International Journal of Mathematical, Engineering and Management Sciences
Vol. 11, No. 2, 525-544, 2026
https://doi.org/10.33889/IJMEMS.2026.11.2.022

# Enhanced Temporal Convolutional Network Based Approach for Degradation Prediction of Reverse Osmosis Systems

**The-Son Phan**
Faculty of Mathematics, Mechanics, and Informatics,
VNU University of Sciences, 334 Nguyen Trai Street, Thanh Xuan District, Hanoi, Vietnam.
E-mail: phantheson@hus.edu.vn

**Thanh-Ha Do**
Faculty of Artificial Intelligence,
Posts and Telecommunications Institute of Technology, Nguyen Trai Street, Ha Dong District, Hanoi, Vietnam.
E-mail: dothanhha@ptit.edu.vn

**Phuc Do**
SyCoIA, IMT Mines Ales, Ales, France.
*Corresponding author*: phuc.do@mines-ales.fr

(Received on July 22, 2025; Revised on October 19, 2025 & November 25, 2025; Accepted on December 20, 2025)

**Abstract**
Reverse Osmosis (RO) degradation underscores the importance of predictive capabilities to develop optimal maintenance strategies that minimize losses. In this study, we develop a Temporal Convolutional Network (TCN) model to predict the RO system states using the primary indicator for RO analysis: the fluctuations in differential pressure across the RO vessel. Specifically, data from a real desalination plant for the period 2015 to 2020 are used. The dataset encompasses 14 RO train operations, including routine operations, significant maintenance events, temporary shutdowns, and element replacements. The proposed approach uses temporal convolutional operations to capture the dynamic pressure behavior at both ends of the membrane, enabling faster, more accurate anomaly detection. A key challenge in applying deep learning to this domain is the heavy reliance on real-world operational data. The approach involves a strong data preprocessing strategy that reveals subtle relationships between operating time and pressure dynamics. Accurate prediction of membrane degradation also enables preventive and recovery actions, which reduce maintenance expenses. The proposed method is evaluated against conventional models, including LSTM, CNN-LSTM, and GRU, using data from the real desalination plant. Experimental results demonstrate that the proposed model achieves the lowest prediction error and shows strong potential for deployment in practical desalination operations.

**Keywords-** Degradation forecasting, Reverse osmosis system, Deep learning, Filters.

## 1. Introduction

Although water covers 71% of the Earth, more than 97% of it is saltwater. However, the world population is increasing rapidly, so the demand for clean water is essential. To meet this demand, various techniques have been developed to produce clean water for homes and industries. Desalination is one such prominent technique (Alsawaftah et al., 2022). Consequently, Reverse Osmosis (RO) technology has become a cornerstone of desalination plants worldwide. However, a significant challenge these plants face is system impairment due to membrane fouling. Based on the type of accumulated residue, membrane fouling is categorized as particle, organic, inorganic, and biofouling. Biofouling is considered the most severe and challenging to mitigate (Alsawaftah et al., 2022; Villacorte et al., 2017).

As documented in Rooij et al. (2021) and Villacorte et al. (2017), Algal blooms significantly contribute to biofouling in water treatment systems. The organic compounds these blooms release form a slippery layer on membrane surfaces, hindering filtration efficiency. This layer increases the passage of salts and pressure

through the membrane, potentially leading to reduced filtration performance or even irreversible membrane damage (Koutsakos & Moxey, 2007). To improve membrane longevity, several strategies can be employed: (a) continuous monitoring of membrane performance, (b) regular chemical cleaning (clean-in-place or CIP) (Koutsakos & Moxey, 2007; Rooij et al., 2021), and (c) timely replacement of fouled membranes (Koutsakos & Moxey, 2007; Rooij et al., 2021). Mastering the evolution of membrane degradation, which is represented through its differential pressure fluctuation, is crucial for optimizing membrane performance and maintenance.

In this paper, we develop an artificial intelligence-based prediction method for forecasting differential pressure fluctuations in RO systems.

## 2. Literature Review and Technical Contributions

### 2.1 Prediction Approaches Differential Pressure Fluctuations in RO Systems

Many research studies have proposed forecasting differential pressure fluctuations in RO systems, some of which have yielded positive results. The researchers have taken three directions: experience-based, physical-based, and data-driven approaches (Guo et al., 2020).

In the first direction, experience-based models integrate experts' expertise with statistical analysis to forecast membrane degradation and suggest corrective measures. These models are advantageous since historical failure data is limited, or real-time interpretability is required. In general, expert experience-based models can easily understand and apply the rules, do not require extensive computation, and the rules help specific RO system configurations and water qualities. However, the fixed rules may not adapt well to new degradation patterns. The effectiveness of the expert experience-based models relies heavily on the accuracy of expert-defined rules. Sometimes, they may struggle with complex, multi-variable interactions without additional support from machine learning.

The second direction is physical-based models. Wreyford et al. (2020) researched on the physical equations involved in the operation of the desalination using RO technology. In general, the approaches use the engineering principles and system parameters, including flow dynamics, membrane properties, and fouling behavior to build the models. Since models depend on parameters, they are sensitive to the initial conditions and parameters. In addition, real-time sensors and Internet of Things technology are widely used in RO systems to monitor pressure. Multiple sensors capture different variables, such as flow rate, feedwater temperature, and membrane pressure, and help improve predictive capability by mining these variables complementary.

In the third direction, the researchers focused on developing machine learning models to predict failures or identify patterns in differential pressure data that precede faults. First, the machine learning models are combined with data assimilation techniques in physical-based approaches to form hybrid models to improve prediction accuracy. The hybrid models (Lim et al., 2019) are effective in forecasting fluctuations in differential pressure since they integrate noisy, sparse data with well-established physical laws, improving model accuracy. Besides, machine learning algorithms (Chandola et al., 2009; Zhou & Paffenroth, 2017), in combination with sensor data, detect abnormal fluctuations that can be the reason for fouling or scaling.

Some other prominent techniques include using time-series forecasting for failure prediction (Masum et al., 2018), the autoencoder-based models for fault diagnosis and anomaly detection (Ahmad et al., 2020), fault classification using supervised learning (Inoue et al., 2017), and recently, reinforcement learning being Q-learning or Deep Q Networks (DQN) models Bonny et al. (2022) and Soleimanzade et al. (2022) are developed and applied. Deep learning models (LSTM, GRU, CNN-LSTM hybrids) have the potential to

forecast pressure fluctuations in RO systems since they effectively capture non-linear dynamics and temporal dependencies. Relevant deep learning models include RNN (Sherstinsky, 2018), LSTM (Staudemeyer & Morris, 2019), GRU, GNN (Zhou et al., 2021), and more recently, the Transformer of Liu et al. (2022) has emerged. The introduction of one-direction convolutional layers (Kiranyaz et al., 2019) and the Temporal Convolutional Network (Lea et al., 2016) (TCN) has also yielded some promising results.

It is also worth mentioning that with the development of artificial intelligence (AI), Niu et al. (2022) provides a detailed overview of applying AI algorithms for predicting fouling in membrane processes over the past 20 years. The results indicated that ANNs predicted effectively membrane permeate flow in Reverse Osmosis (RO) and membrane distillation (MD) systems, achieving $R^2$ values greater than 0.97. However, for membrane bioreactor (MBR) systems, the ANN predictions yielded $R^2$ values between 0.85 and 0.99. Hwang et al. (2010) used a multilayer feedforward neural network (MLP) model to forecasting fouling rates within a pilot-scale MF (microfiltration) unit used to purify water from the Han River. The model accurately predicted fouling rates ($R^2 = 0.92$) and permeability ($R^2 = 0.94$).

Shim et al. (2021) implemented an LSTM recurrent neural network model, a variant of RNN that can store long-term information from past values. The objective is to forecast the evolution of fouling layer thickness and permeate flow rate through the membrane. The model accurately predicted permeate flow rate ($R^2 = 0.9982$) as well as fouling layer thickness ($R^2 = 0.9987$). In addition, Shi et al. (2022) developed the CBAM-MUL-CNN model, that utilizes Convolutional Block Attention Module to overcome the challenge of insufficient feature extraction ability of biofilm in biofilm system, resulting in complex structure of membrane fouling data, making it impossible to identify and classify membrane fouling in biofilm system effectively.

In conclusion, deep learning and reinforcement learning models are highly accurate and adaptable in real time. Hybrid models can work well in dynamic, complex systems, and integrated approaches combining multiple methods (e.g., hybrid neural networks, RL, and ensemble models) are promising for optimizing RO system performance and minimizing time computing.

