Application of Deep Learning in Process Fault Diagnosis, Modelling and Optimization

by

Saman Dehghanbanadaki

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Control Systems

Department of Electrical and Computer Engineering
University of Alberta

© Saman Dehghanbanadaki, 2022

Abstract

Artificial Neural Network (ANN) has gained great interest in industrial applications due to their supremacy in modelling complex process behaviour. Applications of ANN include process modelling, optimization and fault diagnosis. However, pure data-driven approaches that use only observations to approximate the system dynamics may not be ideal candidates, especially when substantial physics-based knowledge is available but not utilized. This study refers to such knowledge as Domain Knowledge (DK).

This M.Sc. research is focused on fault diagnosis, modelling and optimization of industrial plants using ANN techniques. This thesis is mainly divided into two parts. First, Chapter 3 outlines an empirical study of the Long Short-Term Memory (LSTM) network, one of the most well-known Recurrent Neural Network (RNN) algorithms. During this study, we can further understand how the LSTM algorithm works to capture temporal features.Then,we propose adding an Autoregressive Integrated Moving Average with eXogenous input (ARIMAX) on top of the LSTM network to handle faults related to the closed-loop system. The proposed framework is applied to the Wind Turbine System (WTS) to detect blade and pitch system faults. The second part mainly focuses on incorporating DK into the neural network structure. Hence, Chapter 4 describes the Physics-Informed Neural Network (PINN) framework based on Ensemble Sequential Learning and the Mixture Density Network (ESL-MDN). The proposed model estimates the components that build the cost function and constraints of the Differential Evolution (DE) optimizer. Finally, the proposed method has been validated by data collected from Reverse Osmosis Water Desalina-

tion (ROWD) plant to demonstrate a reduction in energy consumption, achieved by the optimization strategy under different scenarios.

Preface

This thesis is an original work by Saman Dehghanbanadaki. No part of this thesis has been previously published.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor, Professor Qing Zhao, for her guidance and support during my graduate studies.

I have been so fortunate for making nice friends during this journey. I especially want to thank my dearest friends Anahita Shervin, and Farzaneh for their friendship, support and encouragement. Last but not least, my deepest love and gratitude goes to my parents and sister for their unconditional love, support and patience in all my lifetime.

Table of Contents

List of Tables

List of Figures

Abbreviations

AE Autoencoder.

ANN Artificial Neural Network.

ARIMAX Autoregressive Integrated Moving Average with eXogenous input.

BNN Bayesian Neural Network.

CGAN Convolutional Generative Adversarial Network.

CNN Convolutional Neural Network.

CSCoh Cyclic Spectral Coherence.

DBN Deep Belief Network.

DK Domain Knowledge.

DNN Deep Neural Network.

ESL-MDN Ensemble Sequential Learning and Mixture Density Network.

FDS Fault Detection System.

FFT Fast Fourier Transform.

GA Genetic Algorithm.

GAN Generative Adversarial Network.

IEMS Intelligent Energy Management System.

IIL Intensified Iterative Learning.

MDN Mixture Density Network.

MLAN Multi-Layer Artificial Neural Network.

MPC Model Predictive Controller.

PINN Physics-Informed Neural Network.

PMU Phasor Measurement Unit.

ReLU Rectified Linear Unit.

RNN Recurrent Neural Network.

ROWD Reverse Osmosis Water Desalination.

SAE Stacked Autoencoder.

Chapter 1

Introduction

The pattern in data could be automatically recognized by deep learning applications that far surpass human ability. Deep learning has provided a solution to the limitation of traditional machine learning algorithms in the past few years. As a result, it has attracted the attention of businesses and researchers who have all been eager to apply and take advantage of it. This thesis explores fault diagnosis, modelling and optimal operation of nonlinear industry process using several deep neural network models. In this chapter, we provide an overview of the literature, define the research motivations, and summarize the main contributions.

1.1 Literature Review

It may be difficult for highly trained professionals to analyze and determine where to begin and effectively implement a deep learning model to deal with their problems. In this thesis we have divided deep learning applications in industrial processes into three significant categories: fault diagnosis, process modelling, and optimization.

- Fault Diagnosis

  An essential component of health management to prevent system failure is fault diagnosis. The deep learning architecture makes it possible to automatically extract a hierarchical representation of the data, which is then used to learn complex features from simpler ones by utilizing the next stacked layers. Fur-

thermore, this approach can be applied to produce a system which, based on raw inputs, automatically identifies features and processes them accordingly. Autoencoder (AE) is one of the most popular techniques for automatic feature extraction, and it has been successfully applied for fault diagnosis. A study published in [1] used frequency spectrum data as input for automatically extracting features that were used for bearing fault detection. [2] transformed raw time-domain data into frequency-domain data using Fast Fourier Transform (FFT). The authors suggested using AE with Rectified Linear Unit (ReLu) activation function and dropout layer (to prevent overfitting) for automatic features extraction and using it for hydraulic pump fault diagnosis. Intensified Iterative Learning (IIL) based on Stacked Autoencoder (SAE) was designed in [3] to intensify the impact of features with the most favourable information in the hidden unit and accomplish a better fault detection for industrial processes. Deep Belief Networks (DBNs) are generative models that learn to reconstruct their inputs probabilistically. In [4] a gearbox fault diagnosis framework was introduced based on DBN structure fed by time-domain and frequency-domain features extracted from load and speed measurements. A Fault Detection System (FDS) for chemical process was designed in [5] with extended DBN to dynamically classify faults using dynamic characteristic of the process. Experts can benefit from the design of feature extraction techniques which are often time and resource consuming, and they produce inconsistent results. In recent years, Convolutional Neural Networks (CNN) techniques have gained much attention, mainly when processing high-dimensional data types like images and time-series. The authors in [6] used Cyclic Spectral Coherence (CSCoh) to map vibration signals into bearing discriminative patterns and used the CNN to classify bearing failures. To investigate the rotary machine malfunctions, the researchers in [7] suggested a sensor fusion algorithm utilizing a 2-D CNN. Recurrent Neural Networks (RNNs) are suited to dealing with sequential data,

making them an ideal candidate when dealing with health management systems due to their time-series nature. The authors of [8] proposed a feature engineering model that incorporates the Continuous Wavelet Transform (CWT), along with the FFT. It was designed to classify rotary machine faults by using the Convolutional LSTM (CLSTM) model. In [9] a supervised fault FDS for the hydraulic system was designed combining LSTM with AE to improve fault classification accuracy. Complete transient data from the pre and post-fault cycle of the Phasor Measurement Unit (PMU) was analyzed in [10] by LSTM based FDS framework that identifies the fault region, classifies the type of fault, and predicts the location of the fault.

- Process Modelling

  For decades researchers have studied first-principle models for industrial processes and used them in control, optimization and other decision-making problems [11]. However, first-principle models are hardly maintainable due to parameter variations. Also, complex applications with sophisticated physical phenomena makes it almost impossible for engineers to establish an explicit mathematical description [12]. Meanwhile, the massive amount of data collected by SCADA systems have let to increased popularity of data-driven approaches, e.g. ANN based, in industries. However, compared to data-centric modeling, physics-based model help reveal inter-relationships among key outputs and intermediate process variables that are not measured. By incorporating these relational information in ANNs, prediction of outputs can be improved in terms of accuracy.

  Lately we have witnessed great advances on DNNs, and their fast adaptation in various areas of research and applications, among which an important problem is process modelling via deep learning. Fitting the complex kinetic data was investigated in [13] using a deep neural network, which was evaluated by applying to

extensive experimental data and compared with several traditional approaches. [14] developed a Conditional Generative Adversarial Network (CGAN) to estimate the fluid flow and heat condition in a purely data-driven approach without relying on the knowledge coming from the model. In [15], process modelling based on DNN was developed to help understand plant-wide processes. The authors suggested a systematic framework containing DK-based data processing and global sensitivity analysis with Monte-Carlo simulations to predict nitrous oxide emission characteristics for wastewater treatment plants. An Intelligent Energy Management System (IEMS) was investigated in [16] for a hybrid grid-connected Reverse Osmosis (RO) desalination process with Photo-Voltaic (PV) and energy storage system. Five-hours-ahead power forecasting was conducted by using CNN and LSTM networks. Combination of Multi-Layer Artificial Neural Network (MLANN) and Genetic Algorithm (GA) has been studied in [17] to find a comprehensive model of RO process and to remove chlorophenol from wastewater.

Although DNN has demonstrated high modelling performance, it is considered a black box model and the interpretability of the created model is a challenging task. There are numerous ways to combine a physical model with DNN to accurately regenerate the behaviour of a complex model at significantly lower computation cost. [18] infused the physics into the RNN model to leverage the complementary knowledge to predict the lake temperature profile. In this work, traditional LSTM was augmented by physical law of thermodynamics to take energy conservation into account. Down-sampling technique was used in [19] to replace the dynamic system with DNN as a surrogate model to map the input-output relationship with higher resolution and speed. However, the main challenge is to ensure that the surrogate model is consistent with the first principle model. Specific complex dynamic process in [20] was replaced

by simpler mathematical equations whose related coefficients were estimated by a neural network. It was indicated that the suggested model is robust, and generalization was increased even for unseen scenarios.

In modelling a system/process, considering the uncertainty is of great concern in many applications, specifically those related to the decision-making problems. The main effort is to draw a distribution that captures all quantiles which are necessary for further research. [21, 22] conducted a model reduction and proposed an DNN surrogate model to quantify the uncertainty in a physics-based model. Also, the Gaussian approach was studied in [23] to consider uncertainty in the simulated process. However, both techniques are not applicable for large data set as the computation cost increases massively. Although DNN models have demonstrated immense successes, some modifications need to be performed to quantify the uncertainty. [24] employed a probabilistic dropout framework to deactivate neurons partially and estimate uncertainty. [25] represented Bayesian Neural Network (BNN) which estimated the distribution of all weights and biases. The major challenge is that real-world processes do not necessarily follow any parametric distribution. The Mixture Density Network (MDN) is a conceivable method that can efficiently address uncertainty estimation. MDN is similar to conventional ANN with only one practical difference: the outcome of the MDN is the parameters related to the distribution instead of point prediction. [26] developed an ensemble MDN applied to the data collected from a wind farm in Taiwan to forecast the power and wind speed. uncertainty estimation was investigated in [27] based on MDN to analyze the wind turbine power. [28] studied the use of MDN to analyze the uncertainty in multi-components and nonlinear control problems. An ensemble ANN-MDN was designed in [29] to simulate the prediction uncertainty as a result of variation of the training data, and it was embedded in the mixed-integer linear optimization problem.

It should be noted that, in most cases, the uncertainty induced by DK infusion has not been investigated.

- Optimization

  An optimization is a vital tool for industrial process management. However, in all control and optimization applications, the performance highly depends on the modelling accuracy. Therefore, DNN techniques have been employed to model the nonlinear process and integrated with MPC [30], and RTO [31]. Using process measurements to build an ANN surrogate model for a chemical process is a common approach. However, accuracy is not guaranteed as the training data may not contain all scenarios [32]. There is a huge effort to involve physics in the optimization, too [33]. [34] has developed a predictive model-based ANN and Response Surface Methodology (RSM) to optimize the RO water desalination process. An Feed-Forward Neural Network (FFNN) model combined with the first-principle model in Real-Time Optimization (RTO) is built to predict the nonlinear reaction rate and determine the optimal set-point for the reactor. Also, a Model Predictive Controller (MPC) is designed in [35] to keep the process stable while changing the operating condition and minimizing the energy cost. In order to find the optimal set-points for the ROWD process, DE is used. DE is a metaheuristic optimization method which uses the same principle as the genetic algorithm (GA). By combining mutation and crossover, DE is intended to produce new vectors. Selection determines which vectors will be retained for the next generation. In [36], a combination of feature extraction techniques and a multi-objective Binary Differential Evolution (BDE) is designed to define the health indicator based on the subset of optimal features. [37] introduced an Intelligent Multi-Objective Optimization Control (IMOOC) based on an Adaptive Multi-Objective Differential Evolution (AMODE) algorithm to find the optimal set-points of the water treatment facility and balance

the performance and operational costs.

1.2 Thesis Organization and Contributions

Due to the importance of fault diagnosis, modelling, and optimization, and the promising results of DNN in numerous applications, we conduct research that is mainly concerned with the application of DNN for these purposes. This thesis is arranged according to the following structure: four chapters with two main research directions. Chapter two provides an overview of deep learning algorithms and methods. Then an application of DNN in fault diagnosis of an industrial plant is investigated in Chapter three. We design an LSTM-MDN framework to model the normal behaviour of a blade and pitch system of a WTS and generate residual to detect faults. In addition, we introduce a second type of residual constructed by comparing parameters of the ARIMAX model, which analyzes the predicted signal of the LSTM-MDN model and actual measurement. The intention of the design of the second type of residual is to detect faults that impact the behaviour of the closed-loop system. Chapter four investigates the application of deep learning algorithms for process modelling and optimization. First, we develop a PINN framework to enhance the model performance by incorporating domain knowledge into the data-driven model. Then, we formulate a hybrid optimization problem and use the proposed PINN model to construct the optimizer's cost function. The validation experiments are conducted on two datasets collected from Reverse Osmosis Water Desalination Plant (ROWD). Finally, Chapter five summarizes the thesis and concludes with a few remarks.

Chapter 2

Deep Neural Network

2.1 Introduction

Machine learning is a general name for a collection of algorithms and tools that assist machines in learning the pattern within data and performing a specific task. Furthermore, it is essential to mention that deep learning is not a competitive technology to the domain of machine learning. As shown in Fig. 2.1, provides a clear explanation of three terms to avoid any misconceptions. First, deep learning is a sub-domain of machine learning. Then, we can conclude that deep learning, inspired by how the human brain works, is a set of machine learning algorithms that applies to a large amount of data to learn a complex pattern using many computational units.

2.2 Artificial Neural Network

As the underlying structure of deep learning is based on the function of the human brain, we start by introducing some fundamental terminology borrowed from neuroscience. Perceptron is a computational block that is responsible for modelling the nonlinear function. Similar to how human brains use neurons to transmit electrical pulses, perceptron maps the input signals to the output signals. The data mapping is performed by stacking many layers (collection of computational units), in which each layer is responsible for learning specific patterns in data. Hence, this network structure inspired by neurons of human brains is called an Artificial Neural Network

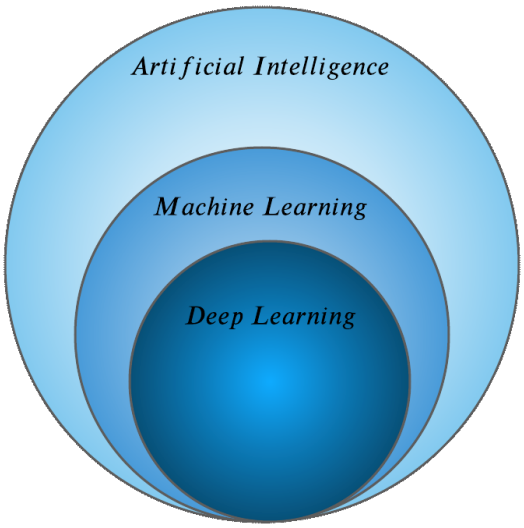


Figure 2.1: Artificial intelligence vs machine learning, and deep learning

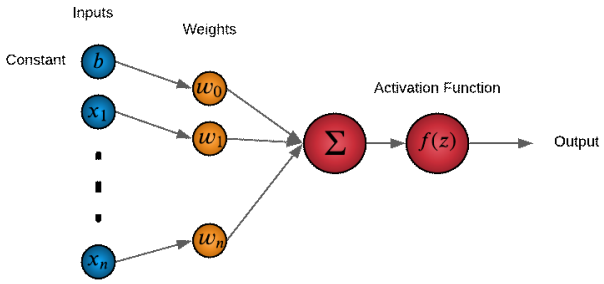


Figure 2.2: Structure of perceptron

(ANN).

Fig. 2.2 indicates the components in each perception that performs the nonlinear mapping of the inputs to the outputs.

The basic form of a neural network contains three layers: input layer, hidden layer, and output layer. A neural network with one hidden layer is called shallow neural network. The mathematical description of this type of network is obtained by the following equations:

$$\begin{aligned} h_j &= f_1(\Sigma_{i=1}^{n} w_{ji}^{(1)} x_i + w_{j0}^{(1)}) \quad j = 1, 2, ..., p \\ y_k &= f_2(\Sigma_{i=1}^{p} w_{ki}^{(2)} h_i + w_{k0}^{(2)}) \quad k = 1, 2, ..., m \end{aligned} \tag{2.1}$$

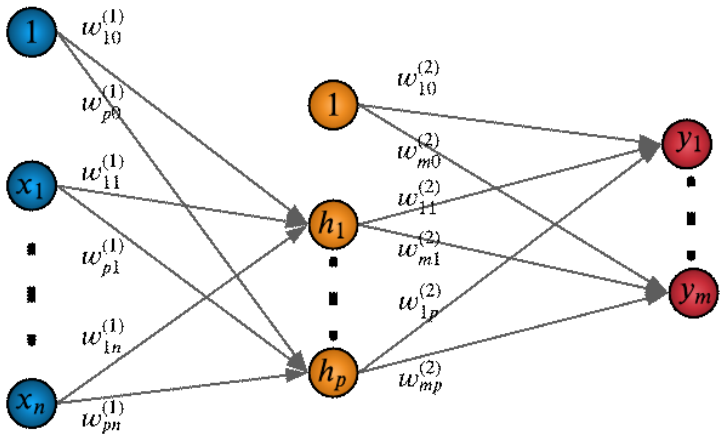


Figure 2.3: A feed-forward neural network n inputs, p neurons in hidden layer and m outputs

where w_{j0}, w_{k0}, $w_{ji}^{(1)}$,and $w_{ki}^{(2)}$ are the biases and weights for the first and second layer,respectively. f_1 and $f2$ are nonlinear function which can be Sigmoid $S(x) = \frac{1}{1+e^{-x}}$ or hyperbolic tangent $tanh(x) = \frac{2}{1+e^{-2x}} - 1$. Fig.2.3 demonstrates the structure of feed-forward neural network n inputs, p neurons in hidden layer and m outputs. Now, we explain the learning process. Given a set of input X and a corresponding set of label Y, the network is trained by minimizing a loss function. Equation 2.2 use a sum of squared error as a loss function:

$$L(w) = \frac{1}{2}\Sigma_{n=1}^{N}||y(X,w) - \hat{Y}||^2 \tag{2.2}$$

Then, the gradient decent optimization is conducted to update the weight and minimize the loss function. This method is calculated as:

$$w^{\tau+1} = w^{\tau} - \eta\nabla L(w^{\tau}) \tag{2.3}$$

Where η is the learning rate, and $\nabla L(w^{\tau})$ (obtained by backpropagation method) is the derivative of the loss function with respect to w. Therefore, the learning process is summarized in 4 steps:

- The network tries to minimize the loss function.

