted and received over long distances [10].

2.1.3 Characteristics of UWSNs

UWSNs operate under significantly different and more challenging conditions compared to terrestrial wireless sensor networks (WSNs). The following points detail the key distinguishing features and challenges:

Acoustic Communication: Radio frequency (RF) signals suffer from severe attenuation in seawater, particularly at higher frequencies, limiting their effective range to just a few meters. Therefore, acoustic waves are the primary medium for underwater communication, capable of propagating over longer distances, albeit with their own limitations such as low data rates and higher susceptibility to noise and multi-path propagation effects [6, 8].

High Latency and Low Bandwidth: Acoustic signals travel much slower in water (1500 m/s) compared to radio waves in air (300,000 km/s). This causes significant propagation delays, especially in long-distance communication. Furthermore, acoustic channels typically offer very limited bandwidth, often in the order of a few kHz, severely restricting data transmission rates [11].

Limited Energy Resources: UWSN nodes are usually powered by batteries, and in most underwater scenarios, recharging or replacing batteries is infeasible due to the remote and submerged environment. Hence, energy conservation becomes a paramount design concern, impacting decisions related to routing, sensing, and transmission protocols [6, 12].

Harsh Environmental Conditions: The underwater environment presents several physical challenges including high pressure, variable salinity, temperature gradients, and dynamic ocean currents. These can affect both signal quality and the reliability of hardware components, demanding robust sensor and communication system designs [10].

Node Mobility and Dynamic Topology: Ocean currents and other underwater dynamics often cause node mobility, leading to frequently changing network topologies. This introduces complexity in maintaining stable routes, increases the need for adaptive routing protocols, and complicates time synchronization across nodes [13].

2.1.4 Applications of UWSNs

Underwater Wireless Sensor Network (UWSN) technology offers real-time monitoring, remote control of underwater systems, and advanced data collection, making it a strong alternative to traditional methods. UWSN applications are generally classified into scientific, industrial, and military/security domains (see Figure 2.2). Military uses include enemy detection, port monitoring, and submarine tracking, while scientific uses involve seismic and marine environment monitoring for disaster prevention. The growing variety of applications calls for ongoing advancement in technologies and standards to support UWSN development.[14]

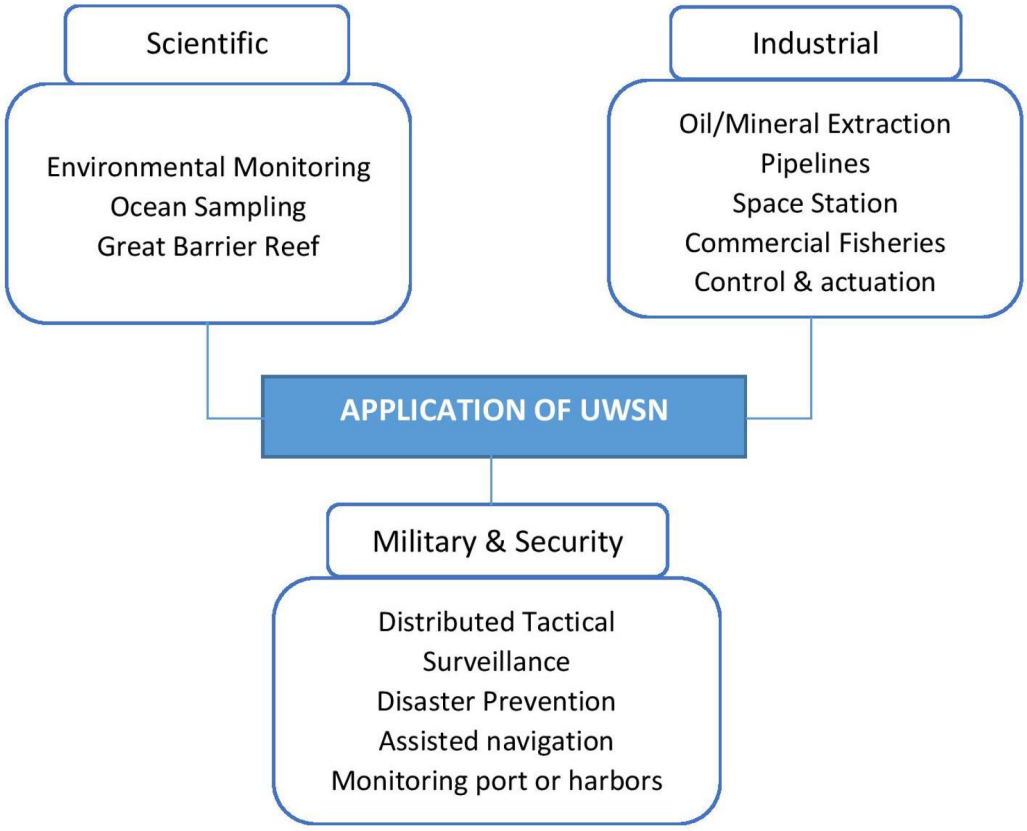


Figure 2.2: Applications of UWSNs.[1]

Scientific: Underwater Wireless Sensor Networks (UWSNs) support a range of scientific applications including pollution monitoring, ocean sampling, and coral reef research. These systems use technologies like robotic fish [15] and autonomous vehicles [16] to collect real-time data on water quality, oxygen levels, and ocean conditions. Advanced approaches also integrate IoT and big data to study coral bleaching [17], while hybrid systems combining different mobile agents have been deployed for long-term environmental monitoring [18].

Industrial: Industrial applications of Underwater Wireless Sensor Networks (UWSNs) play a crucial role in enhancing commercial operations, particularly in oil and gas pipeline monitoring and aquaculture. Researchers have proposed UWSN-based systems for monitoring pipeline health over large underwater areas using actuation components for control [6, 19]. In aquaculture, UWSNs have been used for automated fish farm monitoring, where systems track water quality parameters such as dissolved oxygen, pH, temperature, and humidity. Some implementations use Zigbee-based systems and integrate wireless cameras for real-time, remote access via the Internet [20]. Other works apply acoustic communication to achieve similar objectives in commercial fisheries [21, 7].

Defense and Disaster Prevention Application: Underwater Wireless Sensor Networks (UWSNs) have been extensively utilized in military and defense applications for early threat detection, including port and harbor monitoring [22], sea mine detection [23], border protection against unauthorized naval activities [24], and distributed tactical surveillance [25]. Advanced UWSN technologies, such as mobile underwater sensor networks, are also employed for early warning systems related to natural disasters like underwater seismic activities [22]. For instance, Jain and Virmani [24] developed a real-time tsunami prediction model using data from the 2004 Indian Ocean tsunami. Communication in these networks typically relies on acoustic waves or a combination of acoustic and radio frequency (RF) waves, depending on application-specific factors such as network type, region, water depth, node spacing, sensor type, and total number of nodes, as summarized in.

2.1.5 Challenges in Underwater Wireless Sensor Networks (UWSNs)

Despite their vast potential in fields such as environmental monitoring, military surveillance, underwater exploration, and disaster prevention, Underwater Wireless Sensor Networks (UWSNs) face several significant and persistent challenges. These include harsh and unpredictable underwater environments, limited bandwidth, high latency, and energy constraints, all of which complicate the design and deployment of efficient, scalable, and robust network systems. Moreover, the difficulty of maintaining and securing underwater nodes, coupled with the lack of standardized architectures and real-time responsiveness, further hinders the development of reliable long-term UWSN applications.

Data Loss and Ambient Noise:

Underwater acoustic communication is highly susceptible to various sources of ambient noise, including turbulence, shipping activities, wind, and thermal variations. These noise sources can distort sensor readings and lead to data loss. For instance, shipping noise (10100 Hz) and wind noise (100 Hz100 kHz) can significantly degrade signal quality, affecting the reliability of data transmission [6,

26].

Limited Communication Range and Bandwidth:

Acoustic signals used in UWSNs attenuate rapidly with distance, limiting the effective communication range. Additionally, the available bandwidth is constrained and varies with transmission distance and depth. For example, long-range communications (over several kilometers) may have bandwidths as low as 500 Hz to 10 kHz, resulting in data rates around 10 kbps, whereas short-range communications (less than 100 meters) can achieve higher bandwidths and data rates [11].

Synchronization Issues:

The propagation speed of acoustic waves underwater is approximately 1500 m/s, which is significantly slower than electromagnetic waves in air. This results in long and variable propagation delays, complicating time synchronization among sensor nodes. Accurate synchronization is crucial for coordinated operations and data fusion in UWSNs [26].

Maintenance and Energy Constraints:

Underwater sensor nodes are often deployed in inaccessible locations, making maintenance and battery replacement challenging. The harsh underwater environment can accelerate battery depletion and hardware degradation. Energy efficiency is therefore a critical concern, necessitating the development of energy-aware protocols and hardware [6].

Dynamic Network Topology:

The underwater environment is dynamic, with factors such as water currents and mobile nodes (e.g., Autonomous Underwater Vehicles) causing frequent changes in network topology. This mobility leads to intermittent connectivity and requires robust routing protocols that can adapt to the changing network structure [27].

Multipath Propagation and Doppler Effects:

Acoustic signals in underwater environments often experience multipath propagation due to reflections from the sea surface and bottom. This can cause signal fading and inter-symbol interference. Additionally, relative motion between the transmitter and receiver introduces Doppler shifts, further complicating signal detection and synchronization [11].

Security Vulnerabilities:

UWSNs are vulnerable to various security threats, including denial-of-service attacks, eavesdropping, and spoofing. The unique characteristics of underwater communication, such as low bandwidth and high latency, make it challenging to implement traditional security measures, necessitating specialized security protocols [28].

2.2 Named Data Networking (NDN)

2.2.1 Introduction

Numerous solutions have been proposed to narrow the gap between the Internet design and its current usage. One such potential Future Internet Architecture(FIA), sponsored by NSF, is Name Data Networking (NDN). NDN explicitly addresses the data (content) itself instead of its physical location (i.e., host) in the network, therefore, transforming data into the first-class" entity. In NDN, the receiver directly requests the name of the content by issuing an interest. The network then handles the request by efficiently finding and retrieving back the closest copy of the relevant content. This decoupling of time and space among request resolution and content transfer enables NDN to provide storage, mobility and security as native features belonging to the network architecture[29].

2.2.2 NDN Overview

The current Internet architecture is built on an hourglass model centered around the IP protocol,which provides minimal functionality necessary for global connectivity [30]. Originally designed in the 1970s for host-to-host communication, this model enabled rapid innovation and growth [31]. However, as the Internet evolved into a platform for content distributiondriven by e-commerce, digital media,and mobile applicationsthe host-based model has shown limitations in addressing distribution challenges effectively [30]. To overcome these, the Named Data Networking (NDN) architecture proposes a fundamental shift: instead of sending packets to specific endpoints, NDN enables the retrieval of data based on names [32]. This name-based approach allows packets to identify any object, such as a movie segment, a command, or a document. NDN preserves many strengths of IP-based networking while offering built-in traffic regulation and security through features like Interest/Data flow balance and mandatory data signatures, aiming to support a broader set of applications beyond traditional end-to-end communication [32].