## 2.2 Limits of Existing Models and Scientific Contributions

Bai et al. (2018) provided a comparison for the Temporal Convolutional Networks (TCN) and found promising results when compared with standard models like RNN, LSTM, and GRU in handling sequence data (see **Table 1)**. TCN have demonstrated significant superiority over traditional Recurrent architectures (such as LSTM, GRU, and vanilla RNN) in sequence processing problems. With the ability to capture long-term relationships and avoid the vanishing gradient phenomenon, TCN performs better on various tasks, including language modeling, sequence recognition, and music modeling. MNIST and Permuted MNIST also significantly reduce loss and perplexity in problems such as adding problems, copy memory, and char-level language modeling.

Notably, the causal convolution structure and parallelization capabilities of TCN reduce time computing, overcoming the sequential processing drawbacks of recurrent networks. These advantages present that TCN is an effective and potential solution for long-term and computationally demanding problems. More specifically, Hewage et al. (2020) employed a Temporal Convolutional Network (TCN) to predict ten weather parameters, using data from nearby weather stations that included ten meteorological variables (e.g., barometric pressure, temperature, humidity, precipitation, wind speed, dew point, etc.). Evaluation results demonstrate that the TCN model has superior performance compared to Standard Regression (SR), Support Vector Machine (SVM) and LSTM models in terms of the Mean Squared Error (MSE).

**Table 1.** Performance of different models on sequence modeling tasks (Bai et al., 2018).

| Domain | Task/Dataset | Metric | # Parameters | LSTM | GRU | RNN | TCN |
|---|---|---|---|---|---|---|---|
| Image | Sequence MNIST | Accuracy | 7.0e4 | 87.2 | 96.2 | 21.5 | **99.0** |
| | Permuted MNIST | Accuracy | 7.0e4 | 85.7 | 87.3 | 25.3 | **97.2** |
| Synthetic | Adding problem (T=600) | Loss | 7.0e4 | 0.164 | **5.3e-5** | 0.177 | 5.8e-5 |
| | Copying memory (T=1000) | Loss | 1.6e4 | 0.0204 | 0.0197 | 0.0202 | **3.5e-5** |
| Audio | JSB Chorales | Loss | 30.0e4 | 8.45 | 8.43 | 8.91 | **8.10** |
| | Music Nottingham | Loss | 1.0e6 | 3.29 | 3.46 | 4.05 | **3.07** |
| Language | Penn Treebank (word) | Perplexity | 130.e6 | **78.93** | 92.48 | 114.50 | 88.68 |
| | Wiki-103 (word) | Perplexity | - | 48.4 | - | - | **45.19** |
| | LAMBADA (word) | Perplexity | - | 4186 | - | 14725 | **1279** |
| | Penn Treebank (char) | Bpc | 3.0e6 | 1.36 | 1.37 | 1.48 | **1.31** |
| | Text8 (char) | Bpc | 5.0e6 | 1.50 | 1.53 | 1.69 | **1.45** |

However, pure deep learning models often face cumulative errors in sequential prediction. When using the model output to continue predicting the next step, errors can accumulate quickly because the model does not have a control mechanism based on physical principles. There are some directions to overcome the limitations of pure deep-learning models: using a physical-based combined loss function and enhancing the existing deep-learning model, focusing on understanding/processing data to be used for real data easily. In this paper, we focus on the second direction.

Our research propose to solve complex problems by developing a more detailed description of depth, which is described in Section 4. Furthermore, the collected data exhibits high variability, which mitigates using a Moving Average filter, as proposed by Rooij (2022). Similar research by Rooij (2022) utilized statistical modeling techniques to estimate parameters, achieving an $R^2$ score of up to 98%.

## 3. Desalination Plant Systems and Dataset

The studied desalination plant is a system comprising over 2000 individual pressure vessels. These vessels are grouped into 14 independent operating units known as RO (Reverse Osmosis) trains. These trains operate in parallel, allowing for high operational flexibility and redundancy.

Each vessel within a train houses 8 serially connected sockets (or membrane elements). These sockets accommodate spirally wound membrane elements, thus facilitating the efficient flow of seawater through the multiple membrane layers within each element (see **Figure 1**).

### 3.1 Membrane Biofouling

The authors in Rooij et al. (2021) show that algal blooms significantly negatively impact the performance of Reverse Osmosis (RO) membranes. The quality of the feedwater entering the RO system reflects the presence and intensity of such algal events.

A key driver of this accelerated membrane degradation is algae's organic matter produced during the algal life cycle. This organic matter, collectively termed Algae-Derived Organic Matter (AOM), comprises a complex mixture of substances, with Transparent Exopolymer Particles (TEP) particularly problematic.

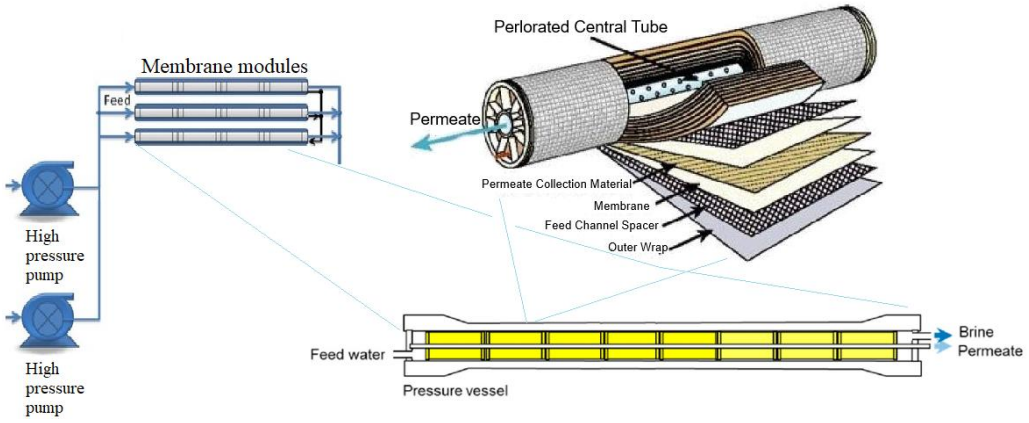



**Figure 1.** Structure of an RO pressure vessel containing eight sockets. High-pressure feed water flows through the series of elements from the first to the tail element (Rooij, 2022).

As algal cells die and decompose, they release AOM into the surrounding water. The AOM deposits on the membrane surface, creating a cohesive fouling layer that hinders water flow and promotes the growth of microorganisms. Furthermore, biofouling on the membrane surface exacerbates the situation, providing an even more favorable environment for AOM accumulation and subsequent biofilm formation.

The adsorption of TEP creates a cohesive film on submerged surfaces. This phenomenon is exacerbated by Dinoflagellates, which release substantial TEPs during phases of nutrient depletion. As a result, the deleterious effects of biofouling extend beyond the duration of the actual algal bloom. Increased fouling will lead to increased pressure difference between the two ends of the RO vessel.

The increasing thickness of the biofouling layer leads to a gradual rise in the transmembrane pressure (TMP), which denotes the pressure gradient existing between the feed and the permeate side. This increased TMP reflects the growing resistance to water flow as the membrane becomes increasingly clogged. **Figure 2** illustrates the growth of the differential pressure over time. The influence of algal blooms rapidly increases the differential pressure.

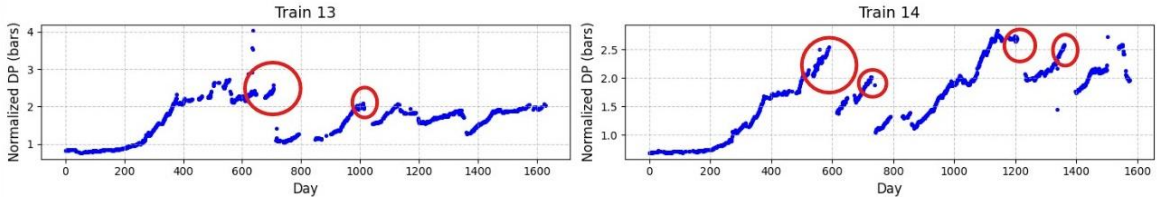



**Figure 2.** Time evolution of pressure differentials in RO trains 13 and 14. The red-highlighted area marks the beginning of severe biofouling, characterized by a sudden increase in hydraulic resistance across the membrane.

