LEARNING BASED ALGORITHMS FOR TEMPERATURE CONTROL AND FOULING PREDICTION IN HEAT-EXCHANGERS

BY

SREENATH SUNDAR

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Mechanical Science and Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Abstract

My thesis broadly explores different data-driven algorithmic frameworks for solving two class of problems pertaining to heat-exchangers: temperature control and fouling resistance modeling and prediction.

Designing robust and accurate temperature controllers for heat exchangers is very challenging primarily because of complexities associated with the synthesis of dynamics of industrial heat exchangers. It is practically infeasible to synthesize an accurate dynamical model of heat-exchanger's flow and heat transfer physics because of unmodeled system dynamics and random noise. The use of non-model based control approaches such as PI control is limited by the challenges faced in the accurate tuning of its parameters. Further, an optimally tuned PI controller will not be universally applicable across the entire operating range of heat-exchanger's outlet temperatures due to changing controller set-points caused by large-scale stochastic fluctuations in the heat exchanger's flow and temperature inputs. We propose a couple of data-driven temperature control solutions for heat-exchangers based on n-step advance deep neural networks and a hybrid control approach based on steady state neural network with a proportional controller. We compared the performance of these two control approaches with PI control and open loop (no control) scenarios on a simulated water-air heat exchanger system. The heat-exchanger control objective was to accurately track the user-defined set-point temperature signal at the air-outlet duct of the heating coil system. We assess the response characteristics of the developed control approaches for two set-point temperature signals- a constant temperature and a high frequency (periodic) square pulse. We obtained over 70% reduction in absolute set-point temperature tracking error compared to a PI controller, over a 50-minute duration for both the developed control approaches. Further, we explained the working of the neural network + proportional control module in great detail along with the rationale behind our design choices for ensuring the controller's stability and robustness.

The complexities involved in fouling prediction in heat-exchangers have deterred traditional model based approaches and other empirical methods. Many existing data-driven prediction approaches are typically application-driven or heat-exchanger specific. Hence, these algorithms are limited either by the scale of data they can handle or by the geometrical and flow configurations of the heat-exchanger for which they were conceived. Here, we develop a generalized and scalable statistical model for accurate prediction of fouling resistance using commonly measured parameters of industrial heat-exchangers. This prediction model is based on deep learning which is a family of computational data-driven methods that allows a scalable algorithmic architecture to learn non-linear functional relationships between a set of target and predictor variables from large number of training samples. The efficacy of this modeling approach is demonstrated for predicting fouling in a simulation of a flue-gas and water cross flow heat exchanger designed for waste-heat recovery. The results demonstrate that the average coefficient of determination, which quantifies the accuracy of predictions is over 99% in predicting flue-gas side, water side and overall fouling resistances. This prediction framework is also evaluated under varying levels of input noise and it was demonstrated that averaging predictions over an ensemble of multiple neural networks achieves better accuracy and robustness to noise. We find that the proposed deep-learning fouling prediction framework learns to follow heat-exchanger flow and heat transfer physics, which we confirm using locally interpretable model agnostic explanations around randomly selected operating points.

To my parents and my brother, for their love and support.

Acknowledgments

I would like to extend my heart-felt gratitude and thanks to my research adviser Prof. Srinivasa Salapaka who has been a constant source of support, guidance and inspiration. He has taught me the art of doing research and molded my personality significantly throughout the time that I spent under his tutelage. I have also benefited immensely through the wisdom he has been imparting me through our discussions over a wide range of topics ranging from research to his personal experiences. I also admire his open-mindedness and his willingness to explore new research frontiers. I hope to inculcate the values he has taught me to become a good model student of his, both from a personal and professional stand-point.

We acknowledge support from the Advanced Manufacturing Office (AMO) of the Office of Energy Efficiency and Renewable Energy (EERE) under the U.S. Department of Energy, through the grant DE-EE0008312 for this work. I have also had the opportunity to collaborate with various faculty members of UIUC through the course of this project- Dr. Sanjiv Sinha, Dr. Nenad Miljkovic, Dr. Placid Ferreira and Dr. Chenhui Shao and I would like to thank them for their guidance and feedback. I would also like to thank various researchers who I had worked with closely during this project- Manjunath C.Rajagopal, Ho Chan Chang, Hanyang Zhao, Gowtham Kuntumalla and Yuquan Meng. I have also had a memorable journey with my fellow research colleagues and would like to acknowledge the contributions of Mayank Baranwal, Ram Sai Gorugantu, Sheikh Mashrafi, Amber Srivastava and Alireza Askarian.

I would like to extend my deepest regards to some of my great teachers at UIUC for their knowledge and for instilling confidence and motivation in me to keep striving for excellence. I would also like to thank the MechSE department at UIUC for supporting my graduate education at UIUC. I would like to thank my department Graduate coordinator, Kathyrn Smith, for her assistance and timely advise through all the paperwork starting from my admission till my graduation.

Finally, my acknowledgement would be incomplete without me expressing gratitude to my amazing parents Sundar and Deepa, my brother Shreecharran for their love, support and encouragement for all my endeavors. Graduate life at UIUC has been a fun-filled and memorable journey thanks to my amazing friends who were always there to discuss ideas, challenges and offer suggestions.

Table of Contents

List of Tables

List of Figures

LIST OF ABBREVIATIONS

HX	Heat Exchanger
NN	Neural Network
SNN	Single Layer Neural Network
DNN	Deep Neural Network
PI	Proportional Integral Controller
P	Proportional Controller
SSE	Steady State Error
Var	Variance
Config	Configuration

Expressions

Nomenclature	
D_o	Pipe Outer Diameter
D_i	Pipe Inner Diameter
t	Pipe Wall Thickness
L	Pipe Length
s	Spacing between pipes(constant spacing along row and column)
k_{tube}	Thermal Conductivity of pipe material
n_r	Number of tubes along flue gas flow direction
n_t	Number of tubes in each row
H	Height of flue gas flow duct
W	Width of flue gas flow duct
T_{fi}	Flue gas Inlet Temperature

T_{fo}	Flue gas Outlet Temperature
T_{wi}	Water Inlet Temperature
T_{wo}	Water Outlet Temperature
$\dot{M}_{flue}$	Mass Flow Rate of Flue gas
$\dot{M}_{w}$	Mass Flow Rate of water
f_1	Flue gas Fouling Factor
f_2	Water Side Fouling Factor
R_1	Flue gas side Fouling Resistance
R_2	Water Side Fouling Resistance
$R_1 + R_2$	Overall Fouling Resistance
U	Mean water flow velocity in pipes
V_{max}	Maximum Flue gas flow velocity for minimum flow area between the tubes normal to flow direction
Re_w	Reynolds Number for Water flow
Re_{flue}	Reynolds Number for Flue flow
Nu_w	Water-side Nusselt Number
Nu_{flue}	Flue-side Nusselt Number
h_w	Water-side heat transfer co-efficient
h_{flue}	Flue-side heat transfer co-efficient
f	Fanning Friction factor for pipe flow
Eu	Euler Number for Flue flow
C_{min}	Minimum fluid heat capacity rate
C_{max}	Maximum fluid heat capacity rate
C_r	Fluid heat capacity ratio
IQR	Inter-Quartile Range
Subscript	
f	physical variable evaluated under fouled-circumstances

Chapter 1

Introduction

1.1 Introduction- Temperature Control Module

Synthesis of accurate dynamic models for complex industrial heat-exchanger systems is extremely challenging due to associated uncertainties and random fluctuations in fluid-flow conditions of the heat exchanger. Hence, typical methods for designing model based feedback controllers often result in sub-optimal performance. The applications of non-model based control algorithms such as Proportional Integral(PI) control module is limited by the degree of uncertainty and random fluctuations in fluid flow-conditions. The process gain of industrial heat-exchanger systems is highly variable often depending on the load of components such as heating and cooling coils and on inlet conditions of the heating and the cooling fluids such as the air temperature and air volume flow-rates. In previous works, some of these issues have been addressed by the use of artifical neural networks and other artificial intelligence techniques to the control of heat and air-conditioning systems [4, 5, 6].

In Chapter 2 of this work, we propose three data-driven approaches for designing a robust control architecture that would control the desired fluid temperature at the heat-exchanger exit in the presence of random stochastic fluctuations of input flow-rate and temperature variables. We have implemented a model of feed-forward neural network in operating in parallel with a proportional feed-back controller as one of the algorithm. The gain of the proportional feedback controller has been identified through a grid-search algorithm over the possible solution space. The feed-forward neural network module operates in a slow-time scale while the proportional controller operates in real-time to provide accurate temperature tracking at the exit of the heat exchanger. Another algorithm that we had developed for accurate temperature control of the heat-exchanger uses a modified version of neural network that incorporates temporal information of the heat-exchanger operation variables known as the n-step ahead neural network. This network takes the state-variable values at time t and controlled parameter's (temperature) value at time t+n to predict optimal control input signal(fluid flow-rate) at time t. The weights of the n-step ahead network are optimized to minimize the error between predicted and actual control input signal for a particular temperature set-point.

The proposed data-driven control algorithms have been tested on an accurate simulated model of air-water heater system. Initially, we came up with a design of proportional-integral (PI) control scheme that would enable us to regulate the air-outlet temperature by controlling the flow-rate of water. The proportional and integral gain parameters of the PI control module was tuned through theoretical methods ensuring best response and minimal steady state error under likely input disturbances and changes in the set-point. The performance of this optimally tuned Proportional Integral (PI) controller in maintaining the set-point temperature form the baseline of comparison for developed data-driven temperature control techniques.

1.2 Introduction- Fouling Resistance Prediction

Fouling is the process of continuous deposition and accumulation of unwanted materials and sediments such as scale, algae, suspended solid and insoluble salts on the interior or exterior surface of the heat exchanger [7]. Fouling on process equipment and heat exchanger-surfaces often have a significant, detrimental impact on the working efficiency and operations of the heat exchangers. Specifically, fouling reduces heat transfer rate, impedes fluid flow, corrodes material surface and contaminates the working fluid. These effects result in financial strain on all major heat and process industries today, typically in terms of installation of additional extended surfaces, increased fuel consumption, production losses from unplanned fouling related shutdowns, and maintenance costs for removal of fouling deposits with chemicals and mechanical anti-fouling devices [8]. Various studies estimate the total fouling related costs for major industrialized nations to be over \$4.4 billion USD with the corresponding fouling related economic losses amounting to about 0.25% to 0.30% of their overall GDP [9, 10]. The fouling related cleaning costs range between \$40,000 to \$50000 per heat exchanger per cleaning [11].