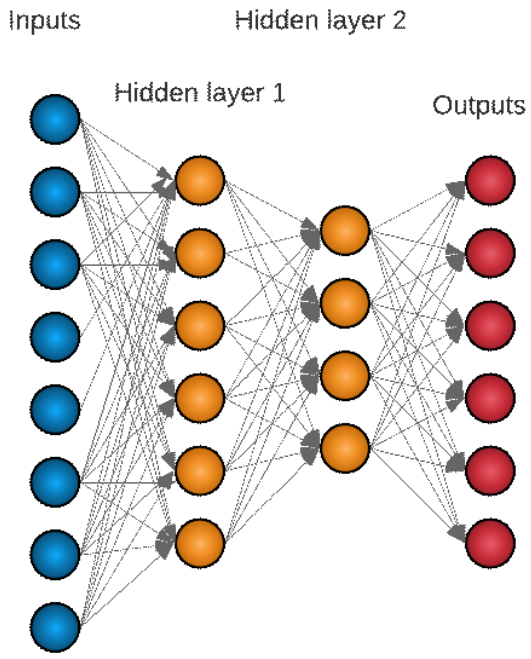


Figure 2.4: Deep structure of neural network with two hidden layers

- The total error is obtained at the output layer by calculating the difference between the original and the predicted values.
- The weights of the last layer are updated using the gradient and the learning rate.
- The exact process is done for the previous layer until it reaches the first layer.

2.3 Deep Neural Network

Depending on the complexity of the problem, we can add more hidden layers to increase the learning capability of the network. The word deep is referred to the multiple hidden layers in the network. Fig. 2.4 shows the deep structure of a neural network with two hidden layers.

In this section, we briefly introduce the most popular deep learning networks and next, we focus on the LSTM network as the primary structure used through this

thesis.

2.3.1 Convolutional Neural Network (CNN)

CNN is one of the most popular deep learning algorithms, utilizing a grid-like topology to process data. It has been widely used in different applications such as Natural Language Processing (NLP) [38], speech processing [39], and machine vision [40]. Like other deep learning techniques, CNNs are biologically-inspired models based on the simulation of the visual cortex in a cat's brain using a complex architecture of neurons. The main modification of CNNs compared to the traditional fully connected network is utilizing local connection and shared weights in the network that let the network fully use the two-dimensional structure of input data, which leads to a faster and easier training stage.

2.3.2 Deep Generative Networks

In this section, we introduce the three most well-known generative networks: Deep Belief Network (DBN), Variational Auto-encoder (VAE), and generative Adversarial Network (GAN). DBN, which was introduced in [41], is a generative model made of stacked layers of Restricted Boltzman Machines (RBM). DBN is suitable for both supervised and unsupervised learning tasks, especially when dimensionality reduction is necessary. VAE [42] is a generative algorithm that assumes a probability distribution for source data. The main application of VAE is to generate fictitious data related to the source data.

Another popular generative model is GAN [43], which contains two models: generative model G and discriminative model D. First, the generator learns to generate fictitious data. Then, the discriminator distinguishes the generator's fictitious data from actual data. Over the training process, generators produce more realistic fake data that can fool the discriminator model.

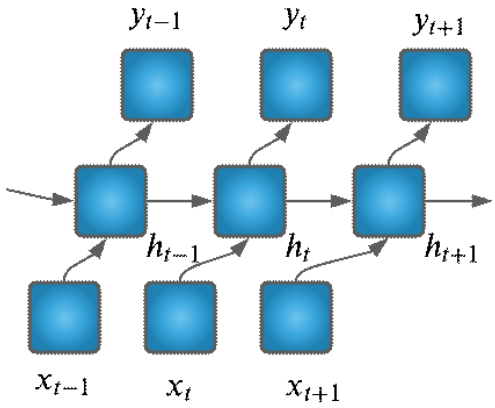


Figure 2.5: RNN unfolded in time

2.3.3 Recurrent Neural Network (RNN)

The most popular deep learning algorithm in time-series analysis is RNN [44]. RNN exploits sequential information, an essential ability to analyze dynamical systems. Therefore, RNNs are well-suited for applications in which sequential data convey valuable information. A typical RNN structure is depicted in Fig. 2.5.

To represent the system with RNN, assume that dynamical system is expressed by:

$$\begin{aligned} h_t &= f_h(x_t, h_{t-1}) \\ y_t &= f_o(h_t) \end{aligned} \tag{2.4}$$

where t represents time. f_h is hidden transition function, and f_o is output function. parameters of each function is indicated by θ_h and θ_o. RNN parameters are estimated by minimizing the following loss function:

$$j(\theta) = \frac{1}{N}\Sigma_{n=1}^{N}\Sigma_{t=1}^{T_n} d(y_t^{(n),f_o(h_t^{(n)})}) \tag{2.5}$$

Given $D = \left[\left((x_1^n, y_1^n), ..., (x_{T_n}^n, y_{T_n}^n)\right)\right]_{n=1}^{N}$ as a set of N training sequences and $d(a, b)$ indicates the divergence measure between a and b.

Long Short-term Memory (LSTM)

There are some problems with RNNs, especially when it comes to vanishing and exploding gradients. This means that the gradient might become exponentially larger or smaller during training because of the multiplication of many derivatives. This sensitivity decreases over time and causes the initial inputs to be forgotten as new ones enter the network. In order to handle this issue, Long Short-term Memory (LSTM) framework is introduced [45]. First, we introduce the original version of

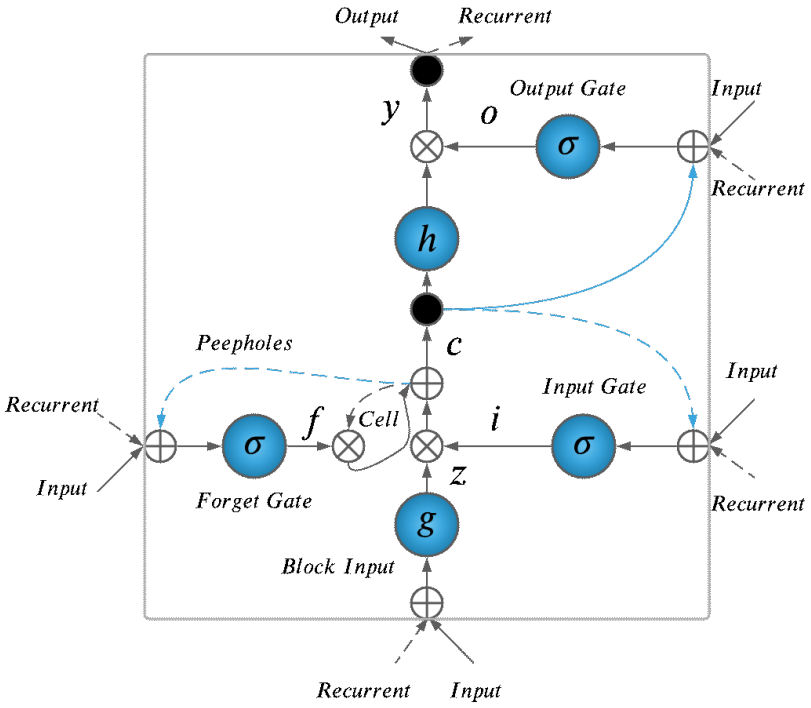


Figure 2.6: Structure of LSTM. The blue line shows the weighted connection, dot line is the connection with lags, and rest of the line are unweighted connection, h is an output activation function, g is input activation function, and σ is gate activation function (always Sigmoid)

LSTM described by [46]. LSTM blocks are building units for layers of RNN. As it is shown in Fig 2.6, the LSTM block contains three gates: input gate, output gate, and forget gate. Also, it includes block input, cell and peepholes connection. In order to investigate how LSTM parameters update, we separate the mechanism into forward-

pass and time-based backpropagation. Assume that the input vector is indicated by x_t at time t. The number of LSTM blocks and the number of inputs are shown by N and M, respectively. Therefore, the forward pass equations are as follows:

The cell is responsible for memorizing values over different time intervals. All other gates are similar to conventional multi-layer neural networks responsible for computing an activation of a weighted sum. The following equations describe the way of updating each LSTM gate.

$$\begin{aligned}
\bar{z}_t &= W_z x_t + R_z y_{t-1} + b_z \\
z_t &= g(\bar{z}_t) \\
\bar{i}_t &= W_i x_t + R_i y_{t-1} + p_i \odot c_{t-1} + b_i \\
i_t &= \sigma(\bar{i}_t) \\
\bar{f}_t &= W_f x_t + R_f y_{t-1} + p_f \odot c_{t-1} + b_f \\
f_t &= \sigma(\bar{f}_t) \\
c_t &= z_t \odot i_t + c_{t-1} \odot f_t \\
\bar{o}_t &= W_o x_t + R_o y_{t-1} + p_o \odot c_t + b_o \\
o_t &= \sigma(\bar{o}_t) \\
y_t &= h(c_t) \odot o_t
\end{aligned} \tag{2.6}$$

Where, $W_i, W_f, W_z, W_o \in \mathbb{R}^{N \times M}$ are input weights, $R_i, R_f, R_z, R_o \in \mathbb{R}^{N \times M}$ are recurrent weights, $p_i, p_f, p_o \in \mathbb{R}$ are peepholes weights, and $b_i, b_f, b_z, b_o \in \mathbb{R}$ are the biases. Then, we can calculate the backpropagation over time using Equations (2.7).

$$
\begin{aligned}
\delta y_t &= \Delta_t + R_z^T \delta z_{t+1} + R_i^T \delta i_{t+1} + R_f^T \delta f_{t+1} + R_o^T \delta o_{t+1} \\
\delta \bar{o}_t &= \delta y_t \odot h(c_t) \odot \sigma'(\bar{o}_t) \\
c_t &= \delta y_t \odot o_t \odot h'(c_t) + p_o \odot \delta \bar{o}_t + p_i \odot \delta \bar{i}_{t+1} + p_f \odot \delta \bar{f}_{t+1} + \delta c_{t+1} \odot f_{t+1} \\
\delta \bar{f}_t &= \delta c_t \odot c_{t-1} \odot \sigma'(\bar{f}_t) \\
\delta \bar{i}_t &= \delta c_t \odot z_t \odot \sigma'(\bar{i}_t) \\
\delta \bar{z}_t &= \delta c_t \odot i_t \odot g'(\bar{z}_t)
\end{aligned}
\tag{2.7}
$$

where, deltas from layer above are represented by Δ_t and gradients are calculated as follows:

$$
\begin{aligned}
\delta W_a &= \Sigma_{t=0}^{T} \langle \delta a_t, x_t \rangle \\
\delta R_a &= \Sigma_{t=0}^{T-1} \langle \delta a_{t+1}, y_t \rangle \\
\delta b_a &= \Sigma_{t=0}^{T} \delta a_t \\
\delta p_i &= \Sigma_{t=0}^{T-1} c_t \odot \delta \bar{i}_{t+1} \\
\delta p_f &= \Sigma_{t=0}^{T-1} c_t \odot \delta \bar{f}_{t+1} \\
\delta p_o &= \Sigma_{t=0}^{T} c_t \odot \delta \bar{o}_t
\end{aligned}
\tag{2.8}
$$

where, a can be any of $\bar{i}, \bar{f}, \bar{z}, \bar{o}$ and $\langle\rangle$ represents the outer product of two vectors.

In general, a peephole connection allows the cell to control all gates. In other words, all gates depend on the previous hidden state and previous internal state. The main reason to augment the LSTM structure with a peephole connection is to make learning precise timing easier. It means that LSTMs with peephole connections between their internal cells and their multiplicative gates are capable of learning the slight differences between sequences of spikes [47].

Chapter 3

Fault Detection Based on Deep Neural Network

3.1 Introduction

In this Chapter, we investigate the application of DNN on fault diagnosis in a wind turbine system (WTS). Wind energy has grown rapidly over the past few decades, and condition monitoring has contributed to more effective operating and maintenance procedures. Additionally, the majority of wind turbines are located offshore, and non-planned visits are costly. Consequently, early detection of faults improves safety and system reliability.

In WTS, sensor data has been used for control and monitoring purposes, especially to detect faults. Usually, the system behaves differently in malfunction conditions compared to the Normal Behaviour Model (NBM) [48]. Various data-driven approaches have been proposed to construct the NBM. However, some of them, do not adequately model the system when operating conditions change. Moreover, failure to consider multi-mode effects can substantially increase False Alarm Rates (FAR) for data-drievn techniques relying on mean-centring and failing to distinguish normal behaviour of the system in different modes from abnormal behaviour. Furthermore, considering the system with a controller in the loop, the controller can compensate for defective actuators' effect, so the system's output still tracks the set-point regardless of actuator faults. Thus, analyzing the observation signal will not detect a significant

change to indicate the fault [49].

To address this research gap, we propose a novel framework for early fault detection within the WTS. First, the NBM of the pitch and blade sub-model of the WTS is represented by an LSTM-MDN model. Then, an adaptive threshold has been designed to detect possible faults based on the MDN parameters. The next step is the real-time system identification using the ARIMAX model, which analyzes the closed-loop system to detect faults with more complicated structures. Last but not least, pruning rules analyze the candidate faults for early detection to reduce FAR. Our work has made the following contributions:

- Develop an FDI framework for early fault detection of the WTS utilizing LSTM-MDN structure
- Identify faults that may affect the closed-loop system by implementing a real-time system identification using the ARIMAX model
- Add pruning rules to reduce the FAR of the proposed framework

3.2 Wind Turbine System

3.2.1 Wind Turbine Model

The WTS considered in this work has a horizontal-axis turbine, and a three-bladed rotor with an active yaw system [50]. Fig. 3.1 illustrates the structure of the sub-models in WTS and their interaction. The WTS model is comprised of three sub-models: pitch and blade, drive train, and power system. Considering that blade and pitch systems account for about $60 - 80\%$ of the malfunctions, we have focused on this sub-model. In the following section, we will briefly describe the model and fault specification of the blade and pitch system.

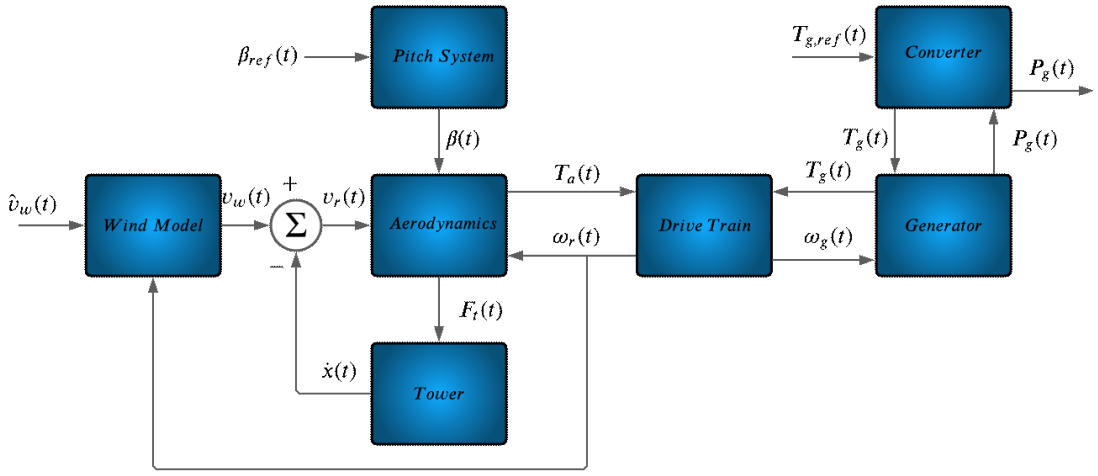


Figure 3.1: Wind Turbine Subsystems including Blade and Pitch, Drive Train, Generator and Converter and Control. The wind speed is the exogenous input, and the generator power is the outlet variable. The rest of the variables are intermediate states.

3.2.2 Blade and Pitch System

This subsystem consists of rotor effective wind speed v_r, pitch angle reference signal β_r, and rotor speed ω_r as inputs, in addition to pitch angle β and aerodynamic torque T_r as outputs. The Pitch system adjusts the blade by manipulating hydraulic actuators controlled by the valves and pumping station. The second-order model of the pitch actuator is as follows:

$$\frac{\beta(s)}{\beta_{ref}(s)} = \frac{e^{-t_d s} w_n^2}{s^2 + 2\xi\omega_n s + w_n^2} \tag{3.1}$$

$$\ddot{\beta}(t) = -2\xi\omega_n\dot{\beta}(t) - \omega_n^2\beta(t) + \omega_n^2\beta_{ref}(t = t_d) \tag{3.2}$$

where t_d, ω_n and ξ are communication delay, natural frequency of the pitch model and damping ratio of the pitch actuator respectively.

3.2.3 Faults Type

This work examines the faults of pitch sensors and pitch actuators. A pitch sensor fault can result in biased output, which can negatively impact both the pitch control

Table 3.1: Description of Wind Turbine Sensors

Sensors	Description
V_w	Wind speed
β_i	Pitch angel i
β_r	Reference for blade angle
ωg	Generator speed
ωr	Rotor speed
τ_g	Generator torque
τ_{gr}	Reference for generator torque
τ_r	Rotor torque
p_g	Generator power

system and the blade angle measurement. Sensor bias is primarily the result of inadequate maintenance, such as inaccurate calibration. This study examines pump actuator faults, including high oil air content and pump wear. The high air content in the oil is causing the closed-loop pitch system to malfunction. There is also a slow wear of the pump that results in low pump pressure. Description of each fault are described in Table 3.3.

Sensor Faults

Any internal fault or malfunction in the pitch sensor may lead to biased output which can affect both the closed loop pitch system and the pitch angle measurement. In normal condition, pitch actuator model is shown in Equations 3.1 and 3.2. However, when the bias is introduced, the model of the pitch actuator is modified.

$$\ddot{\beta}(t) = -2\xi\omega_n\dot{\beta}(t) - \omega_n^2\left(\beta(t) + \beta_{bias}(t)\right) + \omega_n^2\beta_{ref}\left(t = t_d\right) \tag{3.3}$$

Actuator Faults

Actuator faults include hydraulic leakage and high air content in the oil. Parameters of pitch actuator in normal and faulty conditions are shown in Table 3.3.

Table 3.2: Variable values in the benchmark

Fault	Parameters
No Fault	$\omega_n = 11.11 rad/s, \xi = 0.6$
High air content in the oil	$\omega_n = 5.73 rad/s, \xi = 0.45$
Hydraulic Leakage	$\omega_n = 3.42 rad/s, \xi = 0.9$

Hydraulic Leakage is an incipient fault which changes parameter for closed loop pitch system.

$$\ddot{\beta}(t) = -2\tilde{\xi}\tilde{\omega}_n\dot{\beta}(t) - \tilde{\omega}_n^2\beta(t) + \tilde{\omega}_n^2\beta_{ref}(t = t_d) \tag{3.4}$$

$$\tilde{\xi}(t) = (1 - \alpha_{hl}(t))\,\xi + \alpha_{hl}\xi_{hl} \tag{3.5}$$

$$\tilde{\omega}_n(t) = (1 - \alpha_{hl}(t))\,\omega_n + \alpha_{hl}\omega_{n,hl} \tag{3.6}$$

Where α_{hl} is a hydraulic leakage coefficient. It means that in case of hydraulic leakage occurs, it equals to 1, otherwise it is zero.

High air content in the oil is similar to hydraulic leakage which affect on closed loop pitch system.

$$\tilde{\xi}(t) = (1 - \alpha_{ha}(t))\,\xi + \alpha_{ha}\xi_{ha} \tag{3.7}$$

$$\tilde{\omega}_n(t) = (1 - \alpha_{ha}(t))\,\omega_n + \alpha_{ha}\omega_{n,ha} \tag{3.8}$$

Where α_{ha} is similar to α_{hl} and it is high air content in the oil coefficient.