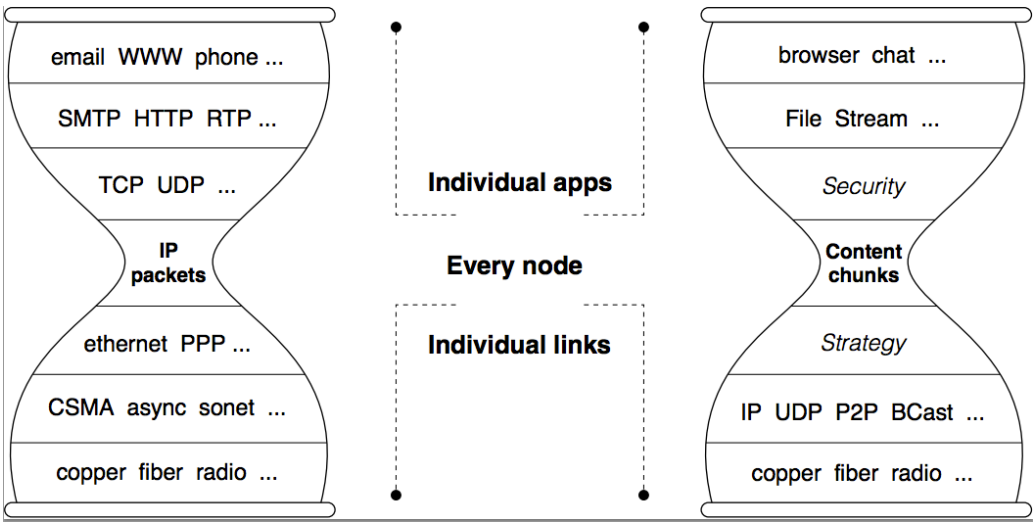


Figure 2.3: Internet and NDN Hourglass Architectures.[2]

2.2.3 NDN Principles

Named Data Networking (NDN) is the most promising and popular among Information Centric Networking (ICN) architectures, it is a research project in the Future Internet Architecture Program funded by the U.S. National Science Foundation (NSF). Its origin is a previous project, Content-Centric Networking (CCN) [33] which Van Jacobson started at Xerox PARC in 2006, Its fundamental concepts were outlined in a Google tech talk by Van Jacobson, long before that exactly in 2009[29]. In order to achieve the objective of addressing the limitations and improving the strengths of the current Internet architecture, the new NDN design must be guided by the following architectural principles[29]:

- Security has to be integrated into the architecture. Security in the present Internet architecture does not meet the requirements of todays increasingly hostile environment. NDN offers the fundamental security building block right in the thin waist by signing all named data [29].

- The end-to-end principle allows solid apps to be developed against network failures. NDN maintains and extends this principle [29].

- Network traffic has to be self-regulating. Flow-balanced data delivery is crucial for stable network operation. Since the IP performs open loop data delivery, the transport protocols have been modified to provide unicast traffic balance. NDN designs the flow-balance into the thin waist [29].

- A separation between the routing and forwarding plane has demonstrated to be essential for the development of the Internet. It allows the forwarding plane to operate while the routing scheme continues to develop over time. NDN adheres to the same principle to allow the implementation of NDN with the best available forwarding technology while we carry out in parallel new routing system research [29].

2.2.4 Advantages of NDN

Unlike the traditional IP-based architecture that relies on end-to-end communication between hosts, the Named Data Networking (NDN) architecture introduces a fundamentally different approach by focusing on content retrieval rather than host location. This paradigm shift brings multiple advantages: it enables efficient in-network caching, reducing latency and redundant data transmissions; enhances scalability in content distribution by allowing the same data to be fetched from multiple sources; and improves security by binding signatures directly to data rather than relying solely on securing communication channels. Moreover, NDN's built-in flow control through Interest/Data exchange helps manage congestion and resource allocation more effectively, making it well-suited for dynamic and mobile environments where traditional IP mechanisms struggle to perform reliably [32, 30].

Improved Efficiency: By shifting from host-based to content-based communication, the NDN architecture improves network efficiency and scalability. It enables in-network caching, allowing intermediate nodes to serve content, thus lowering latency and reducing bandwidth usage [32]. This approach is well-suited for dynamic scenarios like IoT, video streaming, and disaster response. Security is enhanced by attaching trust directly to data using digital signatures [30].

Enhanced Security: Data-centric security in NDN ensures that each content packet is cryptographically signed, enabling verification of data authenticity regardless of its source [32]. This approach protects against tampering and unauthorized access, even in untrusted environments. It shifts the security focus from securing channels to securing the data itself [30].

Scalability: The hierarchical naming and stateful forwarding in NDN enable efficient content routing and retrieval [32]. These features reduce the reliance on large routing tables, supporting network scalability. NDN can adapt to growing numbers of devices and data without significant overhead. This makes it ideal for large-scale and dynamic environments like IoT.

Resilience: NDN's ability to retrieve data from multiple sources and its support for dynamic routing enhance network resilience in the face of failures or congestion [34].

The next Table 2.1 compares the key features of Named Data Networking (NDN) and IP-based Internet architecture. It highlights NDNs data-centric approach, which offers built-in caching, security, and name-based routing. These features improve scalability, efficiency, and adaptability in modern networks. In contrast, IP relies on end-to-end communication and lacks native support for such functions. The comparison underscores NDNs advantages for todays content-driven applications.

Table 2.1: Key Advantages of NDN Not Present in IP-based Internet

Feature	**NDN**	**IP-Based Internet**
Security Model	Data-centric security: each data packet is signed and can be verified independently of the source.	Security is primarily focused on securing the communication channel (e.g., via TLS, SSL).
Caching Mechanism	In-network caching: routers store data and serve requests locally, reducing bandwidth and latency.	No native in-network caching. Data must be fetched from the original source each time.
Name-Based Routing	Routing is based on hierarchical data names, which simplifies mobility and multicast.	Routing is based on IP addresses, which requires complex handling for mobility and multicast.
Forwarding State	Stateful forwarding plane: routers maintain state for each data request, optimizing flow control.	Stateless forwarding: IP routers do not track requests and cannot manage flow directly.
Connection Model	No need for persistent end-to-end connections: data is retrieved from any available copy.	Requires end-to-end communication between hosts, establishing a dedicated path for each session.
Multicast Support	Built-in support for multicast: a single data request can serve multiple consumers simultaneously.	IP multicast is complex and often not deployed, requiring additional protocols like IGMP.

2.3 Addressing UWSN Challenges Through Named Data Networking (NDN)

Named Data Networking (NDN) introduces a data-centric communication paradigm that addresses several of the inherent challenges in Underwater Wireless Sensor Networks (UWSNs). By focusing on the content itself rather than its location, NDN offers multiple advantages for data retrieval, efficiency, and robustness in harsh underwater environments.

Data-Centric Communication: NDN eliminates the need for IP-based addressing by allowing nodes to request data by name. This approach is particularly beneficial in dynamic underwater environments where traditional IP routing fails due to frequent topology changes or node mobility.

In-Network Caching: One of the most powerful features of Named Data Networking (NDN) is its in-network caching capability, where intermediate nodes temporarily store requested data to serve future requests locally. This mechanism reduces redundant transmissions, which is particularly beneficial for minimizing energy consumption in battery-constrained environments such as underwater sensor networks. It also significantly lowers data retrieval latency, which is critical in high-latency acoustic communication channels, and alleviates the load on source nodes by preventing repeated energy-intensive transmissions [32, 30].

Robustness to Disruptions: NDNs name-based routing and content replication across multiple caches ensure continued data access even if certain nodes or paths fail. This property increases reliability in underwater environments prone to disconnections and high error rates.

Built-in Security: NDN secures data at the content level by attaching digital signatures to each packet, ensuring data integrity, authenticity, and resilience against spoofing or tampering [32]. This content-centric security model is particularly advantageous in underwater environments, where establishing and maintaining secure communication sessions is costly due to high latency and limited resources. Unlike IP-based systems, NDNs security remains effective regardless of the datas delivery path or source.

Adaptability to Mobility and Failures: Since NDN operates using content names, it does not rely on fixed node identifiers or routes. This enables:

- Seamless operation despite node movement caused by underwater currents.
- Efficient re-routing through nearby caches when the original data producer becomes unavailable.

Support for Delay-Tolerant Networking (DTN): NDN inherently supports intermittent connectivity due to its caching and data request model. This makes it suitable for sparse UWSNs where continuous end-to-end paths may not always exist.

2.3.1 Summary

Table 2.2 outlines how Named Data Networking (NDN) addresses several critical challenges in Underwater Wireless Sensor Networks (UWSNs), such as high latency, mobility, and limited energy. While NDN offers significant advantages through in-network caching, name-based routing, and data-centric security, it does not address all issues. Limitations remain in areas such as sensor data accuracy, environmental interference, and predictive maintenance, which require complementary solutions beyond the scope of NDN alone.

Table 2.2: NDN-Based Solutions and Limitations for Key UWSN Challenges

UWSN Challenge	**NDN-Based Solution or Limitation**
High Latency	Local caching reduces need for distant communication
Node Mobility	Name-based routing tolerates dynamic topologies
Limited Energy	Fewer transmissions due to caching and multicast
Data Loss	Data replication and cached copies ensure availability
Security	Data packet-level signatures ensure trust and integrity
Missing or Corrupted Sensor Data	Not directly addressed by NDN.
Environmental Noise	NDN does not mitigate physical layer interference.
Predict Sensor Failure	Not supported by NDN.

2.4 Conclusion

Integrating Named Data Networking (NDN) into Underwater Wireless Sensor Networks (UWSNs) forms Underwater Named Data Networking (UNDN), which improves data access, supports mobility, and enhances security through content-centric communication and in-network caching. However, UNDN still faces data-related challenges, such as sensor failures, data loss, and uncertain data quality. These issues limit the reliability and effectiveness of the system. To overcome them, researchers turn to Artificial Intelligence (AI), especially Machine Learning (ML) and Deep Learning (DL), which can predict failures, recover missing data, and assess quality. This work focuses on applying DL models

to improve data reliability in UNDN, offering smarter and more resilient underwater communication systems.

CHAPTER 3

RELATED WORK

Contents

Many researchers have worked on improving underwater sensor networks by using new technologies like Named Data Networking (NDN) and deep learning. These methods help solve problems such as missing data, sensor faults, and real-time monitoring. This section presents some important studies that are related to our work.

3.1 Introduction

Underwater Wireless Sensor Networks (UWSNs) have become an important research area due to their ability to monitor marine environments, detect natural events, and support underwater exploration. However, UWSNs face several challenges such as high latency, limited energy, and difficult communication conditions. To solve some of these issues, researchers have explored the integration of Named Data Networking (NDN), a data-centric communication model, into underwater systems. Additionally, deep learning has gained attention as a powerful tool for predicting sensor behavior and improving fault detection. In this chapter, we review the most relevant studies that combine these technologies to address problems in underwater sensor networks.

3.2 Literature Review

NDN Goes Deep: Foreseeing the Underwater Named Data Networks

Bouk et al. [3] introduced the concept of *Underwater Named Data Networking (UNDN)* by adapting Named Data Networking to address challenges such as high latency and energy constraints in underwater environments. They proposed a hierarchical naming scheme tailored to underwater sensor data, which enables efficient data retrieval and in-network caching. Their work demonstrated the feasibility of using NDN to improve data availability and reliability in underwater networks. **NDN**

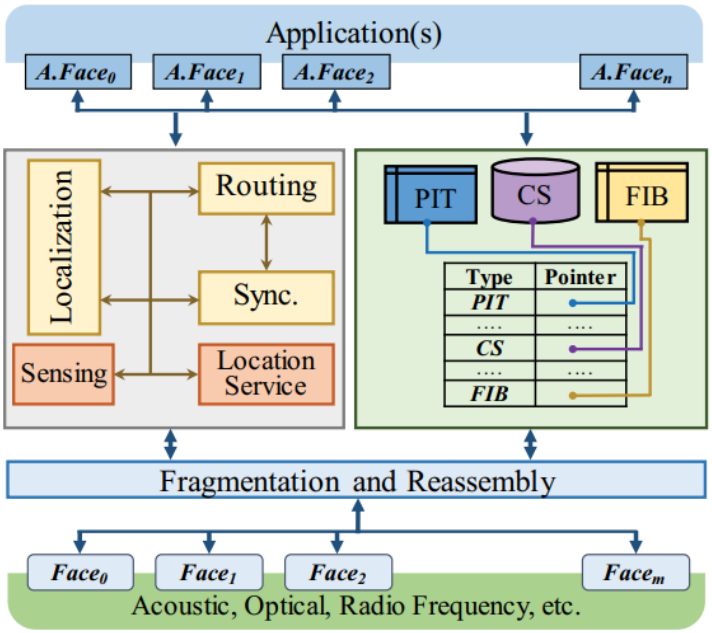


Figure 3.1: UNDN Architecture. [3]

Diving Deeper: A Comprehensive Survey of Named Data Networking for Underwater Sensor Networks (UWSN):

Benarfa et al. [35] present a comprehensive survey on the integration of Named Data Networking (NDN) within Underwater Sensor Networks (UWSNs). The paper explores the unique challenges of underwater environments, such as high latency, limited bandwidth, and energy constraints, and discusses how NDN's data-centric approach can address these issues. It reviews existing NDN-based protocols tailored for UWSNs, highlighting advancements in naming schemes, forwarding strategies, and security mechanisms. The survey serves as a foundational reference for researchers aiming to enhance underwater communication networks using NDN.