The existing research on fouling mitigation that spans several centuries can be primarily categorized as - (1) fouling prevention, or mitigation techniques: These include adding anti-foulant or hydrophobic chemicals [12, 13] to the operating fluid for interrupting fouling-inducing mechanisms, real-time control of heat-exchanger environment and fluid composition for minimizing empirical fouling risks [14], engineering the surfaces on heat transfer equipment [15, 16], and regular cleaning and optimal maintenance scheduling of heat exchangers during the operation cycle [17, 18]. While these research techniques [19] can control the rate of fouling to a considerable extent, they can significantly be improved upon with better fouling modeling and prediction algorithms that lead to better planning and maintenance control strategies. Fouling prediction can potentially savesave billions of dollars lost annually on fouling related losses through significant reductions in heat exchanger downtime.

(2) Fouling modeling and prediction algorithms: These algorithms have predominantly been based on empirical approaches or specific heat-exchanger dynamical models. Previous analytical models of fouling processes are based on rate equations [20] where the rate of fouling deposition growth is estimated to be a deterministic function (linear [21, 22], asymptotic [23, 24], non-linear [25]) or a stochastic approximation [26]. A non-linear model was conceived in Ref.[25] to predict the threshold fouling conditions as a function of Reynolds and Prandtl numbers. The functional parameters were then estimated from experimental coking data of pre-heat train and process heaters used in crude-distillation units. In Ref.[26], a heat-exchanger maintenance strategy was developed based on the scatter parameter of time to fouling distribution. Model based heat-exchanger fouling estimation methods have centered around the use of Kalman filters [27] and development of fuzzy polynomial observers for state-estimation [28] based on non-linear heat exchanger fouling dynamical model such as the one proposed by Jonsson [29]. The limitations of empirical and model-based fouling estimation techniques have led to the development of data-driven prediction approaches in the recent literature.

The process of fouling formation and material deposition happens through crystallization of salts, particulate deposit from fluid stream, deposition and growth of biological matter, chemical reaction by-products from various reactants in the working fluids and corrosion. Each of these individual phenomena operates at different rates(time scales) which makes the creation of an all- encompassing dynamic model for fouling monitoring and prediction infeasible. These challenges make it very difficult to develop accurate models for predicting fouling from first principles. This has resulted in a surge of data-driven approaches for fouling prediction from easily measurable heat-exchanger parameters. Fouling prediction and detection algorithms based on Support Vector Machines(SVM) [30], Autoassociative Kernel Regression(AAKR) [31] and artificial neural networks [32, 27, 33, 34, 35] have been successfully developed and deployed in various process applications. The extensive use of artificial neural networks for fouling prediction and detection is inspired by their utility in solving inverse problems pertaining to modeling and evaluation of heat transfer coefficients for fin-tube heat exchangers [36] and fluid-particle systems [37]. These data-driven algorithms for fouling prediction, despite achieving good fouling predictive accuracies for their respective applications, face issues of scalability and generalization. These approaches that fall under the category of classical machine learning techniques are often limited by the size of data and require extensive feature reduction either by visual identification or by mathematical methods (such as Principal Component Analysis(PCA) [33]) to reduce model complexity. These challenges have often become significant impediments to their deployment in large-scale industrial applications, where copious amounts of process data is generated.

In Chapter 3, we present a generalized approach for modeling and predicting fouling in heat-exchangers using deep-learning based feed-forward ensemble neural network architectures. Deep learning, the primary driver of our model, utilizes neural network often consisting of a few thousand parameters to learn the non-linear functional relationship existing between fouling resistances (target variable) and heat-exchanger operational data (causal variables) [38]. Our framework ensures that the fouling model can be powered by large datasets with very little pre-processing [38]. We tested our fouling prediction algorithm on a simulated model of cross-flow heat exchanger designed for waste-heat recovery. The choice of application and selection of heat-exchanger type for the simulated system is motivated by United States's annual industrial energy consumption, which is approximately 32 quadrillion Btu (10^{15} Btu), of which a further 1,478 trillion Btu of energy from exhaust gas thermal emissions remain unrecovered [1, 39]. Acidic condensation, increased chemical activity and fouling are often cited as significant barriers affecting the economy and effectiveness of waste-heat recovery units (WHRU) which results in their long-payback periods [1, 39, 40]. Further, industrial cross-flow heat exchangers are widely used for transferring heat between two gas streams or between a liquid and gas such as in stack gas heat recovery, convective recuperators or air preheaters in WHR [41]. The high fouling rates typically experienced by these waste heat recovery units underline the need for an accurate fouling prediction algorithm, where its real-time deployment can create maximum working impact and cost reduction.

Chapter 3 is organized as follows. In Section 3.1, we develop a robust data-generation scheme for simulating the chosen heat-exchanger system under fouled circumstances by incorporating state of the art models for both hydrodynamic and thermal fouling effects. Details of the neural network architecture used in our fouling prediction algorithm is laid out in Section 3.2. We address historically faced training challenges in fouling prediction networks such as lower precision, slower network convergence and unstable model performance [30] through our choice of network activation functions and output data transformation techniques. Further, we demonstrate the versatility of the proposed deep-learning framework by adapting the network architecture to predict individual flue-gas side and water-side fouling resistances in addition to the overall fouling resistance, using same operating input samples. Here, we outline the neural training and hyper-parameter tuning strategies for the optimization algorithm used for updating network weights. In Section 3.3, we present the training of our deep neural network using the data samples derived from the data-generation scheme of Section 2. The accuracy of fouling predictions is quantified using three test datasets, each with its unique sample distribution. We obtained over 99% model fit accuracy(R^2) between predicted and actual fouling resistance in each of these data-sets. In Section 3.4, we highlight the important features of our deep-learning model. First, we quantify the predictions from our module under varying levels of practically relevant input noise. Finally, we analyze our fouling prediction module using local interpretable model-agnostic explanations(LIME) [42]. We show that the predictions agree with results from high fidelity heat exchanger flow and heat-transfer fouling physics models around randomly chosen operating points. With the rapid advances in high-speed computing, the proposed fouling module is a generalized yet scalable framework that can be adapted to any heat-exchanger environment to make real time predictions.

Mathematical Preliminaries

Our solution approaches for tackling the challenges faced in the temperature control and fouling monitoring of heat-exchangers, in the absence of well-defined mathematical dynamical models, revolve around the use of data-driven approaches. Learning functional maps between a set of input features and the output of interest is central to the working of data-driven algorithms. Formally, any data-driven algorithm, also called as a *learning algorithm* seeks to *learn* the inherent patterns and structures present in the input features $x \in \mathbb{X} \subseteq \mathbb{R}^m$ to predict a variable of interest $y \in \mathbb{Y}$. We refer to the task of a learning algorithm as a regression or classification task depending on whether y is discrete valued or continuous valued.

The objective of any data-driven algorithm is to learn the functional mapping $f : x \to y$. The data-driven algorithm searches for a optimal representative function from a functional space f(x;w) characterized by a parameter w. This parameter w is tuned based on an optimization objective so that the learnt function $\hat{y} = f(x; w^*)$ predicts the output y accurately. This optimization requires a metric that quantifies the model performance i.e, an error measure based on the predicted value $\mathbf{\hat{y}} = \mathbf{f}(\mathbf{X}; \mathbf{W})$ and the true observation $\mathbf{y}$. Formally, this is captured through a functional representation of the learning algorithm's error called the *loss function* $\mathscr{L} : \mathbb{Y}*\mathbb{Y} \to \mathbb{R}$. The choice of the loss function depends on the desired objective (for instance, whether we need lesser absolute errors or penalize false positives,etc) and nature of prediction task (e.g. classification or regression). Thus, the objective of tuning of weights in a learning model reduces to minimizing the loss function $\mathscr{L}(y, \hat{y})$ over a set of n training samples which yields the following optimization problem:

$$W^* = argmin_{W \in \mathbb{W}} \sum_i L(y_i, \hat{y_i}) \qquad \text{with} \qquad L(y_i, \hat{y_i}) = (y_i - \hat{y_i})^2, \qquad (1.2.1)$$

where $\mathbb{W}$ is the parameter space of the *learning* algorithm. We use deep neural networks as the key *learning* algorithmic component of our temperature controller and fouling prediction module designs. In the coming sections of this chapter, we seek to define an optimization objective for a deep learning task and explore algorithms for tuning the weights of deep neural network. We also define how local interpretable model agnostic explanations (LIME) is used to effectively explain the predictions made by the neural network and understand its working.

1.3 Neural Networks and Deep Learning

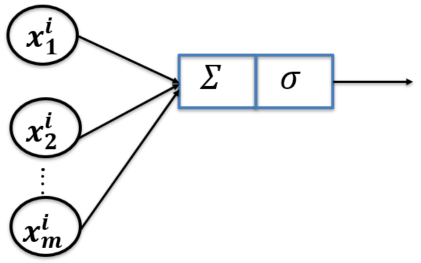
Deep learning is a family of computational data-driven methods that allows a scalable and tunable algorithmic architecture to learn non-linear functional relationships between a set of target and predictor variables from large number of training samples. The nature of heat-exchanger's flow and heat-exchanger physics is very complex. The fundamental equations defining the heat-exchanger's flow and heat-exchanger physics such as the Navier Stokes equation, conjugate heat-transfer and fluid flow equations are non-linear in nature. Further, the modeling complexity of heat-exchanger systems is greatly enhanced by the presence of noise and unmodeled dynamics. Hence, we adopt deep learning in our solution approach for heat exchanger temperature control and fouling prediction due to its ability to learn non-linear functional maps between input and target variables. In this section, we review *shallow* neural networks and proceed to *deep* neural networks [43, 44, 38, 45].

1.3.1 Single-layer neural networks

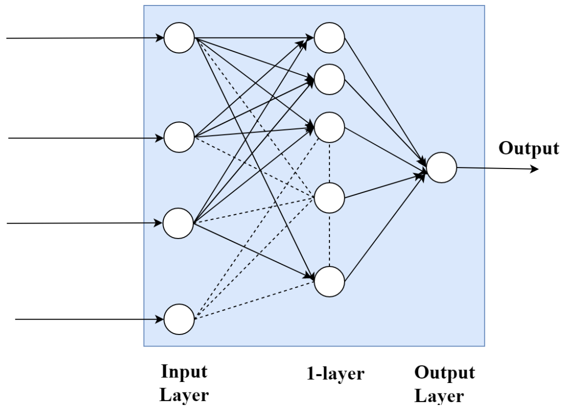
A single layer neural network, also called an artificial neural network, consists of connected units called neurons. A neuron is a computing unit that takes the inputs from other neurons, computes the weighted sum of its inputs based on neural parameters w, and transforms the sum via an activation function to the output (Fig 1.1(a)). This output is passed to the next layer of connected neurons as their input (Fig 1.1(b)). This neural network is also called a feed-forward neural network because the network does not have any cycles or feedback connections. Mathematically, we represent the operations of single neuron as:

$$f_{SNN}(x; W, b) = \sigma(Wx + b) \tag{1.3.1}$$

where W is the network parameters, b is the bias. The activation function (σ) defines the nature of non-linear relationship between the inputs and outputs. Commonly used activation functions are- sigmoid $\frac{1}{1+e^{-x}} \in (0, 1)$, hyperbolic tangent $\frac{2}{1+e^{-2x}} - 1 \in [-1, 1]$ and rectified linear unit (RELU) $\sigma(x) = max(0, x) \in [0, \infty)$. It is evident that these activation functions allow values to pass through only after its inputs clear a certain threshold. These activation functions are central to neural networks modeling non-linear and non-convex functions. However, they increase the complexity of the optimization problem in eq. (1.0.1) and make it difficult to solve. There is no closed form solution for obtaining the global minima of the loss function in eq (1.0.1). It is solved through iterative gradient descent techniques that yield solutions close to the global minima ([46]).