3.3 Proposed Method

We proposed a novel two-stage fault detection algorithm based on LSTM-MDN model and ARIMAX real-time system identification. The suggested framework is expressed in Fig. 3.2. The following is a summary of the general steps:

Table 3.3: Description of wind turbine faults

Fault No	Description
Fault 1	Fixed value on Pitch 1 position sensor 1
Fault 2	Scaling Error on Pitch 2 position sensor 1
Fault 3	Fixed value on Pitch 3 position sensor 1
Fault 4	Changed pitch system response pitch actuator 2 – high air content in oil
Fault 5	Changed pitch system response pitch actuator 3 – low pressure

- Train LSTM-MDN model using NBM data
- Extract pruning rules to identify modes detected in NBM data
- Design real-time system identification based on ARIMAX to extract parameters of the closed-loop system

The proposed method is divided into two stages: The offline stage constructs NBM using LSTM-MDN model and data collected under normal operation. Then, extracts pruning rules to determine the WTS operating conditions. The second phase provides real-time fault detection using an LSTM-MDN model that has been pre-trained and an online system identifier associated with ARIMAX.

LSTM-MDN is employed in this framework in order to estimate the outlet variable of the pitch model. In the offline phase, historical Normal data is used to model the NBM of the WTS. Additionally, wind turbine operation includes startup, transition state, and operation. As a result, it may be considered a bi-modal distribution with two modes and transition states. Assigning a Gaussian distribution to the entire population is not possible. Nevertheless, two or more normal distributions may be sufficient to describe the whole population, and MDN can synthesize the underlying distribution from a mixture of multiple Gaussian distributions. Using DNNs in a black-box fashion to make point predictions is not sufficient, despite the fact that DNN algorithms outperform most traditional modelling methods. Greater trans-

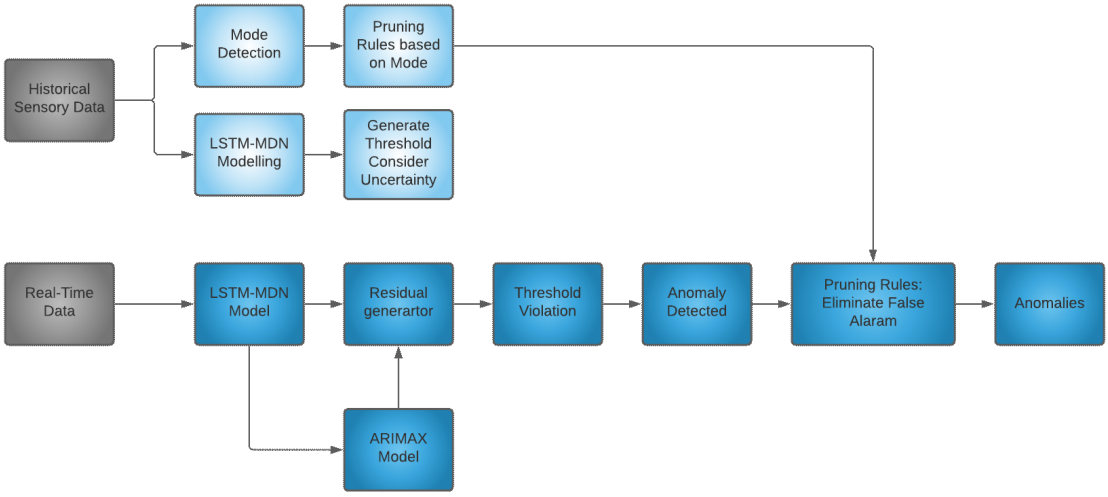


Figure 3.2: Wind turbine fault detection framework

parency is required for critical applications such as fault detection. By considering the uncertainty and constructing the prediction interval, a more robust framework can be established in the presence of noise, which enhances the interpretability of the method. When Normal data is used to train the model, The constructed NBM behavior will be different in fault scenarios. As a result, prediction interval adds more flexibility for the fault detection system to ensure that any deviation from the NBM will not be identified as a fault. In the event that the residual analysis unit shows any samples that exceed the threshold, a possible fault is initially detected. Finally, pruning rules are used to make a final decision, and any fault discovered after the pruning phase is a real fault. Additionally, faults that modify the closed-loop parameters but not the measurement are hard to detect. We introduce a second residual type that investigates any deviation between the ARIMAX parameters obtained from the output of the LSTM-MDN model and the actual measurements. By using the ARIMAX structure, we can visualize the approximation of closed-loop parameters.

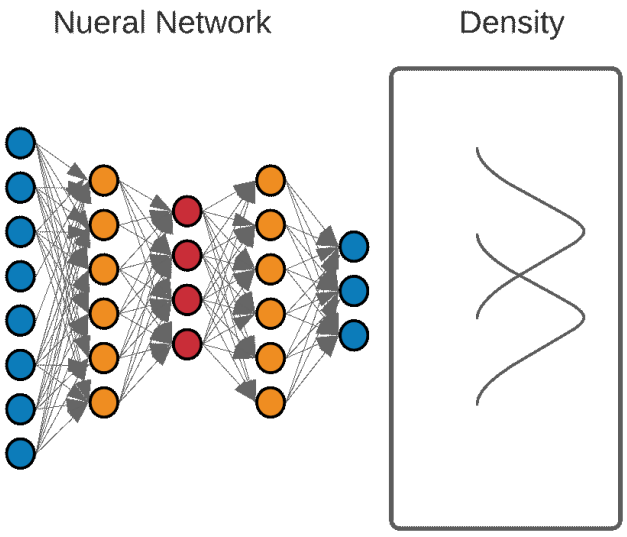


Figure 3.3: MDN structure including input,hidden and output layer

3.3.1 LSTM-MDN Framework

LSTM-MDN provides a subset of values based on their probability density function (PDF), unlike traditional LSTM networks that only provide a point prediction of the target variable. There are many variations of LSTM architecture that have been proposed since its introduction. However, the vanilla LSTM continues to be the most commonly used. An alternative architecture to the LSTM, called gated recurrent unit (GRU) also known as coupled input forget gate (CIFG), was proposed in [51]. The design made no use of peepholes nor output activation functions, and instead integrated the input with the forget gate into an update gate. The proposed structure is illustrated in fig. 3.4. In other words, it would be equivalent to setting $f_t = 1 - i_t$ instead of learning the forget gate weights separately. Generally, uncertainty in the data-driven approach derives from two sources, modelling error and noisy data. Also, reliability is the critical factor for any model fitting application that combines with the decision-making process. Moreover, the neural network model cannot make a reliable prediction if the sample is out of the training distribution. Here, we quantify

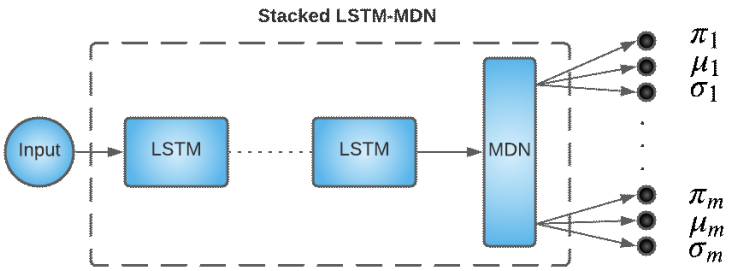


Figure 3.4: Stacked LSTM-MDN structure

the uncertainty in prediction related to the noisy input and model mismatch. As indicated in Fig. 4.3, we will consider the conditional distribution for the target variable to represent the uncertainty in the prediction using mixture density network [52]. The assumption is that the target does not follow any parametric distribution and a probability distribution model generated by the sum of simple distributions. Therefore, the mixture of density distribution can be written as:

$$p(y|x) = \sum_{i=1}^{N} (\pi_i(x)\phi_i(y|\mu_i(x), \sigma_i^2(x)) \tag{3.9}$$

where $\pi_i(x)$ for $i = 1, ..., N$ is the mixture coefficients considered as a prior probability of the outlet variable y built by i_{th} component of the mixture. $\phi_i(y|\mu_i(x), \Sigma_i(x))$ is the kernel distribution of the mixture model and N is the total number of kernels. Although there are myriad choices for kernel distribution function, Gaussian distribution has been selected.

$$\phi(y|x) = \mathcal{N}(y|\mu_i(x), \sigma_i^2(x)) \tag{3.10}$$

Where c is the dimensionality of output vector y and $\mu_i(x)$ is the centre of the i_{th} kernel. The assumption in Eq. (4.4) is that each component of the distribution is statistically independent within the component of the output vector. A full covariance matrix for each Gaussian kernel can relax this assumption. However, As [52] has shown that the Gaussian mixture model (GMM) brought in Eq. (4.4) can approximate any given density function to arbitrary accuracy, provided correct mixing

coefficients and Gaussian parameters. Mixing coefficients must be positive to ensure that the $p(y|x)$ is a valid representation of y. Also, the mixing coefficients $\phi_i(x)$ should satisfy $\sum_{i=1}^{N} \pi_i(x) = 1$. This is guaranteed by selecting $\phi_i(x)$ as output of a network with a Softmax function [53].For the N-components mixture model and c-dimension y, Parameters of the GNN model calculate as follow [52]:

$$\pi_i = \frac{e^{(a_i^\pi)}}{\sum_{j=1}^{N} e^{(a_j^\pi)}} \tag{3.11}$$

Additionally, the variance must be strictly positive, so it can be driven as:

$$\sigma_i = e^{(a_i^\sigma)} \tag{3.12}$$

Finally, μ_i is not restricted by any constraint and defines as:

$$\mu_{ni} = a_{ni}^\mu \tag{3.13}$$

Where $a_i^\pi, a_i^\sigma, a_{ni}^\mu$ are the raw outputs of the MDN model derived by the neural network, the parameters w of the MDN model learns using maximum likelihood. The loss function is as follows:

$$E(w) = -\sum_{c=1}^{C} ln \left(\sum_{i=1}^{N} \pi_i(x_c, w) \mathcal{N}(y_c|\mu_i(x_c, w), \sigma_i^2(x_c, w) \right) \tag{3.14}$$

Mixing coefficients $\pi_i(x)$ can be considered as an x-dependent prior probability. Therefore, the corresponding posterior probability introduces as:

$$\delta_i(y|x) = \frac{\pi_i \mathcal{N}_{ic}}{\sum_{l=1}^{N} \pi_l \mathcal{N}_{il}} \tag{3.15}$$

where $\mathcal{N}_{ic}$ defines as:

$$\mathcal{N}_{ic} = \mathcal{N}(y_c|\mu_i(x_c), \sigma_i^2 x_c) \tag{3.16}$$

Consider the derivative of the loss function with respect to the network outputs, it is obtained by:

$$\frac{\partial E_c}{\partial a_i^\pi} = \pi_i - \gamma_i \tag{3.17}$$

$$\frac{\partial E_c}{\partial a_{ic}^{\mu}} = \gamma_i \left(\frac{\mu_{ic} - y_c}{\sigma_i^2} \right) \tag{3.18}$$

$$\frac{\partial E_c}{\partial a_i^{\sigma}} = -\gamma_i \left(\frac{||\boldsymbol{y} - \boldsymbol{\mu_i}||^2}{\sigma_i^3} - \frac{1}{\sigma_i} \right) \tag{3.19}$$

$p(y|x)$ provides more detailed information to approximate the interval using the specific quantile. Interval width (IW) is approximated by mean, variance and confidence level(α). IW can be written as:

$$\begin{aligned} I_{Upper} &= \mu + \alpha\sqrt{\sigma} \\ I_{Lower} &= \mu - \alpha\sqrt{\sigma} \\ IW &= I_{Upper} - I_{Lower} \end{aligned} \tag{3.20}$$

Regression confidence level relates to the IW. It means that if the model is not confident due to the noisy data or model mismatch, IW will be wider.

3.3.2 Dynamic Identification

In this section, we investigate the effect of faults that change the parameters of the closed-loop system and do not directly impact the measurement. Detecting faults that change the closed-loop system is a difficult task because the controller in the loop can compensate for the impact of faults. The NBM constructed by the LSTM-MDN model is trained under normal operation. So, further study of the estimated signal in the presence of faults is conducted by adding a new layer to the framework. Motivated by that, we propose to employ a fast online system identification using an auto-regressive moving average with an exogenous input (ARMAX). We aim to investigate the estimated signal of the LSTM-MDN model and identify the parameters of the closed-loop system to see if there are any changes in the presence of faults. ARMAX incorporates lags to model the dynamics of the process. The structure of the ARMAX is as follows:

$$A(z^{-1})y_k = B(z^{-1})u_k + \epsilon_k \tag{3.21}$$

where A, B are nominator and denominator of the transfer function of the input to the process output. ϵ is a measurement noise and assumed to be a white noise. However, to consider non-stationary assumption, we use the described method with a noise filter to define the integrated term and express ARIMAX model. Therefore, Eq 3.21 can be rearranged as follows:

$$A(z^{-1})y_k = B(z^{-1})u_k + \frac{1-\alpha z^{-1}}{1-z^{-1}}\epsilon_k \tag{3.22}$$

Then, we have:

$$A(z^{-1})y_k(1-z^{-1}) = B(z^{-1})u_k(1-z^{-1}) + (1-\alpha z^{-1})\epsilon_k \tag{3.23}$$

where α indicates the level of activity of integrator in the system. In the case of $\alpha = 1$, the integrator is inactive and ARIMAX turns to be ARMAX model. Then model polynomial can be identified by:

$$\begin{aligned} \tilde{A}(z^{-1}) &= (1-z^{-1})A(z^{-1}) \\ \tilde{B}(z^{-1}) &= (1-z^{-1})B(z^{-1}) \\ \tilde{C}(z^{-1}) &= 1-\alpha z^{-1} \end{aligned} \tag{3.24}$$

The structure of the polynomials in Eq. (3.24) is as follows:

$$\begin{aligned} \tilde{A}(z^{-1}) &= 1 + a_1 z^{-1} + ...a_n z^{-n} \\ \tilde{B}(z^{-1}) &= 1 + b_1 z^{-1} + ...b_n z^{-n} \\ \tilde{C}(z^{-1}) &= 1 + c_1 z^{-1} + ...c_n z^{-n} \end{aligned} \tag{3.25}$$

In the final step, instances are replaced by δ variables to consider the non-stationary. So, Eq. (3.21) cane be rewritten as:

$$A(z^{-1})\Delta y_k = B(z^{-1})\Delta u_k + \frac{1-\alpha z^{-1}}{1-z^{-1}}\epsilon_k \tag{3.26}$$

where Δ calculates the difference of the two consecutive instances.

Detect Operating Mode

if $\mu_0 - \alpha\sigma_0 < P_g < \mu_0 + \alpha\sigma_0$ **then**
  if $V_w <$ *effective wind* **then**
    Mode: **startup**
  end
  if $V_w >$ *effective wind* **then**
    Power sensor malfunction
  end
end
if $\mu_1 - \alpha\sigma_1 < P_g < \mu_1 + \alpha\sigma_1$ **then**
  if $V_w >$ *effective wind* **then**
    Mode: **Operational**
  end
  if $V_w <$ *effective wind* **then**
    Power sensor malfunction
  end
end
if P_g *out of two mentioned range* **then**
  Mode: **Transition**
end

Figure 3.5: Algorithm 1

3.3.3 Pruning Rules

Pruning rules are intended to reduce the number of false alarms. While the wind velocity is less than the effective wind, which is the wind that rotates the turbine blades, the system is at its startup stage. Accordingly, the data collected during that period does not provide operational information. The fault detection system is designed to operate in real-time, but since the wind velocity varies over time, many rising and falling moments can be considered a transition between startup and operation. Because there are fewer data over this period than the other two modes, it is not easy to find an accurate model for the transition mode. For this reason, the fault detection system must be aware of the operating mode in order to minimize false alarms. Using the Algorithm (1), the pruning rules can determine the operating mode. Therefore, any possible fault detected by the FDS will be analyzed using pruning rules to determine the actual fault.

3.4 Experimental Setup

We evaluate our proposed FDI framework on data collected from the wind turbine test bench of kk-electronic a/s [54]. The test bench contains five different types of pitch and blade system faults.

3.4.1 Preprocessing

The test bench incorporates wind turbulence and sensor noise to create a realistic test scenario. Thus, performing a data preprocessing step is essential to remove invalid data and reduce noise prior to modelling. The data must be extracted from the contaminated signal at a high resolution. It is essential to separate the valuable information from the noise while retaining the most relevant information. Additionally, the fluctuation of the variable makes it difficult to filter the noise in multi-stage applications. In order to reduce the noise in the input signal while capturing valuable information, an exponential weighted moving average (EWMA) is applied. EWMA can be written, as shown in Eq. (3.27):

$$\hat{x}_i(t) = \lambda x_i(t) + (1-\lambda)\,\hat{x}_i(t-1) \tag{3.27}$$

where $\hat{x}_i(t)$ is the EWMA of the signal $x_i(t)$ and λ is the forgetting factor. As λ increases, the contribution coming from the current state of the signal will rise. The initial value $\hat{x}_i(0)$ is computed based on the average of historical data.

3.4.2 Performance Metrics

One of the most common measures for evaluating FDS algorithms is the F score.

Before calculating the F score, recall and precision must be calculated. Hence, they are based on two criteria: 1) a True Positive (TP) when the fault is correctly detected, and 2) a True Negative (TN) when no fault is detected. TP and TN are then added together to determine the correct decision. False Positive (FP) occurs when a fault is incorrectly detected . A False Negative (FN) error occurs when a

fault is incorrectly identified as normal. Then, precision is defined in 3.28:

$$Precision = \frac{TP}{TP + FP} \tag{3.28}$$

Where TP and FP are total number of true positive and false positive respectively. Also, recall is as follows:

$$Recall = \frac{TP}{TP + FN} \tag{3.29}$$

Finally, F score is calculated using recall and precision. Then, F score is:

$$F = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.30}$$

F score is the harmonic average of precision and recall and reaches its best value at one.