A Conceptual Framework for Predictive Maintenance of Underwater Sensors Using Named Data Networking and Machine Learning

Benarfa et al. [36] propose an innovative framework that integrates Named Data Networking (NDN) with Machine Learning (ML) to enhance predictive maintenance in Underwater Sensor Networks (UWSNs). The framework aims to proactively detect and address sensor faults, thereby improving data reliability and reducing maintenance costs. By leveraging NDN's data-centric communication model alongside ML algorithms, the authors envision a system capable of real-time fault detection and efficient data dissemination in challenging underwater environments. This work lays the groundwork for future implementations of intelligent maintenance systems in UWSNs.

Predictive Maintenance for UWSNs: A Practical Implementation of Fault Detection Using NDN and Deep Learning

Benarfa et al. [37] present a practical implementation of an AI-driven fault detection system for Underwater Wireless Sensor Networks (UWSNs), integrating Named Data Networking (NDN) with deep learning techniques. Utilizing the "Underwater Sensor Dataset," the authors developed a feed-forward neural network to classify sensor readings as healthy or faulty. The model achieved impressive performance metrics: 99.9% accuracy, 100% precision, 99.1% recall, and a 99.6% F1-score. This approach demonstrates the feasibility of employing AI-based predictive maintenance in UWSNs, aiming to reduce downtime, minimize maintenance costs, and extend the operational lifespan of underwater sensor networks.

A Novel Approach for Missing Data Recovery and Fault Nodes Detection in Wireless Sensor Networks

Thiyagarajan and Nagabhooshanam [38] propose an integrated framework combining fault node detection and missing data recovery in Wireless Sensor Networks (WSNs). The approach employs

Fuzzy Density-Based Spatial Clustering of Applications with Noise (FDBSCAN) to identify and isolate faulty nodes by analyzing deviations from expected network behavior. Subsequently, a Bidirectional Long Short-Term Memory (Bi-LSTM) neural network is utilized to reconstruct missing data, leveraging temporal patterns in sensor readings. Experimental results demonstrate high data reliability (96--98%) even with up to 80% missing data, and fault detection accuracy of 97.4%, outperforming existing methods. This work underscores the efficacy of combining clustering algorithms with deep learning for robust WSN data integrity.

Deploying Deep Learning for Real-Time Tsunami Monitoring: A CNN-LSTM Model for Sensor-Based Prediction

Karthihadevi et al. [39] introduced a hybrid CNNLSTM deep learning framework for real-time tsunami monitoring based on sensor data. Their model combines convolutional neural networks (CNNs) for spatial feature extraction and long short-term memory (LSTM) units for capturing temporal dependencies. This approach enables the system to detect tsunami events from continuous ocean sensor readings and issue early warnings efficiently. The authors emphasize the models real-time applicability and potential to reduce false alarms, although specific dataset details and evaluation metrics were not provided in the abstract.

3.3 Summary

To establish the foundation for our proposed approach, we reviewed several significant contributions in the domains of underwater sensor networks (UWSNs), Named Data Networking (NDN), and deep learning for real-time monitoring. The selected works explore a range of methods including NDN-based data dissemination, machine learning-driven predictive maintenance, and deep learning models for environmental event prediction such as tsunami detection. Table 3.1 summarizes these studies by highlighting their core techniques, targeted applications, and limitations. This comparative analysis reveals key research gapssuch as the lack of integrated frameworks combining NDN and deep learning for real-time, predictive analysis in underwater environmentswhich our work aims to address.

Table 3.1: Summary of Related Work and Research Gaps

Title	Authors	Approach	Application	Identified Gaps
NDN Goes Deep: Foreseeing the Underwater Named Data Networks	Bouk et al. [3]	NDN with hierarchical naming	Data retrieval and caching in UWSNs	Lacks integration with AI for event prediction or anomaly detection
NDN Diving Deeper: A Comprehensive Survey of NDN for UWSNs	Benarfa et al. [35]	Survey on NDN protocols for UWSNs	Naming, forwarding, and security in UWSNs	No practical implementation or real-time decision models
A Conceptual Framework for Predictive Maintenance using NDN and ML	Benarfa et al. [36]	NDN + ML (conceptual)	Predictive maintenance in UWSNs	No implementation or experimental validation
Predictive Maintenance for UWSNs using NDN and Deep Learning	Benarfa et al. [37]	NDN + Feedforward Neural Network	Fault detection using real dataset	Focused only on static fault detection, no spatiotemporal event prediction
Missing Data Recovery and Fault Nodes Detection in WSNs	Thiyagarajan and Nagabhooshanam [38]	FDBSCAN + Bi-LSTM	Fault detection and data recovery	Not designed for underwater or tsunami-specific scenarios
Deploying Deep Learning for Real-Time Tsunami Monitoring	Karthihadevi et al. [39]	CNNLSTM	Tsunami detection using sensor data	Dataset/method details unclear; lacks integration with NDN or predictive maintenance

3.4 Conclusion

This chapter has shown that combining NDN with deep learning holds great potential for improving underwater sensor networks. However, many existing solutions focus only on one part of the problemeither communication or prediction. In the next chapter, we introduce our proposed approach, which brings together both technologies in a simple, effective way to predict sensor values and validate their reliability in real-time.

CHAPTER 4

DEEP LEARNING DUAL-MODEL FOR PREDICTING AND VALIDATING SENSOR DATA IN UNDN

Contents

4.1 Introduction

In underwater environments, sensors are exposed to harsh conditions that often lead to malfunctions, such as missing data, irregular measurements, or complete node failures. These issues can severely impact the reliability of collected data and reduce the effectiveness of underwater monitoring systems.

To address this, researchers are increasingly turning to Deep Learning (DL) methods to analyze sensor behavior over time and predict failures before they occur. DL models can automatically learn patterns in historical data and detect subtle changes that may indicate future faults.

In this work, we apply DL models to predict sensor failures in underwater sensor networks. Our goal is to improve data quality and network reliability by identifying at-risk sensors early and enabling proactive actions, such as data correction or alert generation. This approach offers a data-driven solution to one of the key challenges in maintaining robust underwater monitoring systems.

4.2 Problem Definition

Despite the advancements brought by Named Data Networking (NDN) in underwater wireless sensor networks (UWSNs), several critical challenges remain unresolved especially those related to the quality, availability, and reliability of the data collected by underwater sensors.

In real-world underwater environments, sensors are exposed to harsh conditions that may lead to hardware failures, data loss, noisy or inaccurate readings, and inconsistent data sampling. NDN focuses on data routing and delivery, but it cannot detect faulty sensors or compensate for missing or poor-quality data.

Furthermore, traditional systems do not have the ability to predict environmental changes or sensor faults before they occur. This results in delayed responses and data gaps, which negatively affect underwater monitoring missions.

The table below 4.1 highlights key data-related challenges in UWSNs and compares how traditional NDN-based methods and Deep Learning (DL) techniques handle them.

Table 4.1: Limitations of traditional methods and advantages of DL in UWSNs

Challenge	**NDN / Traditional Methods**	**Deep Learning (DL)**
Sensor failure or breakdown	Not detected automatically	Can predict failure in advance
Missing or corrupted data	No compensation mechanism	Can estimate or reconstruct missing values
Environmental anomaly detection	Hard to detect dynamically	Learns patterns and detects anomalies
Poor data quality	Not evaluated at the source	Can evaluate and filter low-quality data
Reactive behavior	Responds after problem occurs	Enables proactive response and planning

To address these limitations, we propose using Deep Learning models capable of forecasting sensor behavior and environmental trends, making UWSNs smarter and more resilient.

4.3 Proposed Approach

In this work, we propose a dual-model architecture to improve underwater data collection in challenging environments. Traditional systems often struggle when sensor data is missing, delayed, or affected by noise. Our solution uses two main components to handle these issues more effectively. The first is a Time Series Prediction Model, which estimates missing or future sensor values based on patterns learned from past data. This helps maintain a complete and continuous data stream, even when some sensors fail or data is lost. The second component is a Result Validation Model, which checks the accuracy and reliability of the predicted data. By combining these two models, our architecture improves both the quality and trustworthiness of sensor data in Underwater Named Data Networks (UNDNs). This approach makes it easier to support real-time decision-making, early warning systems, and reliable underwater communication even under harsh or unpredictable conditions.

4.3.1 Time Series Prediction Model

Underwater sensors often encounter harsh conditions leading to data loss, irregular sampling, or hardware failure. To address these challenges, we employ deep learning models specifically suited for time series forecasting. These models are trained to predict continuous sequences of parameters such as temperature, salinity, and dissolved oxygen levels. The architectures used include:

- **Long Short-Term Memory (LSTM):** Designed to capture long-range dependencies in sequential data, LSTM networks are effective in modeling temporal patterns even with irregular sampling.
- **Gated Recurrent Unit (GRU):** A simpler alternative to LSTM with fewer parameters, GRUs offer competitive performance with reduced training time, making them suitable for resource-constrained underwater environments.
- **1D Convolutional Neural Networks (CNN-1D):** CNN-1D models apply convolutional filters over time steps to capture local temporal patterns and trends in the sensor data.
- **Transformer-Based Models:** Leveraging self-attention mechanisms, transformers are capable of learning global dependencies in time series data and have recently shown high accuracy in many forecasting tasks.
- **Temporal Convolutional Networks (TCN):** TCNs use dilated causal convolutions to model temporal relationships with a large receptive field while maintaining sequence order. In our experiments, the TCN model demonstrated the best performance in terms of prediction accuracy and stability.

Each model is trained on sequences of historical sensor readings using sliding windows and standard time series preprocessing techniques. Model performance is evaluated using metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE).

4.3.2 Validation Model For Obtained Results

While prediction models can fill gaps in data, they may also produce inaccurate outputs due to noise, sensor drift, or anomalous conditions. To mitigate this risk, we introduce a lightweight classification model that acts as a validation gate.

This model utilizes a simple Multi-Layer Perceptron (MLP) architecture to classify predicted sequences as either valid or invalid. It is trained on labeled data containing both normal and faulty prediction scenarios, which may result from sensor noise or out-of-distribution inputs. The model helps filter out unreliable forecasts and enhances the overall robustness of the system.

The classifier is designed to be computationally efficient and interpretable. It ensures that only trustworthy predictions are passed to higher-level applications or stored for long-term use. Incorrect predictions flagged as invalid can either be discarded or flagged for human inspection.

4.3.3 Architecture Summary

Figure 4.1 illustrates the proposed dual-model architecture. The forecasting model receives time-series data and outputs predicted sensor values. These predictions are then passed to the classification model, which determines their reliability. This pipeline enhances the overall data quality and supports more resilient decision-making in underwater monitoring systems.

4.3.4 Advantages of the Proposed Approach

The proposed dual-model approach offers several important advantages for underwater monitoring systems. First, it improves data completeness by using prediction models to estimate missing or corrupted sensor data, ensuring continuous and reliable information. Second, it enhances data reliability through a validation layer that checks and confirms the accuracy of predicted data, allowing only trustworthy results to be retained or transmitted. Third, the system is scalable, meaning it can be extended to monitor other underwater parameters or incorporate additional deep learning models without major changes. Fourth, it promotes energy efficiency by avoiding repeated transmissions of low-quality data, which helps preserve battery life in resource-limited underwater environments. Finally, by identifying faulty sensors early and enabling predictive maintenance, it becomes possible to repair or replace multiple failing sensors at the same time thereby minimizing operational costs and downtime. Altogether, this dual-model system supports the development of more intelligent, efficient, and autonomous Underwater Named Data Networks (UNDNs) that are well-suited for real-world deployment.