(a) Single Neuron



(b) Single Layer Network Architecture

Figure 1.1: **Zoomed in view of a neuron in a network (a), Single layer neural network (b)**

1.3.2 Deep neural networks

One can extend the idea of single layer neural network to multiple layers as well. The term *deep* neural network can be used to specify a neural network with an arbitrarily large number of layer ($n_{layers} \geq 1$). The advantage that a deep neural network offers is that it possesses a large number of parameters that can be tuned based on the observed data samples. Thus, a deep neural network can learn highly complex, non-linear maps accurately. The output of a deep neural network is obtained by horizontally stacking the outputs of several single-layered neural network. Mathematically, this can be written as:

$$f_{DNN}(x) = f_{SNN}(f_{SNN}(.......(f_{SNN}(x)) \ \ (n \text{ layers in total}) \tag{1.3.2}$$

The choice of number of layers and number of neurons in each layer is challenging. We adopt the approach suggested by [47] which recommends adding layers to the deep network until the generalization error stops improving. For deep neural networks, the commonly used activation function is RELU as this leads to the creation of sparse matrices that are easier to compute. The optimization objective of (1.0.1), rewritten for the deep neural network case, is as follows:

$$W^* = argmin_{W \in \mathbb{W}} \sum_i L(y_i, f_{DNN}(x_i; W)) \tag{1.3.3}$$

where $\mathbb{W}$ is the weight space of the deep learning algorithm. It is shown in [48] that it is NP-hard to optimize a neural network to produce accurate predictions for all training samples (i.e, 100 % training accuracy). The optimization objective in eq. 1.1.3 is solved using gradient descent based iterative optimization algorithms. The class of gradient descent algorithms used for tuning the weights of the deep neural networks has two standard steps- (1) Partial gradient computation of the loss function w.r.t the parameters of the network, (2) An update rule for the network parameters based on the magnitude of the gradient. These steps are expressed as shown:

$$\nabla_W L(W; X; Y) = \frac{1}{n} \sum_{i=1} \nabla_W L(y_i, f(x_i; W)), \tag{1.3.4}$$

$$W \leftarrow W - \alpha \nabla_W L(W; X; Y) \tag{1.3.5}$$

where α is the learning rate parameter which controls the extent of descent along decreasing gradient direction. These steps are repeated over several iterations. For this update rule to be applied, the gradient of the loss function has to be calculated w.r.t all the parameters of the network. This is done using back-propagation algorithm which is an efficient way of calculating neural network gradients by re-using computation already done for the previous layer. We explain the back-propagation algorithm and review the various gradient descent algorithms in the coming sections.

1.3.3 Network Gradient Computation using Back-Propagation (BP) algorithm:

The weights of a deep neural network are tuned so that the network prediction error is minimum. The back-propogation algorithm has been derived for a 3 layer network here and the same principle can be extended to a network with any number of layers. The back-propagation algorithm can be seen as gradient descent on sum of squared error. The single sample squared prediction error(L) for a network with n_k neurons in the output layer can be written as:

$$L = \frac{1}{2}\sum_{k=1}^{k=n_k}(t_k - y_k)^2 \tag{1.3.6}$$

The incremental adjustment of weights is directly proportional to rate of change of network prediction error w.r.t layer weights which can be written as:

$$\delta W \propto -\frac{\partial L}{\partial W} = \alpha\frac{\partial L}{\partial W} \tag{1.3.7}$$

$$\text{where } \alpha \text{ is the network learning rate} \tag{1.3.8}$$

The change in weights from layer j to layer k using the above established idea followed by the application of chain-rule of differentiation can be written as:

$$\triangle w_{jk} = -\alpha\frac{\partial L}{\partial w_{jk}} \tag{1.3.9}$$

$$\triangle w_{jk} = -\alpha\frac{\partial L}{\partial y_k}\frac{\partial y_k}{\partial x_k}\frac{\partial x_k}{\partial w_{jk}} \tag{1.3.10}$$

$$\triangle w_{jk} = \alpha(t_k - y_k)\dot{f}(x_k)y_j = \alpha\delta_k y_j \tag{1.3.11}$$

$$\text{where } \delta_k = (t_k - y_k)\dot{f}(x_k) \tag{1.3.12}$$

Similarly, the adjustment in weights from layer i to layer j can be derived as:

$$\triangle w_{ij} = -\alpha\frac{\partial L}{\partial w_{ij}} \tag{1.3.13}$$

$$\triangle w_{ij} = -\alpha[\sum_k \frac{\partial L}{\partial y_k}\frac{\partial y_k}{\partial x_k}\frac{\partial x_k}{\partial y_j}]\frac{\partial y_j}{\partial x_j}\frac{\partial y_j}{\partial w_{ij}} \tag{1.3.14}$$

$$\triangle w_{ij} = \alpha[\sum_k \delta_k w_j k]\dot{f}(x_j)y_i = \alpha\delta_j y_i \tag{1.3.15}$$

$$\text{where } \delta_k = [\sum_k \delta_k w_{jk}]\dot{f}(x_j) \tag{1.3.16}$$

1.3.4 Review of Gradient Descent based Optimization Algorithms

The three variants of gradient descent algorithm that can be used for updating the weights of a network with the objective of minimizing the loss function i.e. the network prediction error are:

Batch Gradient Descent Batch gradient descent computes the gradients of the loss function with respect to the weights W of the network for the entire training dataset:

$$W = W - \alpha \nabla_W L(W) \tag{1.3.17}$$

A single update using the batch gradient descent algorithm would involve gradient computations for the whole dataset using the backpropagation algorithm which makes this algorithm slow, computational intensive and intractable for large datasets.

Stochastic and Mini-Batch Gradient Descent Stochastic gradient descent(SGD) updates the weights based on a randomly chosen sample from the data-set in contrast to gradient descent algorithm. Stochastic gradient descent algorithm does away with much of the redundant gradient computations of gradient descent algorithm by performing one update at a time. This makes stochastic gradient descent algorithm(SGD) faster than gradient descent algorithm however SGD performs weight updates with a high variance that causes the objective function to fluctuate heavily after each update. Mini-batch gradient descent algorithm combines the best of gradient descent and stochastic gradient descent algorithms by performing an update for every mini-batch of 'n' samples randomly sampled from the training dataset. Common mini-batch size varies between 16 to 256 depending on the type of application. The advantage of using mini-batch gradient descent is that it (1) Stabilizes the convergence of the loss function to local or global minima by reducing the variances of weight updates (2) Utilizes highly optimized state of the art matrix optimization techniques.

$$\text{Stochastic Gradient Descent: } W = W - \alpha \nabla_W J(L; x^{(i)}; y^{(i)}) \tag{1.3.18}$$

$$\text{Mini-batch Gradient Descent: } W = W - \alpha \nabla_W J(L; x^{(i:i+n)}; y^{(i:i+n)}) \tag{1.3.19}$$

Mini-batch Gradient Descent with Nesterov Momentum Stochastic gradient descents and its variants such as mini-batch gradient descent algorithms has potential issues in navigating ravines [49], i.e. areas where the surface curves much more steeply in one dimension than in another, which are found frequently near local minima. Nesterov accelerated gradient used in our implementation for fouling resistance prediction network helps to adjust the weight updates of a network depending on the direction and magnitude of future gradient of the loss function. Nesterov gradient helps to attain faster convergence and dampens the oscillations in the loss function.

$$\text{Momentum Term: } v_t = \gamma v_{t-1} + \alpha \nabla_W J(W - \gamma v_{t-1}; x^{(i:i+n)}; y^{(i:i+n)}) \tag{1.3.20}$$

$$\text{Weight Update: } W = W - v_t \tag{1.3.21}$$

where γ is the damping co-efficient

In our implementation of deep neural network for temperature controller and fouling prediction modules, the damping co-efficient in the momentum term has been set to 0.9 and the hyper-parameters step-size(α) and batch-size(n) were optimized using exhaustive grid-search algorithm over their possible solution space. When the learning rate (α) decreases at an appropriate rate every iteration, the stochastic gradient descent is guaranteed to converge to the global optimum under mild mathematical assumptions [50].

1.4 Local Intepretable Model Agnostic Explanations (LIME)

$$explanation(x) = argmin_{g \epsilon G} L(f, g, \pi_x) + \omega(g) \tag{1.4.1}$$

A mathematical representation for local surrogate models with interpretability constraint is shown above. The explanation model for a test sample x is the local linear model (such as weighted linear regression model) that minimizes a loss function L which is a measure of how close the explanation is to the prediction of our fouling resistance prediction model f. The model complexity of the local linear model($\omega(g)$) is kept low by controlling the number of features used for local explanation. G is the family of possible linear models that can be used for local approximation of fouling resistance prediction network. The proximity measure(π_x) defines the size of the local neighborhood around the chosen sample. Typically, the LIME algorithm optimizes the loss-function part while the local model surrogate complexity is user-defined depending on the problem.

The family of possible local linear models used for obtaining a local explanation of the fouling resistance prediction network around a local test sample is locally weighted linear regression model. Various test inputs and their outputs for learning the parameters of the local surrogate model are obtained by perturbing the chosen data-sample and getting their corresponding predictions from the fouling resistance prediction network. The parameters of the local surrogate model for fouling resistance prediction network is then obtained by minimizing the squared prediction error weighted by the distance of the perturbations from the original test sample as shown:

$$\text{Loss Function: } L = min_{\theta,\tau} \sum_{i=1}^{i=n} exp\left(-\frac{\left|x^{(i)} - x\right|^2}{2\tau^2}\right)(y^{(i)} - \theta^T x^{(i)}) \tag{1.4.2}$$

$$\text{where } x^{(i)}, y^{(i)} \text{are the perturbed sample points and the corresponding network predictions} \tag{1.4.3}$$

Chapter 2

Data-driven Temperature Controller Approaches for Heat Exchangers

In this chapter, we introduce the simulated model of air-water heat-exchanger system. We then implemented an optimally tuned PI control design for air outlet temperature control at the exit of simulated air-water heat-exchanger system. The working of data-driven control design frameworks for n-step ahead neural networks, hybrid control models based on PI controller with steady neural network and reinforcement learning algorithms are introduced later in the chapter. We have also compared the performance of the proposed data-driven control approaches with optimally tuned PI design in terms of response curve characteristics and steady state error.