## 3.2 Membranes Restoration

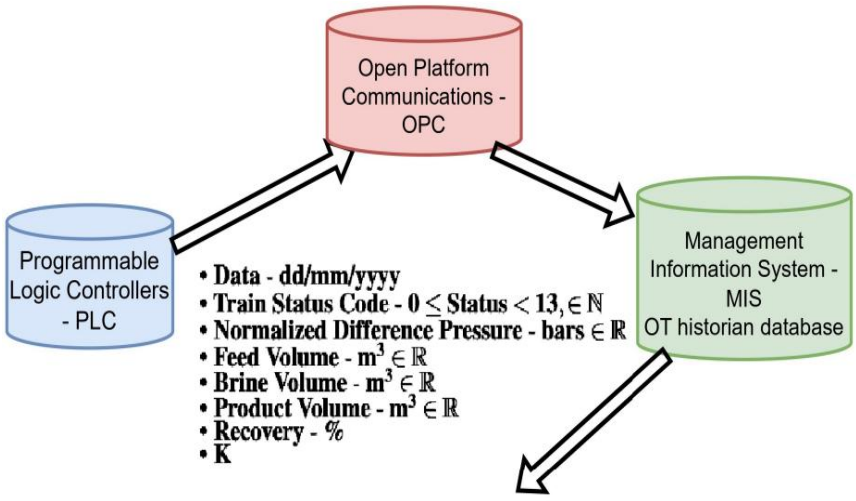
To minimize performance degradation due to fouling, specific remediation strategies are implemented in reverse osmosis (RO) vessels. These maintenance measures are systematically categorized into four distinct maintenance operations:

1) **Chemical Cleaning (Clean-In-Place or CIP)**
   - *C*1 is a two-step process involving a high-pH wash (using a sodium hydroxide solution) followed by an acid wash (using hydrochloric acid) (Rooij, 2022).
   - *C*2 is a more intensive approach. The elements are treated with a sodium bisulfate soak subsequent to the C1 cleaning phase.
2) **Element Redistribution**: *C*3 is a method that involves physically repositioning the elements within the vessel. The element with maximum biomass (Si) is removed for cleaning and relocated to the end of the pressure vessel. The elements S2 to S8 are shifted forward by one position (Rooij, 2022).
3) **Vessel Restructuring**: involves rearranging the element positions within the vessel, sometimes replacing some elements with new ones (Rooij, 2022).
4) **Other Corrective Measures** encompasses some corrective actions, such as low salinity flushing, depressurization, and instrument adjustments.

The effectiveness of each cleaning method varies, emphasizing the importance of a well-defined maintenance plan to optimize operational efficiency and cost-effectiveness.

### 3.3 Dataset and Problem Formulation

Data is collected from multiple sources, including the Historian database for temporal records and the Computerized Maintenance Management System (CMMS) for tracking and incorporating maintenance activities, the durations of algal algae bloom, element condition assessments, output requirements, and supply logistics derived from periodic reports submitted to interested parties.




| Date | Day | week | Train Status Code | Normalized DP (bars) | Feed Volume ($m^3$) | Brine Volume ($m^3$) | Product Volume ($m^3$) | Recovery (%) | K | Train Status |
|---|---|---|---|---|---|---|---|---|---|---|
| 11/7/2015 | 1 | 0.14 | 5 | 0.780 | 21243 | 10375 | 10867 | 51.2 | 1.0 | Operation |
| 11/8/2015 | 2 | 0.29 | 5 | 0.776 | 30060 | 14709 | 15351 | 51.1 | 1.0 | Operation |
| 11/9/2015 | 3 | 0.43 | 5 | 0.776 | 30065 | 14712 | 15353 | 51.1 | 1.0 | Operation |
| 11/10/2015 | 4 | 0.57 | 5 | 0.775 | 29762 | 14562 | 15201 | 51.1 | 1.0 | Operation |
| 11/11/2015 | 5 | 0.71 | 5 | 0.773 | 29371 | 14352 | 15019 | 51.1 | 1.0 | Operation |
| 11/12/2015 | 6 | 0.86 | 5 | 0.778 | 29563 | 14443 | 15120 | 51.1 | 1.0 | Operation |
| 11/13/2015 | 7 | 1.00 | 5 | 0.779 | 11067 | 5411 | 5656 | 51.1 | 1.0 | Operation |
| 11/14/2015 | 8 | 1.14 | 5 | 0.763 | 16893 | 8256 | 8637 | 51.0 | 1.0 | Operation |
| 11/15/2015 | 9 | 1.29 | 5 | 0.769 | 29278 | 14038 | 15240 | 52.1 | 1.0 | Operation |
| 11/16/2015 | 10 | 1.43 | 5 | 0.769 | 29215 | 14002 | 15214 | 52.1 | 1.0 | Operation |
| 11/17/2015 | 11 | 1.57 | 5 | 0.773 | 29235 | 14001 | 15234 | 52.1 | 1.0 | Operation |
| 11/18/2015 | 12 | 1.71 | 5 | 0.775 | 29761 | 14251 | 15510 | 52.1 | 1.0 | Operation |

**Figure 3.** The diagram illustrates the process of collecting secondary data, through MIS.

Input and output data are saved temporarily in Programmable Logic Controllers (PLCs). In this study, the raw analog signals are converted into a digital format by an Analog-to-Digital Converter. (ADC) located at the Remote Input/Output (RIO) modules. Subsequently, the data is aggregated via an independent Open Platform Communications (OPC) server. The Management Information System (MIS) is where the data is stored and used for this research. The entire data collection process is illustrated in **Figure 3.**

CMMS (Computerized Maintenance Management System) is a maintenance management system that helps monitor, plan, and manage equipment maintenance in the RO seawater filtration system. The status of key equipment such as the seawater intake pump (Intake Pump), low-pressure pump (LPB Pump), high-pressure pump (HPB Pump), and CIP (Clean-in-Place) system is monitored every hour. Maintenance staff manually record maintenance tasks such as filter membrane replacement or CIP cleaning in the CMMS system.

Differential pressure (DP) is sampled hourly. To minimize signal noise and ensure stability over time, these raw data are aggregated into a daily normalized differential pressure (NDP) average. The collected data is secondary and is stored in an Excel file containing key fields such as date, train status code, normalized differential pressure (bars), feed volume ($m^3$), brine volume ($m^3$), product volume ($m^3$), and membrane recovery - R (%). Operational states are encoded as integers in the interval [0, 13); specifically, code 5 denotes online production, while code 12 indicates Clean-in-Place (CIP) protocols.

## 4. Temporal Convolutional Network-based Differential Pressure Forecasting

The proposed approach is illustrated in **Figure 4**. It consists of four stages: (i) data preprocessing: perform transformations to represent the system's relationship between normalized differential pressure (NDP) and the operational state (Train Status), (ii) data smoothing: evaluate the smooth method to stabilize the data, which improves the expected accuracy, (iii) model training: propose the seq-to-seq TCN and describe the testing process with different forms to evaluate the performance, and (iv) evaluation: test the expected results on several RO trains. For each RO train, the time series is split chronologically into Train/Validation/Test. The predictor is NDP.

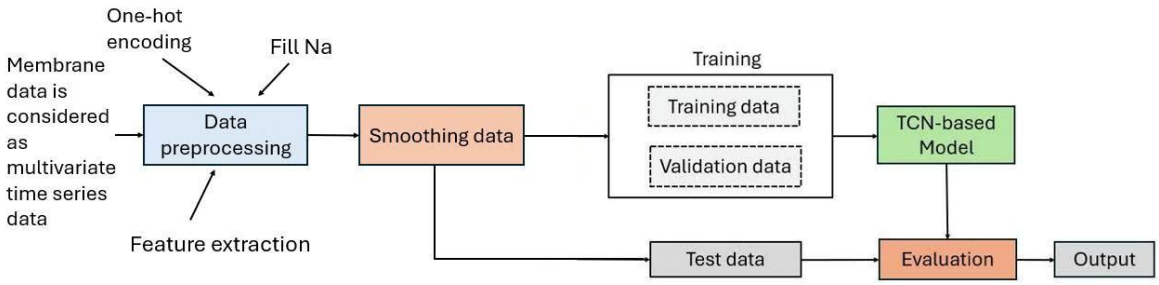



**Figure 4.** Illustration of the proposed prediction method, the data are split by time into three separate sets: train, validation and test.

### 4.1 Data Preprocessing

This study utilizes a dataset sourced from a desalination plant. The data collected from 14 operational RO trains over six years (2015-2020) comprises two kinds of variables:

**(1) Normalized Differential Pressure (NDP):** is the target variable representing the pressure difference across the RO membranes.

**(2) Train Status:** reflects the operational condition of each train, including three following states:

- ***Operation*:** Indicates regular operation.

- ***C*1, *C*2, *C*3:** Represent different chemical cleaning procedures.
- ***Permutations*:** Denotes the sequence of membrane elements within the vessel. The permutation state is encoded as a string of numbers (1-8), and the character 'N' denoted the installation new element at that specific position.

For instance, the sequence '324N5678' indicates that the element originally in the third position has shifted to the first, while element 4 has moved to the third. A new element ('N') is installed at the fourth position, leaving element 2 and elements 5 through 8 in their original locations.