This dual-model system represents a novel step toward more intelligent and autonomous underwater monitoring systems, making UNDNs more robust in real-world deployments.

4.4 Implementation

In this section, we describe the implementation details of our dual-model approach for enhancing the reliability and quality of underwater sensor data. The implementation is divided into several key components including data preparation, training procedures, and evaluation.

4.4.1 Development Setup and Computational Resources

The implementation of the proposed system was carried out on a standard personal laptop without a dedicated graphics card (GPU). The environment was set up using Miniconda to manage depen-

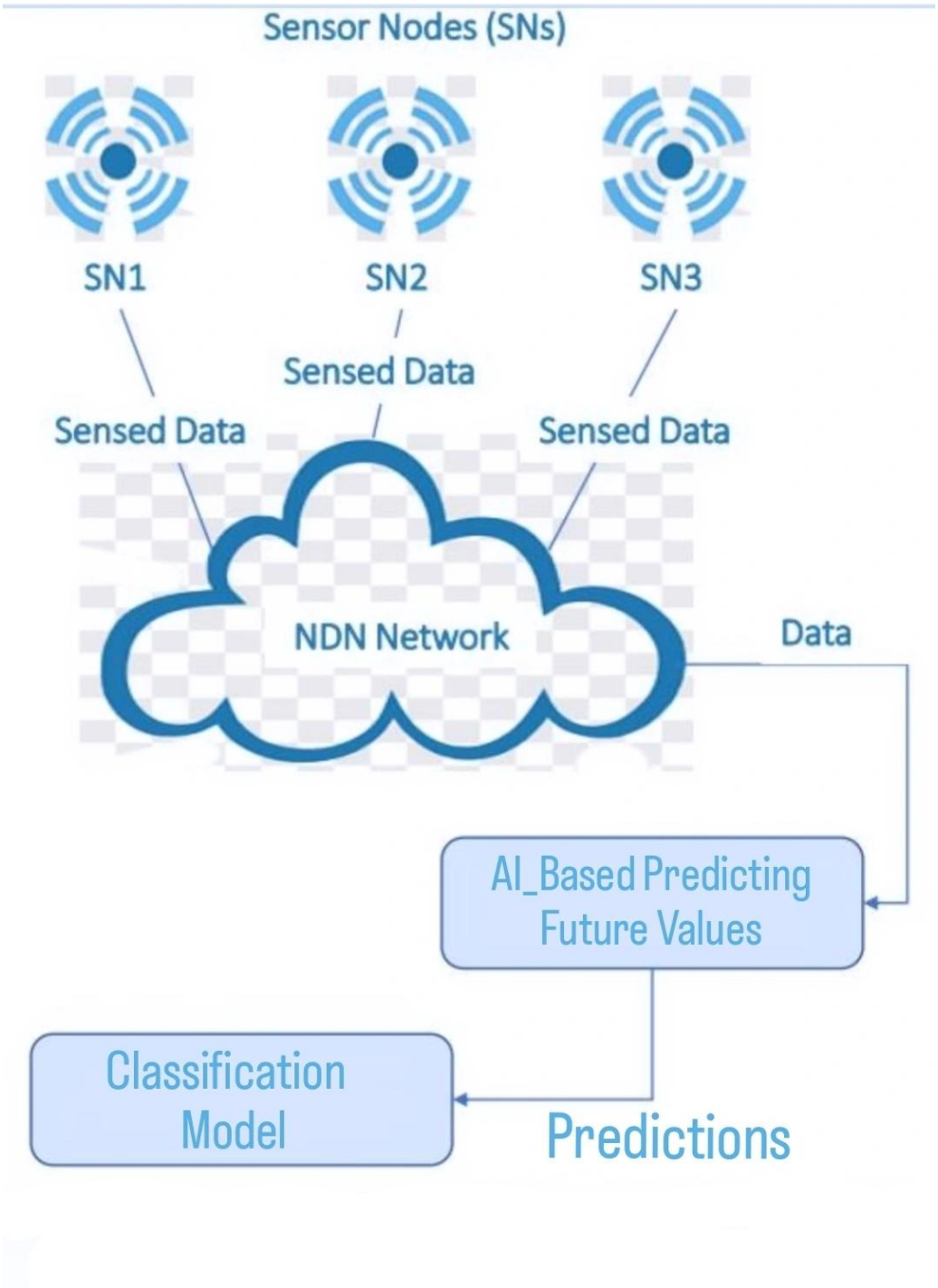


Figure 4.1: Dual-model architecture for prediction and validation in UNDN.

dencies and virtual environments efficiently. Python was the primary programming language, and the models were developed using the TensorFlow deep learning framework. Development and testing were performed using Jupyter Notebook for its interactive features, alongside Visual Studio Code (VS Code) as a lightweight code editor. Despite limited hardware resources, the setup was sufficient for training and evaluating the proposed models using moderate-sized datasets.

4.4.2 Data Preparation

For developing and testing our deep learning models, we used the Underwater Surface Temperature Dataset collected along the Santa Catarina coast in Brazil. This dataset contains temperature measurements from seven islands and two submerged rocky reefs. The data was gathered every 20 minutes between December 2012 and July 2014 using HOBO Pendantő Temperature Data Logger UA-002 devices, which were installed underwater by SCUBA divers. Each record in the dataset includes critical information such as the site name, geographical coordinates (latitude and longitude), the date and time of the measurement, the water temperature in degrees Celsius, and the depth at which it was taken. This detailed dataset provides a valuable view into underwater environmental conditions over time and was essential for training and evaluating our predictive models [40].

4.4.2.1 Selection and Preprocessing

We selected data from one sensor site at a fixed depth of 22.1 meters. Two distinct preprocessing pipelines were used depending on the model architecture. The data splitting was performed during the preprocessing phase to ensure consistency throughout the pipeline.

A. Preprocessing for Time-Series Models (TCN, LSTM, GRU, CNN-1D, Transformer)

For the time-series forecasting models, we performed a structured preprocessing pipeline tailored to preserve temporal relationships within the sensor data. The dataset was first filtered to include only measurements from a specific underwater sensor site (*Parcel do Xavier_Alalunga*) at a fixed depth of 22.1 meters, ensuring spatial and contextual consistency. The `Date` and `Time` columns were then merged into a single unified `Timestamp` field to facilitate chronological ordering. Temperature values were normalized using Min-Max scaling, transforming the input range to [0, 1] to stabilize and accelerate the training process across different models.

To prepare the data for time-series forecasting, we applied a sliding window technique to generate input sequences. Specifically, we used a window size of 64, meaning each input sample consists of 64 consecutive time steps of sensor readings. This method allows the models to learn patterns and

dependencies over time. To maintain the integrity of temporal dependencies during model training and evaluation, we split the dataset chronologicallyreserving the first 85% of the sequences for training and the remaining 15% for testing. This approach avoids data leakage and ensures that the models are evaluated on future, unseen data in a manner consistent with real-world forecasting scenarios.

B. Preprocessing for the Second Model (MLP)

For the Multi-Layer Perceptron (MLP) model, we adopted a distinct preprocessing pipeline aimed at constructing a robust classification framework that can differentiate between correct and faulty sensor readings. Initially, we removed irrelevant or redundant columns such as sensor `ID` and `Site` to avoid introducing noise into the model. The `Time` and `Date` fields were simplified`Time` was converted to represent only the hour of the reading (023), and `Date` was decomposed into two separate features: `Month` and `Day`. This transformation helps reduce overfitting by discarding overly granular timestamp information while preserving seasonal and daily patterns.

An `Output` column was added and initialized to 1 for all existing entries, representing true (valid) data samples. To simulate slightly erroneous but realistic measurements, we duplicated a subset of the true samples and subtracted the mean squared error (1.13řC), previously calculated from the time-series forecasting model, from the temperature values. These modified samples were also labeled with `Output = 1` to indicate they remain within acceptable variance.

In order to train the model to detect faulty readings, additional synthetic or clearly erroneous data points were introduced and labeled with `Output = 0`. These samples serve as negative examples, helping the model learn to distinguish between normal and abnormal behavior.

We selected the following features for model training: `Latitude`, `Longitude`, `Month`, `Day`, `Time`, `Temp (řC)`, and `Depth`. All features were normalized using z-score normalization via `StandardScaler` to ensure uniform scaling across the input space. Finally, the dataset was randomly split into 80% for training and 20% for testing, assuming an independent and identically distributed (i.i.d.) structure in the data. This process ensures that the MLP model receives a diverse and well-balanced set of training samples.

4.4.2.2 Summary

To make sure the results are fair and the models work as well as possible, we used different data preparation steps for each type of model. Time-series models like TCN, LSTM, GRU, CNN-1D, and Transformer need the data to stay in the correct time order so they can learn patterns over time. On the other hand, the MLP classification model treats each row of data separately and doesnt need the time

order. We customized the preprocessing steps for each case based on these needs. Table 4.2 shows a clear side-by-side comparison of the main steps we followed for each model type.

Table 4.2: Summary of Data Preprocessing Steps for Time-Series and MLP Models

Step	**Time-Series Models (TCN, LSTM, GRU, etc.)**	**MLP Model (Classifier)**
1. Data Source	One specific sensor site at depth 22.1 meters	Mixed data (real, modified, and fake samples)
2. Time Handling	Merged `Date` and `Time` into one `Timestamp`	Converted `Time` to hours only; `Date` to month and day
3. Filtering	Only rows with depth = 22.1m	Removed unnecessary columns (e.g., `ID`, `Site`)
4. Scaling	Min-Max normalization of temperature (0 to 1)	Z-score normalization (StandardScaler) for all features
5. Feature Engineering	Created sequences of 64 time steps using sliding window	Selected 7 features: Latitude, Longitude, Month, Day, Time, Temp, Depth
6. Output Labeling	Not needed at this stage (forecasting task)	Added `Output` = 1 (real); duplicated/shifted rows also = 1; fake data = 0
7. Data Split	85% training, 15% testing (chronological order)	80% training, 20% testing (random split, i.i.d. assumption)

4.5 Experimental Results

In this section, we present the results of our experiments, which are divided into two main parts. The first part focuses on the performance of various time-series models used for predicting underwater sensor data. We discuss both the strengths and weaknesses observed during their training and evaluation. The second part covers the results of the MLP classification model, which was designed to identify real versus fake sensor data. By separating these parts, we aim to clearly show the differences in performance and behavior between predictive models and classification models.

4.5.1 Results From Time-Series Models (LSTM, GRU, CNN-1D, Transformer, TCN)

In this section, we present the evaluation of several time-series modelsTCN, LSTM, GRU, CNN-1D, and Transformerusing Mean Squared Error (MSE) as the key metric. While most models showed similar performance levels, the TCN model performed differently, achieving a noticeably better score. The following results will detail these findings.

- **LSTM and GRU:** These recurrent neural networks are designed to capture temporal dependencies and are widely used for time-series prediction tasks. The internal architectures of the LSTM and GRU models used in this study are illustrated in Figure **??**, highlighting their key components and the flow of information through their recurrent units.