2.1 Air-Water Heater-Model

Underwood and Crawford [51] developed a model of a water-air heater system by fitting a set of second-order, non-linear equations to measurements of air and water temperatures and flow-rates obtained from experiments on an actual water-air heater-system. A representative figure of the water-air model with the proposed data-driven control and learning module is as shown below:

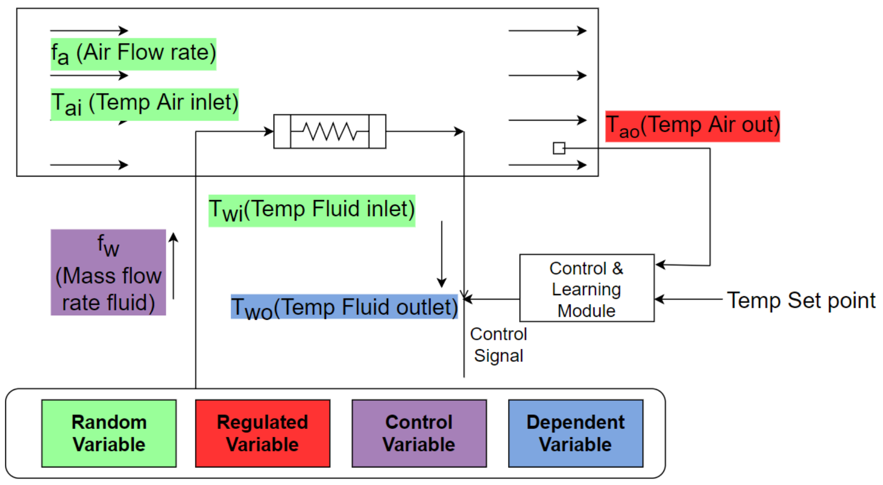


Figure 2.1: **Simulated heater system with Controller+Learning Module**

In Fig 2.1, the cold air from the surrounding environment enters the duct at a temperature T_{ai}, exchanges heat with hot water flowing over the pipes to exit the duct at a temperature T_{ao}. Likewise, the hot-water, supplied from a reservoir through a water-pump, flowing through the pipes loses heat to cold air to eventually exit the pipe at a temperature T_{wo}. The assumptions that were made in the development of a non-linear dynamic model for this heat-exchanger are:

1. The tube, water and air have constant specific heats and densities throughout the tube length and over the ranges of temperature encountered.
2. The heat conduction in the axial direction is negligible.
3. The inlet air temperature at any cross section is taken to be the effective air temperature for convective heat transfer purposes
4. The mean temperature difference between the two fluids driving the overall heat UA is the difference of inlet air temperature and average water temperature.

The water-side and air-side energy balance equations for the heat-exchanger derived with the above assumptions are:

$$\dot{m}_w(t)C_{pw}(t)[T_{wi}(t) - T_{wo}(t)] + UA(t)[T_{ai}(t) - \bar{T}_w(t)] = C_w \frac{d}{dt}[T_{wo}(t)] \tag{2.1.1a}$$

$$\dot{m}_a(t)C_{pa}(t)[T_{ai}(t) - T_{ao}(t)] + UA(t)[\bar{T}_w(t) - T_{ai}(t)] = C_a \frac{d}{dt}[T_{ao}(t)] \tag{2.1.1b}$$

$$UA(t) = a + b\dot{m}_w(t) + d\dot{m}_a(t) \tag{2.1.1c}$$

The dynamical model of the water-air heater system has air and water input and output temperatures, flow rates($T_{ai}, T_{ao}, T_{wi}, T_{wo}, m_a, m_w$) as the state variables. The control signal (c) is the model input which determines the pump flow-rate. The overall heat transfer co-efficient is written in terms of water and air flow rates(m_w) as shown in eq. 2.1.1c. Applying Euler forward rule for discretizing eqs. (2.1.1a) and (2.1.1b), the final model equations with their coefficients determined from actual heating coil experiments through least-square fit to data is obtained as:

$$f_w(t) = 0.008 + 0.00703(-41.29 + 0.30932c(t-1) - 3.2681 \times 10^{-4} \times c(t-1)^2 + 9.56 \times 10^{-8} \times c(t-1)^3), \tag{2.1.2a}$$

$$T_{wo}(t) = T_{wo}(t-1) + 0.64908 \times f_w(t-1)(T_{wi}(t-1) - T_{wo}(t-1)) + (0.02319 + 0.10357 f_w(t-1) + 0.02806 f_a(t-1))(T_{ai}(t-1) - \frac{(T_{wi}(t-1) + T_{wo}(t-1))}{2}), \tag{2.1.2b}$$

and $$T_{ao}(t) = T_{ao}(t-1) + 0.19739 \times f_a(t-1)(T_{ai}(t-1) - T_{ao}(t-1)) + (0.03184 + 0.15440 f_w(t-1) + 0.04468 f_a(t-1))(\frac{(T_{wi}(t-1) + T_{wo}(t-1))}{2} - T_{ai}(t-1)) + + 0.20569 \times (T_{ai}(t) - T_{ai}(t-1)) \tag{2.1.2c}$$

The total experimental duration was 50 minutes with input-output data sampled every 5 seconds. The same conditions were recreated in simulated model of heat-exchanger. The random variables in the simulation (highlighted 'green' in Fig 2.1) were sampled from a random distribution lying within an appropriate range every 50 seconds to model disturbances and changing operating conditions that occur in real-world heat exchanger systems. The range of operating variable conditions used in the simulated heat-exchanger model are summarized below:

Heat Exchanger Variable	**Operating Range**
Air Flow rate(f_a)	$0.7kgs^{-1} \leq f_{flue} \leq 0.9kgs^{-1}$
Air inlet Temperature(T_{ai})	$4°C \leq T_{fi} \leq 10°C$
Water inlet Temperature(T_{wi})	$73°C \leq T_{wi} \leq 81°C$

Table 2.1: Range of Input Parameter Values in Water-Air heater system simulation

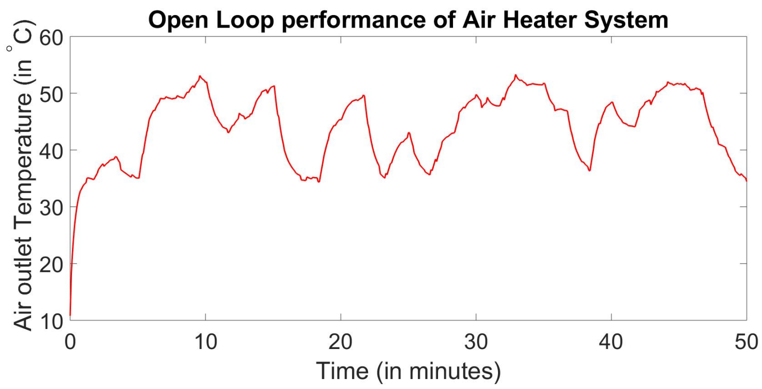


Figure 2.2: **Simulated heater system's in absence of water-flow rate control**

Open Loop System Performance There are large fluctuations observed in air-outlet temperature at simulated heat-exchanger's exit (Fig 2.2) in the absence of water flow-rate control. These fluctuations in outlet temperature of the heat-exchanger system is caused by the stochastic variations in input flow-rate and temperature conditions. The fluctuations in air-outlet temperature with time induced by input variables and model uncertainties often result in sub-optimal heat exchanger performance. The sub-optimal heat exchanger performance leads to loss in energy that is otherwise recoverable if the heat-exchanger operates at its maximal rating. There are specific requirements on outlet air temperature for certain process applications that cannot be met in the absence of robust and accurate air-exit temperature control. Thus, there arises a need for accurate temperature tracking at heat-exchanger's exit in the presence of disturbances and variations in input temperature and flow rates.

2.2 PI Control design and its limitations

A standard SISO (single input single output) PI controller was tuned to control the PI controller implemented on the heat-exchanger. The air-outlet temperature is the regulated variable (Fig 2.1) while the input to the water-flow rate control value is the control signal (Fig 2.1). The PI control equation is:

$$\text{Continuous time: } c'(t) = k_p e(t) + k_i \int e dt, \tag{2.2.1a}$$

$$\text{Discrete time: } c'(t) = k_p e(t) + k_i \delta t \sum_{j=0}^{t} e_j, \tag{2.2.1b}$$

where: $c'(t) -$ normalized control output signal to valve,,

$e(t) - T_{set}(t) - T_{ao}(t)$ difference between temperature set-point and measured value of control variable,

$K_p -$ proportional gain constant,

$K_i -$ integral gain constant$[1/s]$,

$\delta t -$ sampling rate$[s]$

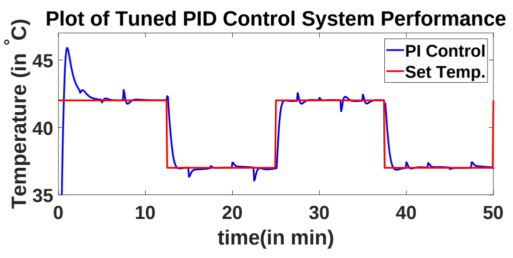
Discretizing equations in eq. (2.2.1b) using Euler forward method and taking the z-transform of that expression, we get:

$$\frac{u(t+1)-u(t)}{\delta t} = k_p\left[\frac{e(t+1)-e(t)}{\delta t}\right] + k_i e(t) \tag{2.2.2}$$

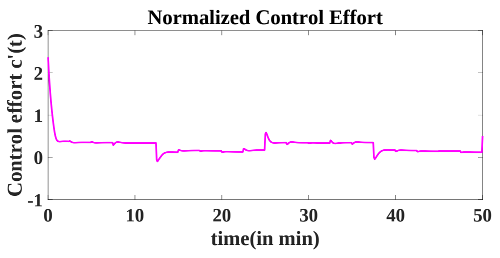
$$\frac{c'(z)}{e(z)} = \left[k_p + \frac{k_i}{z-1}\delta t\right] \tag{2.2.3}$$

The transfer function in eq. (2.2.3) is implemented as the control block in dynamical model of water-air heater system created in MATLAB-SIMULINK. The best proportional and integral gains were empirically determined by minimizing the sum of the absolute value of the difference between the set point and actual existing air temperature subject to variety of disturbances. The PI control constants were tuned using trial and error method as suggested by Haines and Hittle ([52]) and later optimizing the PI control gains through grid-search around chosen values. The choice of control parameters that produced the best response and minimal steady state error is $K_p = 0.0759, K_i = 0.015$. The normalized control signal is transformed to the actual control signal ($c(t)$) of the model by:

$$c(t) = 1400 - 730c'(t)$$



(a)



(b)

Figure 2.3: **Control response and Control effort Plot for low-frequency temperature tracking**

We studied the performance of the tuned controller under the objective of tracking a square air outlet-temperature signal. The square wave switched from a temperature of $37°C$ to $42°C$ and vice-versa every 12.5 minutes. From PI control response plot in Fig 2.3(a), we infer that the tuned PI control precisely tracks the set-air outlet temperature. The PI response overshoots the set-temperature initially by a large margin before adjusting the control input to give very low tracking errors. We also observe small wiggles in the PI response curve once it settles down to a steady-state value in the vicinity of set-point temperature. These small wiggles are caused by controller over-compensating for temperature tracking error at certain instants

(Fig 2.3(b)). The performance of PI control signal in tracking high frequency air-outlet temperature has also been studied (Fig 2.4). In Fig 2.4, the set-temperature fluctuates between $37°C$ to $42°C$ and vice-versa every 100 seconds. There is a large over-shoot in the initial phase after which the control output stabilizes to track air-outlet temperature set-point effectively.