Our study employs a rule-based approach to present the relationship between element positions, train operating status, and the target variable (Normalized Differential Pressure—NDP). A rule-based approach is described as follows:

**(1) Feature Engineering**:
- Additional features *Si,* (*i* = 1...8) denote the indices of the membrane elements within the train. For instance, new element replacements ('N') are assigned the value 0, converting '324N5678' into '32405678'.
- The 'Train Status' is transformed to the 'Permutation' format for all element restructuring methods.

**(2) *C*3 Method Representation**:
- The *C*3 method, involving element redistribution, is parameterized by the variables *S*1 through *S*8 by the sequence ”23456781,” indicating the new order of the elements after the procedure.
- When the system is in Operation, the socket values are derived from the most recent membrane restoration event.

***Handling system downtime***
- To account for system downtime during maintenance (typically 3-5 days), these periods are merged into a single day, and NaN values are assigned to the target variable for these merged days.
- Subsequently, the missing NDP values are imputed by filling them with the NDP value of the preceding operational period.

**(3) One-Hot Encoding for Train Status**: The 'Train Status' variable is transformed using One-Hot Encoding. This technique converts categorical variables defined by the set $L = \{l_1, l_2, \ldots, l_n\}$ into binary vectors *Vi* of length *n*, where the index of the active label is marked with a 1, and all remaining elements are set to 0. This binary representation simplifies data processing and subsequent model training.

**(4) Data Normalization**: All numerical features are normalized to a range of 0 to 1.

In addition, advanced smoothing techniques are applied to improve both accuracy and reliability of the data obtained after a rule-based technique. Specifically, a Savitzky-Golay filter (Gallagher, 2020; Savitzky & Golay, 1964) and a Simple Moving Average (Rooij, 2022) are used to mitigate the effects of noise and fluctuations on the values of normalized differential pressure (NDP) (**Figure 5)**. Comparing the smoothed series with the original series helps to identify underlying trends and patterns more effectively. Data smoothing facilitates the learning process, enabling greater accuracy in capturing the inherent dynamics of the system.

A Savitzky-Golay filter (S-G filter) is a digital signal processing technique developed to smooth data while preserving the underlying signal trends. The S-G filter functions by performing a local polynomial regression to smooth the signal. Specifically, for a dataset *Xj*, (*j* = 1, *...n),* the filter fits a polynomial of degree k utilizing a sliding window of size *w*. The approximating equation is expressed as follows:

$$\hat{y}_i = a_0 + a_1 z + a_2 z^2 + \ldots + a_k z^k \quad (1)$$

where, $y_i$ represents the smoothed data point at position $i$; $a_0$, $a_1$, ..., $ak$ are the coefficients of the polynomial, and $z$ is an independent variable (a vector) that takes values from $\frac{1-w}{2}$ to $\frac{1+w}{2}$.

This relationship can be concisely represented in matrix form as:

$$\hat{Y} = J \bullet a \tag{2}$$

where, $Y$ is a vector that contains the smoothed data points within the window; $J$ is the Vandermonde matrix, which depends on the values of $z$ within the window, and $a$ presents the polynomial's coefficients. The optimal values $a$ are determined by solving equations defined in Equation (3).

$$\hat{a} = \underset{a}{\operatorname{argmin}} ||J \bullet a - Y||^2 \tag{3}$$

The second filter employed is the Simple Moving Average (SMA). Technically, SMA computes the unweighted mean of the preceding $k$ data points at each timestamp t. While SMA introduces a slight phase lag, it is highly effective at suppressing high-frequency noise, thereby stabilizing the underlying trend line.

A standard formulation of an equally weighted moving average involves calculating the mean of the most recent $k$ data points within a dataset comprising $n$ observations. For the sequence $\{p_1, p_2, \ldots, p_n\}$, the SMA over the last $k$ points, denoted as MA, is given in Equation (4):

$$SMA = \frac{1}{k}\sum_{i=n-k+1}^{n} p_i \tag{4}$$

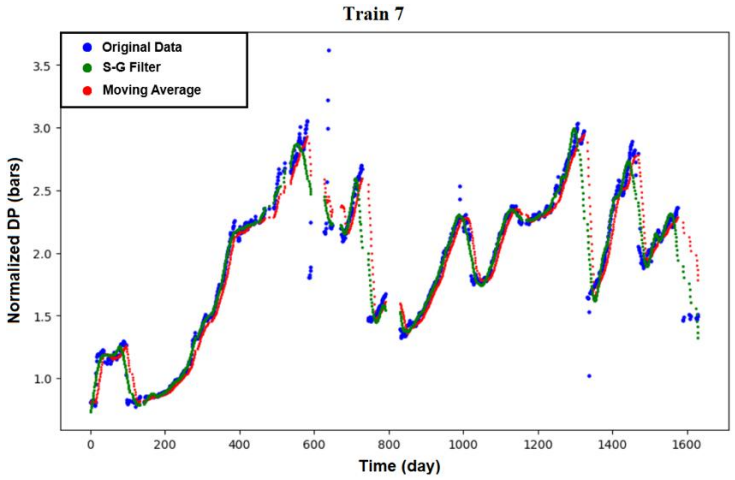



**Figure 5.** Normalized Differential Pressure (NDP) trends in RO Train 7 from 2015 to 2020. The image shows the comparison of smoothing methods. The blue line represents the raw data, while the green line indicates the output of the S-G filter with day = 120 and 4-th degree polynomial, and the red line being the moving average method with window size = 10.

## 4.2 Enhanced Temporal Convolutional Networks

Besides LSTM and GRU, which are famous and have become the dominant choices for forecasting pressure fluctuations in RO systems, TCN (Temporal Convolutional Network) (Bai et al., 2018) is a robust architecture for modeling time series data and has demonstrated effective in tasks such as time series forecasting and sequence classification. This research proposes an improved TCN model consisting of two sequential TCN blocks, as indicated in **Figure 6.** The architecture has an Encoder-TCN and a Decoder-TCN arranged

in sequence. The Encoder compresses a past window of length T into a context vector; the Decoder receives the context and simultaneously generates H future steps in one inference (non-autoregressive), thereby reducing the accumulation of errors over the horizon and shortening the inference time. The input data are initially processed by the first TCN layer, and its output is then duplicated and passed to the second TCN layer. Each TCN layer utilizes 80 convolution filters with a kernel size of 3, and dilation rates of [1,2,4,8,16,32]. The final connected layer operates across all temporal positions. With increasing dilation rates, the proposed hierarchical structure enables the capture of both short-term and long-term dependencies within the time series data. Finally, a fully connected layer is applied to generate the final output.

The core component of this model is the TCN block, which represents a specialized Temporal Convolutional Network (TCN) architecture. TCNs effectively address time series data by processing through the integration convolutional layers with mechanisms such as causal convolutions and residual connections.

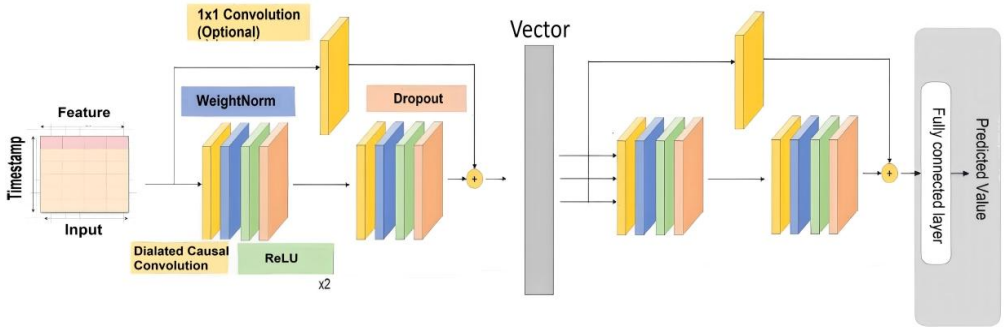



**Figure 6.** Proposed enhanced TCN model with 2 TCN blocks.

TCN blocks adhere to two fundamental principles Bai et al. (2018): 1) *size invariance*: the TCN block's input and output dimensions must remain identical to ensure consistent data flow throughout the network; and 2) *causality*: the model must strictly adhere to the temporal order of the data. In other words, the prediction at any given time step must not be influenced by information from the future. This principle is essential for maintaining the integrity of time series analysis. The convolutional layers within the TCN are strategically padded to fulfill the two fundamental principles of *size invariance* and *causality.* To maintain equivalence between the input and output sequence lengths, (*kernel-size* - 1) zeros are added to the beginning of the input sequence. In addition, by placing the padding exclusively on the left side of the input sequence, the model is restricted to using only past information for its predictions. A TCN can be described as a fully convolutional network (FCN) that incorporates the principle of causality through left-sided padding.