Listing 4.1: Architecture of the LSTM and GRU models

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, GRU, Dense

# Build LSTM model
lstm_model = Sequential([
    LSTM(16, return_sequences=True, ),
    LSTM(16, return_sequences=False),
    Dense(8, activation='relu'),
    Dense(1)
])

lstm_model.compile(optimizer="adam", loss="mse")

# Build GRU model
gru_model = Sequential([
    GRU(16, return_sequences=True, ),
    GRU(16, return_sequences=False),
    Dense(8, activation='relu'),
    Dense(1)
])

gru_model.compile(optimizer="adam", loss="mse")
```

During the testing phase, both the LSTM and GRU models exhibited reasonable performance, with their outputs closely matching the expected trends. The models appeared to capture short-term temporal dependencies effectively, and the predicted sequences aligned well with the test data, as shown in Figure 4.2. However, when applied to forecast future, unseen values beyond the scope of the training and test sets, both models failed to generalize adequately. Visual inspection of the prediction plots, presented in Figure 4.3, revealed that the output sequences diverged significantly from the actual values, often displaying unrealistic or unstable patterns. This stark contrast between testing and prediction behavior suggests that the models may have overfitted to the training data, learning specific patterns rather than underlying dynamics. The lack of robustness in real-world forecasting highlights a generalization gap, possibly due to limited data variability or the absence of strong regularization techniques.

The testing results are illustrated in Figure 4.2, where the LSTM and GRU models show outputs

that follow the actual data with reasonable accuracy. use it with this

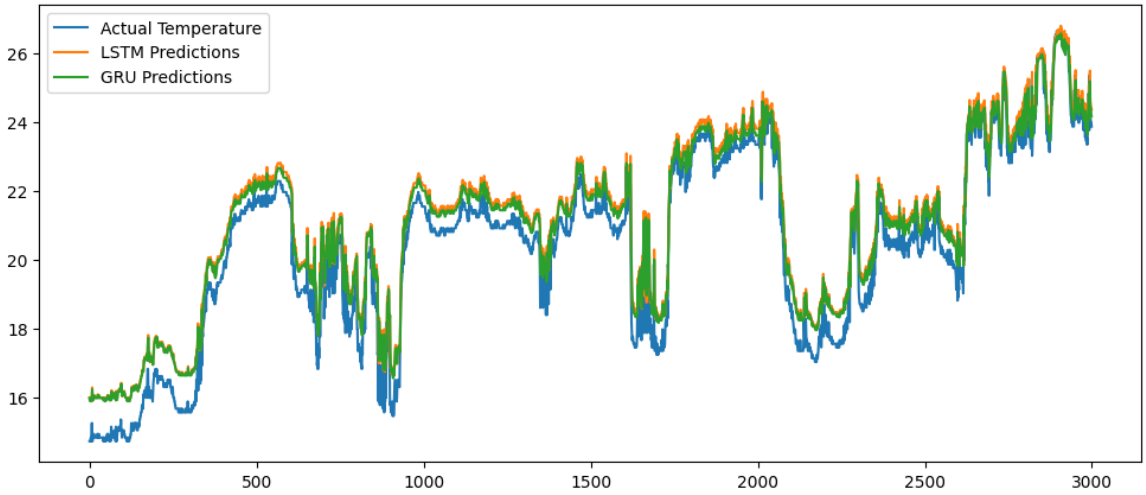


Figure 4.2: LSTM and GRU predictions during the testing phase.

The prediction performance on future unseen data is shown in Figure 4.3, where the models fail to maintain stability and diverge significantly from the true values.

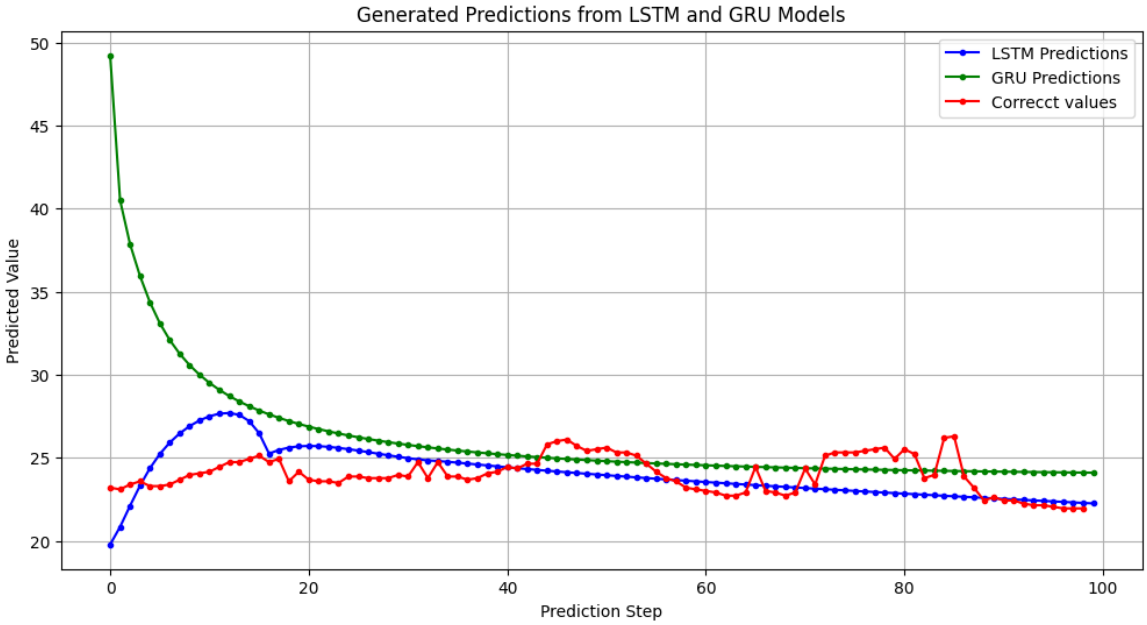


Figure 4.3: LSTM and GRU predictions on future unseen data.

- **CNN-1D:** A one-dimensional convolutional neural network that extracts local patterns from sequential data. The structure of the 1D CNN model implemented in this work is shown in Figure **??**, illustrating how convolutional and pooling layers operate on sequential input to extract temporal features.

Listing 4.2: Architecture of the 1D Convolutional Neural Network (CNN-1D) used for time-series data.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten,
   Dense, Dropout

# Define model
model = Sequential([
    Conv1D(filters=128, kernel_size=3, activation='relu', input_shape
       =(input_seq_len, 1)),
    MaxPooling1D(pool_size=2),

    Conv1D(filters=64, kernel_size=3, activation='relu'),
    Conv1D(filters=32, kernel_size=3, activation='relu'),
    MaxPooling1D(pool_size=2),

    Flatten(),

    Dense(32, activation='relu'),
    Dropout(0.4),
    Dense(1)  # Predict a single value (sequence-to-one)
])

model.compile(optimizer='adam', loss='mse')
model.summary()
```

After presenting the model architecture, we now evaluate its performance during the testing phase to assess how well it learned from the training data.

During the testing phase, the CNN-1D model demonstrated strong predictive performance, as illustrated in Figure 4.4. The predicted values closely followed the actual ground truth, with minimal deviation across most time steps. This alignment indicates that the model was effective in learning local temporal patterns from the data, thanks to its ability to extract features through convolutional layers. The consistency between the predicted and actual values reflects the model's strength in short-term forecasting within the range it was trained on.

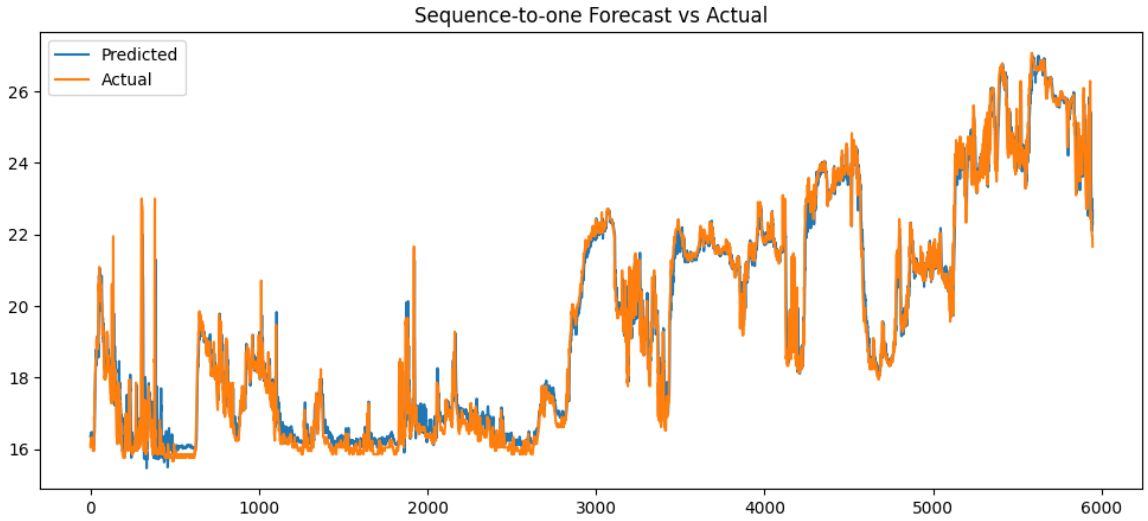


Figure 4.4: CNN-1D predictions during the testing phase showing close alignment with actual values.

The CNN-1D model worked well during testing, but when it was used to predict new future values, the results were much worse. As we see in Figure 4.5, the predicted line does not match the real values and looks very rough and jumpy. This is different from the other models, which also made mistakes but gave smoother curves. The CNN-1D model could not understand the full pattern of the data, so it gave bad results for future predictions. One reason could be that CNN-1D focuses on small local features and misses the long-term trends. Because of this, it had more trouble than the other models in predicting new data.

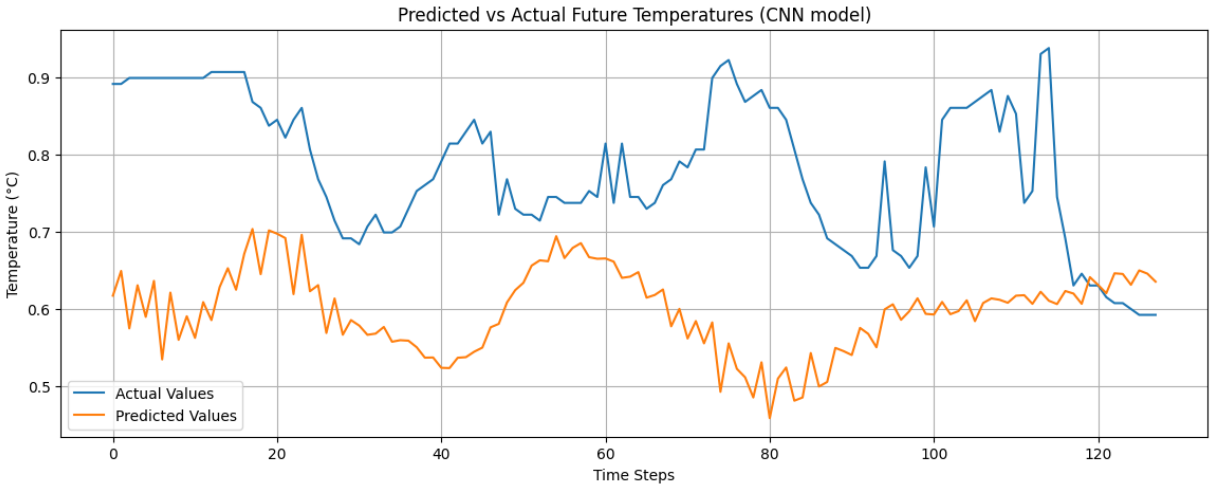


Figure 4.5: CNN-1D predictions for future unseen data, showing rough and unstable behavior.

- **Transformer:**The Transformer model is a powerful deep learning architecture originally designed for sequence-to-sequence tasks, and it has recently been adapted for time-series forecasting. Unlike traditional recurrent models, the Transformer relies entirely on attention mechanisms to learn relationships between time steps, allowing it to handle long-term dependencies more effectively. Its architecture includes an input embedding layer, positional encoding to retain the order of data, a multi-head self-attention mechanism, a feed-forward neural network, and normalization layers. Figure **??** illustrates the custom architecture used in this work for time-series prediction.