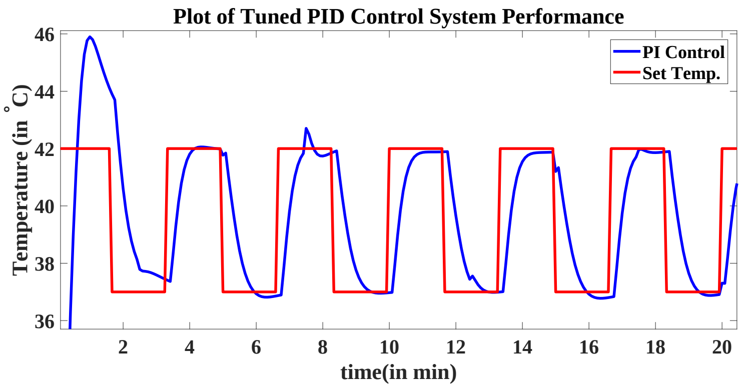


Figure 2.4: **PI Control's high frequency air-outlet temperature Tracking response**

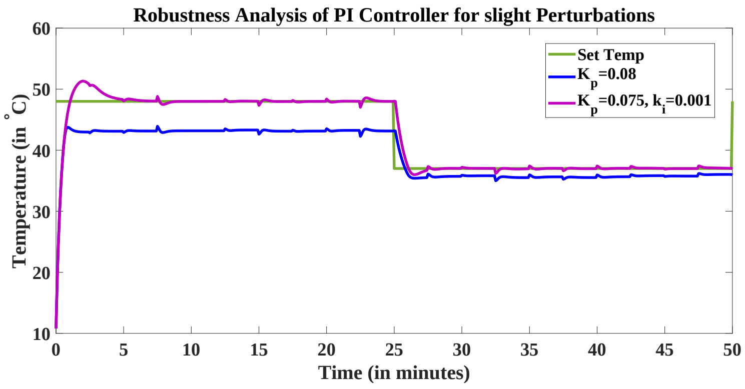
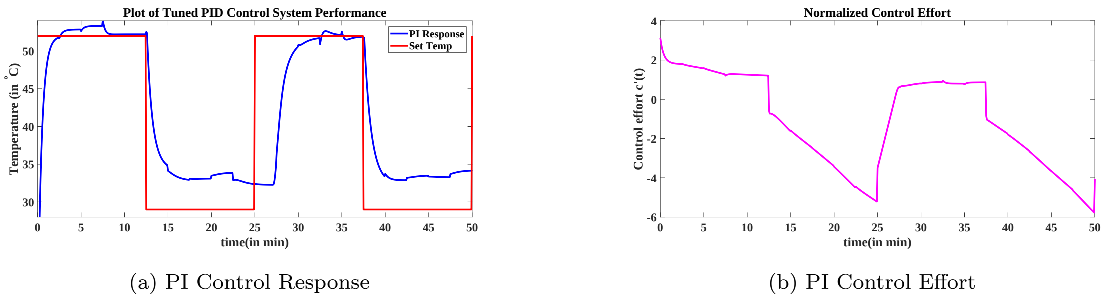


Figure 2.5: **Slight perturbation of optimally tuned values of PI control parameters result in sub-optimal system performance. The value of proportional gain(K_p) is changed to 0.08 from 0.075 with integral gain(K_i) set to 0 from 0.001. The slight change of control parameters makes the set-temperature tracking highly ineffective. This illustrates the need for determining the PI control parameters exactly for industrial systems with large model uncertainities and input fluctuations.**

Need for alternate control designs The performance of PI control response in tracking air outlet set-point temperature pulses of various frequencies was studied in the previous section. There is a need for alternate control design approaches despite effective temperature control of water-air heater system by PI controller. PI control design has certain fundamental limitations in its applicability to non-linear system with unmodeled dynamics and large input fluctuations. The major challenge faced in PI control design is in tuning the proportional and integral control gains. The existing approaches to parameter tuning in

PI controllers are largely empirical and trial and error based. The tuned PI parameters even if uniquely determined, are largely limited to a narrow range of input variable operating conditions. This is because heat-exchangers have large process gains and input variable fluctuations typically and the controller gains can be tuned effectively only for a small range of process gains and variable fluctuations. Conventional control approaches dealt with this PI design setback by tuning multiple controllers for various regimes of process gains and switching between the controllers based on estimates of current process gains. These approaches often require frequent human interventions and the presence of large number of controllers often result in high power consumption.



(a) PI Control Response (b) PI Control Effort

Figure 2.6: **The validity of tuned control parameters is only for small range of water-air heater process gains. This is seen in Fig 2.6(a) where the set-point tracking error is high because of limited applicability of tuned control parameters to specific air-outlet temperature range. The controller effort in Fig 2.6(b) shows the aperiodic nature of input flow-rate signal despite the periodic nature of set-point temperature.**

2.3 Evaluating Controller Performance

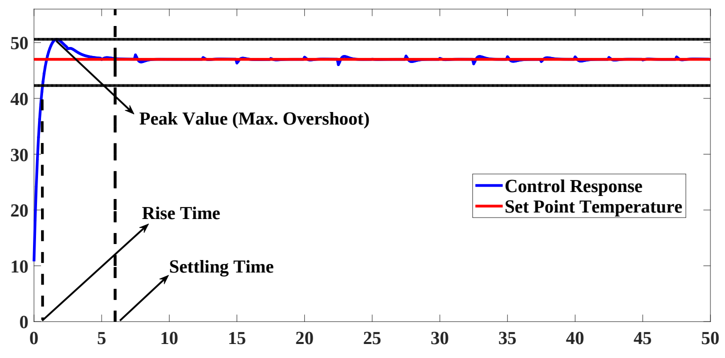


Figure 2.7: **Evaluating Control Response to Set-point Temperature**

Control Response Characteristics Each of the discussed fluid-outlet temperature controller frameworks for heat exchanger has been evaluated based on their response to different set-point temperature signals. The controller response curve parameters of each of the control strategy have been measured for constant and pulse based set-point temperature signals. The key parameters measured for quantifying transient control response are:

1. **Rise-Time** Rise-time is the time taken for the control response to reach from 0% to 90% of constant temperature value or the difference in height of set-point temperature pulse.

2. **Peak Value** Peak value measures the global maximum of temperature control response curve. Maximum overshoot is a measure by which the peak value exceeds the set-point temperature.

3. **Settling Time** Settling time is the time taken by the temperature control response to reach the vicinity of set-point temperature and remain within the permissible error band around set-point temperature for the reminder of time. The error-band is defined as:

$$|T(t) - T_{set}(t)| \leq \epsilon \tag{2.3.1}$$

   where ϵ is about 2% of constant set-point temperature value or the different in height of set-point temperature pulse.

4. **Steady-State Error** The mean of absolute difference between the desired set-point temperature and the actual fluid outlet temperature once the system reaches steady state after which the control response stays in the error band defined by eq (2.3.1) if the heat-exchanger system remain undisturbed.

Evaluating Data-driven algorithms The objective of the data-driven algorithms in temperature control module is to predict the input signal to water-pump as a function of heat exchanger operating variables. These data-driven algorithms have been used in a stand-alone manner or in tandem with a proportional controller to form a hybrid controller interface. The predictive performance of data-driven algorithmic component in the heat-exchangers temperature controller models was evaluated by looking at the co-efficient of determination(R^2) [53].The co-efficient of determination(R^2) is defined as the proportion of variance in the dependent variable (normalized control signal to water-pump) that is predictable from the predictor variables(heat-exchanger real-time operations data)(eq 2.3.2) or the reduction in error over a mean-only model. A higher co-efficient of determination(R^2) of a data-driven temperature control approach is an indication of its high predictive power with the model successfully explaining a significant fraction of variations observed in water-flow rate control signal values present in the dataset. R^2 values typically range between $[0, 1]$ though it may be negative in certain cases when the developed learning model performs worse than mean-only model.

Let $y_1, y_2, y_3, ...y_n$ be the n values of target flow-rate signal variable in the dataset and $\hat{y}_1, \hat{y}_2, \hat{y}_3, ...\hat{y}_n$ be the corresponding values predicted by the data-driven control algorithm then:

$$\textbf{Residuals}:\ e_i = y_i - \hat{y}_i \tag{2.3.2a}$$

$$\textbf{Absolute prediction error}:\ |e_i| = |y_i - \hat{y}_i| \tag{2.3.2b}$$

$$\textbf{Sample Mean}:\ \bar{y} = \frac{1}{n}\sum_{i=1}^{i=n} y_i \tag{2.3.2c}$$

$$\textbf{Total Sum of squares}:\ TSS = \sum_{i=1}^{i=n}(y_i - \bar{y})^2 \text{ (Proportional to data variance)} \tag{2.3.2d}$$

$$\textbf{Residual Sum of squares:}\ RSS = \sum_{i=1}^{i=n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{i=n} e_i^2 \tag{2.3.2e}$$

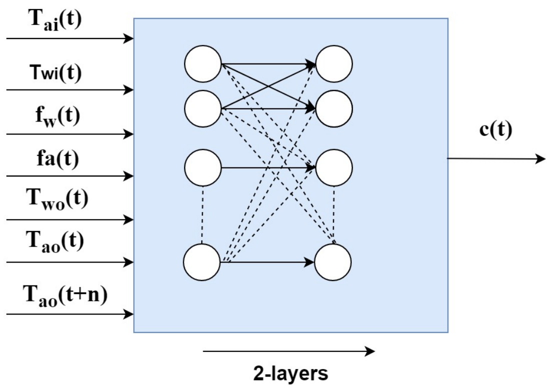
$$\textbf{Coefficient of Determination}(R^2)\ = 1 - \frac{RSS}{TSS} = 1 - \text{Fraction of Explained Variance(FVU)} \tag{2.3.2f}$$

2.4 Deep-Learning Controller based on Temporal Information

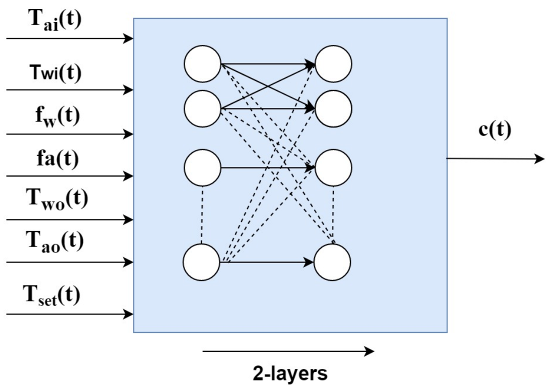
2.4.1 Neural controller configuration

We propose a deep neural network based control design for controlling the outlet air temperature of air-water heater system. The fundamental challenge encountered in the design of a neural network based controller is that the desired water-flow rate control output is not available for a particular air-outlet temperature set-point. A conventional control-system based approach to deal with this problem is to optimally learn a water-flow rate input signal by computing the error in controlled state variable which would be the difference between the temperature set-point and the observed air-outlet temperature in this problem setting. Further, the objective of the deep neural network will then be to minimize control error over time. This approach can be implemented using recurrent neural networks however this would require an inverse heat-exchanger model for generating the training and test datasets. It is challenging to obtain an inverse model of an industrial heat-exchanger whose approximate dynamics are synthesized from input-output data or simulations.