However, standard causal convolutions are limited by network depth, where each neuron can only perceive a restricted number of past elements. This creates a bottleneck for tasks requiring extended historical information. Dilated convolutions are employed to overcome the limitation of causal convolutions.

Using a parameter named "dilation factor" allows the network's receptive field to grow exponentially by systematically increasing the dilation factor in successive layers. Therefore, the model effectively captures information from a much wider range of input elements, enabling it to learn complex long-term dependencies within the time series data. Given 1-D input sequence $x$ and a filter $f$, the dilated convolution at index $s$ is computed as a weighted sum of the filter elements $k$ and the input values located at positions determined by the dilation factor $d$:

$$F(s) = \sum_{i=1}^{k-1} f(i) \cdot X_{s-d \cdot i} \quad (5)$$

The receptive field within a TCN block can be effectively expanded using two distinct strategies. The first one employs larger filter sizes ($k$), so the TCN gets more local information within each convolutional operation, directly capturing a more significant number of consecutive elements within the input sequence. The second one introduces the dilation factor ($d$). An increase in dilation factor $d$ results in an exponential expansion of the TCN's receptive field. It enables the model to capture long-range dependencies within the time series data, allowing the model to consider information from distant past time steps.

The incorporation of dilated convolutions within the TCN architecture provides several significant advantages: By effectively empowering the model to learn information from distant points within the input sequence and adjusting the dilation factors across different layers, the TCN can selectively focus on the most relevant time steps while effectively ignoring less important information.

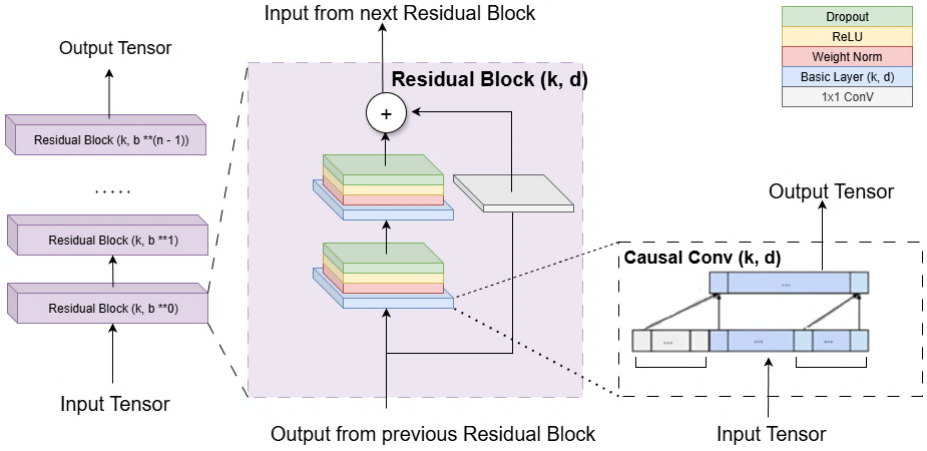



**Figure 7.** TCN architecture input length l, kernel size $k$, dilation base $b$ ($k \geq b$), and the minimum number of residual blocks $n$ required for full history coverage.

The concept of the residual block originated from the groundbreaking ResNet architecture (He et al., 2015). In ResNet, residual blocks mitigate the vanishing gradient problem, a common challenge in deep neural networks. Characteristic of a residual block (He et al., 2015) is the inclusion of a shortcut connection. This shortcut path directly connects the block's input ($x$) to its output. In parallel, a separate branch applies a series of transformations ($F$) to the input, and the output of this transformation branch is then added to the original input. Through its additive structure, the network learns residual functions, effectively capturing the deviation between the input and the target output.

$$\sigma = activation\ (x + F(x)) \quad (6)$$

Each residual block within the proposed architecture comprises two sequential layers. Each layer is structured with a dilated causal convolution and a rectified linear unit (ReLU) activation function to incorporate non-linearity. Weight normalization (Salimans & Kingma, 2016) is applied to the convolutional filters to enhance model stability. In addition, spatial dropout is also applied after each dilated convolution layer to prevent overfitting and improve model robustness.

**Figure 7** indicates the overall architecture of the TCN model, in which l represents the input length; $k$ denotes the kernel size of the convolutional filters must be greater than or equal to the base dilation factor $b$.

## 5. Experiment Results

This research utilizes a real dataset from a desalination plant, comprising data from 14 operational trains. As detailed in Section 4.1, the data underwent proposed rule-based techniques.

A split-sample approach was adopted for model training and evaluation: 80% of the data was utilized as a training and validation set (60% train and 20% validation), and the remaining 20% was reserved as a dedicated testing set. That ratio ensures enough training data to stabilize parameter determination, still covers all system maintenance actions, while remaining a sufficiently long validation/test period, thus allowing for a more comprehensive evaluation.

The model was trained for 400 epochs with early stopping to prevent overfitting. A batch size of 64 was used, and the Adam optimizer (learning rate = 0.001) was employed. Non-linear behavior was achieved through the ReLU activation.

The performance assessment of the proposed enhanced TCN model compared to established deep learning architectures in forecasting pressure fluctuations in RO systems, LSTM, and GRU. All models were trained and evaluated on the same pre-processed dataset, and hyperparameter tuning identified optimal values for each model, which were 80 units, a 0.1 dropout rate, and a kernel size of 3 for the convolutional layers. In addition, we use three metrics to assess how good the system is: Root Mean Squared Error (*RMSE*) quantifies the overall magnitude of prediction errors, Mean Absolute Percentage Error (*MAPE*) evaluates the average percentage difference between the estimated and true values, and R-squared ($R^2$) denotes the fraction of variance in the target accounted for by the model. For each metric, the individual values were calculated for each of the 14 trains, and the final performance was determined by averaging these individual values across all trains.

In this paper, three experimental groups are presented. The first experiment investigates the influence of window size $t$ on the prediction accuracy of the target variable. Notably, smoothing techniques, such as the Savitzky-Golay (S-G) filter (Rooij, 2022) and Simple Moving Average (SMA) filter, were not applied in this phase. As shown in **Table 2**, a window size of $t$ = 120 yields relatively stable performance, achieving *RMSE* = 0.161 and $R^2$ = 0.879.

**Table 2.** Impact of input sequence length on predictive performance.

| Timestamp | 90 | 100 | 110 | 120 | 130 | 140 |
|---|---|---|---|---|---|---|
| *RMSE* | 0.257 | 0.229 | 0.198 | **0.161** | 0.173 | 0.172 |
| $R^2$ | 0.707 | 0.742 | 0.801 | **0.879** | 0.872 | 0.872 |

The second experiment focused on assessing the proposed model's performance using different smoothing techniques to evaluate the impact of data smoothing and noise reduction. We performed a hyperparameter sensitivity analysis for two filters: the Savitzky-Golay (S-G) with window sizes $w$ {90, 120, 150, 180}, and the Simple Moving Average (SMA) ∈ {5, 10, 20, 30}.

Targeting a 14-day forecast horizon, the results depicted in **Figure 8** represent the aggregated mean performance across all 14 RO trains within the dataset. Furthermore, the proposed model's performance was

compared against other deep learning models, including Long-Short-Term Memory (LSTM), Convolutional Neural Network-Long-Short-Term Memory (CNN-LSTM), and Gated Recurrent Unit (GRU) models.

Across the 14 RO trains, the three models—LSTM, GRU, and CNN-LSTM—show fairly consistent rankings and errors across both smoothing methods. CNN-LSTM tends to provide slightly lower *RMSE* and higher $R^2$ than regular LSTM and GRU on RO Trains, suggesting that shallow convolutional feature extraction prior to the recurrent layer can stabilize the sequence model. GRUs have not shown a consistent advantage over LSTM and CNN–LSTM. All three baselines benefit from smoothing: Savitzky–Golay and Moving Average reduce residual noise and tighten dispersion across sequences.

**Figure 8** demonstrates the significant effect of data smoothing filters on model prediction accuracy. The choice of window size ($w$) for smoothing methods affects the performance: a small window size may not effectively mitigate noise and fluctuations, while a large window size may result in the loss of important information within the data.