  This architecture setup provided a strong foundation for time-series learning, as evidenced by the models testing performance. During the testing phase, the Transformer model demonstrated very good performance. Its predictions closely followed the actual time-series data, capturing both the overall trend and the finer variations. As shown in Figure 4.6, this strong alignment between the predicted and actual values suggests that the model was able to effectively learn the temporal patterns within the training data. The attention mechanism likely played a key role in

this success by allowing the model to consider relationships across a wider range of time steps. Overall, the Transformer showed promising results in terms of accuracy and stability during the test phase.

When used to predict future unseen data beyond the test set, the performance of the Transformer model significantly dropped. As shown in Figure 4.7, the output values deviated strongly from the expected trend, especially in the first 25 predicted steps, where the predictions were highly inaccurate and unstable. This sharp mismatch indicates that the model struggled to generalize to new input sequences that were not part of the training or testing process. Due to this noticeable failure in capturing realistic future patterns, the prediction process was stopped early after observing these initial errors. This result highlights a limitation of the model in real-world forecasting scenarios, despite its strong testing performance.

The following figures provide a visual summary of the Transformer models performance. First, the architecture used in this work is presented, followed by the results during the testing phase and then the predictions on future unseen data. These images are placed one below the other to do an easily compare the models structure with its performance in both phases.

Listing 4.3: Architecture of the custom Transformer model used for time-series forecasting.

```
import tensorflow as tf
from tensorflow.keras import layers, Model

class TimeSeriesTransformer(Model):
    def __init__(self, input_len, d_model=64, num_heads=6, ff_dim=64,
        dropout_rate=0.3):
        super().__init__()
        self.embedding = layers.Dense(d_model)
        self.pos_encoding = self.positional_encoding(input_len,
           d_model)
        self.attention = layers.MultiHeadAttention(num_heads=
           num_heads, key_dim=d_model)
        self.ffn = tf.keras.Sequential([
            layers.Dense(ff_dim, activation='relu'),
            layers.Dense(d_model)
        ])
        self.dropout = layers.Dropout(dropout_rate)
        self.norm1 = layers.LayerNormalization()
        self.norm2 = layers.LayerNormalization()
        self.out = layers.Dense(1)

    def positional_encoding(self, length, depth):
        pos = np.arange(length)[:, np.newaxis]
        i = np.arange(depth)[np.newaxis, :]
        angle_rates = 1 / np.power(10000, (2 * (i//2)) / np.float32(
           depth))
        angle_rads = pos * angle_rates
        angle_rads[:, 0::2] = np.sin(angle_rads[:, 0::2])
        angle_rads[:, 1::2] = np.cos(angle_rads[:, 1::2])
        pos_encoding = angle_rads[np.newaxis, ...]
        return tf.cast(pos_encoding, dtype=tf.float32)

    def call(self, x):
        x = self.embedding(x)
        x += self.pos_encoding
        attn_output = self.attention(x, x)
        x = self.norm1(x + attn_output)
        ffn_output = self.ffn(x)
        x = self.norm2(x + ffn_output)
        x = tf.reduce_mean(x, axis=1)
        return self.out(x)

# Reshape input for Transformer: (samples, timesteps, features)
X_train_tf = X_train.reshape((-1, input_len, 1))
X_test_tf = X_test.reshape((-1, input_len, 1))

model = TimeSeriesTransformer(input_len)
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
model.build(input_shape=(None, input_len, 1))
model.summary()
```

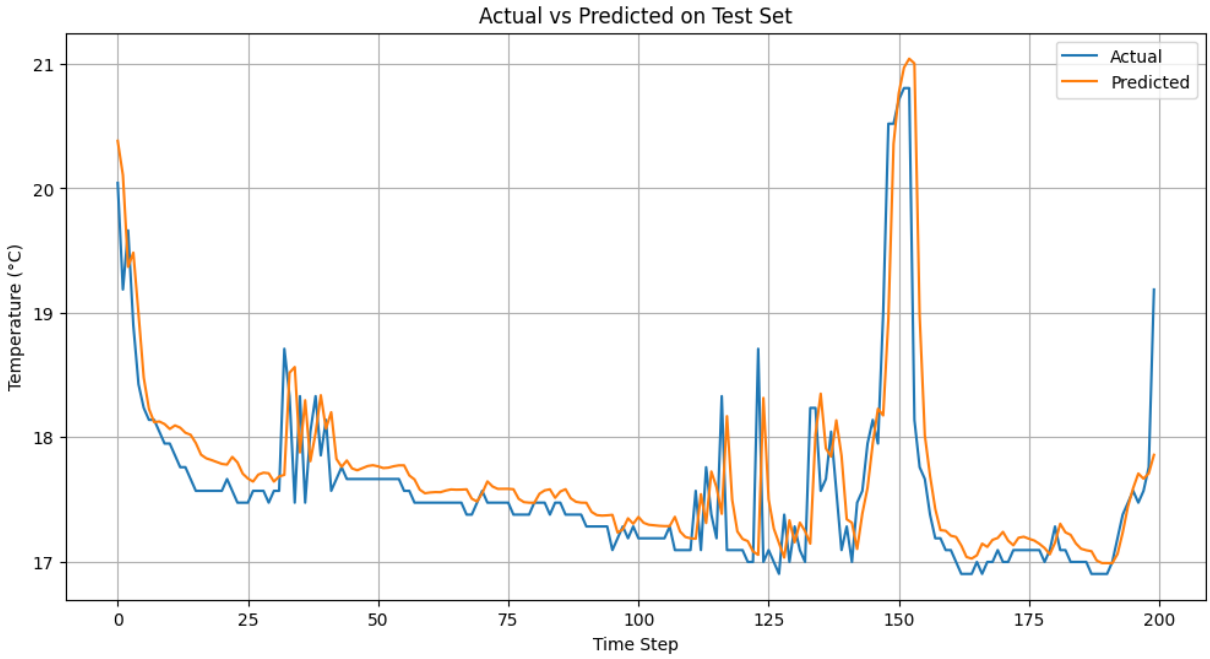


Figure 4.6: Transformer predictions during the testing phase.

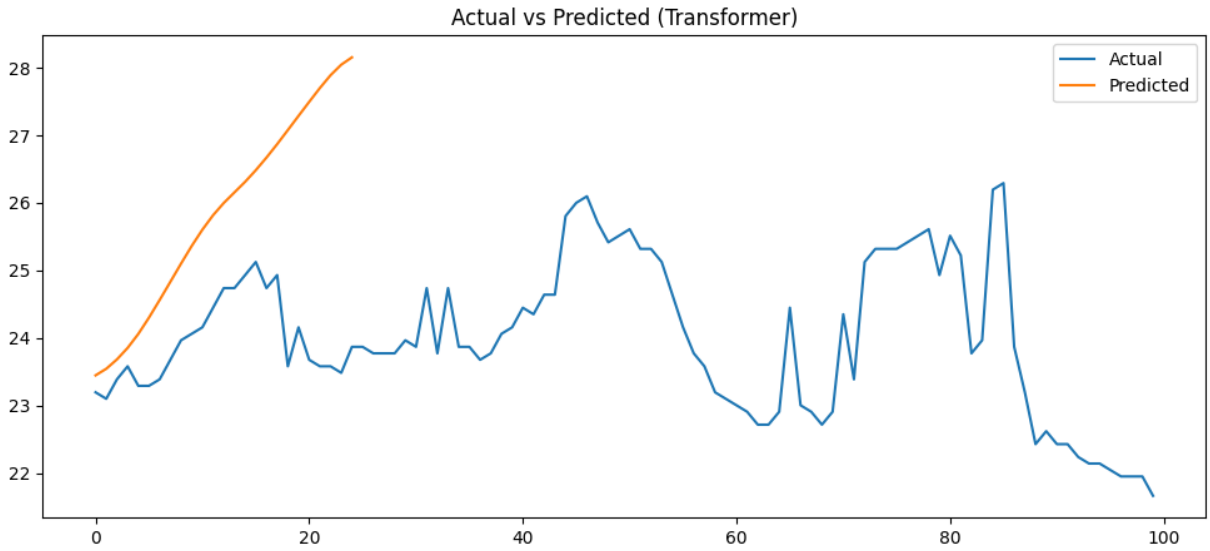


Figure 4.7: Transformer predictions on future unseen data(First 25 Value).

- **TCN:** The Temporal Convolutional Network (TCN) is a deep learning architecture designed to handle sequence data using stacks of causal 1D convolutional layers. Unlike recurrent networks, the TCN processes sequences using parallel operations, which speeds up training and avoids issues like vanishing gradients. Its ability to model long-term dependencies comes from dilated convolutions, which allow the network to capture wide temporal patterns efficiently while maintaining the causal structure of the data.

  When applied to the testing dataset, the TCN model did not perform particularly well. As shown in Figure 4.8, the predicted values followed the general direction of the actual data but lacked accuracy in capturing finer details. The model was able to understand the global shape of the sequence, such as rising or falling trends, but failed to reflect the local fluctuations present in the real time series. This limited resolution may suggest that the TCN learned a smoothed approximation of the data without fully understanding its short-term variations. However, such behavior is not necessarily a flaw in all contextsespecially when broad patterns are more important than exact short-term accuracy.

  The most surprising outcome came during the prediction phase on future unseen data. Despite its underwhelming performance in testing, the TCN model delivered excellent results when forecasting beyond the available time range. As shown in Figure 4.9, the model produced a smooth and stable curve that closely followed the expected trend. Remarkably, the Mean Squared Error for this phase dropped to just 1.13 degrees, making the TCN the top-performing model in this comparison. This significant improvement suggests that TCN may have captured the essential underlying dynamics of the data even though it didnt manifest clearly in the testing step. With further tuning and optimizationsuch as deeper layers, better regularization, or more training dataits forecasting ability could be enhanced even more.

  The figures below provide a visual comparison of the TCN models behavior in both phases. First, Figure 4.8 shows the testing results, followed by Figure 4.9, which displays the prediction performance on future data.

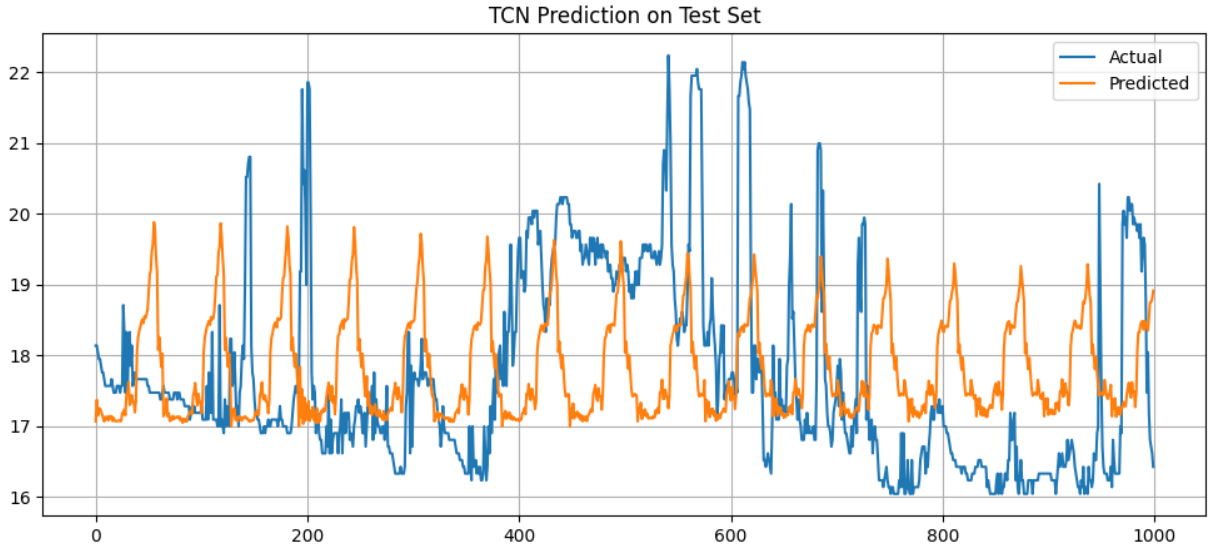


Figure 4.8: TCN predictions during the testing phase. The model captures only the global trend but misses finer variations.