In our solution approach, we adopt a deep-learning based methodology which is the process of training a neural network (typically a large-network with millions of parameters) to perform the prescribed task of predicting water-flow rate signal for a set air-outlet temperature. The neural network module learns the function approximating the target variable (water- flow rate) through observed values of predictor variables (heat-exchanger operating conditions data). Creating or ”training” the neural network controller module would require a large data-set of values connecting the operating conditions of heat-exchanger and user-defined set-point temperature with known values of water flow-rate input signal. The training data-set consisted of water-air heater system variables such as $T_{ai}, T_{wi}, f_a, f_w, T_{wo}, T_{ao}$ measured at a time instant t and regulated variable T_{ao} measured $'n'$ steps after current time instant t for a chosen normalized control signal value $c'(t)$ (Fig 2.8(a)). The training data was generated through approximate dynamical model of water-air heater system developed from input-output experimental data as shown in eqs (2.1.2a,2.1.2b,2.1.2c). A *step* corresponds to experimental sample time of 5 secs for water-air heating coil model. The evolution of control variable air-outlet temperature (T_{ao}) due to the causal effects of heat-exchanger operating conditions and normalized control signal (c') at time instant t is captured with the inclusion of air outlet temperature at a time instant $t + n$ steps later ($T_{ao}(t + n)$).



(a) Training



(b) Test

Figure 2.8: **Neural network controller's training (a) and test dataset (b) variables configuration**

The neural network controller is trained on this training dataset to output the control signal for the water-air heater model. The performance of neural network controller to various set-point air-outlet temperature signals is then evaluated on a test data-set where the user defined signal $T_{set}(t)$ replaced the temperature variable $T_{ao}(t + n)$ of the training dataset. The objective of this replacement of temperature variable in the test dataset is to ensure that the regulated variable air-outlet temperature ($T_{ao}(t + n)$) reaches the set-point temperature ($T_{set}(t)$) $'n'$ steps in future through the regulation of water-flow rate control signal ($c'(t)$). Thus, we succeed in overcoming the challenge of synthesizing an inverse model of a heat-exchanger under consideration. Moreover, this control design is entirely data-driven and can be deployed in real-time with controller inputs obtained in real-time from flow-rate and temperature sensors of the heat-exchanger. The *step* used in this algorithm will suitably be replaced with the sampling interval of digital controller that would be implementing the developed control scheme.

Neural Network Control Architecture: The neural network control architecture (Fig 2.7) has entities called neurons that are stacked horizontally into layers. The first layer called the input layer takes in the inputs of the neural network controller. The neural network control architecture has seven neurons in our implementation with six neurons corresponding to water-air heater operating conditions $(T_{ai}, T_{ao}, T_{wi}, T_{wo}, f_a, f_w)$ and the final neuron taking in either air-outlet temperature at time instant $t+n$ $(T_{ao}(t+n))$ or the set-point temperature (T_{set}) depending on training or test configuration (Fig 2.7). Every neuron in subsequent layers of the network takes a weighted linear combination of all neuron outputs in the previous layers and subjects it to a non-linear activation function to create an output that the next layers of neurons act on. The discussed neural controller architecture uses rectified linear units (RELU) activation function in the first two layers and sigmoid activation function in the output layer. The definition of these activations functions are as follows:

$$\textbf{RELU: } f(x) = max(0, x) \tag{2.4.1a}$$

$$\textbf{Sigmoid: } g(x) = \frac{1}{1+e^{-x}} \tag{2.4.1b}$$

$$\textbf{Tanh: } h(x) = \frac{2}{1+e^{-2x}} - 1 \tag{2.4.1c}$$

The choice of rectified linear units as activation function in the intermediate layers of the neural network over traditionally used sigmoid and hyperbolic tangent activation functions is because network computations involving RELU result in the creation of sparse matrices. Sparse matrix computations are quicker and computationally inexpensive over dense matrix computations. The use of rectified linear units (RELU)as activation functions greatly accelerates the convergence of optimization algorithms (often by a factor of six or more in certain cases) used for tuning the parameters of the neural network [54]. The sigmoid activation function was used at the network output layer as it is required that the neural network controller outputs a normalized water-flow rate control signal $(c'(t))$ lying in [0,1].

Dataset Generation and Neural Training: The training and test datasets are generated from the approximate dynamical model of the heat-exchanger developed from experimental input-output data (eqs (2.1.2a,2.1.2b,2.1.2c)). Separate training dataset is generated for every value of $'n'$ (number of steps in advance) (Fig 2.7(a)) ranging between 0 to 9. The random variables (T_{ai}, T_{wi}, f_a) and control variable $(c'(t))$ in the training dataset were randomly changed after a few time steps (randomly every 10 to 30 samples) to account for fluctuating input conditions and entire range of water-air heater's operation. The inputs to the neural network are unit normalized (range: [0,1]) for achieving faster training rates and neural network convergence. The optimization algorithm used to train the neural network controller module through back-propagation algorithm [55] is the distributed stochastic gradient descent implementation [56, 57, 58] with Nesterov momentum [59].

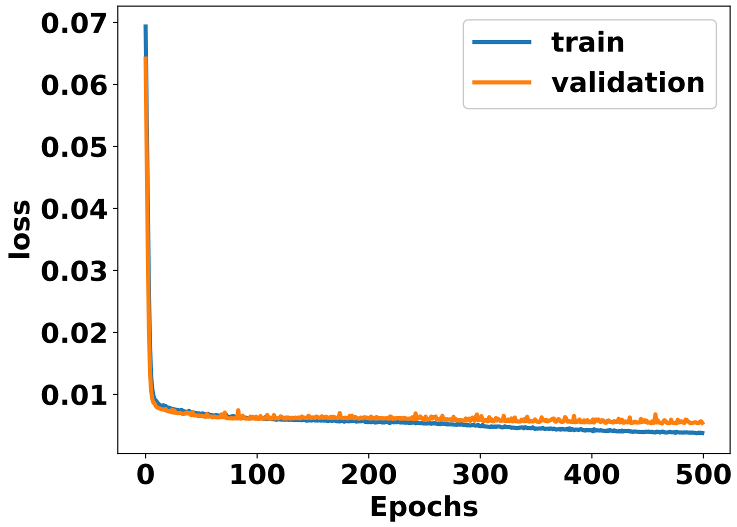
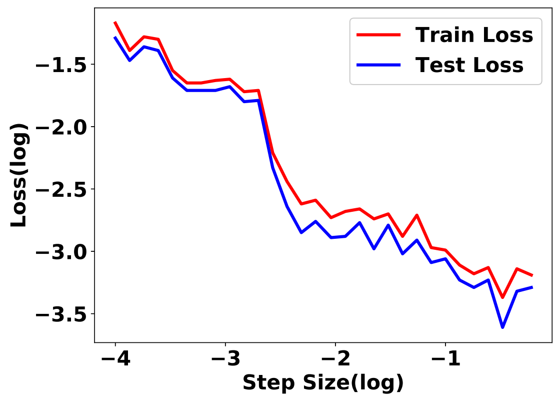


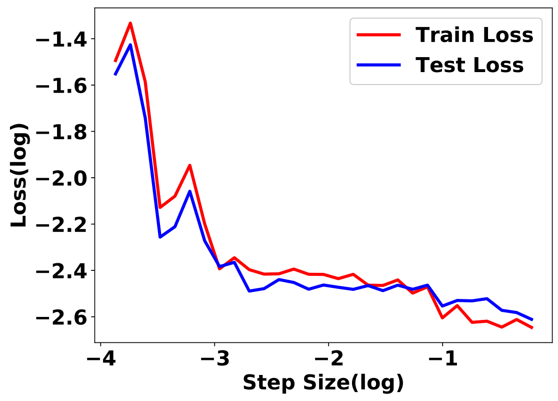
Figure 2.9: **Test-Train loss curve for neural controller training with parameter n = 7**

The dataset generated for training the neural controller is split into two parts: (1) training dataset: About 80% of dataset was randomly selected to train network parameters (2) Validation data-set: About 20% of data was used to optimize the hyper-parameters such as optimization algorithm's step-size, batch-size and stopping criterion. Fig 2.8 shows the training-validation model loss curve against the number of training iterations (epochs). The model loss curve plots depict the decrease in loss function or prediction error (defined as $(c(t) - c'(t))^2$ where $c(t)$ is the network prediction and $c'(t)$ is the *ground truth* of water flow rate-signal) as the neural network controller module is trained on training dataset samples repeatedly. The training process is stopped at the instant when both the training and validation loss are small with subsequent iterations only leading to only marginal reduction or increase in training and validation loss. This was done in order to prevent the neural model from either over-fitting or under-fitting the training data-set.