Both the SMA and Savitzky–Golay (S-G) methods help reduce noise and narrow the dispersion between time series, especially at short to medium windows

The two filters have different characteristics and therefore different effects on the forecast. MA is a simple, stable filter with few parameters: when the window is chosen moderately (e.g., $w = 10$), it effectively removes high-frequency noise while preserving the slow variation of the series, thereby reducing the *RMSE* and increasing $R^2$ compared to $w$ that is too small (not enough denoising) or too large (signal flattening). In contrast, S-G fits local polynomials to preserve the shape/derivative, which is useful when preserving peaks and edges. However, S-G is sensitive to the choice of windows and polynomial degree: small windows allow noise to enter, while large windows lead to over-smoothing, which smooths out predictive fluctuations (e.g., pre-CIP drifts), thereby increasing the error. The results in **Figure 8** show that SMA (especially $w = 10$) achieves a good balance between noise removal and dynamics preservation. At the same time, the current S-G configuration performs slightly worse due to the strong smoothing.

The results show that an SMA filter with a window size of 10 improved performance to around 6% compared to the size of 5. This filter achieves robust smoothing, preserves key information and reduces the sensitivity to outliers. The experimental results also indicated that the S-G filter exhibited suboptimal performance in its current configuration due to over smoothing of the data.

A third group of experiments was conducted to investigate further the impact of smoothing techniques on individual RO trains. **Figure 10** presents the detailed results. Overall, the SMA filter demonstrated superior performance across most RO trains. However, the results exhibited some variability across the 14 trains. Train 9 displayed less favorable performance, with an RMSE of 0.669, a MAPE of 18.331, and an $R^2$ of 0.331. These results suggest that the optimal smoothing parameters may vary across different trains within the desalination plant.

Experimental results revealed that the proposed TCN-based approach achieved better performance relative to LSTM, CNN-LSTM, and GRU models.

We conducted an experiment to investigate the impact of error accumulation on multi-step predictions. The results of the model are visualized in **Figure 9.** Initially, data partitioning was performed such that 60% of the RO dataset was used for training, while validation and testing sets each comprised 20%.

Additionally, the entire time series for each RO train is forecasted using the first 120 timesteps, and the results show that there is a significant error accumulation over long periods. Based on these findings, we recommended prioritizing short-term forecasts, with the forecast horizon ideally aligned with the model's typical output range.

A negative R-squared value for the GRU model, as observed in **Figure 8**, indicates that the its predictive performance is less accurate than those obtained by simply using the target variable's mean as a baseline. This indicates that the model may require further refinement, or adjust in data handling process. When a regression model exhibits a negative R-squared value, it implies that its predictions deviate further from the actual values than the simple average of the target variable. This occurs when the sum of squared prediction errors exceeds the sum of squared deviations of the actual values from the target variable's mean.

$$R^2 < 0 if \sum(y_i - \hat{y}_i)^2 > \sum(y_i - \overline{y_i})^2 \quad (7)$$

With the preamble consisting of historical NDP, train state (train state, one-hot encoding), and membrane position vectors $S1$–$S8$, we develop the following scenarios to evaluate and illustrate the industrial usefulness of the model:

- Adjust the CIP schedule $\pm k$ days early/late and observe the performance on the forecast trajectory.
- Partial or complete replacement: switch on/off possible positions across $S1$–$S8$ (partial replacement) to replace all (complete).
- Track the NDP variation by changing the vector position or training state at a given time and predict the regeneration.

For each scenario, we report the metrics evaluation and warning index. The results provided to the model directly support the planning of continuous-day scheduling, while maintaining the reliability needed for actual production deployment (see **Figure 9**).

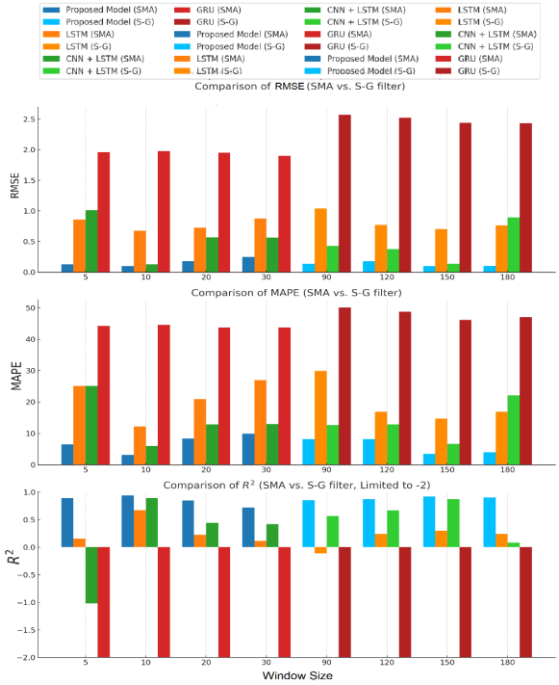



**Figure 8.** Evaluation of prediction methods with alternative smoothing strategies.

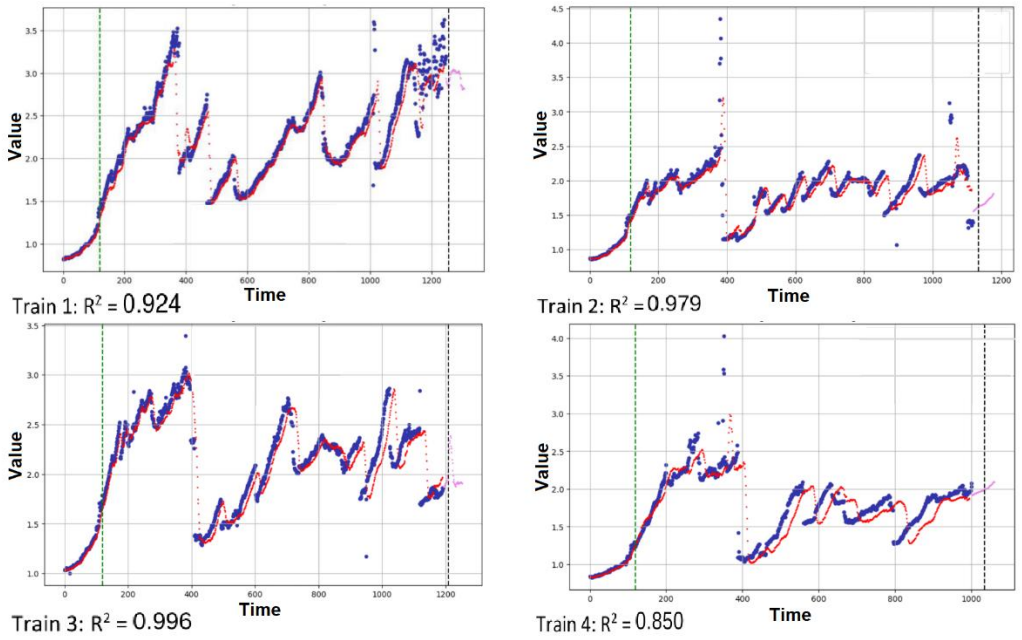
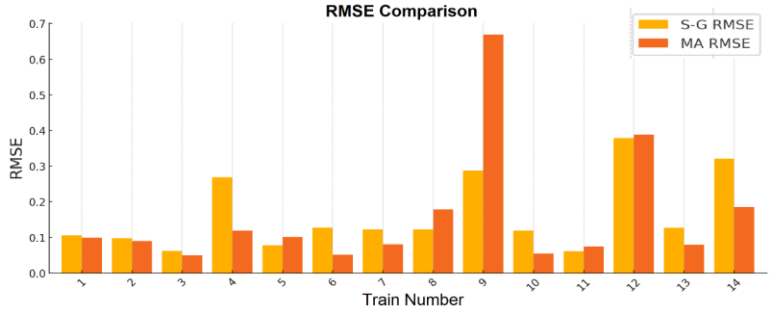
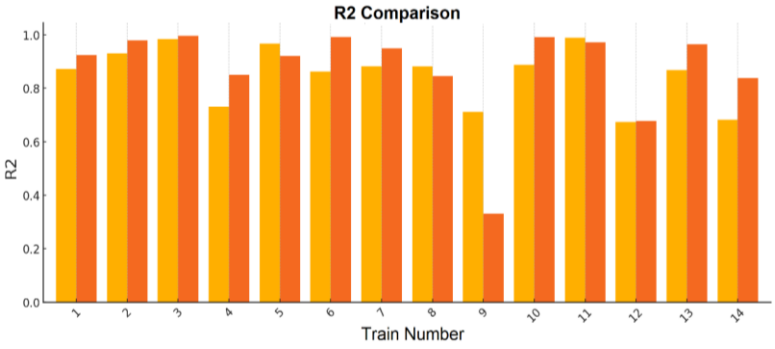



**Figure 9.** Predicting results of the proposed model after applying a simple moving average (window size = 10); blue line denotes the actual values; red line denotes predicted values.




(a) Comparison of *RMSE* values between different RO trains.




b) Comparison of $R^2$ values between different train filters.