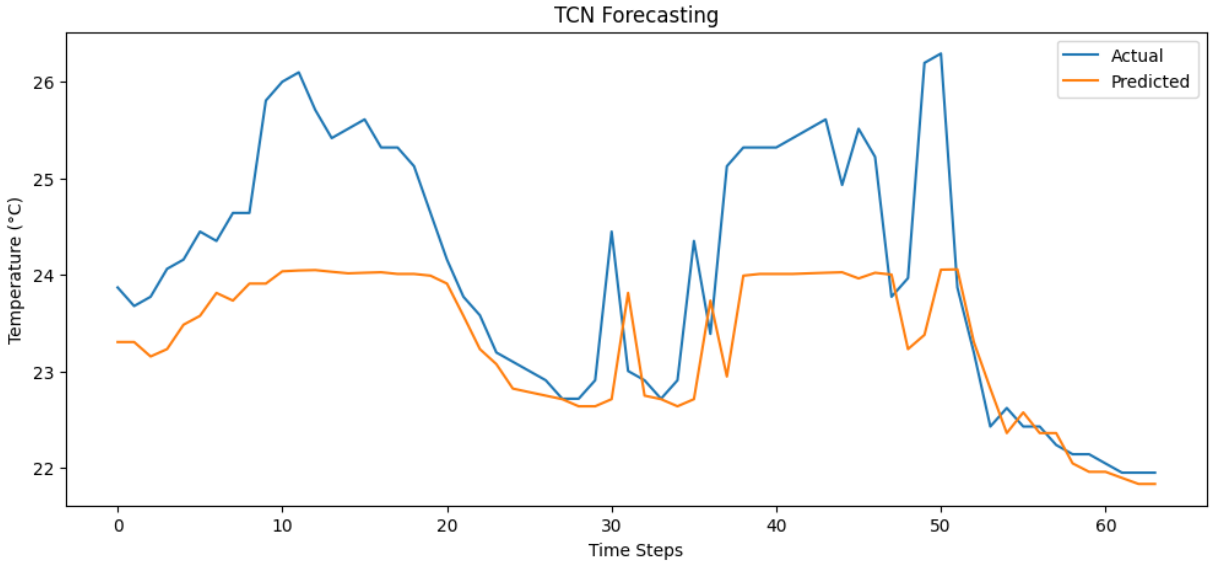


Figure 4.9: TCN predictions on future unseen data. The model achieved the best performance among all tested models with a MSE of 1.13 degrees.

4.5.1.1 Summary:

This section presents the results of five time-series models: LSTM, GRU, CNN-1D, Transformer, and TCN. Most models gave good results during the testing phase, closely following the real data. However, when used to predict future values, only the TCN model gave strong and reliable results. LSTM, GRU, and Transformer worked well on test data but performed poorly on future predictions. CNN-1D also struggled with smooth predictions. Surprisingly, the TCN model, which had weak test performance, gave the best results for forecasting, showing its potential for future time-series tasks.

4.5.2 Results of The MLP Classification Model

In this part, we present the results obtained from the Multi-Layer Perceptron (MLP) model used for a binary classification task. The MLP is a simple yet effective feedforward neural network capable of learning complex patterns from input data. In our case, the model is applied to classify the sensor status as either normal or faulty based on the extracted features. To evaluate the performance of the MLP model, we rely on several metrics, including accuracy, recall, specificity, and the F1-score. These metrics provide a more complete understanding of the models strengths and weaknesses, especially in handling class imbalances and real-world prediction reliability. Both visual outputs and numerical results are presented to support the evaluation. This simple MLP classification model delivered encouraging results in both the training and testing phases, especially considering the limited size and variability of the dataset. The model was trained for 100 epochs with a batch size of 32, using the Adam optimizer and binary cross-entropy as the loss functioncommonly used settings for binary classification problems. By the end of training, the model achieved a training loss of 0.1538 and an accuracy of 93.08%, indicating that it was able to learn the patterns in the data effectively without significant overfitting.

When evaluated on the testing data, the model maintained a strong performance with an accuracy of 93%, confirming its ability to generalize beyond the training set. This level of accuracy, achieved with a relatively simple architecture, suggests that the MLP model is capable of capturing the key features that distinguish between the two classes: normal and faulty sensor status.

To further assess the model's performance and provide a more complete evaluation, we include the confusion matrix and several important metrics: recall, specificity, and F1-score. These metrics help us understand how well the model performs in identifying true positives, avoiding false positives, and maintaining a balance between precision and recall. This detailed evaluation is particularly important in binary classification tasks, where imbalanced data or incorrect predictions can signifi-

cantly impact real-world performance. Overall, the results from this model serve as a strong baseline and demonstrate that even with limited data, meaningful classification performance can be achieved. To evaluate the performance of the MLP classification model, we use the confusion matrix shown in Figure **??**. The results include 9709 true negatives, 882 false positives, 643 false negatives, and 9993 true positives. From this, we calculate important evaluation metrics. The model achieved an accuracy of 93%, with a recall of 0.9395, a precision of 0.9189, and an F1-score of 0.9291. These values show that the model performs well in detecting both classes and provides a good balance between false positives and false negatives.

As shown in Figure 4.10, the confusion matrix provides a detailed view of the model's classification performance.

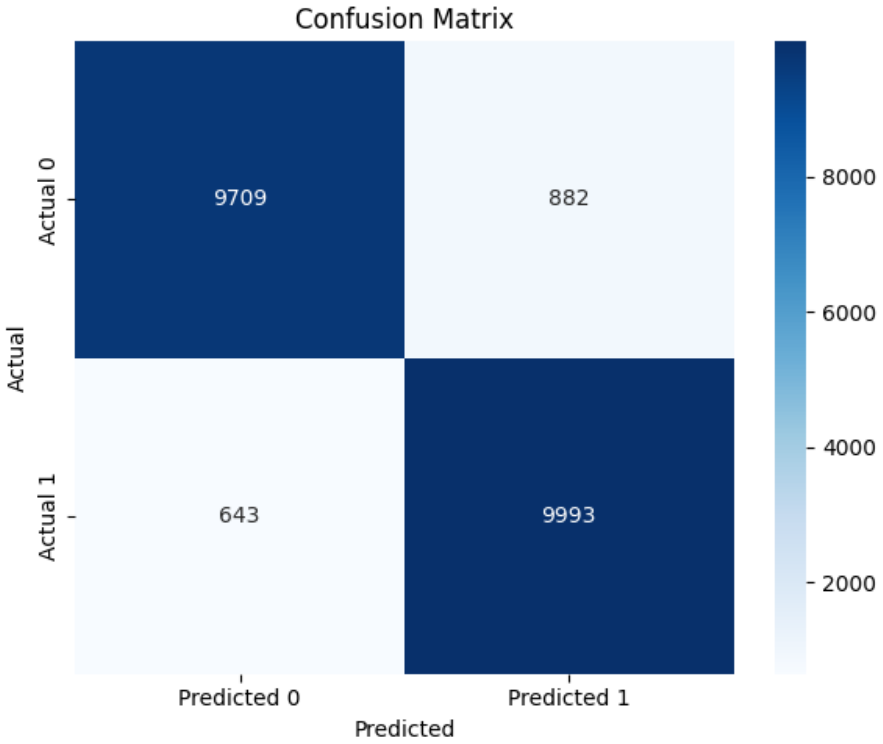


Figure 4.10: Confusion matrix of the MLP classification model.

From the confusion matrix in Figure 4.10, we define the components as follows:

- **True Positives (TP):** The number of instances correctly predicted as class 1.
- **True Negatives (TN):** The number of instances correctly predicted as class 0.
- **False Positives (FP):** The number of instances incorrectly predicted as class 1 while they are actually class 0.

- **False Negatives (FN):** The number of instances incorrectly predicted as class 0 while they are actually class 1.

Using these components, the following evaluation metrics are calculated:

- **Accuracy:**

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Recall (Sensitivity):**

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **Specificity:**

$$\text{Specificity} = \frac{TN}{TN + FP}$$

- **F1-Score:**

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \quad \text{where Precision} = \frac{TP}{TP + FP}$$

4.6 Discussion

In this section, we compare the deep learning models used in this study. These include the models for time-series forecasting (LSTM, GRU, CNN-1D, Transformer, and TCN) and the MLP model used for classification. The goal is to understand how each model behaves, what works well, and where problems appear. This helps us learn which models are better at handling unseen data and which ones may be more useful in real-life situations.

We start by talking about why we only showed the architecture of some models. The models that had poor prediction results were explained in more detail, including their structure. This is because when a model doesnt perform well, its important to see how it works inside. Other researchers might be able to fix the problem or improve the model. On the other hand, the models that gave good results were either simple or based on well-known designs. So, we didnt show their architecture, as they are already easy to understand or repeat.

Next, one major issue we noticed is that most models failed when trying to predict future values. They performed well in the testing phase, but when we gave them new data, the results were not good. This shows that the models didnt really learn the patterns in the datathey just memorized what they saw during training. This is like overfitting, where a model does well on known data but can’t handle

new information. So, even if the testing results looked good, the prediction results tell a different story.

One model that behaved differently is the TCN. It didnt do very well during testing (see Figure 4.8), but when it came to predicting future values, it performed better than all the others (see Figure 4.9). The TCN was able to learn the overall pattern of the data, not just memorize it. This helped it make better predictions and shows that learning general patterns is more useful than remembering data points.

A big reason why TCN works well is because of its special design. It uses something called **causal** and **dilated convolutions**. Causal convolutions make sure the model only uses past data to make predictions, which is important for time-series. Dilated convolutions help the model see a bigger range of past data without needing too many layers or parameters. This helps TCN handle long patterns in the data efficiently. **Other models**, like LSTM, GRU, and CNN-1D, either forget long-term patterns or dont handle time order well. Transformers can also work, but they need extra tools like positional encoding. These advantages make the TCN more powerful in tasks where time and sequence are very important.

Now let's talk about the MLP classification model. It gave good results, with 93% accuracy on the test set. While this is a strong result, in many important applications we need even higher accuracy, closer to 99%. One reason the model didnt reach higher accuracy is the nature of the dataset. The dataset contains 106,132 samples with only 7 features, and many data points are very similar. This makes it hard for the model to tell the difference between correct and incorrect cases. The models low loss value of 0.1538 means it was confident in its answers, but sometimes it still got things wrong because the data is complex and overlapping. This tells us that to improve results, we may need more data or better quality data that is easier to separate.

In summary, this discussion shows that each model has its own strengths and weaknesses. Some models worked well in testing but failed in prediction. Others, like TCN, showed better ability to generalize. The MLP model gave good classification results but couldnt reach perfect accuracy due to the dataset limitations. After reading this section, the reader can now look back at the graphs and plots to better understand the behavior of each model.

4.7 Real-World Applications

The proposed deep learning dual-model system for predicting and validating sensor data in Underwater Named Data Networks (UNDN) has several potential applications across environmental, industrial, and defense domains.

One major use is in the detection and monitoring of **natural phenomena**. The system can help identify early signals of underwater earthquakes, tsunamis, or volcanic activity by recognizing irregular patterns in the data collected from underwater sensors. This can significantly improve the performance of early warning systems and reduce risks to coastal populations and infrastructure.

In **environmental monitoring**, the system can support efforts to track ocean pollution, monitor temperature and salinity changes, and observe marine life behavior. The ability to validate and forecast data reliably is essential for long-term environmental research and effective response planning.

In the **oil and petroleum industry**, the system can play a key role in **detecting leaks and equipment failures**. Changes in sensor readingssuch as sudden shifts in pressure, temperature, or chemical concentrationscan indicate the presence of a leak or abnormal event. Early and accurate detection of such incidents helps prevent environmental disasters, reduces economic loss, and enhances operational safety in offshore drilling platforms or underwater pipelines.