Hyper-parameter tuning: The optimization algorithm's learning rate and batch-size are two major hyper-parameters that need to be optimally tuned. The neural network controller's set-point tracking characteristics depend on these two parameters. We have tuned these parameters using grid search algorithm [60] over possible solution spaces for stepsize and batchsize. The network is then trained for every combination of hyper-parameters returned by the grid-search algorithm with every training run totaling 200epochs (iterations). Model loss is computed after each run and the set of values of stepsize and batchsize that gave low training and validation loss are taken to be the optimal hyper-parameter values for subsequent algorithmic studies. This process is repeated for various training datasets corresponding to different values of $'n'$. (Fig 2.9)



(a) 5-step advance neural controller



(b) 6-step advance neural controller

Figure 2.10: **Neural network controller's loss curves at various learning rates for training datasets with $n = 5$ (a) and $n = 6$ (b). The 5-step neural network controller's training and test loss curves achieve their minimum for a stepsize of 0.33 with the corresponding batchsize of 8 samples. The 6-step neural network loss curves reach their minimum for a learning rate of 0.6 and batch size of 64. These optimal parameter values are determined by training the network for 200 epochs on their corresponding datasets**

2.4.2 Neural controller performance characterization

The effectiveness of deep neural controller is assessed by evaluating its co-efficient of determination (R^2) on training and validation datasets. Further, the neural controller's transient response characteristics were quantified on water-air heater system by subjecting it to two test set-point air-outlet temperature signals over a 50 minute duration- (1) a constant temperature of $47°C$ and (2) 40-sample periodic square pulse signal switching between $40°C - 45°C$ every 20 steps (200s period). The controller transient response parameters are evaluated using the methodology described in section (2.3). The performance of series of deep neural controller models with low training and validation residual sum of square errors is evaluated to determine the best controller for various $'n'$ (number of steps in advance) ranging from one to eight. We observed marginal differences in transient response characteristics of neural controllers whose coefficient of determination (R^2) on training and validation datasets differed by a maximum of 2% for the same $'n'$. Subsequently, the best controller models for each $'n'$ are compared to determine the optimal value of number of advance steps in neural training configuration ($'n'$).

Model Accuracy(R^2)	Rise-Time(s)	Mean Tracking Error($^{\circ}C$)	Max. Overshoot(%)
98.47%	32.5	0.14	2.5%
98.95%	31.87	0.08	1.78%
99.21%	32.5	0.13	2.12%
99.41%	28.75	0.04	2.41%
99.52%	30.63	0.11	2.58%

Table 2.2: **Variation of control response characteristics with 4-step advance ($n = 4$) neural controller's coefficient of determination(R^2) for a set-point air-outlet temperature square pulse signal**

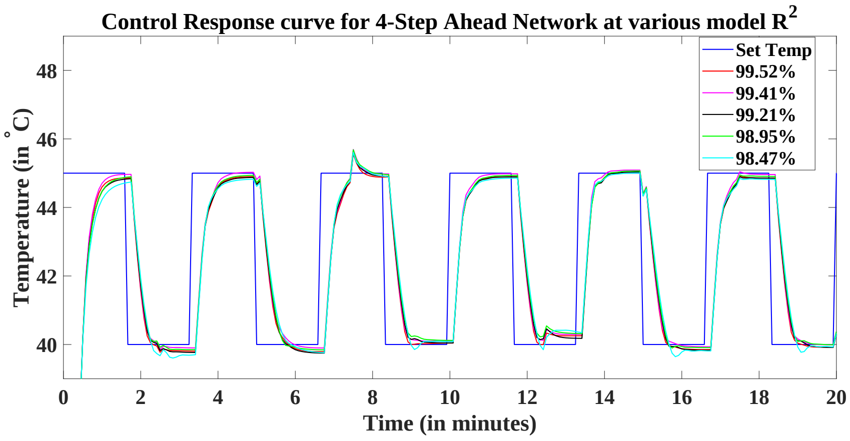


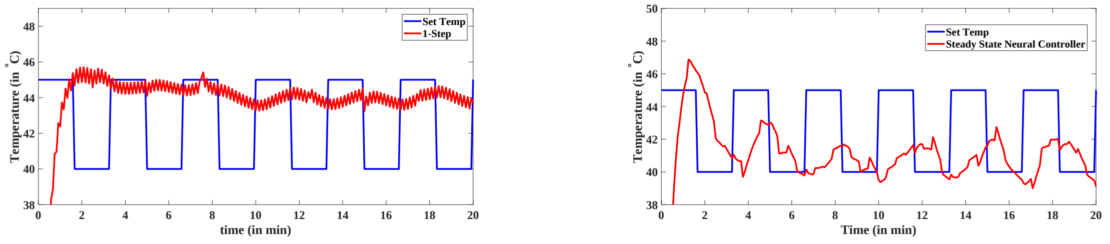
Figure 2.11: **Control response characteristics of neural controller with different model accuracies(R^2) for a 20-step test square pulse over 20 minutes duration:** The control response for a 4-step advance neural controller with model accuracy of 99.41% exhibits the fastest rise time and least mean set-point temperature-tracking error at air-outlet.

The variations observed in the control response parameters of 4-step advance ($n = 4$) neural controller module with different model accuracies(R^2) evaluated on a test set-point air-outlet temperature signal are minimal (Table 2.2 and Fig 2.11). The neural controller configurations with $n = 0, 1$ fail to effectively track the set-point temperature signal despite achieving high training and validation accuracies in the training dataset. The limitations of static ($n = 0$) and one-step advance ($n = 1$) neural controller in tracking set-point temperature signal is due to the dependency of the current air-outlet temperature of water-air heater system on flow-rate and temperature values from previous time steps. The analysis of water-air heater's model equations (eq 2.1.2) bring out that temporal dependance on air-outlet temperature's evolution.The optimal value of n in the neural controller framework (Fig (2.8)) has to be at least greater than 1 for accurate prediction of water flow-rate signal from heater's operating conditions and air-outlet set-point temperature signal. It is seen from eq. (2.4.2a) and (2.4.2b) that the normalized flow-rate signal ($c'(t)$) is dependent on the air outlet temperature atleast $n = 2$ steps ahead of time. This functional dependence of the normalized flow-rate signal ($c'(t)$) is not modeled by a static ($n = 0$) and one step ahead ($n = 1$) neural controller. This results in a sub-optimal controller performance while tracking various temperature set-points (Fig 2.11).

$$T_{ao}(t) = g_1(T_{ao}(t-1), f_a(t-1), T_{ai}(t-1), f_w(t-1), T_{ai}(t), T_{wi}(t-1), T_{wo}(t-1)), \tag{2.4.2a}$$

$$f_w(t-1) = g_2(c'(t-2)), \tag{2.4.2b}$$

where g_1, g_2 are functions modeled in eq (2.1.2c) and eq (2.1.2a) respectively.

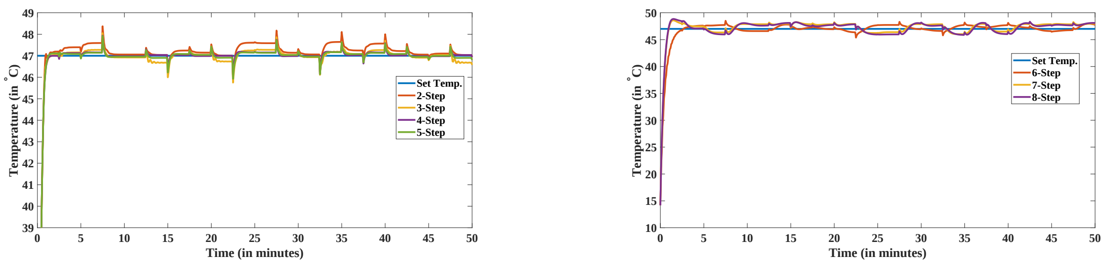


(a) **Steady State Controller** ($n = 0$) (b) **One Step ahead Controller** ($n = 1$)

Figure 2.12: **Both steady state and one step ahead controllers fail to effectively track the periodic square pulse signal. This happens because of insufficient temporal information provided to the neural controller for learning the functional map from the heat-exchanger's flow-rate and temperature variables to the input water-flow rate signal, for a given set-point temperature signal.**

Config (n)	**Rise-Time (s)**	**Max. Overshoot (%)**	**SSE ($°C$)**	**Var ($(°C)^2$)**
2	45s	2.92%	$0.27°C$	0.05
3	45s	2.24%	$0.17°C$	0.03
4	45s	1.98%	$0.09°C$	0.02
5	45s	1.96%	$0.11°C$	0.02
6	70s	3.20%	$0.48°C$	0.33
7	45s	3.37%	$0.75°C$	0.07
8	45s	3.91%	$0.77°C$	0.10

Table 2.3: **Performance characterization of different n-step advance controller configurations for constant temperature set-point signal of 47°C**



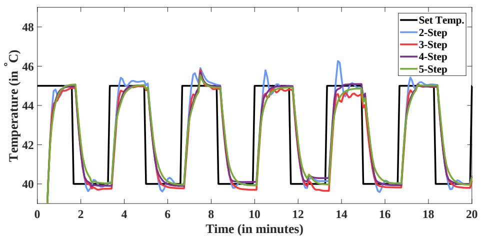
(a) **Configurations** $n = 2, 3, 4, 5$ (b) **Configurations** $n = 6, 7, 8$

Figure 2.13: **Constant temperature tracking characteristics of n-step advance neural controller for different values of n ranging from two to eight.**

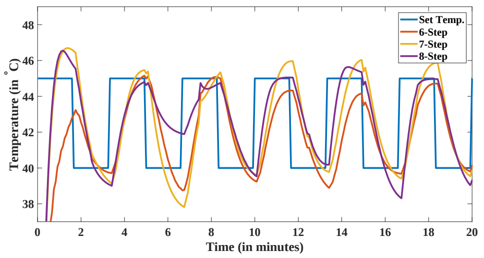
The controller response characteristics for tracking a constant temperature of $47°C$ (Fig 2.13 and Table 2.3) demonstrate that the steady state tracking error and the error variance decreases as the value of parameter n increases from two till four. A steep increase is observed in the tracking error from $n = 6$ to $n = 8$ after a brief plateau from $n = 4$ to $n = 5$. These computations of controller response metrics have been performed on the heat-exchanger system that is allowed to settle for a minute after the start of the neural controller module (steady state is reached after around 10 sampling time steps). A similar trend to steady state error is observed for the maximum overshoot of the controller response over the set-point temperature for various controller configurations. The speed of controller response to the constant set-point air-outlet temperature is largely the same for various controller designs based on $'n'$, as characterized by their equal rise times except for $n = 6$ controller configuration.

Configuration (n)	**Rise-Time (s)**	**Max. Overshoot (%)**	**Steady State Error ($°C$)**
2	23.1s	2.86%	$0.11°C$
3	26.9s	3.01%	$0.12°C$
4	28.8s	2.41%	$0.04°C$
5	35.6s	1.99%	$0.03°C$
6	63.1s	5.3%	$0.30°C$
7	58.1s	5.6%	$0.49°C$
8	50s	4.3%	$0.62°C$

Table 2.4: **Performance characterization of different n-step advance controller configurations for square pulse temperature set-point signal between** $40°C$ **and** $45°C$



(a) **Configurations** $n = 2, 3, 4, 5$



(b) **Configurations** $n = 6, 7, 8$

Figure 2.14: **Control response curves of n-step advance neural controller for a 20-step set-point air-outlet temperature square pulse signal zoomed over a 20 minute duration, for different values of n ranging from two to eight. The combination of controller input variables such as inlet temperatures and flow-rates are different at various instances of time. This causes the control response curve to slightly differ along different periods of the set-point square pulse temperature signal. We remove the variance of these stochastic fluctuations in neural controller response by computing their performance characteristics for every period of this chosen set-point temperature signal and averaging out the numbers, thus obtained. We obtain a measure of the average controller performance to a high frequency periodic square set-point temperature signal over a span of 50 minutes through this exercise.**

The trends observed in the average control response parameters, such as maximum overshoot and steady state error, for various neural configurations, agree with the inferences from the controller's response to constant set-point temperature. The minimum average steady state tracking error of $0.03°C$ over a 50 minute duration of the test set-point temperature signal, is observed for $n = 5$ controller configuration along with minimum overshoot of 1.99%. The controller rise times increase with the increase in the parameter (n) from one to five after which there is no clear trend observed. Thus, we conclude that 5-step advance neural controller offer the best controller response to both constant and square pulse set-point temperature signals. We also infer that a neural controller with any chosen value of n between 2 to 8 offer better controller response characteristics compared to the tuned PI control response (Comparison details in section 2.6).