3.4.3 Performance Evaluation

Our aim is to demonstrate the effectiveness of the proposed approach by comparing it with traditional multivariate approaches that are commonly applied,i.e., Dynamic Principal Component Analysis (DPCA), LSTM, stacked LSTM (SLSTM), LSTM autoencoder (LSTMAE),and GRU. For fair comparison all LSTM models use same parameters. Wind turbine pitch and blade variables are used in constructing the pitch and blade LSTM-MDN network, as well as other variables closely interdependent with it. Table 3.4 lists the simulator variables used as inputs to the blade and pitch LSTM-MDN models. A simulator dataset is used to train the pitch and blade LSTM-MDN models, and then the parameters of the MDN model are used to calculate prediction error and interval. According to Fig 3.6, the residual pitch and blade values fluctuate within an adaptive threshold range determined by MDN variance for fault-free scenario. Also, generated power is estimated using proposed structure to determine the mode change for pruning rule phase. Fig 3.7 demonstrates the model performance for generated power using NBM data. For comparison, we generated 100 test data-sets with different wind patterns and fault scenarios. Table 3.5 describes

Table 3.4: LSTM-MDN inputs and outputs variable

Inputs	Outputs
Wind Velocity	Blade Angle i
Rotor Speed	Rotor Torque
Reference for Blade Angle	
Generator Speed	

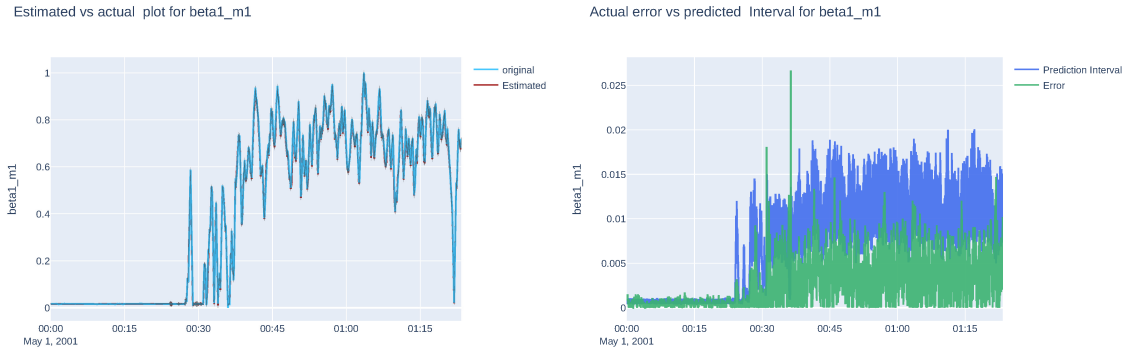
Table 3.5: Hyper Parameter of the Method

Hyper Parameters	Value
Number of Features	6
Number of Samples	1,000,000
Number of Stacked Network	1
Number of Neurons	20
Number of Mixture Density	2
Length of Input Sequence	5
Learning Rate	0.001
dropout rate	0.10
Number of Epoch	20

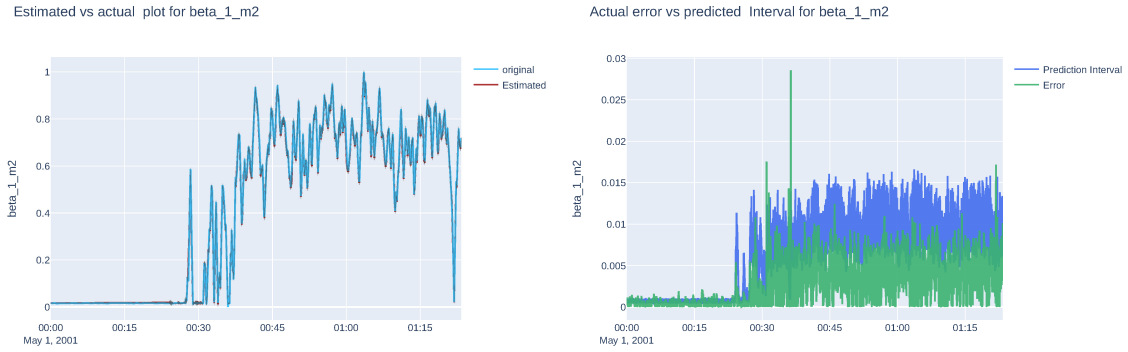
the inputs and outputs parameter using to build LSTM-MDN network for pitch and blade sub-system. Also, a comprehensive evaluation is shown in Table 3.6.

Analysis of Pitch and Blade Sensor Fault

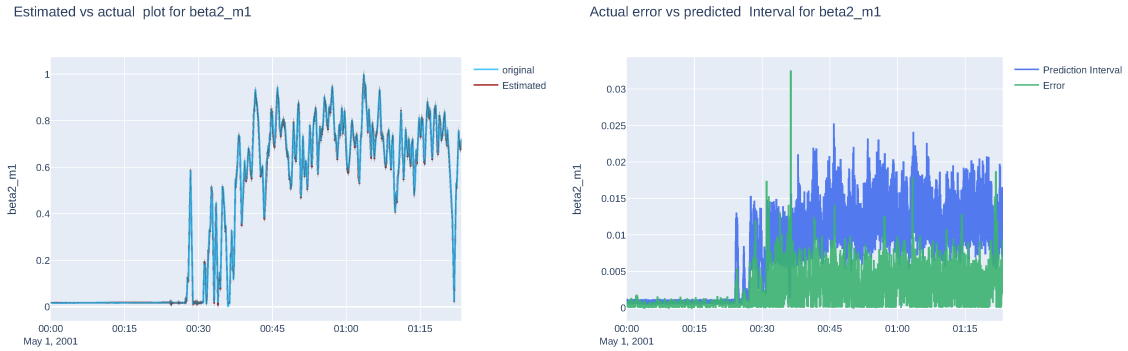
In the previous section, we discussed two types of sensor faults: Fixed values on sensors and scaling factors. It is easier to detect the first fault since a malfunctioning sensor does not show any variation. However, the second type, a scaling factor on the sensor output, is more difficult to detect. Fig 3.8 shows how residual fluctuates in case of fault one. As the scaling factor changes the expected output pattern, the model has a different response when fault two occurs. Fig 3.9 illustrates the generated residual



(a) left: Angle sensor of blade 1 actual vs estimated value. Right: Prediction error and interval.

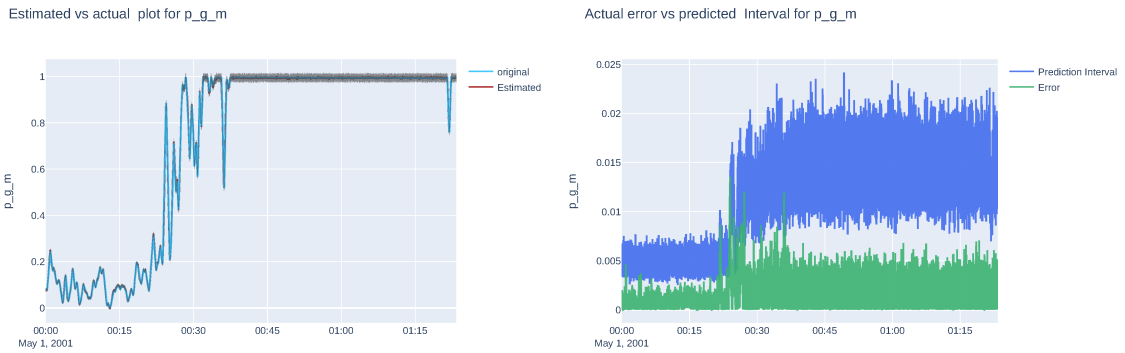


(b) left: Angle sensor of blade 2 actual vs estimated value. Right: Prediction error and interval.



(c) left: Angle sensor of blade 3 actual vs estimated value. Right: Prediction error and interval.

Figure 3.6: Left figures, prediction vs actual values of the angle sensor , right figures: Prediction error and interval



(a) left: Generated power actual vs estimated value. Right: Prediction error and interval.

Figure 3.7: Left figures, prediction vs actual values of the generated power , right figures: Prediction error and interval

for fault two scenario. Finally, the last scenario is a fixed value on the angle sensor of the third blade. Fig 3.10 shows how model detects this fault too. FDI is more sensitive to scaling factor errors by using a threshold to detect sensor malfunction, which is one of the main advantages of the proposed method. The proposed method can detect scaling factor faults despite a small scaling value.

Analysis of Pitch and Blade Actuator Fault

The actuator fault described in the previous section does not directly affect the measurement. However, it can be determined by analyzing the closed-loop system behaviour. As a result, we designed an ARIMAX model to analyze the estimated signal and compare the identified parameters with the parameters obtained from the actual measurements. It can be seen in Fig. 3.11 that there is an abrupt change in the residual resulting from the subtraction of the first and second autoregressive parameters derived from the ARIMAX model applied to actual and estimated data. Presently, residual shows a change between expected and actual model outputs, which is indicated by red. Thus, we conclude that high-pressure oil is present on the actuator of the second blade, while the rest of the residuals appear to be normal. As a final step, fault five is detected in Figure 3.12 by following the same procedure as fault 4.

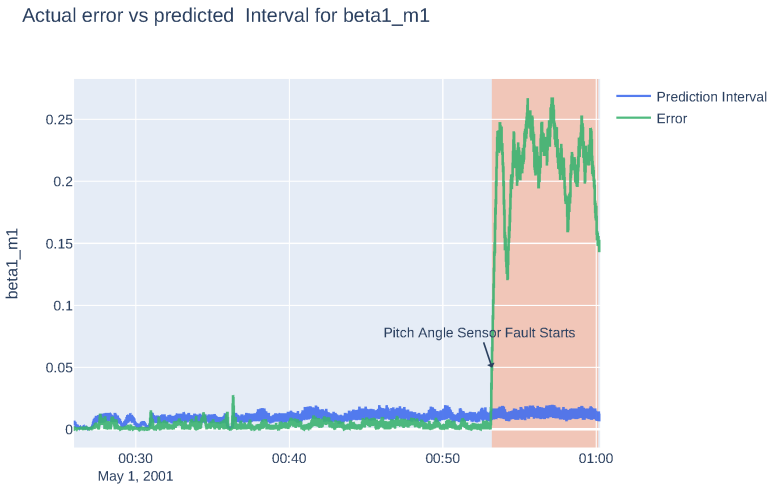


Figure 3.8: Violation of prediction interval by generated residual for angle sensor of blade one in case of fixed value on angle sensor fault one.

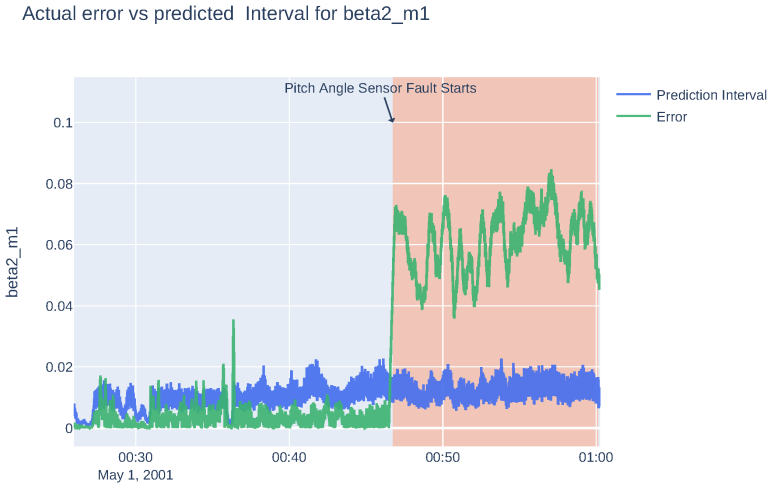


Figure 3.9: Violation of prediction interval by generated residual for angle sensor of blade two in case of scaling factor on angle sensor fault one.

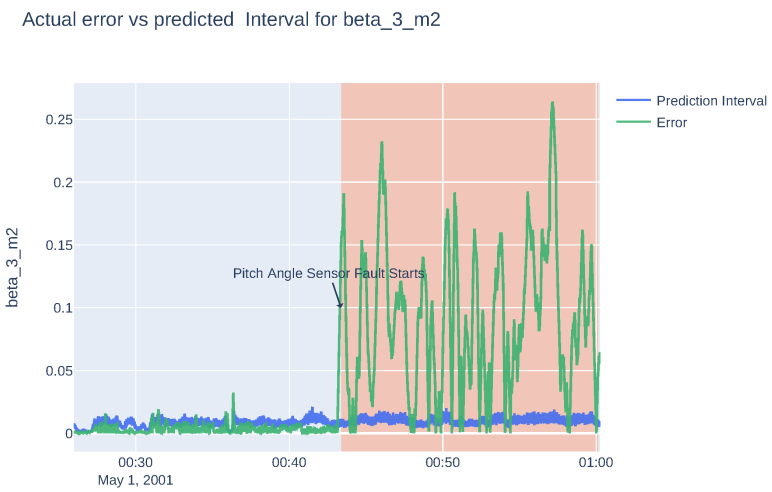
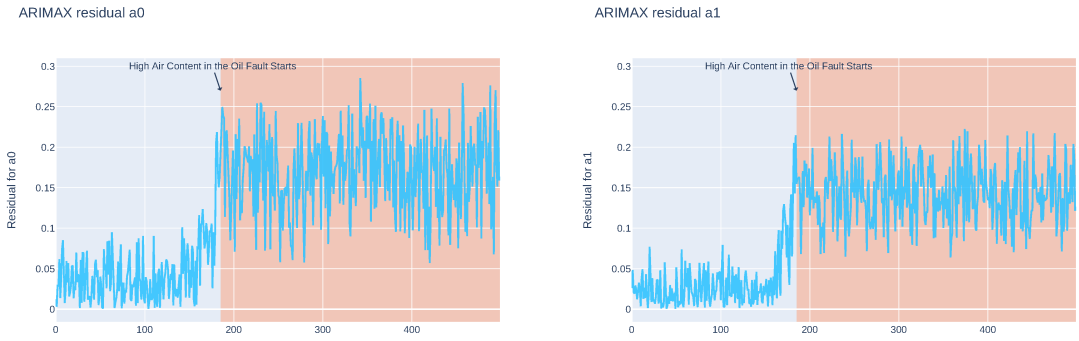


Figure 3.10: Violation of prediction interval by generated residual for angle sensor of blade three in case of fixed value on angle sensor fault 3.

As show in this figure, there is an abrupt change the generated residual of the third blade angle that indicate the low pressure (hydraulic leakage).

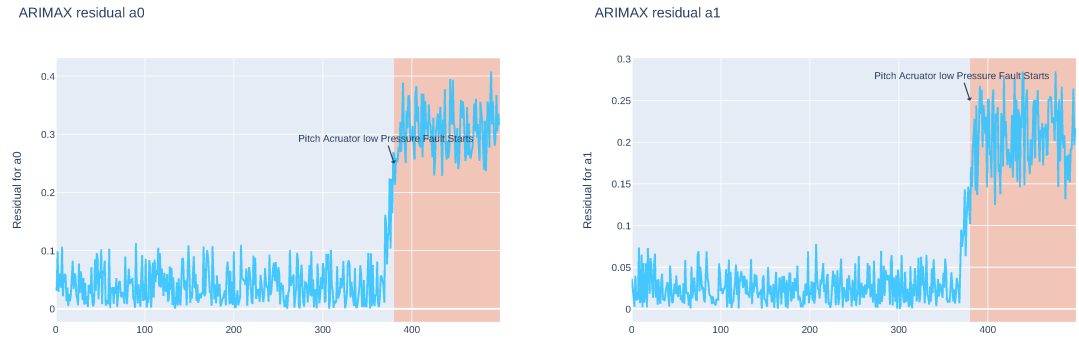
Table 3.6: Evaluation of the Proposed Method Versus Benchmarks

Algorithms	Precision	Recall	F score
DPCA	0.948	0.9125	0.9299
LSTM	0.948	0.925	0.9367
SLSTM	0.9358	0.9125	0.9240
LSTMAE	0.974	0.950	0.9620
GRU	0.974	0.9625	0.9685
Proposed Method	0.987	1	0.9937



(a) Left: a0, Right: a1

Figure 3.11: Residual generated by subtracting the auto-regressive coefficient obtained from ARIMAX model applied on the estimated data and actual data. high air content in the oil fault.



(a) Left: a0, Right: a1

Figure 3.12: Residual generated by subtracting the auto-regressive coefficient obtained from ARIMAX model applied on the estimated data and actual data. Low pressure fault

Chapter 4

Modelling and Optimal Operation of A Nonlinear Industry Process via the Physics-Informed Neural Network Approach

4.1 Introduction

Modelling and optimization of reverse osmosis systems are investigated in this work. We propose a framework to combine ESL-MDN with the DK to model the RO process behaviour and formulate an optimization problem considering variations in feed-water features and customer demand. However, there is a widespread problem in water treatment plants. Design engineers spend considerable time optimizing the plant's energy, but operators rarely have time to tune the plant to optimize this energy use. An essential task in a water treatment the plant is to keep the plant running to avoid water shortages for the public the plant serves. Therefore, a storage tank is considered to meet the customer's demand at all times. The hybrid model integrates with optimizer and feed-water dynamic profile to save energy while ensuring freshwater availability. The algorithm plans how much water should be produced in a day over several days to save energy but produce the minimum water. The main contribution of our work summarizes as follows:

- Propose PINN structure by introducing new features obtained from the math-

ematical description of the physical model. The intermediate states are non-measurable process variables that can be calculated using inlet and outlet conditions.

- Develop ESL-MDN- model to enhance the generalization of the framework and make it feasible for a process with variables that do not follow any parametric distribution.

- Consider uncertainty caused by training set variations by estimating the distribution's parameters instead of a single prediction.

- Design an innovative dropout mechanism that adopts the information obtained from uncertainty estimation to update the masking rule and prevent large errors caused by DK infusion propagating through the system.

- Design an optimization framework based on differential evolution algorithm for RO process considering feed-water characteristic variations and customer's demand.

4.2 Proposed Approach

Motivated by the challenges mentioned in the above section, a hybrid approach of combining a mathematical model and DL in the optimization problem has been studied in this work. In the next part, we first explain the PINN framework that is designed to replace the nonlinear model of an industrial process by using the ESL-MDN structure, and at the same time considering the mathematical description of the process. Also, we introduce a novel deterministic dropout approach to reduce the induced error caused by combining DK with DL. Finally, we design an optimizer to employ the obtained ESL-MDN model to manage the process energy consumption.

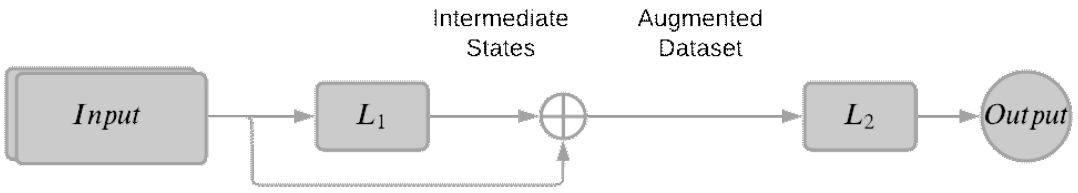


Figure 4.1: PINN based on ESL to infuse attributes related to the physical domain into DL model

4.2.1 Physics-informed Neural Network (PINN) based on Ensemble Sequential Learning (ESL)

Finding the most influential attributes that lead to the best model performance and low computation cost is critical for developing a machine learning model. Many industrial processes, including thermal, hydraulic, chemical, and etc, follow the fundamental physical laws. Therefore, extensive knowledge has been developed through the decades that can be used to generate the most important features. One of the PINN structures to incorporate knowledge into machine learning is Ensemble Sequential Learning (ESL). As it is shown in Fig 4.1, unobservable features obtained from physics are estimated first and used to generate augmented data-set to estimate the outputs. ESL falls within the concept of the stacking procedure. The main idea is to connect multiple estimators in a sequential order to conduct an extension of the input. It means that the previous layer's output is added to the input of the next layer. ESL has been used in many applications since its introduction. This study used ESL to incorporate physics into machine learning and enhance estimation performance. In this sub-section, the components in ESL are explained, including intermediate states, uncertainty based on MDN structure and the proposed deterministic dropout.

Let us start with the features that cannot be measured directly, and they reflect useful information related to the process modelling. Two types of intermediate states have been investigated through this work. The assumption for the first type is that the inlet conditions can directly calculate these unmeasurable variables. However,

the second type is more challenging to formulate as the outlet variables in association with input data need to infer the intrinsic physical parameters. Consider a nonlinear function $\tilde{f}(u)$ that calculates intermediate variable Y_1^c based on the physical equations:

$$\begin{aligned} Y_1^c &= \tilde{f}(u) \\ y &= F_{NN}(Y_1^c, u) \end{aligned} \tag{4.1}$$

It is difficult to extract the second type of intermediate states while outlet conditions are unknown during the operation. Therefore, we need to estimate them based on the inlet data. The suggested framework utilizes training data to generate labels and deploy a NN to learn the model that maps input variables to intermediate states. Assume that nonlinear function $\tilde{g}(u, y)$ describes the relationship of intermediate states with process inputs and outputs, and indicates by Y_2^c. We need to rewrite Eq. (4.1) as follows:

$$\begin{aligned} Y_1^c &= \tilde{f}(u) \\ Y_2^c &= \tilde{g}(u, y) \\ \hat{Y}_2^c &= G_{NN}(u) \\ y &= F_{NN}\left(\hat{Y}_2^c, Y_1^c, u\right) \end{aligned} \tag{4.2}$$

This step is important as it sheds light on high-value information that is impossible to measure. In the following section, we will describe the uncertainty quantification approach using MDN structure.