The model is also valuable for supporting **autonomous underwater vehicles (AUVs)** and **remotely operated vehicles (ROVs)** by improving the accuracy of the sensor data they rely on. This is critical for tasks like underwater inspection, exploration, and infrastructure maintenance.

In the **military and defense** field, reliable underwater data is essential for missions involving surveillance, submarine tracking, or mine detection. The ability to validate sensor outputs in real-time improves situational awareness and supports strategic decision-making.

Overall, this work can be deployed in any application where underwater sensor reliability and prediction accuracy are criticaloffering strong advantages in terms of safety, efficiency, and long-term data integrity in challenging marine environments.

4.8 Conclusion

In this chapter, we showed how important it is to choose the right deep learning model for the task you are working on. Even if many models seem similar, each one behaves differently depending on the type of data and the goal. A good model can save a lot of time and give better results, while a wrong choice can lead to poor performance. Our experiments proved that some models work well for

predicting the future, while others are better at learning from past data. This shows that deep learning models are not one-size-fits-alleach has its own strengths, and selecting the most suitable one is key for successful results. .

CHAPTER 5

CONCLUSION AND PERSPECTIVES

In this work, we explored the field of Underwater Wireless Sensor Networks (UWSNs), which are essential for monitoring and collecting data from underwater environments. While UWSNs are important, they face many challenges, such as limited energy, difficult communication, and unstable connections. To solve some of these problems, we introduced a new networking approach called Underwater Named Data Networking (UNDN), which changes how data is requested and delivered. Unlike traditional systems that focus on where data is stored, UNDN focuses on what data is needed, making it more flexible and reliable in complex underwater conditions.

However, UNDN also comes with its own challenges, especially in terms of detecting failures and ensuring that the data collected is accurate and trustworthy. To address these issues, we applied deep learning techniques, which are powerful tools that can learn from data and detect patterns that are hard to find using traditional methods. We used a combination of time-series forecasting models to predict sensor behavior and a classification model to detect sensor failures. The results showed that deep learning can be very helpful in improving the performance and reliability of UNDN, even with limited and complex underwater data.

Overall, this work demonstrates the value of combining advanced networking concepts with deep learning models to solve real problems in underwater environments. While there is still room for improvement, especially with better data and optimized models, this approach offers a strong foundation for future research and practical applications in underwater sensor networks.

Future Work

In the future, we aim to improve the accuracy of the prediction and classification models by collecting more high-quality data and exploring deeper architectures to enhance model performance and stability.

Additionally, we plan to introduce a new model focused on computer vision tasks. This model will process underwater imagery and video data to explore and analyze marine life and environmental conditions in deep-sea regionsareas that remain largely unexplored by scientists. Integrating vision-based analysis with our current sensor data framework could open new pathways for studying biodiversity, detecting anomalies in marine habitats, and understanding the dynamic interactions in the deep ocean environment. This step will significantly extend the capabilities of our system beyond sensor validation to real-time ecological discovery and monitoring.

Another exciting direction is using this technology to better understand what is happening deep in the oceans and seas. By improving our models and collecting more useful data, we could detect unusual behavior in the environment. This could help researchers study natural events like tsunamis, underwater earthquakes, and other important phenomena. In the long term, these systems could become powerful tools for early warning and environmental monitoring, which could save lives and protect ecosystems. This vision shows how deep learning and underwater sensor networks can work together to create smart and safe solutions for the future.

BIBLIOGRAPHY

[1] Shaikh Fattah, Abdullah Gani, Izzatdin Ahmedy, Mohd Yamani Idna Idris, and Ibrahim Abaker Targio Hashem. A survey on underwater wireless sensor networks: Requirements, taxonomy, recent advances, and open research challenges. *Sensors*, 20(18):5393, 2020.

[2] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D. Thornton, Diana K. Smetters, Beichuan Zhang, Gene Tsudik, Dan Massey, Christos Papadopoulos, et al. Named data networking (ndn) project. Technical Report NDN-0001, Xerox Palo Alto Research Center - PARC, Palo Alto, CA, USA, 2010. Technical Report.

[3] Safdar Hussain Bouk, Syed Hassan Ahmed, and Dongkyun Kim. NDN Goes Deep: Foreseeing the Underwater Named Data Networks. In *Proceedings of the 32nd Annual ACM Symposium on Applied Computing (SAC)*, pages 642--646. ACM, 2017.

[4] Esraa Eldesouky, Mahmoud Bekhit, Ahmed Fathalla, Ahmad Salah, and Ahmed Ali. A robust uwsn handover prediction system using ensemble learning. *International Journal of Advanced Computer Science*, 2021.

[5] Yuxuan Zhao, Syed Saad Afzal, Waqas Akbar, Osvaldo Rodriguez, Fan Mo, Dave Boyle, Fadel Adib, and Hamed Haddadi. Towards battery-free machine learning and inference in underwater environments. *arXiv preprint arXiv:2202.08174*, 2022.

[6] Ian F Akyildiz, Dario Pompili, and Tommaso Melodia. Underwater acoustic sensor networks: research challenges. *Ad Hoc Networks*, 3(3):257--279, 2005.

[7] A.A. Maindalkar and S.M. Ansari. Design of robotic fish for aquatic environment monitoring. *International Journal of Computer Applications*, 117(23):31--34, 2015.

[8] M.C. Domingo. An overview of the internet of underwater things. *Journal of Network and Computer Applications*, 35(6):1879--1890, 2012.

[9] Ian F. Akyildiz, Dario Pompili, and Tommaso Melodia. A survey on routing techniques in underwater wireless sensor networks. *IEEE Communications Magazine*, 47(1):97--102, 2005.

[10] John Heidemann, Milica Stojanovic, and Michele Zorzi. Underwater sensor networking: Research challenges and potential applications. *Ad Hoc Networks*, 9(7):1051--1071, 2012.

[11] Milica Stojanovic. Underwater acoustic communications: Design considerations on the physical layer. *Wireless on Demand Network Systems and Services, 2008. WONS 2008. Fifth Annual Conference on*, pages 1--10, 2009.

[12] John Heidemann, Milica Stojanovic, and Michele Zorzi. Research challenges and applications for underwater sensor networking. *IEEE Wireless Communications*, 13(4):12--21, 2006.

[13] Lin Pu, Ying Li, and Jian Xu. Mobility-aware clustering algorithm for underwater wireless sensor networks. In *2013 IEEE International Conference on Communications (ICC)*, pages 225--230. IEEE, 2013.

[14] Shaikh Fattah, Abdullah Gani, et al. A survey on underwater wireless sensor networks: Requirements, taxonomy, recent advances, and open research challenges. *Sensors*, 20(18):5393, 2020.

[15] A.A. Maindalkar and S.M. Ansari. Robotic fish for aquatic monitoring. *International Journal of Computer Applications*, 117(23):31--34, 2015.

[16] C. Author and D. Author. Underwater vehicles for ocean sampling and monitoring. *Ocean Engineering*, 2018.

[17] E. Author and F. Author. Iot and big data in coral reef monitoring. *Sensors*, 2021.

[18] G. Author and H. Author. Hybrid agents for marine environmental monitoring. *Journal of Marine Science and Engineering*, 2022.

[19] Syed A. A. Shah and M. Anwar Hossain. Pipeline monitoring systems and the application of wireless sensor networks: A review. *Sensors*, 8(1):713--730, 2008.

[20] Jungwoo Park, Daeho Lee, and Hojung Kim. Design and implementation of a zigbee-based fish farm monitoring system. *Sensors*, 12(12):16240--16251, 2012.

[21] Zhen Jiang, Jingsheng Wang, and Bin Wang. Underwater acoustic communication in sensor networks. *Wireless Communications and Mobile Computing*, 8(8):977--994, 2008.

[22] Gianluca Antonelli, Andrea Caffaz, Giuseppe Casalino, Nicola C Volpi, Ivo B De Jong, Domenico De Palma, Henrique Duarte, James Grimsdale, Giovanni Indiveri, Sergio Jesus, et al. The widely scalable mobile underwater sonar technology (wimust) h2020 project: First year status. In *OCEANS 2016-Shanghai*, pages 1--8. IEEE, 2016.

[23] S Kumar, A Perry, C Moeller, D Skvoretz, M Ebbert, R Ostrom, S Bennett, and P Czipott. Real-time tracking magnetic gradiometer for underwater mine detection. In *Oceans'04 MTS/IEEE Techno-Ocean'04*, volume 2, pages 874--878. IEEE, 2004.

[24] Usha Jain and Muzzammil Hussain. Security mechanism for maritime territory and frontier surveillance in naval operations using wireless sensor networks. *Concurrency and Computation: Practice and Experience*, 33(17):e6300, 2021.

[25] Irfan Ahmad et al. Analysis of security attacks and taxonomy in underwater wireless sensor networks. *Wireless Communications and Mobile Computing*, 2021:1--20, 2021.

[26] John Heidemann, Milica Stojanovic, and Michele Zorzi. Research challenges and applications for underwater sensor networking. *IEEE Wireless Communications*, 20(3):12--18, 2012.

[27] Ling Pu, Zhi Peng, Jun-Hong Cui, and Zhi Wang. Challenges and opportunities in underwater sensor networks. *Wireless Communications and Mobile Computing*, 11(8):1013--1023, 2011.

[28] Shengming Jiang, Jie Wang, Zhi Wang, and Jun-Hong Cui. Security in underwater sensor networks. *IEEE Wireless Communications*, 15(5):52--59, 2008.

[29] Benarfa Abdelmadjid. *Communication security in the Future internet architectures*. PhD thesis, Ammar Telidji University, 2021.

[30] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. Networking named content. *Communications of the ACM*, 55(1):117--124, 2012.

[31] David D Clark. The design philosophy of the darpa internet protocols. *ACM SIGCOMM Computer Communication Review*, 18(4):106--114, 1988.

[32] Lixia Zhang, Alexander Afanasyev, Jeff Burke, Van Jacobson, kc Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66--73, 2014.

[33] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nick Briggs, and Rebecca L Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1--12, 2009.

[34] Alexander Afanasyev, Illya Moiseenko, and Lixia Zhang. A brief introduction to named data networking. *IEEE Communications Magazine*, 56(9):38--43, 2018.

[35] Abdelmadjid Benarfa, Sofiane Dahmane, Bouziane Brik, and Mauro Conti. Ndn diving deeper: A comprehensive survey of named data networking for underwater sensor networks (uwsn). *Journal of Network and Computer Applications*, 200:102345, 2024.

[36] Abdelmadjid Benarfa, Sofiane Dahmane, and Bouziane Brik. A conceptual framework for predictive maintenance of underwater sensors using named data networking and machine learning. In *Proceedings of the 2024 International Conference on Internet of Things and Intelligence (ICITI)*. IEEE, 2024.

[37] Abdelmadjid Benarfa, Sofiane Dahmane, Bouziane Brik, and Mohamed Bachir Yagoubi. Predictive maintenance for uwsns: A practical implementation of fault detection using ndn and deep learning. *Internet Technology Letters*, 7(6):e630, 2024.

[38] R. Thiyagarajan and N. Nagabhooshanam. A novel approach for missing data recovery and fault nodes detection in wireless sensor networks. *International Journal of Communication Systems*, 37(3):e5924, 2024.

[39] M. Karthihadevi, J. Relin Francis Raj, N. Sumi, S. Vanitha, et al. Deploying deep learning for real-time tsunami monitoring: A cnn-lstm model for sensor-based prediction. In *2025 International Conference on Inventive Computation Technologies (ICICT)*. IEEE, 2025.

[40] F. L. S. de Matos, A. R. de Paula, C. L. S. Sampaio, C. E. L. Ferreira, J. Claudet, and A. C. Bastos. Data from: Temporal variability of water temperature in rocky reefs in the coast of santa catarina, brazil. *Dryad Digital Repository*, 2015.