**Figure 10.** Model performance on RO datasets (>1300 days) with S-G smoothing ($w$ = 150, $k$ = 4) and SMA techniques ($w$ = 10).

## 6. Conclusion

This research gained a deep understanding of the impact of the algal bloom phenomena on seawater-on-seawater desalination membranes using the RO system. This knowledge is the basis for us to continue research and propose a new model based on a TCN network to forecast fluctuations in differential pressure in RO systems. In addition, to enhance data quality and reveal underlying trends, the collected data is smoothed using Savitzky-Golay and Moving Average filters. The proposed model, trained on a comprehensive dataset encompassing 14 RO trains from an industrial desalination plant (over 16,000 data points collected between 2015 and 2020), demonstrated promising predictive capabilities.

While the current model effectively predicts future pressure differentials, it does not fully understand the decisions made; it only learns from the data. Thus, in the future, we will develop a model based on Physics-Informed Machine Learning (PIML) for RO degradation prediction. This method helps train models based on pre-existing physical laws, unlike standard neural networks trained based on minimizing errors between model output and target output. The PIML model will predict membrane degradation specifically for applications in predictive maintenance decision-making. This model will incorporate physical knowledge to make more accurate predictions.

### *Appendix A. Long Short-Term Memory - LSTM*

Long short-term memory (LSTM) (Graves, 2012 and Hochreiter & Schmidhuber, 1997) is recurrent neural network (RNN) (Sherstinsky, 2018) used to deal with the vanishing gradient problem. LSTM is an advantage over other RNN-based models by providing long-term memory capabilities that often last thousands of timesteps. It is applied for tasks involving sequence data, including classification, processing, and prediction data in domains such as handwriting recognition, speech recognition, machine translation, and speech activity detection.

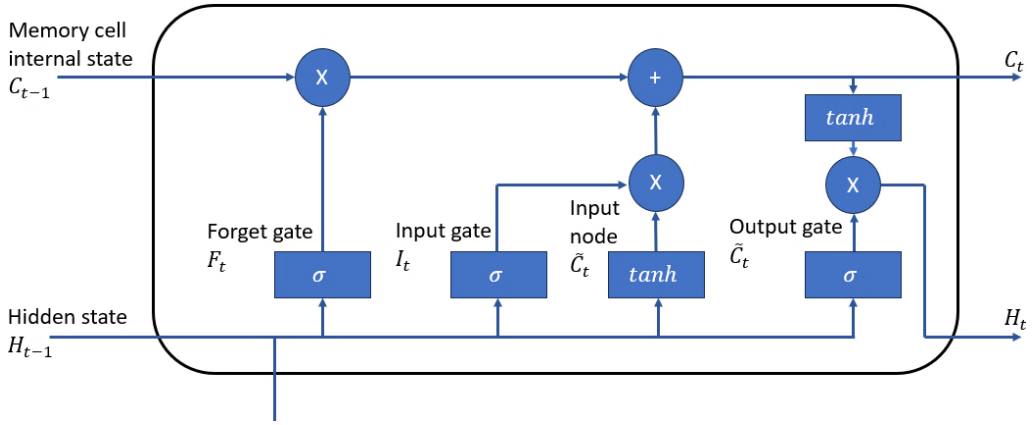



**Figure A1. Illustration of LSTM architecture**, information is stored in the cell state, enabling long-term memory and hidden stage operations that address the vanishing gradient problem, unlike standard RNNs.

The cell state within an LSTM network (see **Figure A1)** can be conceptualized as selectively retaining and integrating information from previous time stepson from previous time steps selectively. The key LSTM combines partial "forgetting" and "increment" operations in cell states to preserve relevant information from earlier time steps while discarding less important details. Specifically, the LSTM operation follows three stages: forget gate, update cell state, and output state.

***Forget gate*:** determines the proportion of information retained from the previous cell state. Using the sigmoid activation function $\sigma$ (range [0,1]), the forget gate is calculated as in Equation (A1).

$$f_t = \sigma(W_{xf} X_t + W_{hf} h_{t-1} + b_f) \quad \text{(A1)}$$

***Update cell state*:** combines forget gate output and current input (see Equation (A2)):

$$i_t = \sigma(W_{xi} X_t + W_{hi} h_{t-1} + b_i),$$
$$g_t = tanh(W_{xg} X_t + W_{hg} h_{t-1} + b_g) \quad \text{(A2)}$$
$$c_t = f_t \odot C_{t-1} + i_t \odot g_t$$

***Output state*:** computes the hidden state value, or the final prediction result if the cell is the last in the sequence (see Equation (A3)):

$$o_t = \sigma(W_{xo} X_t + W_{ho} h_{t-1} + b_o),$$
$$h_t = o_t \odot tanh(c_t). \quad \text{(A3)}$$

## *Appendix B. Gated Recurrent Unit - GRU*

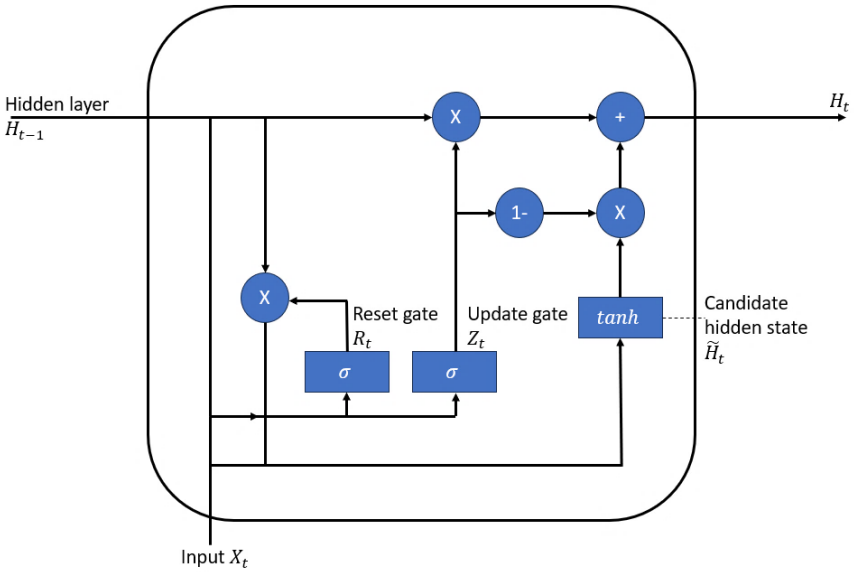



**Figure B1. Gated recurrent unit**, the main difference between a standard RNN and a GRU is that the GRU supports hidden state control. This means that it has learned mechanisms to decide when to update and when to reset the hidden state.

***Gated recurrent unit*:** GRU has the reset and update gates (see **Figure B1)**, where the reset gate (B1) determines how much past information is maintained.

$$r_t = \sigma(W_{xr} X_t + W_{hr} h_{t-1} + b_r) \quad \text{(B1)}$$

Similarly, the update gate (B2) controls how much of the new state incorporates the previous state, based on the current input $x_t$ and the prior hidden state $h_{t-1}$.

$$Z_t = \sigma(W_{xz} X_t + W_{hz} h_{t-1} + b_z) \quad \text{(B2)}$$

***Operation of reset gate*:** Similar to RNNs (Sherstinsky, 2018), GRUs maintain a hidden state and a candidate hidden state $h$ (see Equation (B3))

$$h_t = tanh(W_{xh} X_t + W_{hh} (r_t \odot h_{t-1}) + b_h) \quad \text{(B3)}$$

The information from $h_{t-1}$ passes through the reset gate $r_t$, which modulates the retained information. When $r_t$ is close to 1, the result is similar to an RNN: $h_t = tanh(W_{xh} \bullet X_t + h_{t-1} \bullet W_{hh} + b_h)$. Conversely, when $r_t$ is close to 0, the hidden state reflects a transformation of $X_t$ via a feedforward network.

***Operation of update gate*:** determines the similarity between the new state $H_t$ and the previous state $H_{t-1}$, as well as the contribution of the potential hidden state $H_t$ (see Equation (B4))

$$h_t = (1 - Z_t) \odot h_t + Z_t \odot h_{t-1} \quad \text{(B4)}$$

Computation of $h_t$ is determined by update gate ($z_t$) and the candidate state ($h_t$). When $Z_t$ is close to 1, the previous state is retained, effectively ignoring $x_t$ and skipping time step $t$. Conversely, when $z_t$ is close to 0, $h_t$ closely approximates the candidate state. These designs address the vanishing gradient problem and enhance modeling of long-term dependencies.

### Conflicts of Interest

The authors confirm that there is no conflict of interest to declare for this publication.