Uncertainty Estimation

Generally, uncertainty in the data-driven approach derives from two sources, modelling error and noisy data. Also, reliability is the critical factor for any model fitting application that combines with the decision-making process. Moreover, the neural network model cannot make a reliable prediction if the sample is out of the training distribution. Here, we quantify the uncertainty in prediction related to the noisy input and model mismatch. As indicated in Fig. 4.3, we will consider the conditional distribution for the target variable to represent the uncertainty in the prediction using

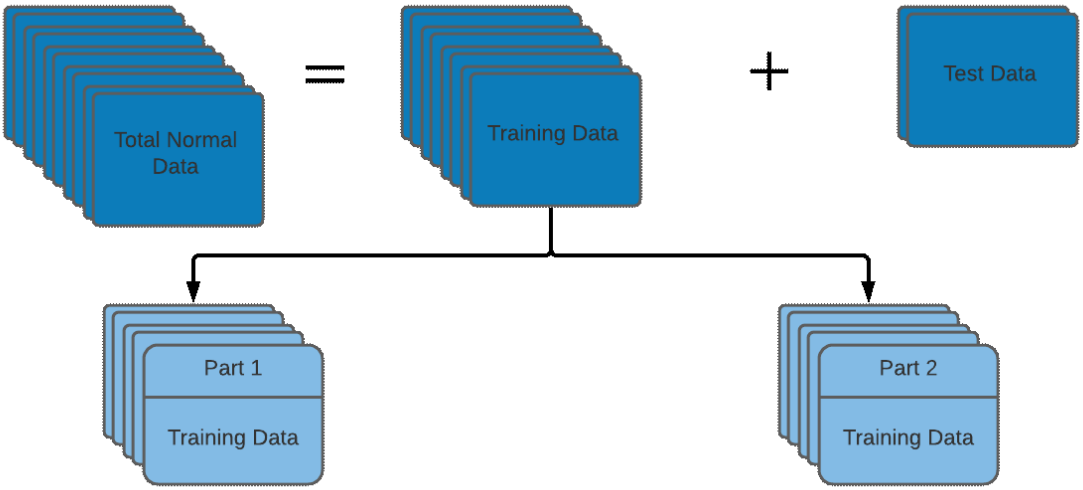


Figure 4.2: Data allocation process

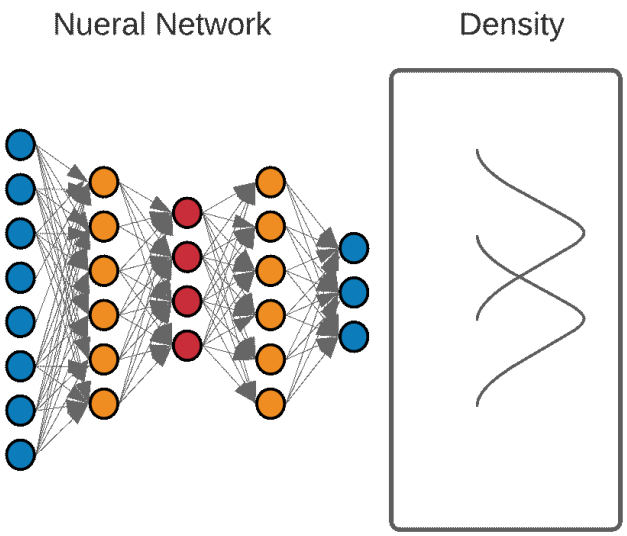


Figure 4.3: MDN structure is the new layer on top of any NN model to represent the conditional probability density function of the targets

mixture density network [52]. The assumption is that the target does not follow any parametric distribution and a probability distribution model generated by the sum of simple distributions. Therefore, the mixture of density distribution can be written as:

$$p(y|x) = \sum_{i=1}^{N} (\pi_i(x)\phi_i(y|\mu_i(x), \sigma_i^2(x)) \tag{4.3}$$

where $\pi_i(x)$ for $i = 1, ..., N$ is the mixture coefficients considered as a prior probability of the outlet variable y built by i_{th} component of the mixture. $\phi_i(y|\mu_i(x), \Sigma_i(x))$ is the kernel distribution of the mixture model and N is the total number of kernels. Gaussian distribution is a suitable candidate to select as a kernel for MDN.

$$\phi(y|x) = \mathcal{N}(y|\mu_i(x), \sigma_i^2(x)) \tag{4.4}$$

Where c is the dimensionality of output vector y and $\mu_i(x)$ is the centre of the i_{th} kernel. $p(y|x)$ provides more detailed information to approximate the interval using the specific quantile. Interval width (IW) is approximated by mean, variance and confidence level (α). IW can be written as:

$$\begin{aligned} I_{Upper} &= \mu + \alpha\sqrt{\sigma} \\ I_{Lower} &= \mu - \alpha\sqrt{\sigma} \\ IW &= I_{Upper} - I_{Lower} \end{aligned} \tag{4.5}$$

IW can be considered as a confidence level index of the model. In other words, if the model is not confident enough due to the noisy data or model mismatch, IW will be wider. Therefore, in the next session, we explain how to use this fact to update the masking rule for the proposed dropout.

Deterministic Dropout

The ESL-MDN model has been built to inject physics-based knowledge into DL while preventing errors related to introducing new features. According to Fig. 4.2, training data are divided into two halves; the first half is responsible for obtaining data for

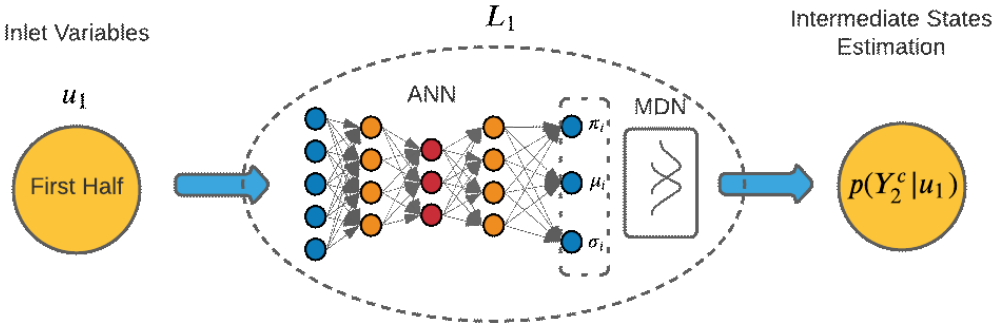


Figure 4.4: L_1 is the first leaner in PINN sequential structure to estimate unobservable attributes that come from the physical law

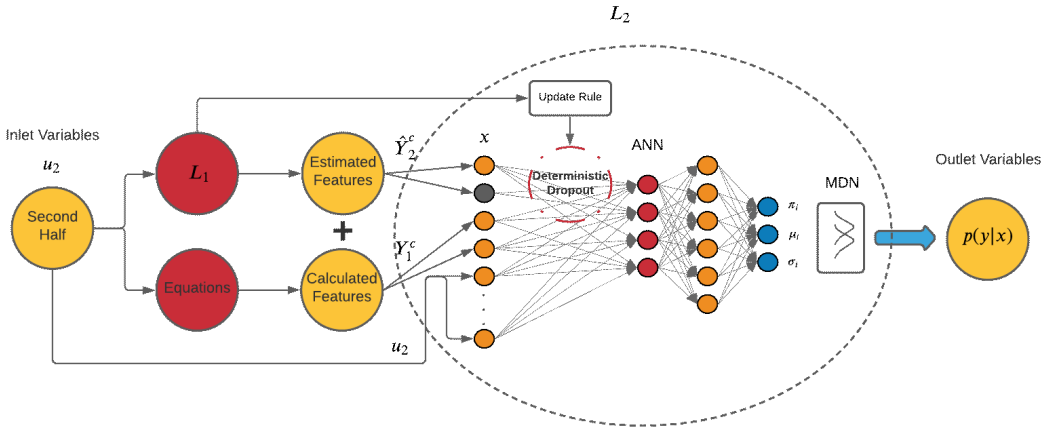


Figure 4.5: L_2 is the second learner in PINN sequential structure to estimate targets

unobservable states and calculating physics-based features. Then, the second half aggregates with a new feature estimated by L_1 and train L_2 by mapping the augmented dataset to the process targets. The details of the two learners L_1 and L_2 are shown in Figs. 4.4 and 4.5. In order to consider the impact of the induced error caused by engineered features, We propose a deterministic dropout mechanism to prevent the error propagates through the system. Let us start with a brief introduction of the random dropout technique brought in [55]. Assume that data set U, an input for neural network, is composed of N number of sensors and i samples associated with each sensor. Consider each layer of the network created by Z_l as input and Y_l as outputs. So, in this network, the relationship between layer l and $l+1$ is expressed

as:

$$Z_{l+1} = W_l Y_l + b_l \tag{4.6}$$

where W_l and b_l are weight matrix and bias to connect layer l to $l+1$. Consider $\Phi(l+1)$ as an activation function for layer $l+1$, then

$$Y_{l+1} = \Phi_{l+1}(Z_{l+1}) \tag{4.7}$$

Dropout is the mechanism to terminate some connection between layer l and $l+1$ randomly by generating a mask rule D_l. So, Eq. (4.6) can be modified as:

$$Z_{l+1} = (W_l \odot D_l) Y_l + b_l \tag{4.8}$$

where $D_l(i,j) = Bernouli(p)$ for i,j are nodes in layer l and $l+1$. Also, $\odot$ is the Hadamard product that calculates product of two matrices term by term. So, Eq. (4.9) can be written as:

$$Y_{l+1} = \Phi_{l+1}((W_l \odot D_l) Y_l + b_l) \tag{4.9}$$

Dropout applies to any layer, including input, hidden, and output layers. The idea is to use deterministic dropout placed at the second learner's input layer. The first learner provides intermediate variables and their estimation error. We can use this information to eliminate those connections related to the intermediate state instance with a high estimated variance. In other words, we drop features affected by noise and keep the rest of the important variables. Consider two consecutive neural network models in which the first learner's output is used as an input for the second learner. The first learner may generate unreliable outputs in noisy samples, drastically reducing the second learner's performance. Consider $D^d(i,j)$ as a dropout mask between input of second learner and its layer 1 where the (i,j)th element $(i \in [i_1, .., i_k, i_{k+1}, ...i_{k+s}])$.Where k and s are the total numbers of intermediate and input variables. D^d can be expressed as:

$$D^d(t) = \begin{cases} 0 & IW(t) \geq \delta \\ 1 & \text{Otherwise} \end{cases} \tag{4.10}$$

Threshold δ is a hyper-parameter obtained by tuning. Therefore, the output of the second learner is as follows:

$$Y_2 = \Phi_2 \left(\left(\left[W_2^1, W_2^2 \right] \odot \left[D^d, 1 \right] \right) X + b_2 \right) \tag{4.11}$$

Where X is the augmented training data (intermediate states estimated + second portion of inlet variables),W_2^1 is the weight for intermediate variables, and W_2^2 is the weight for process inputs. As it is shown in Eq. (4.11), we modify the forward propagation by adding a deterministic dropout which can be updated using the information obtained from the first learner.

The proposed model has the following advantages compared with black-box neural network modelling: 1) The hybrid model can work reliably without data as it is integrated with physical knowledge, while the black-box neural network cannot be functional. 2) In the case of uncertainty, the model prevents significant error propagation through the network because if there is a corrupted sample, the error estimator's outputs will show high variance. 3) network parameters are increased in minor ways by doubling the number of output variables in the first learner to estimate intermediate states and their estimation error.

4.2.2 Optimization with Neural Network Model

A valid, comprehensive mathematical model is a crucial step for process optimization. However, a representative model is not always accessible. Due to the non-stationary nature and high level of non-linearity for complex processes, we develop an optimizer whose components are derived by the DNN model. In general, an optimization problem is formulated as shown in Eq. (4.12):

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & \text{Constraints} \end{aligned} \tag{4.12}$$

However, it is not easy to formulate the whole system. We intend to estimate the process's nonlinear and most complex components using the DNN technique.

Therefore, constraints contain some elements estimated by the DNN model. Then, the optimization problem is developed as Eq. (4.13):

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & g(F_{NN}(x), x) \\ & \text{other constraints} \end{aligned} \tag{4.13}$$

Also, the cost function is the combination of the element predicted by the NN model. So, Eq. (4.13) is rewritten as:

$$\begin{aligned} \min_{x} \quad & f(F_{NN}(x), x) \\ \text{s.t.} \quad & g(F_{NN}(x), x) \\ & \text{other constraints} \end{aligned} \tag{4.14}$$

While F_{NN} is the ESL-MDN model introduced in the previous section and it maps the inputs and intermediate features to the process outputs.

Designing an optimizer for practical application is very challenging as we can face one of the following issues [56]:1) multimodality, 2) non-differentiable objective function or constraints, 3) non-convex problem, 4) mixed variables, 5) application with large dimension, 6) multi-objectives. Although gradient-based optimization is a powerful tool to find local minima, they cannot deal with multi-modal and non-convex problems. As a result, gradient-free algorithms are introduced to address this issue.

Differential Evolution

Differential evolution (DE) which is introduced by [57], is a population-based metaheuristic search algorithm developed to optimize problems over the continuous domain. Making few or no assumptions about the underlying problem, simple structure, robustness and speed candidates DE as a powerful tool for optimization. Better solutions have been derived by the search over the design space in a stochastic way. The summary of DE has been shown in Algorithm (2) [58].

DE is in a category of the gradient-free method, which makes it a perfect tool for the problem formulated in Eq (4.36) as the objective function and some constraints

Generate the initial population of individuals of size $NP \geq 4$
Initialize DE parameters including CR and F
while *i in the population* **do**
  Pick three random integers $a, b, c \in (1, NP)$
  $a \neq b \neq c \neq i$
  Pick a random integer $j_{int} \in \{1, n\}$
  for *each parameter j* **do**
    $x'_{j,i} =$
    $\begin{cases} \mathrm{x}_{j,c} + F(x_{j,a} - x_{j,b}), \\ \text{if } rand(0,1) < CR \text{ or } j = j_{int} \\ \mathrm{x}_{j,i} \qquad \text{otherwise} \end{cases}$
  end
  if $f(x'_i) < f(x_i)$ **then**
    Replace the agent with the improved solution
  end
end

are the outcome of a neural network and the differentiability of the problem is not the requirement. DE generates new solutions by combining the existing ones according to the mutation process, and it keeps the population of candidate solutions while searching the design space. In the next iteration, candidates with the best objective value are considered new solutions. The process iterates until it satisfies the termination criterion. Therefore, the detailed strategy is as follows:

According to Algorithm. (2), we select a population of D dimensional population vector with NP population size. Each target vector for iteration j indicates as $(x_{j,i})$ for $i \in 1, 2, ..., NP$ and $x_{j,i}$ includes all parameters must be optimized. The donor vector, $v_{j,i}$ is built by picking three random vectors as follows:

$$v_{j,i} = x_{j,c} + F(x_{j,a} - x_{j,b}) \tag{4.15}$$

F is a vector called mutation factor, and it is a random number obtained from a uniform distribution in a range of $[0, 2]$. The random integer numbers a, b, c must be different from i. The next step is to define a trial vector using the crossover, a

random number in the range of $[0, 1]$. The operation of crossover is as follows:

$$x'_{j,i} = \begin{cases} x_{j,c} + F(x_{j,a} - x_{j,b}), & \text{if rand(0,1)} \leq \text{CR or } j = j_{int} \\ x_{j,i} & \text{otherwise} \end{cases} \tag{4.16}$$

Then, the selection process finds a better solution by minimizing the problem. The loss function for the problem denoted by f. As a result, if $f(x'_{j,i}) \leq f(x_{j,i})$, target vector will be replaced by trial vector as it improves the solution by minimizing the loss function.

4.3 Experiment

4.3.1 Reverse Osmosis Water Desalination Plant

Water scarcity, which results from a mismatch between water resources and water demands, is not limited to arid areas. The water quality has decreased in many regions due to the pollution and exploitation of groundwater and surface water. Furthermore, the growth of the population leads to an increase in water demand for industry and agriculture. As a result, exploring an alternative resource for freshwater has gained many interests. Desalination is the answer to this, such as alternative resources. Desalination systems are divided into two main groups based on the separation process. Thermal desalination exploits evaporation and condensation to separate solid material from water. The other separation process is the membrane desalination technique. RO is the most popular technique in which water diffuses through the membrane, and it retrains solid particles. Fig. 4.6 demonstrates the process of water desalination using RO, which includes three stages, such as pre-treatment, RO system and post-treatment.

A mathematical description given by [59] is widely accepted and most used by RO system designer. the water flux J_ω is as follows:

$$J_\omega = A_w(\Delta P - \Delta\pi) \tag{4.17}$$

A_w is the solvent permeability coefficient which depends on the temperature and

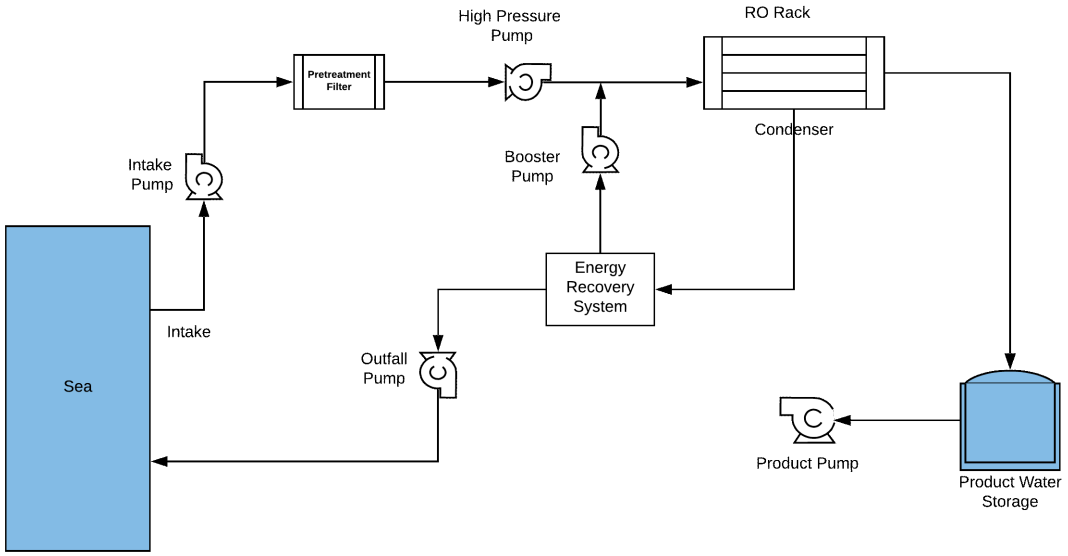


Figure 4.6: Structure of RO plant contains three sections, pre-treatment, RO desalination system and post-treatment.

membrane's features. ΔP is pressure drop and $\Delta\pi$ is the osmotic pressure difference across the membrane element. Solute flux can be modeled as:

$$J_s = A_s(X_m - X_p) \tag{4.18}$$

Where A_s is solute permeability coefficient which depends on the temperature too. As a result, permeate flow and concentration can be derived by:

$$X_p = \frac{J_s}{J_\omega} \tag{4.19}$$

$$Q_p = SJ_\omega \tag{4.20}$$

Using above equation, we can rewrite the permeate flow as:

$$Q_p = SA_w(\Delta P - \Delta\pi) \tag{4.21}$$

So, water permeability can be derived by:

$$A_w = \frac{Q_p}{S\,(\Delta P - \Delta\pi)} \tag{4.22}$$

Also, the volume and mass balance relationship around the membrane layers is estimated as follows:

$$Q_f = Q_p + Q_b \tag{4.23}$$

$$Q_f X_f = Q_p X_p + Q_b X_b \tag{4.24}$$

Water recovery is the measure to evaluate the process performance and it can be written as follow:

$$R = \frac{Q_p}{Q_f} \times 100 \tag{4.25}$$

Rest of the equations expressed in Table 4.1 are suggested by Filmtec company [60].