### Acknowledgments

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors. The authors would like to thank the editor and anonymous reviewers for their comments that help improve the quality of this work. Author Contributions- Conceptualization: DO-Phuc; methodology: Thanh-Ha DO and The-Son PHAN; val¬idation: The-Son PHAN; formal analysis: Phuc-DO, The-Son PHAN, Thanh-Ha DO; investigation: Thanh-Ha DO, The-Son PHAN; data curation: Phuc-DO; writing— original draft preparation: Thanh-Ha DO and The-Son PHAN; writing—review and editing: Thanh-Ha DO and Phuc-DO; visualization: The-Son PHAN; supervision: Phuc-DO; review and revision, project administration: Phuc-DO and Thanh-Ha DO. All authors have read and agreed to the published version of the manuscript "Degrada¬tion Forecasting in Reverse Osmosis Systems using Enhanced Temporal Convolutional Network". Data Availability Statement- The datasets analyzed during the current study are available from the corresponding author and her colleague upon reasonable request.

### AI Disclosure

During the preparation of this work the authors used generative AI in order to improve the language of the article. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

## References

Ahmad, S., Styp-Rekowski, K., Nedelkoski, S., & Kao, O. (2020). Autoencoder-based condition monitoring and anomaly detection method for rotating machines. In *2020 IEEE International Conference on Big Data* (pp. 4093-4102). IEEE. Atlanta, GA, USA. https://doi.org/10.1109/BigData50022.2020.9378015.

Alsawaftah, N., Abuwatfa, W., Darwish, N., & Husseini, G.A. (2022). A review on membrane biofouling: prediction, characterization, and mitigation. *Membranes, 12*(12), 1271. https://doi.org/10.3390/membranes12121271.

Bai, S., Kolter, J.Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *Computation and Language.* https://doi.org/10.48550/arXiv.1803.01271.

Bonny, T., Kashkash, M., & Ahmed, F. (2022). An efficient deep reinforcement machine learning-based control reverse osmosis system for water desalination. *Desalination, 522,* 115443. https://doi.org/10.1016/j.desal.2021.115443.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: a survey. *ACM Computing Surveys, 41*(3), 1-58. https://doi.org/10.1145/1541880.1541882.

Gallagher, N.B. (2020). Savitzky-golay smoothing and differentiation filter. *Eigenvector Research Incorporated*. https://doi.org/10.13140/RG.2.2.20339.50725.

Graves, A. (2012). Long short-term memory. In *Supervised Sequence Labelling with Recurrent Neural Networks.* Springer, Berlin, Heidelberg, pp. 37-45. https://doi.org/10.1007/978-3-642-24797-2.

Guo, J., Li, Z., & Li, M. (2020). A review on prognostics methods for engineering systems. *IEEE Transactions on Reliability, 69*(3), 1110-1129. https://doi.org/10.1109/TR.2019.2957965.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *Computer Vision and Pattern Recognition.* http://arxiv.org/abs/1512.03385.

Hewage, P., Behera, A., Trovati, M., Pereira, E., Ghahremani, M., Palmieri, F., & Liu, Y. (2020). Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station. *Soft Computing, 24*(21), 16453-16482. https://doi.org/10.1007/s00500-020-04954-0.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Hwang, T.M., Choi, Y., Nam, S.H., Lee, S., Oh, H., Hyun, K., & Choung, Y.K. (2010). Prediction of membrane fouling rate by neural network modeling. *Desalination and Water Treatment, 15*(1-3), 134-140. https://doi.org/10.5004/dwt.2010.1677.

Inoue, J., Yamagata, Y., Chen, Y., Poskitt, C.M., & Sun, J. (2017). Anomaly detection for a water treatment system using unsupervised machine learning. *In 2017 IEEE International Conference on Data Mining Workshops* (pp. 1058-1065). IEEE. New Orleans, LA, USA. https://doi.org/10.1109/ICDMW.2017.149.

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D.J. (2019). 1D convolutional neural networks and applications: a survey. *Signal Prosessing.* https://doi.org/10.48550/arXiv.1905.03554.

Koutsakos, E., & Moxey, D. (2007). Membrane management system. *Desalination, 203*(1-3), 307-311. https://doi.org/10.1016/j.desal.2006.02.023.

Lea, C., Flynn, M.D., Vidal, R., Reiter, A., & Hager, G.D. (2016). Temporal convolutional networks for action segmentation and detection. *Computer Vision and Pattern Recognition.* http://arxiv.org/abs/1611.05267.

Lim, S.J., Kim, Y.M., Park, H., Ki, S., Jeong, K., Seo, J., Chae, S.H., & Kim, J.H. (2019). Enhancing accuracy of membrane fouling prediction using hybrid machine learning models. *Desalination and Water Treatment, 146,* 22-28. https://doi.org/10.5004/dwt.2019.23444.

Liu, Y., Wu, H., Wang, J., & Long, M. (2022). Non-stationary transformers: exploring the stationarity in time series forecasting. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds) *Advances in Neural Information Processing Systems* (Vol. 35, pp. 9881-9893). Curran Associates, Inc. https://doi.org/10.48550/arXiv.2205.14415.

Masum, S., Liu, Y., & Chiverton, J. (2018). Multi-step time series forecasting of electric load using machine learning models. In: Rutkowski, L., Scherer, R., Korytkowski, M., Pedrycz, W., Tadeusiewicz, R., Zurada, J.M. (eds) *Artificial Intelligence and Soft Computing* (pp. 148-159). Springer International Publishing, Cham. https://doi.org/10.1007/978-3-319-91253-0_15.

Niu, C., Li, X., Dai, R., & Wang, Z. (2022). Artificial intelligence-incorporated membrane fouling prediction for membrane-based processes in the past 20 years: a critical review. *Water Research, 216,* 118299. https://doi.org/10.1016/j.watres.2022.118299.

Rooij, F.V. (2022). *Managing the restoration of membranes in reverse osmosis desalination using a digital twin*. The University of Salford. [Doctoral dissertation]. https://salford-repository.worktribe.com/output/1322881/managing-the-restoration-of-membranes-in-reverse-osmosis-desalination-using-a-digital-twin.

Rooij, F.V., Scarf, P., & Do, P. (2021). Planning the restoration of membranes in RO desalination using a digital twin. *Desalination, 519,* 115214. https://doi.org/10.1016/j.desal.2021.115214.

Salimans, T., & Kingma, D.P. (2016). Weight normalization: a simple reparameterization to accelerate training of deep neural networks. *Neural and Evolutionary Computing.* http://arxiv.org/abs/1602.07868.

Savitzky, A., & Golay, M.J.E. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry, 36*(8), 1627-1639. https://doi.org/10.1021/ac60214a047.

Sherstinsky, A. (2018). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Machine Learning.* http://arxiv.org/abs/1808.03314.

Shi, Y., Wang, Z., Du, X., Gong, B., Lu, Y., & Li, L. (2022). Membrane fouling diagnosis of membrane components based on multi-feature information fusion. *Journal of Membrane Science, 657,* 120670. https://doi.org/10.1016/j.memsci.2022.120670.

Shim, J., Park, S., & Cho, K.H. (2021). Deep learning model for simulating influence of natural organic matter in nanofiltration. *Water Research, 197,* 117070. https://doi.org/10.1016/j.watres.2021.117070.

Soleimanzade, M.A., Kumar, A., & Sadrzadeh, M. (2022). Novel data-driven energy management of a hybrid photovoltaic-reverse osmosis desalination system using deep reinforcement learning. *Applied Energy, 317,* 119184. https://doi.org/10.1016/j.apenergy.2022.119184.

Staudemeyer, R.C., & Morris, E.R. (2019). Understanding LSTM - a tutorial into long short-term memory recurrent neural networks, *Neural and Evolutionary Computing.* http://arxiv.org/abs/1909.09586.

Villacorte, L.O., Ekowati, Y., Calix-Ponce, H.N., Kisielius, V., Kleijn, J.M., Vrouwenvelder, J.S., Schippers, J.C., & Kennedy, M.D. (2017). Biofouling in capillary and spiral wound membranes facilitated by marine algal bloom. *Desalination, 424,* 74-84. https://doi.org/10.1016/j.desal.2017.09.035.

Wreyford, J.M., Dykstra, J.E., Wetser, K., Bruning, H., & Rijnaarts, H.H.M. (2020). Modelling framework for desalination treatment train comparison applied to brackish water sources. *Desalination, 494,* 114632. https://doi.org/10.1016/j.desal.2020.114632.

Zhou, C., & Paffenroth, R.C. (2017). Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 665-674). Association for Computing Machinery. New York, USA. https://doi.org/10.1145/3097983.3098052.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2021). Graph neural networks: a review of methods and applications. *Machine Learning*. http://arxiv.org/abs/1812.08434.