4.3.2 Preprocessing

We propose a two-stage DNN model to estimate the process output. Data pre-processing is a crucial step to achieving a high-quality model. Significant steps include process shutdown detection, outlier removal, and noise reduction, have been taken into account to clean the data. Scheduled maintenance and abrupt malfunction in the system are two primary reasons for the plant's shutdown. However, data acquisition systems run continuously and collect the data during this time. Shutdown mode is detected using feed-water flow profile when it is below a specific number and 10 m^3/hr is selected as a threshold.

The primary step of pre-processing for industrial applications is outlier removal. Data collected from an actual plant contains normal and abnormal samples. However, the absence of labels for anomalies makes the data cleaning section the most challenging part. Our mission is to clean data and detect outliers in an unsupervised way. We designed the Outlier Removal (OR) based on the double rolling aggregate. Comparing each sample with its previous value is the core idea of this technique to detect an abrupt change that is not consistent with the sequence behaviour. In contrast to sensor surges which may occur abruptly and disappear, sensor malfunction can stay for a while and transfer wrong information. Three-stages OR has been introduced

Table 4.1: Mathematical relationship for seawater RO plant suggested by Filmtec company

Description	**Equation**
Permeate Flow	$Q_p = SA_w\,(TCF)\,(FF)\left(P_f - \frac{\Delta P_{fb}}{2}P_p - \pi_f\right)\left(\frac{X_{fb}}{X_f}P_f - (1-SR)\right)$
Average concentration-side osmotic pressure	$\bar{\pi} = \pi_f\left(\frac{X_{fb}}{X_f}PF\right)$
Average concentration-side concentration	$\mathrm{X}_{fb} = \frac{X_b + X_f}{2}$
Brine Concentration	$\mathrm{X}_b = \left(\frac{1-R(1-SR)}{1-R}\right)X_f$
Feed-water osmotic pressure	$\pi_f = 1.12\,(273+T)\sum m_j$
Temperature correction factor	$TCF = e^{\left(\frac{2640}{298} - \frac{1}{273+T}\right)} \quad T \geq 25^0C$
	$TCF = e^{\left(\frac{3020}{298} - \frac{1}{273+T}\right)} \quad T \leq 25^0C$
Polarization factor	$\mathrm{PF} = \mathrm{e}^{0.7R}$
conductivity-side pressure drop	$\Delta P_{fb} = 0.01\left(\frac{Q_f + Q_b}{2}\right)$
membrane permeability	$A_w = 0.125 \quad \bar{\pi} \leq 25$
	$A_w = 0.125 - 0.011\left(\frac{\bar{\pi} - 25}{35}\right) \quad 25 \leq \bar{\pi} \leq 200$
	$A_w = 0.070 - 0.0001\,(\bar{\pi} - 200) \quad 200 \leq \bar{\pi} \leq 400$
Permeate Concentration	$\mathrm{X}_p = A_s X_{fb} PF\,(TCF)\left(\frac{NS}{Q_p}\right)$

Table 4.2: Nomenclature

Symbol	Description
J_ω	Water flux m/s
A_w	Water permeability $m/bar.s$
J_s	Solute flux m/s
A_s	Salt permeability m/s
X	Concentration ppm
Q	Flow Rate m^3/hr
C	Conductivity S/cm
P	Pressure kpa
π	Osmotic Pressure Kpa
ΔP	Pressure drop Kpa
R	Recovery
S	Area m^2
SR	Salt Rejection
PF	Polarization Factor
FF	Fouling Factor
ΔP_{fb}	Average concentration-side Pressure Drop
m_j	Molar Concentration of jth ion

to address this issue, which compares each instance with the previous one, 24 and 500 moving average samples. OR eliminates those instances if there is a difference more than 5%, 20% and 40% with the previous, 24 and 500 moving average samples, respectively. Table. 4.3 shows the percentage of samples eliminated in each stage.

Plant	Total Samples	Stage One	Stage Two	Stage Three
A	12859478	4.88%	1.29%	0.08%
B	9856732	2.94%	0.86%	0.015%

Table 4.3: Percentage of samples eliminated in each stage

As a feature engineering step, SCADA data is plugged into the equations in Table 4.1 to calculate intermediate states. However, feature selection is of great importance to keep the dimensionality of training data as low as possible. Two intermediate states, average concentration salt conductivity and polarization factor demonstrate almost constant behaviour. Due to the variable fluctuations, we can remove them from the augmented training data as they cannot contribute to the necessary information for the training session. Therefore, the temperature correlation factor (TCF) is the intermediate variable calculated using the feedwater temperature profile. Also, water permeability and feed osmotic pressure can be considered the second type of intermediate variables estimated by the ESL-MDN framework.

Using Table. 4.1, TCF is calculated by feed water temperature. Also, training labels for feed osmotic pressure and water permeability are generated using the first half of the dataset. Fig. 4.7 and 4.8 provide the estimation of the water permeability and feed osmotic pressure with their variance.

Description of the hybrid model has been brought in Table 4.4. ESL-MDN applies to the augmented training data at stage two to predict the main target variables of the RO plant. As it is shown in Table 4.4, six variables (feed-water flow, concentration valve, feed-water temperature, feed-water conductivity, feed-water pressure and differential pressure) incorporated with intermediate states (feed-water osmotic pres-

Figure 4.7: Predicted water permeability and its variance

sure, water permeability, TCF) are considered as inputs and permeate concentration, permeate pressure, and permeate flow are defined as outputs. Also, as mentioned earlier, two variables, average concentration side conductivity and polarization factor, are eliminated from the augmented data during the feature selection step.

4.3.3 Cost Function for optimal Operation

The objective of the optimizer is to calculate the optimal set-points profile that minimizes the energy consumption in the RO desalination process. Specific energy consumption (SEC), in kWh per m^3, is an essential element to evaluate the performance of the RO process. Many factors from different parts of the plant contribute to the SEC. For example, 1) facility provides feed-water, 2) Pre-treatment and filtering, 3) the main section of the process such as high-pressure pump, membrane and energy recovery device, 4) post-treatment and chemical adjustment, 5) disposal facility.

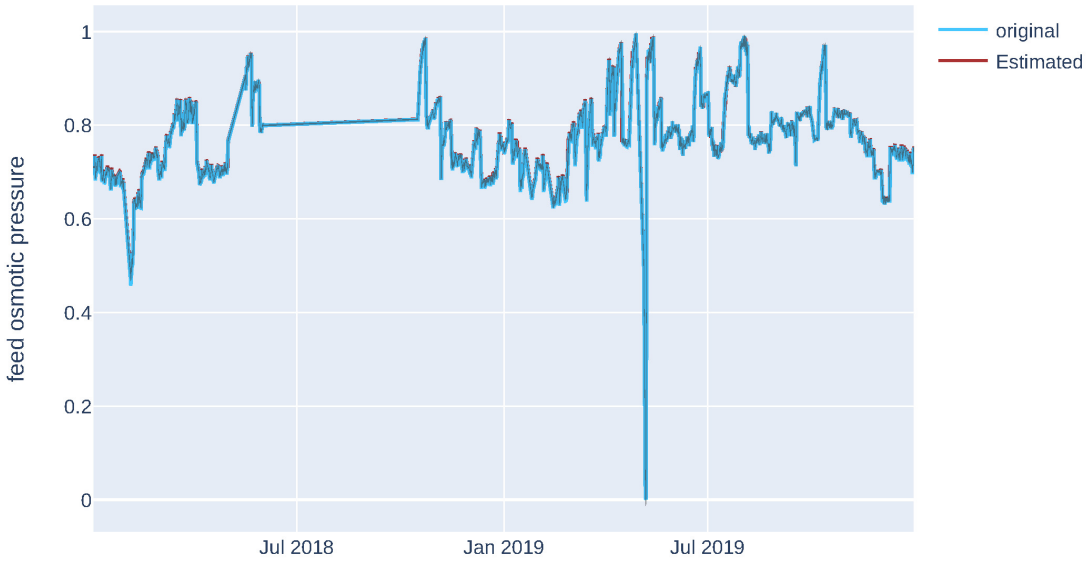


Figure 4.8: Predicted feed osmotic pressure and its variance

However, studies show that the main section is responsible for 60-80% of the total SEC of the desalination system. We focus on the pumps and their contribution to SEC in this work. A general expression for SEC is as follows [61]:

$$SEC = \frac{1}{R}\left[\frac{P_f - P_p}{\eta}\right] + (1-\beta)\left[\frac{1-R}{R}\right]\left[\frac{P_p + \eta_{ERD}\Delta P - \eta_{ERD}P_f}{\eta}\right] \tag{4.26}$$

Where η is pump efficiency, η_{ERD} and β are the efficiency and leakage ratio of energy recovery device, respectively. Equ. (4.27) indicates the value of ideal operation:

$$\eta = \eta_{ERD} = 1 \quad \beta = 0 \tag{4.27}$$

Hence, SEC for ideal process can be rewritten as:

$$SEC_{ID} = \left[\frac{1-R}{R}\right]\Delta P + (P_f - P_p) \tag{4.28}$$

Therefor, inefficient pump and ERD can lead to energy loss, which can be calculated as:

$$j = |SEC - SEC_{ID}| \tag{4.29}$$

4.3.4 Process Constraints

Customer's Demand

In order to provide the customer with fresh water at all times, we consider the daily customer demand profile, which is suggested by [62]. Fig. 4.9 indicates the daily discharge flow rate requested by the customer.

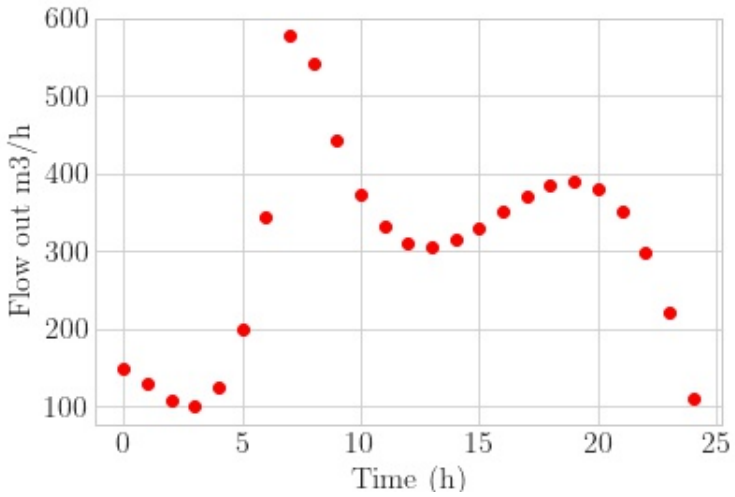


Figure 4.9: Daily fresh water demand

Storage Tank

Product water storage is considered at the last stage of the RO process to reserve water during the low demand and provide more water when the demand is high. Also, for safety reasons, tank level H should not be overflowed. As a result, we define the tank's high and low constraints $(high, low)$. Therefore, the violation can be formulated as:

$$inlet\ flow_t - outlet\ flow_t = A \times h_t \tag{4.30}$$

Where A an h are the area and height of the tank, respectively. So, the total height of the tank after T hours is calculated as:

$$h_T = h_0 + \sum_{t=0}^{T} h_i \tag{4.31}$$

where h_0 is the initial height of the storage tank. so the constraint can be written as:

$$h_{low} < h_T < h_{high} \tag{4.32}$$

Product Water Quality

In addition to tank level, water quality is another constraint defined by the customer. In general, permeate water concentration is a measure to evaluate water quality, and there is an acceptable range for different purposes. These constraints can be written as:

$$\alpha_l < X_p < \alpha_h \tag{4.33}$$

Furthermore, physical limitations such as the size of valves and pumps add more constraints. Physical constraints are as follow: Linear bound on P_f,Q_f and Q_b. Finally, safety constraints are considered to protect the devices installed in the plant. This constraint adds more limitations on sudden change for set points. In other words, the difference between two consecutive set-points must not exceed the predefined threshold. The summary of the optimizer structure is demonstrated in Fig. 4.10.

Therefore, the optimization problem can be rewritten as:

$$\begin{aligned} &\min_{Q_f, P_f, Q_b} \quad \text{j} \\ &Q_{f_{low}} < Q_f < Q_{f_{high}} \\ &P_{f_{low}} < P_f < P_{f_{high}} \\ &Q_{b_{low}} < Q_b < Q_{b_{high}} \\ &h_{low} < h_T < h_{high} \end{aligned} \tag{4.34}$$

However, water characteristics such as feed temperature and conductivity profile must be provided to find the set-point profile for one day ahead. Also, the total

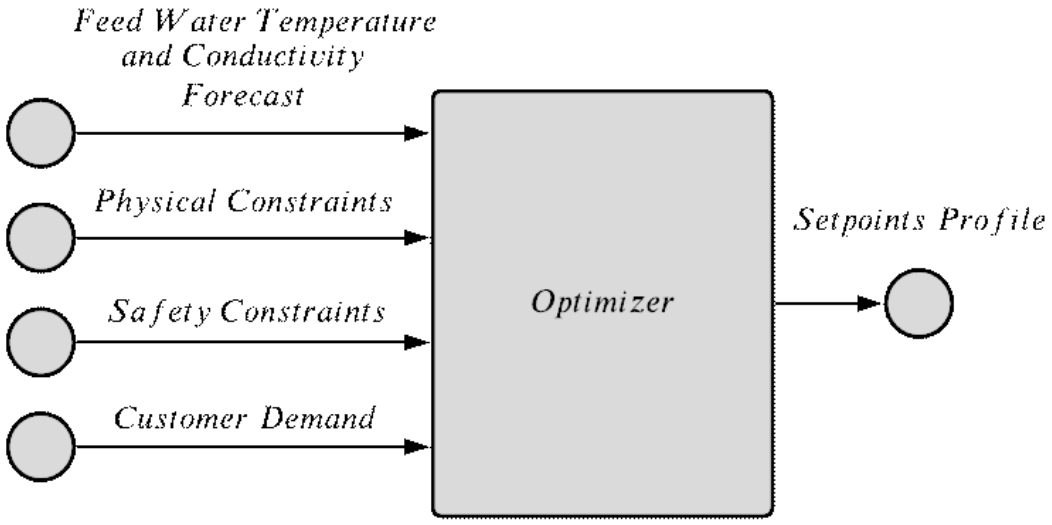


Figure 4.10: Structure of the optimizer and all constraints related to the process

SEC is equal to the aggregate of the predicted SEC for the next 24 hours. So, the optimization problem can be rewritten as:

$$\begin{aligned} \min_{Q_{f_t}, P_{f_t}} \quad & \sum_{t=1}^{i} (j_t) \\ Q_{f_{low}} < Q_f < Q_{f_{high}} \\ P_{f_{low}} < P_f < P_{f_{high}} \\ h_{low} < h_T < h_{high} \\ \alpha_{min} \leq X_{p_t} \leq \alpha_{max} \end{aligned} \tag{4.35}$$

Additionally, we must add a water demands quality constraint estimated by the ESL-MDN model to meet the customer requirement. SEC is the function of feed and permeate pressure. Furthermore, permeate pressure can be substituted by the ESL-MDN model too. Also, X_p is the outcome of the mentioned neural network model. As the predicting SEC depends on the forecasting uncertainty, we use the forgetting factor γ_t, to give the highest weight for the present data and decrease over time. Therefore, the optimization problem is formulated as:

$$
\begin{aligned}
&\min_{Q_{f_t}, P_{f_t}, Q_{b_t}} \quad \sum_{t=1}^{T} (\gamma_t j_t) \\
&Q_{f_{low}} < Q_f < Q_{f_{high}} \\
&P_{f_{low}} < P_f < P_{f_{high}} \\
&Q_{b_{low}} < Q_b < Q_{b_{high}} \\
&h_{low} < h_T < h_{high} \\
&\alpha_{min} \leq X_{p_t} \leq \alpha_{max}
\end{aligned}
\tag{4.36}
$$

4.4 Presentation of the Experiment Analysis

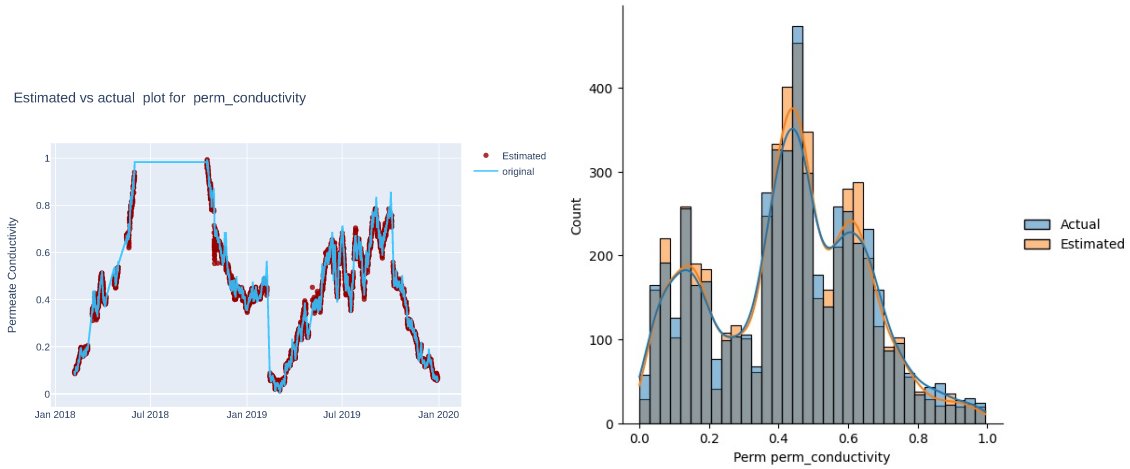
Six process variables, including feed-water flow, feed-water temperature, feed-water conductivity, differential pressure, feed-water pressure, and concentration water valve position, have been chosen to conduct a feature engineering and obtain non-measurable parameters such as feed-water osmotic pressure, water permeability, salt rejection, recovery and average concentration side conductivity. The augmented training data predicts the outlet variables, including permeate flow, permeate conductivity and permeate pressure, directly impacting SEC. The DE-based optimizer has been designed to find the profile of set-points and minimize the SEC loss over a day.

4.4.1 Performance Evaluation of Neural Network Model

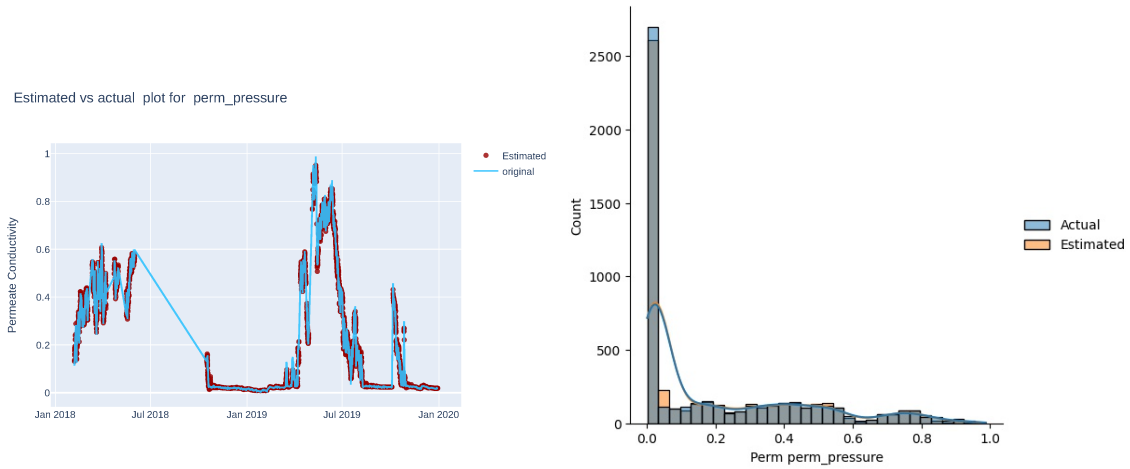
An evaluation has been done on experimental data to demonstrate the model's effectiveness. Data was collected from two different anonymous sites placed in Asia. The description of the neural network models is shown in Table 4.9.

PINN Hyper Parameters

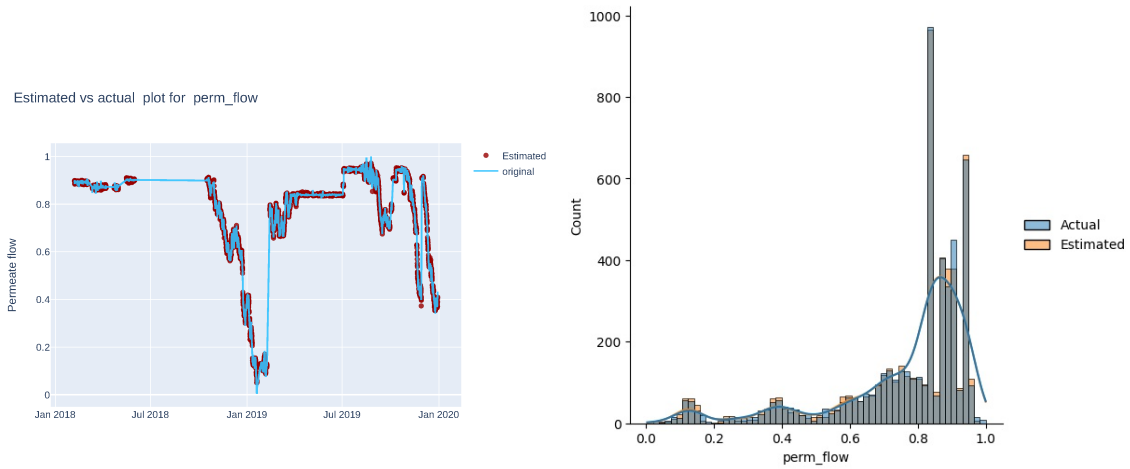
In this work, the PINN model represents the output variable of the RO process. k-fold validation is used to tune the hyperparameters such as the number of neurons in the



(a) Permeate Conductivity

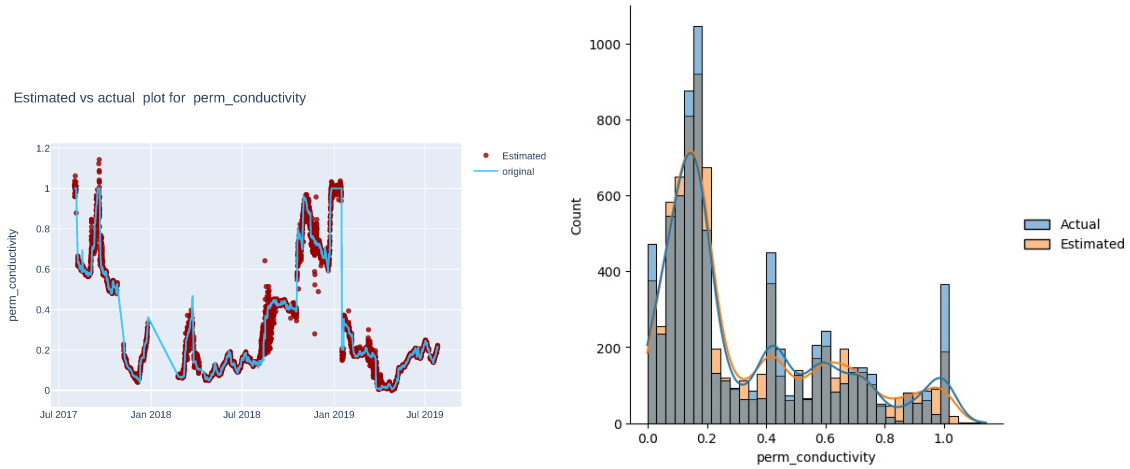


(b) Permeate Pressure

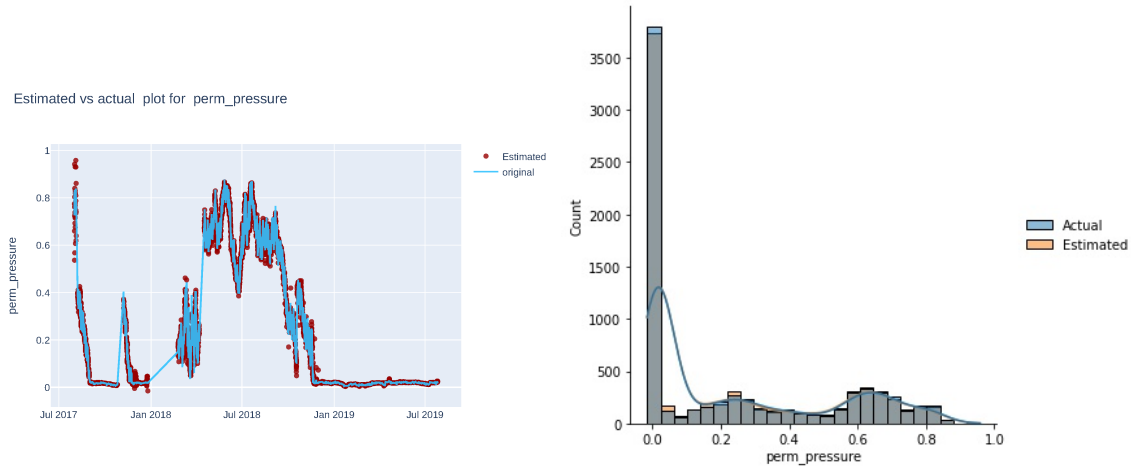


(c) Permeate Flow

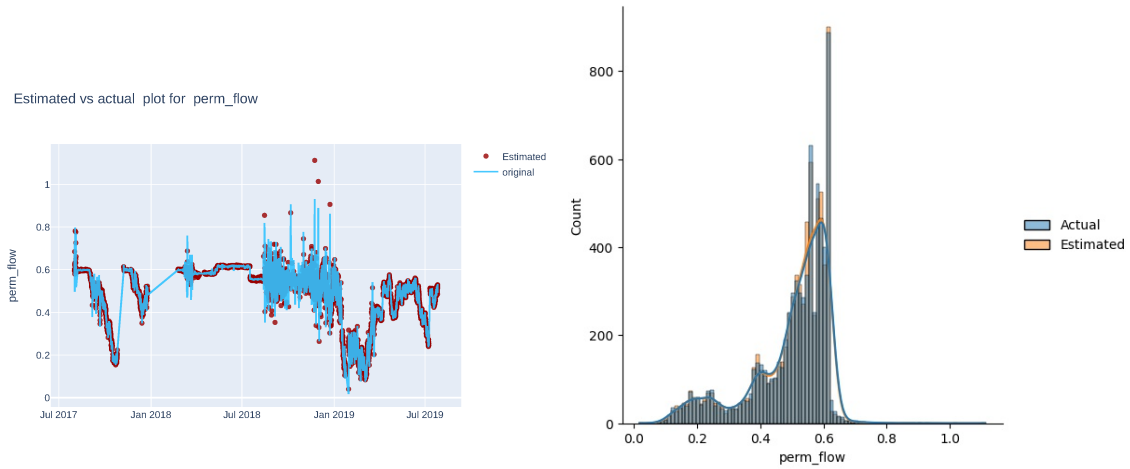
Figure 4.11: Left figures, prediction vs actual values of the outlet variables, right figures: Distribution plot for Plant A



(a) Permeate Conductivity



(b) Permeate Pressure



(c) Permeate Flow

Figure 4.12: Left figures, prediction vs actual values of the outlet variables, right figures: Distribution plot for Plant B

hidden layer, activation function in the hidden layer and output layer, optimizer type and learning rate. Table 4.9 describes the best values for the network's hyperparameters. Although parameter tuning has found similar values for most hyper-parameters, the number of mixture distributions differs for each parameter. The main reason for that is the non-stationary nature of the process, and some parameters need more mixture distribution to have a high estimation accuracy. The number of mixture distributions has been indicated in Table. 4.6.

We considered different activation functions such as linear, Sigmoid, Relu, LeakyRelu and tangent hyperbolic. The linear activation function has a constant derivative for the input. It means that the change caused by backpropagation is independent of estimation error. If we have DNN with multiple hidden layers and a linear function is chosen for all layers, the target is just a linear function of the input. As we know that the Ro is a complex and nonlinear process, the linear activation function is removed from the potential candidate. The rest of the activation functions, even Relu, are nonlinear and suitable for nonlinear regression. Relu is a proper activation function when there are numerous training data available. Sigmoid and tangent hyperbolic activate almost all neurons in the DNN model. So, for the DNN structure, Relu can be a suitable candidate by making the activation sparse and efficient. However, leakyRelu has a wider range of learning because of solving the dying Relu problem that happens due to the appearance of non-responding neurons to the change in input. However, it has more computation cost compared to Relu. Therefore, we used K-fold validation to find the proper activation function for each target. Table. 4.7 summarises the final decision for activation function found by K-fold validation.

Impact of uncertainty on modelling

The threshold that is employed by determinsitic dropout is computed as follows:

$$\delta = \nu * mean(IW) \tag{4.37}$$

Table 4.8 indicates the best value for ν obtained from parameter tuning. multiple tests reveal that incorporating water permeability and feed osmotic pressure with the PINN model improves permeate pressure, conductivity, and flow accuracy. Wider interval means that the model has less confidence regarding the prediction result. If the IW violates the threshold, that sample will be eliminated from the augmented training data as the predictor has low confidence. However, choosing a small value for an acceptable range eliminates a huge part of data, leading to lower accuracy. In contrast, a large value for ν can cause error propagates through the modelling and decrease the overall accuracy.

Modelling Results

Fig. 4.11 and 4.12 show the prediction results and distribution plot for both plants A, and B. PINN model outperforms other conventional DL algorithms as the mean percentage error (MPE) for all variables lies in an acceptable range.

Comprehensive studies have been done to show the effectiveness of the proposed method. First, it has shown that the hybrid model enhanced the accuracy compared to conventional and deep learning techniques. Second The proposed method shows remarkable improvement. The main reason is that the mathematical model includes the steady-state of the system and shows a weak performance during the transition states. The hybrid model can capture both steady and transition states as it incorporates the features of both model-based and data-driven techniques. However, as the mean percentage error gives us general information regarding the performance, we introduce another measure: the percentage of the samples with an error over 10%. Adding deterministic dropout reduces the number of samples with significant error. It is based on the fact that it eliminates estimated samples with significant predicted errors and does not use them as a feature for second learners.

4.4.2 Performance Analysis of the Optimizer

Two scenarios are investigated based on the feed-water characteristic prediction in which the customer demand profile remains fixed. In the first scenario, the optimizer does not have access to feed-water prediction, and it finds the set-points based on the present data. However, the optimizer is provided by dynamic water profile results of the next 24 hours in the second scenario.

Scenario 1: feed-water characteristics prediction NOT available

The optimizer finds the set-points using the present data without future feed-water variation information. Table. 4.13-f illustrates the SEC loss in the presence of the optimizer and compares it with the actual loss. In this scenario, the plant has been given an optimal set-point every 24 hours. It reveals that the SEC loss is reduced using the optimizer design. In the following scenario, the impact of future knowledge will be investigated.

Scenario 2: feed-water characteristics prediction provided

by Designing this scenario, we intend to provide the plant with the profile of optimal set-points, which also meet all requirements and constraints. Feed-water variation for the next 24 hours is provided, and the optimizer finds the profile of set-points every 6 hours. The total energy loss reduction is lower than the actual loss without an optimizer, demonstrating that the optimizer associated with the water variation predictor has better performance to save energy. Fig. 4.14c shows the comparison of SEC loss for both plants using the second scenario.

SEC loss comparison is indicated in Table 4.11. It reveals that adding the proposed optimizer improved the energy loss compared to the conventional optimization technique used in the actual plant. Furthermore, consider future feed water characteristics and manage the water production accordingly, reducing energy loss in the plant.

However, the quality of permeate water must be in a predefined range. The threshold of water conductivity's acceptance is 800ppm. Fig. 4.14a shows that the conductivity of freshwater provided by the optimizer is lower than the threshold, which is the indicator of higher quality compared with the case without the optimizer.

Fig. 4.14b depicts the storage tank level over a cycle of 24 hours. The tank store more water during the first 6 hours of a day due to the lower demand. In contrast, the tank level is shrinking during peak hours to provide the user with fresh water at all times. It is assumed that the tank's level should not reach the bottom 4 and top 18 meters.

The presented work shows the ability of the optimizer to provide the optimal operation while satisfying operation and physical constraints.

set-points and process output are brought in Fig. 4.13 for the second scenario. Although the optimizer introduces more fluctuations than the scenario with no suggested optimizer, all variables lie in the normal range, and due to the safety constraints, there is no abrupt and unacceptable spike.

Table 4.4: Description of features using in learning stage one and two

Stage	Input	Calculation	Estimation
One	Feed Flow	Polarization Factor	Water Permeability
	Concentration Valve	Average Concentration Side Conductivity	Feed Osmotic Pressure
	Feed Temperature	Temperature Correction Factor	
	Feed conductivity		
	Feed Pressure		
	Differential Pressure		
Two	Feed Flow		Permeate Pressure
	Concentration Valve		Permeate Conductivity
	Feed Temperature		Permeate Flow
	Feed Conductivity		
	Feed Pressure		
	Differential Pressure		
	Water Permeability		
	Feed Osmotic Pressure		

Table 4.5: Description of PINN sequential model for outlet variables

Network parameters	**Values**
Number of neurons in the input layer	9
Number of hidden layers	2
Number of neurons in the hidden layer	256-128
Number of neurons in the output layer	1
Loss function	MAE
Optimizer type	Adam
Maximum number of epoch	1000
Batch size	50
Learning rate	0.001
Dropout	10

Table 4.6: Number of mixture density distribution for each target

Plants	**Variables**	Number of Mixture
A	Flow	2
	Pressure	2
	Conductivity	3
B	Flow	2
	Pressure	2
	Conductivity	3

Table 4.7: Activation function for each target

Plants	**Variables**	Activation Function
A	Flow	Relu
	Pressure	Relu
	Conductivity	tangent hyperbolic
B	Flow	Relu
	Pressure	Relu
	Conductivity	tangent hyperbolic

Table 4.8: Best value for ν obtained during parameters tuning process

Plants	**Targets**	Feed Osmotic Pressure	Water Permeability
A	Flow	4	5
	Pressure	5	5
	Conductivity	5	5
B	Flow	4.5	5
	Pressure	5	5
	Conductivity	4.5	5

Table 4.9: Mean percentage error comparison of various ML technique and proposed method

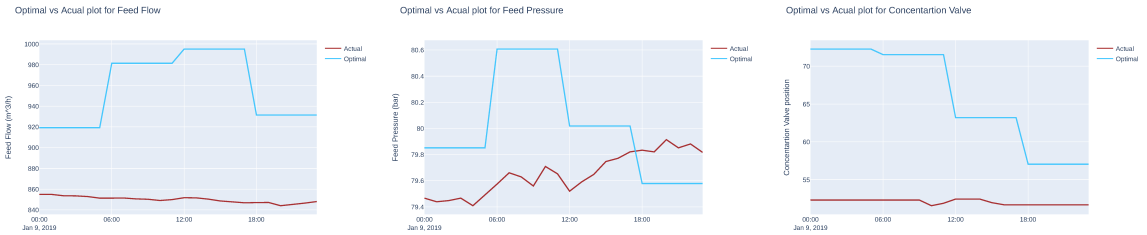
Plants	**Variables**	Linear Regression	MLP	ARIMA	AE
A	Flow	5.3621	3.9452	4.2147	3.0249
	Pressure	12.3598	8.2219	9.2984	6.9436
	Conductivity	13.2641	11.0846	9.2159	10.8436
B	Flow	6.1637	5.9543	5.2374	4.2179
	Pressure	11.2158	7.7624	8.08468	7.1683
	Conductivity	16.4158	10.9462	11.2895	9.2541
Plants	**Variables**	ELM	LSTM	AELSTM	Proposed Method
A	Flow	3.8964	5.2194	4.8206	0.9728
	Pressure	7.8546	8.2597	6.9607	5.66748
	Conductivity	11.9216	8.1598	8.0913	2.4753
B	Flow	4.1039	3.4698	2.6548	0.8834
	Pressure	7.0036	6.5608	6.1975	5.1976
	Conductivity	8.2963	8.0984	7.2083	6.0159

Table 4.10: Percentage of large error (over 10%) comparison with and without deterministic dropout

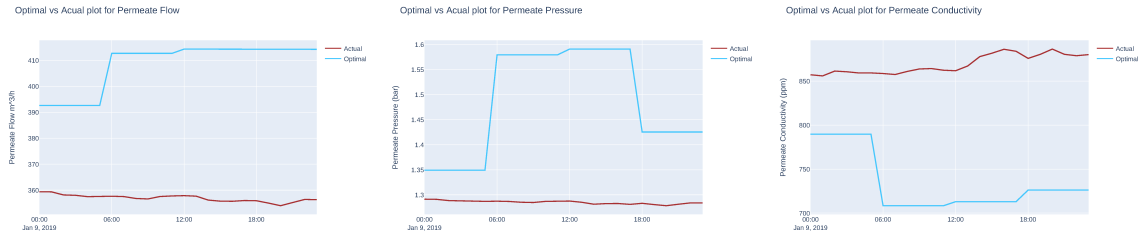
Plants	**Variables**	without dropout	with dropout
A	Flow	1.27	1.09
	Pressure	9.43	2.27
	Conductivity	10.84	3.47
B	Flow	1.02	0.98
	Pressure	6.28	1.27
	Conductivity	7.91	3.22

Table 4.11: SEC loss (kWh) comparison of three scenarios (Without proposed optimizer, scenario one and scenario two)

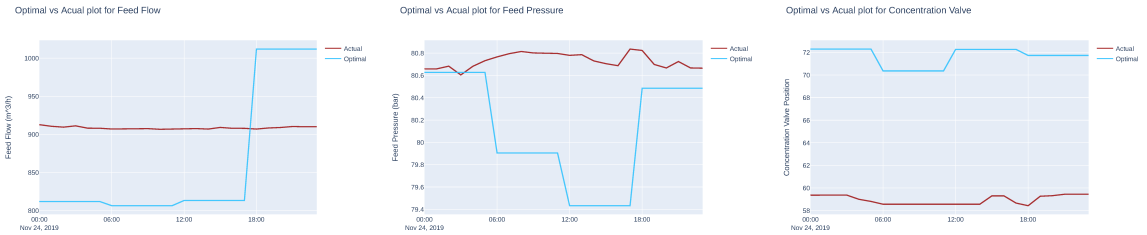
Plants	No Optimizer	Scenario One	Scenario Two
A	7.21	6.46	6.22
B	7.38	6.95	6.78



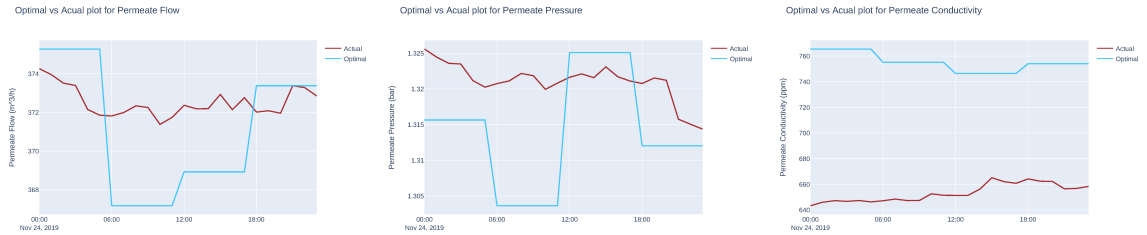
(a) set-points with and without optimizer Plant A



(b) Process outlet variables with and without optimizer Plant A

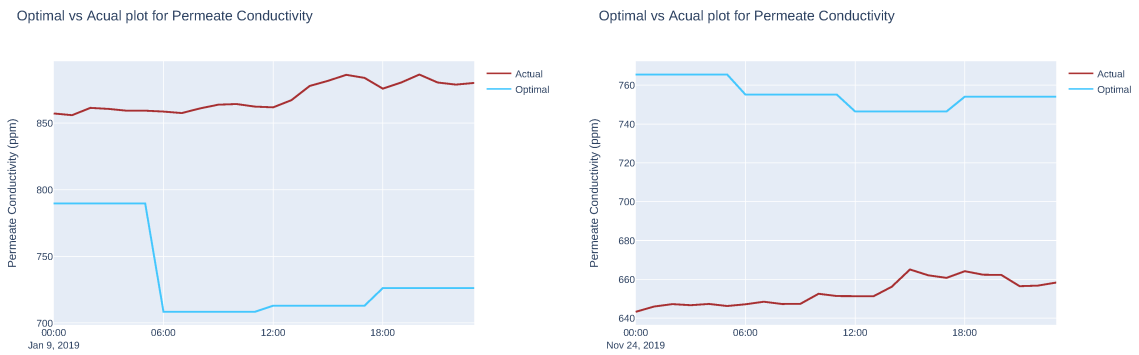


(c) set-points with and without optimizer Plant B

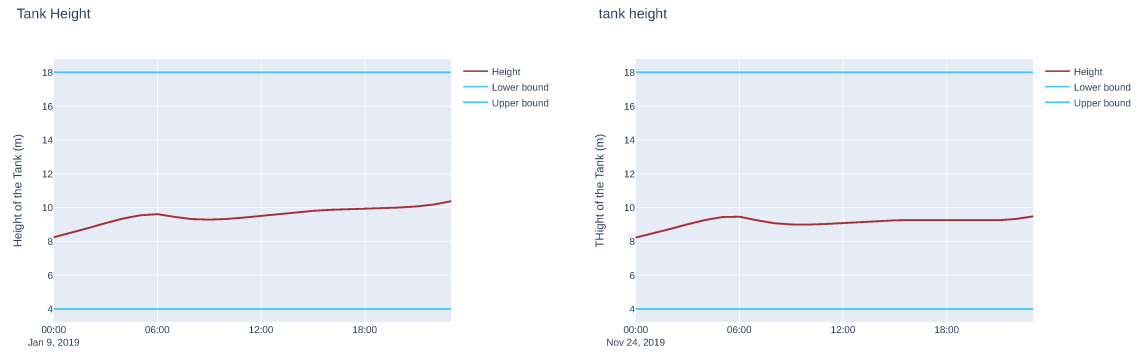


(d) Process outlet variables with and without optimizer Plant B

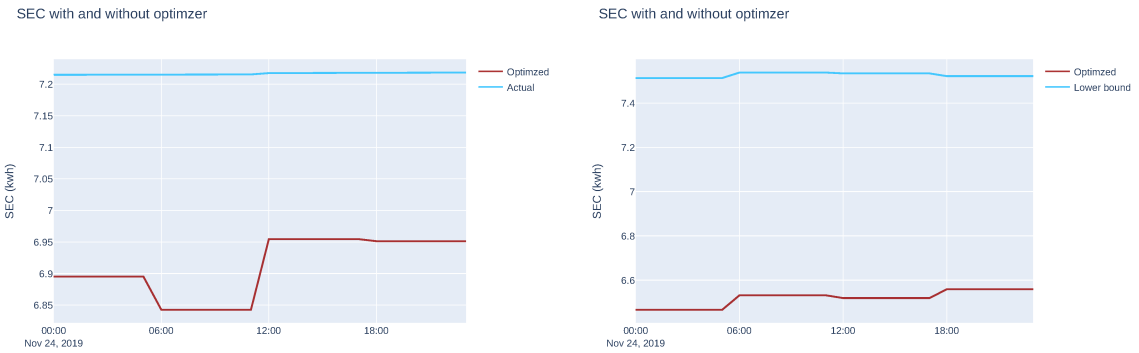
Figure 4.13: Comparison of the process set-points and output variables with and without optimizer for plants A and B in second scenario



(a) Conductivity comparison for plant A (left) and B (right) with and without optimizer



(b) Tank level in the presence of optimizer for plant A (left) and B (right). The blue line indicate the upper and lower threshold



(c) SEC loss comparison for plant A (left) and B (right)

Figure 4.14: Optimal result for second scenario in comparison with the scenario without proposed optimizer

Chapter 5

Conclusions & Future Work

This thesis mainly focuses on three promising applications of AI for industrial applications: fault diagnosis, modelling and optimization. We presented an LSTM-MDN model integrated with ARIMAX system identification for component early fault detection based on simulated data from wind turbines in Chapter 3. First, the LSTM-MDN network model was trained using NBM data. Then, the prediction error, adaptive interval, and residuals were used for fault detection. According to the actual analysis results, the following conclusions were reached: It has been confirmed that the developed framework for early fault detection of wind turbine pitch and blade components is valid based on analysis results generated by the proposed model, which agree with the actual fault scenario generated by the simulator. Additionally, once the LSTM-MDN model has detected a possible fault, pruning rules investigate the situation to ensure that the candidate fault is real and keep the FAR as low as possible. Chapter 4 presented an ESL-MDN with an infused DK framework that enhanced model performance, which is critical for the plant optimizer. Also, to avoid the induced error caused by incorporating Dk into DNN, we introduced a deterministic dropout mechanism based on the uncertainty estimation. Finally, we designed an optimizer based on the DE algorithm to detect the optimal operating set-point of the RO process while satisfying all physical and customer constraints. Also, we investigate the impact of adding the information of feed-water characteristics to

manage energy loss. The results revealed that integrating the ESL-MDN model with physics improved the modelling accuracy. Also, a new design optimizer can reduce the energy loss caused by pumping energy. The results revealed that the suggested framework could reduce energy loss while meeting customer demand. The primary step for future work is to design an optimizer that considers uncertainty estimated by the model. It is important to note that the framework applied to the raw data collected from real plants and the optimizer has been customized. However, coming up with a more general approach applicable to the different industrial processes would be considered another aim for future work.

Bibliography

[1] G. S. Galloway, V. M. Catterson, T. Fay, A. Robb, and C. Love, "Diagnosis of tidal turbine vibration data through deep neural networks," 2016.

[2] Z. Huijie, R. Ting, W. Xinqing, Z. You, and F. Husheng, "Fault diagnosis of hydraulic pump based on stacked autoencoders," in *2015 12th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, IEEE, vol. 1, 2015, pp. 58–62.

[3] J. Yu and X. Yan, "A new deep model based on the stacked autoencoder with intensified iterative learning style for industrial fault detection," *Process Safety and Environmental Protection*, vol. 153, pp. 47–59, 2021.

[4] Z. Chen, C. Li, and R.-V. Sánchez, "Multi-layer neural network with deep belief network for gearbox fault diagnosis," *Journal of Vibroengineering*, vol. 17, no. 5, pp. 2379–2392, 2015.

[5] Y. Wang, Z. Pan, X. Yuan, C. Yang, and W. Gui, "A novel deep learning based fault diagnosis approach for chemical process with extended deep belief network," *ISA transactions*, vol. 96, pp. 457–467, 2020.

[6] Z. Chen, A. Mauricio, W. Li, and K. Gryllias, "A deep learning method for bearing fault diagnosis based on cyclic spectral coherence and convolutional neural networks," *Mechanical Systems and Signal Processing*, vol. 140, p. 106 683, 2020.

[7] M. Azamfar, J. Singh, I. Bravo-Imaz, and J. Lee, "Multisensor data fusion for gearbox fault diagnosis using 2-d convolutional neural network and motor current signature analysis," *Mechanical Systems and Signal Processing*, vol. 144, p. 106 861, 2020.

[8] M. Jalayer, C. Orsenigo, and C. Vercellis, "Fault detection and diagnosis for rotating machinery: A model based on convolutional lstm, fast fourier and continuous wavelet transforms," *Computers in Industry*, vol. 125, p. 103 378, 2021.

[9] A. Mallak and M. Fathi, "Sensor and component fault detection and diagnosis for hydraulic machinery integrating lstm autoencoder detector and diagnostic classifiers," *Sensors*, vol. 21, no. 2, p. 433, 2021.

[10] S. Belagoune, N. Bali, A. Bakdi, B. Baadji, and K. Atif, "Deep learning through lstm classification and regression for transmission line fault detection, diagnosis and location in large-scale multi-machine power systems," *Measurement*, vol. 177, p. 109 330, 2021.

[11] N Bhutani, G. Rangaiah, and A. Ray, "First-principles, data-based, and hybrid modeling and optimization of an industrial hydrocracking unit," *Industrial & engineering chemistry research*, vol. 45, no. 23, pp. 7807–7816, 2006.

[12] S. R. Islam, W. Eberle, S. Bundy, and S. K. Ghafoor, "Infusing domain knowledge in ai-based" black box" models for better explainability with application in bankruptcy prediction," *arXiv preprint arXiv:1905.11474*, 2019.

[13] D. Baş, F. C. Dudak, and
bibinitperiodI. H. Boyacı, "Modeling and optimization iv: Investigation of reaction kinetics and kinetic constants using a program in which artificial neural network (ann) was integrated," *Journal of Food Engineering*, vol. 79, no. 4, pp. 1152–1158, 2007.

[14] A. B. Farimani, J. Gomes, and V. S. Pande, "Deep learning the physics of transport phenomena," *arXiv preprint arXiv:1709.02432*, 2017.

[15] S. Hwangbo, R. Al, and G. Sin, "An integrated framework for plant data-driven process modeling using deep-learning with monte-carlo simulations," *Computers & Chemical Engineering*, vol. 143, p. 107 071, 2020.

[16] M. A. Soleimanzade and M. Sadrzadeh, "Deep learning-based energy management of a hybrid photovoltaic-reverse osmosis-pressure retarded osmosis system," *Applied Energy*, vol. 293, p. 116 959, 2021.

[17] A. T. Mohammad, M. A. Al-Obaidi, E. M. Hameed, B. N. Basheer, and I. M. Mujtaba, "Modelling the chlorophenol removal from wastewater via reverse osmosis process using a multilayer artificial neural network with genetic algorithm," *Journal of Water Process Engineering*, vol. 33, p. 100 993, 2020.

[18] X. Jia *et al.*, "Physics-guided machine learning for scientific discovery: An application in simulating lake temperature profiles," *ACM/IMS Transactions on Data Science*, vol. 2, no. 3, pp. 1–26, 2021.

[19] P. Gentine, M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis, "Could machine learning break the convection parameterization deadlock?" *Geophysical Research Letters*, vol. 45, no. 11, pp. 5742–5751, 2018.

[20] K. J. Bergen, P. A. Johnson, V Maarten, and G. C. Beroza, "Machine learning for data-driven discovery in solid earth geoscience," *Science*, vol. 363, no. 6433, 2019.

[21] D. Galbally, K. Fidkowski, K. Willcox, and O. Ghattas, "Non-linear model reduction for uncertainty quantification in large-scale inverse problems," *International journal for numerical methods in engineering*, vol. 81, no. 12, pp. 1581–1608, 2010.

[22] R. K. Tripathy and I. Bilionis, "Deep uq: Learning deep neural network surrogate models for high dimensional uncertainty quantification," *Journal of computational physics*, vol. 375, pp. 565–588, 2018.

[23] M. M. Rajabi and H. Ketabchi, "Uncertainty-based simulation-optimization using gaussian process emulation: Application to coastal groundwater management," *Journal of hydrology*, vol. 555, pp. 518–534, 2017.

[24] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, PMLR, 2016, pp. 1050–1059.

[25] D. J. MacKay, "A practical bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

[26] Z. Men, E. Yee, F.-S. Lien, D. Wen, and Y. Chen, "Short-term wind speed and power forecasting using an ensemble of mixture density neural networks," *Renewable Energy*, vol. 87, pp. 203–211, 2016.

[27] B. Gu, T. Zhang, H. Meng, and J. Zhang, "Short-term forecasting and uncertainty analysis of wind power based on long short-term memory, cloud model and non-parametric kernel density estimation," *Renewable Energy*, vol. 164, pp. 687–708, 2021.

[28] R. Herzallah and D. Lowe, "A mixture density network approach to modelling and exploiting uncertainty in nonlinear control problems," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 2, pp. 145–158, 2004.

[29] S.-B. Yang, Z. Li, and W. Wu, "Data-driven process optimization considering surrogate model prediction uncertainty: A mixture density network-based approach," *Industrial & Engineering Chemistry Research*, vol. 60, no. 5, pp. 2206–2222, 2021.

[30] M. Lee and S. Park, "A new scheme combining neural feedforward control with model-predictive control," *AIChE Journal*, vol. 38, no. 2, pp. 193–200, 1992.

[31] W. J. Lee, J. Na, K. Kim, C.-J. Lee, Y. Lee, and J. M. Lee, "Narx modeling for real-time optimization of air and gas compression systems in chemical processes," *Computers & Chemical Engineering*, vol. 115, pp. 262–274, 2018.

[32] C. Agbi, Z. Song, and B. Krogh, "Parameter identifiability for multi-zone building models," in *2012 IEEE 51st IEEE conference on decision and control (CDC)*, IEEE, 2012, pp. 6951–6956.

[33] A. Y.-D. Tsen, S. S. Jang, D. S. H. Wong, and B. Joseph, "Predictive control of quality in batch polymerization using hybrid ann models," *AIChE Journal*, vol. 42, no. 2, pp. 455–465, 1996.

[34] M Khayet, C Cojocaru, and M Essalhi, "Artificial neural network modeling and response surface methodology of desalination by reverse osmosis," *Journal of Membrane Science*, vol. 368, no. 1-2, pp. 202–214, 2011.

[35] Z. Zhang, Z. Wu, D. Rincon, and P. D. Christofides, "Real-time optimization and control of nonlinear processes using machine learning," *Mathematics*, vol. 7, no. 10, p. 890, 2019.

[36] P Baraldi, G Bonfanti, and E. Zio, "Differential evolution-based multi-objective optimization for the definition of a health indicator for fault diagnostics and prognostics," *Mechanical Systems and Signal Processing*, vol. 102, pp. 382–400, 2018.

[37] J.-F. Qiao, Y. Hou, and H.-G. Han, "Optimal control for wastewater treatment process based on an adaptive multi-objective differential evolution algorithm," *Neural Computing and Applications*, vol. 31, no. 7, pp. 2537–2550, 2019.

[38] M. E. Basiri, S. Nemati, M. Abdar, E. Cambria, and U. R. Acharya, "Abcdm: An attention-based bidirectional cnn-rnn deep model for sentiment analysis," *Future Generation Computer Systems*, vol. 115, pp. 279–294, 2021.

[39] S. Kwon *et al.*, "Mlt-dnet: Speech emotion recognition using 1d dilated cnn based on multi-learning trick approach," *Expert Systems with Applications*, vol. 167, p. 114 177, 2021.

[40] R. Yang *et al.*, "Cnn-lstm deep learning architecture for computer vision-based modal frequency detection," *Mechanical Systems and signal processing*, vol. 144, p. 106 885, 2020.

[41] G. E. Hinton, "Deep belief networks," *Scholarpedia*, vol. 4, no. 5, p. 5947, 2009.

[42] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[43] I. Goodfellow *et al.*, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[44] C. Smith and Y. Jin, "Evolutionary multi-objective generation of recurrent neural network ensembles for time series prediction," *Neurocomputing*, vol. 143, pp. 302–311, 2014.

[45] X. Li and X. Wu, "Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2015, pp. 4520–4524.

[46] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[47] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[48] Y. Himeur, K. Ghanem, A. Alsalemi, F. Bensaali, and A. Amira, "Artificial intelligence based anomaly detection of energy consumption in buildings: A review, current trends and new perspectives," *Applied Energy*, vol. 287, p. 116 601, 2021.

[49] R. Dahiya *et al.*, "Condition monitoring of wind turbine for rotor fault detection under non stationary conditions," *Ain Shams Engineering Journal*, vol. 9, no. 4, pp. 2441–2452, 2018.

[50] S. Krohn, "Danish wind turbines: An industrial success story," *Available at the www-site of the Danish Wind Industry Association: www. windpower. org*, 2002.

[51] K. Cho *et al.*, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[52] C. M. Bishop, "Mixture density networks," 1994.

[53] J. S. Bridle, "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition," in *Neurocomputing*, Springer, 1990, pp. 227–236.

[54] P. F. Odgaard, J. Stoustrup, and M. Kinnaert, "Fault-tolerant control of wind turbines: A benchmark model," *IEEE Transactions on control systems Technology*, vol. 21, no. 4, pp. 1168–1182, 2013.

[55] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[56] A. D. Belegundu and T. R. Chandrupatla, *Optimization concepts and applications in engineering*. Cambridge University Press, 2019.

[57] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[58] M. J. Reddy and D. N. Kumar, "Multiobjective differential evolution with application to reservoir system optimization," *Journal of computing in civil engineering*, vol. 21, no. 2, pp. 136–146, 2007.

[59] K Jamal, M. Khan, and M Kamil, "Mathematical modeling of reverse osmosis systems," *Desalination*, vol. 160, no. 1, pp. 29–42, 2004.

[60] *Filmtec™ reverse osmosis membranes technical manual*.

[61] M. Li, "Reducing specific energy consumption in reverse osmosis (ro) water desalination: An analysis from first principles," *Desalination*, vol. 276, no. 1-3, pp. 128–135, 2011.

[62] Y. Kawai, K. Ando, and H. Kawamura, "Distortion of near-surface seawater temperature structure by a moored-buoy hull and its effect on skin temperature and heat flux estimates," *Sensors*, vol. 9, no. 8, pp. 6119–6130, 2009.