2.5 Hybrid Control Algorithm based on neural network and proportional controller

2.5.1 Motivation for alternate hybrid control design

Based on our observations in the previous section, a neural network based controller can be set-up to accurately predict the water flow-rate input signal for a given combination of air-water heater system's fluid flow rates, input, output and set-point temperatures. The neural network controller suffer from certain shortcomings despite good control response characteristics. The robustness and stability of an independently operating neural network based controller is not mathematically guaranteed. This poses potential challenges in the implementation of this control design for real-time control of heat exchangers. Conventional control designs such as proportional integral controller requires proper tuning of their parameters and are limited by large-scale stochastic fluctuations in the input variables (Fig 2.6).

The hybrid neural network + proportional controller design, discussed here, is inspired by the need to adopt the better characteristics of each of the individual controllers while nullifying their shortcomings. The objective of this control design approach is to achieve excellent control tracking characteristics with guaranteed stability.

2.5.2 Control Architecture Design

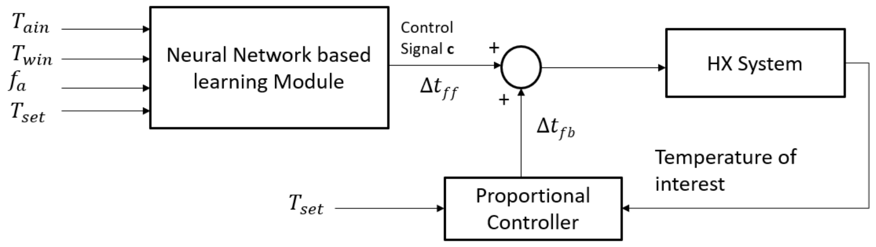


Figure 2.15: **Neural Network + Proportional Controller hybrid design**

The hybrid control design (Fig 2.15) consists of a steady state feed forward neural network controller in operation with a proportional controller. It is observed that the static neural network controller discussed in Section 2.4.2 (Fig 2.12.(a)) delivers poor set-point air-outlet temperature tracking performance. Hence, a steady state (static) neural controller cannot be used alone and needs to be combined with a suitable controller for delivering robust tracking performance. Another issue which the static controller design in Section 2.4.2 (configuration in Fig. 2.8(a) (with $n = 0$) suffers from is that the proposed design would operate in the same time-scales as the proportional controller. Thus, adopting a network design similar to the one shown in Fig (2.8(a)) (Section 2.4.2) for the steady-state neural controller, to be used in our hybrid control algorithm, does not guarantee stability. The time-scale of operation of the neural controller has to be larger than the time scales at which the proportional controller operates for ensuring stable operation of the hybrid control-algorithm on the heat-exchanger system (i.e., $\triangle t_{ff} > \triangle t_{fb}$).

We achieve this control objective by using a different design for the neural controller in our hybrid control algorithm compared to the static network design in Fig. 2.8 (with $n = 0$). We set-up the neural controller to output the steady state PI control response of a controller whose proportional (k_p) and integral gains (k_i)

are designed for some nominal operating condition. The neural network controller is then trained to predict the steady state control output with the air-water heater system's stochastic and set-point flow-rate and temperature variables as inputs $(f_a, f_w, T_{ai}, T_{set})$. These stochastic and set-point system variables do not fluctuate every second but rather at time scales much slower. Hence, we ensure that the neural controller operates at a time scale slower than the proportional controller with our choice of network inputs. This is why we also do not include any of the dynamically evolving output temperature variables like water and air-outlet temperatures (T_{wo}, T_{ao}) in our neural network controller design. The proportional controller performs real-time corrections to the neural predictions to avoid large steady state-errors while tracking set-point air-outlet temperature. The data-generation pseudo-code for training the neural network based controller is as shown below:

1. Generate arrays of size corresponding to total number of chosen training samples for each of the network's input temperature and flow variables- $f_a, T_{ai}, T_{wi}, T_{set}$. The ranges for the inlet air flow-rate (f_a), inlet air temperature (T_{ai}) and inlet water temperature (T_{wi}) are chosen from Table 2.1. The range of set-point temperature (T_{set}) is taken to be between $34°C - 47°C$ corresponding to the observed range of air-outlet temperature possible for this air-water heater system.

2. Choose nominally tuned proportional (k_p) and integral (k_i) control gains for the PI control whose steady-state control output will be predicted by the neural controller

3. Repeat the steps 3- for every input data sample

4. Read the values of the input temperatures, flow-rates and set-point temperature from the generated array of samples for every iteration

5. Initialize air and water-outlet temperatures at time step $t = 0$

6. Initialize error at time $t = 0$ as $T_{set} - T_{ao}(t = 0)$

7. Repeat the following steps for a few time steps till the PI control output converges (15 steps for this air water-heater system)

    (a) The outlet temperature variable (T_{ao}, T_{wo}) evolves according to eq. (2.1.2(a)-2.1.2(c)) based on the input system variables and controller input from previous time step $(t - 1)$

    (b) Compute the integral error component at time step t as $int(t) = int(t-1) + error(t-1)$ (This follows from eq. 2.2.1(a),2.2.1(b))

(c) Calculate the PI control output for the next time step as $c(t) = k_p * e(t) + k_i * int(t)$. Bound the control input to lie between the permissible lower and upper limits. (This constraint is imposed by the limitations of the physical system such as the water-pump in this case)

8. Record the steady state controller output for the given combination of inputs.

Neural Controller Architecture Design and Training The architecture of the steady-state deep neural controller (Fig 2.16) is similar to that of $n-$ step advance neural controller discussed in the previous section. The fundamental difference is in the number of neurons in the input-layer. For the steady state neural controller, we use 4 neurons in the input layer which corresponds to the chosen inputs $f_a, T_{ai}, T_{wi}, T_{set}$. Similar to our $n-$step advance neural controller, a neuron in this steady-state neural network controller receives inputs from other neurons, calculates the weighted sum of the inputs (based on neural controller parameters W) and maps the sum through an activation function to the output of steady state PI controller. This output is passed as the input to the connected neurons in the next layer of the network. As with other networks in our work, we use a 2-layered network with twelve and eight neurons respectively in the hidden layers and one neuron in the output layer. RELU ($f(x) = max(0, x)$) is the choice of activation function for the neurons in the hidden layers. We chose sigmoid ($\sigma(x) = \frac{1}{1+e^{-x}} \in (0, 1)$) as the activation function for the final layer as the objective of the neural controller is to predict the normalized PI control output ($c'(t)$ which maps to the control output as $c(t) = 1400 - 730c'(t)$ for this water-air heater system).

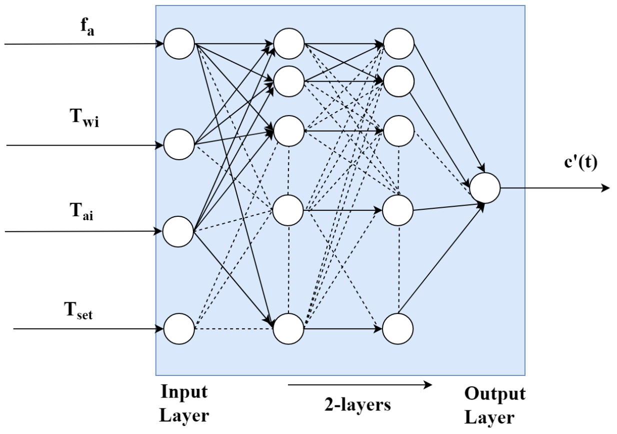


Figure 2.16: **Architecture of Steady state deep neural controller**

The objective of training the steady-state neural controller module is to learn the optimal weights between the neurons in various network layers so that the prediction error ($(c_i - \hat{c}_i)^2$) is the prediction error for the i^{th} data sample) is minimized. Mathematically, this objective is the equivalent of minimizing a loss function ($\mathscr{L} : \mathbb{R}^4 \to \mathbb{R}$) that maps from the input space ($\mathbb{X} := \{X \in \mathbb{R}^4 : X = [T_{ai}, T_{wi}, f_a, T_{set}]\}$) to the output space ($\mathbb{Y} := \{Y \in R : Y = c',\ c'$ is the steady state controller output of PI controller$\}$) over the parameter space $\mathbb{W}$ of the neural controller. This objective is formulated as:

$$W^* = argmin_{W \in \mathbb{W}} \sum_i L(c_i, \hat{c}_i) \qquad \text{with} \qquad L(c_i, \hat{c}_i) = (c_i - \hat{c}_i)^2 \tag{2.5.1}$$

This minimization problem is solved using Stochastic gradient descent algorithm with Nesterov momentum (Chapter 1 eq. 1.3.20-1.3.21). As seen before, this optimization algorithm requires the computation of the loss function gradient with respect to the parameters of the steady-state neural controller. This gradient is calculated using back-propagation algorithm (Chapter 1 eq. 1.3.6-1.3.16) which is an efficient algorithm for computing gradients of a loss function with respect to its parameters. We generate the training dataset for the steady state deep neural controller based on the data-generation scheme presented earlier (Iterative scheme (1)-(8)). Before training, we randomly partition the dataset into 2 parts- (1) training dataset: About 80% of dataset was randomly selected to train network parameters (2) Validation data-set: About 20% of data was used to optimize the hyper-parameters such as optimization algorithm's step-size, batch-size and stopping criterion.

As the neural controller was trained, we monitored the validation loss and stopped training after 100 epochs after which there was no further improvement in its value. We tuned the hyper-parameters (learning rate α, batch-size n) of stochastic gradient descent algorithm used for training this neural controller using an approach similar to that of $n-$ step advance neural controller (refer Section 2.4.1 hyper-parameter tuning, Fig. 2.10). The optimal values for these hyper-parameters- learning rate (α), batch-size (n) is determined to be 0.1 and 32 respectively, for the chosen water-air heater system. The effectiveness of the trained model is evaluated by computing its co-efficient of determination (R^2) (eq 2.3.2(f)). The model R^2 for the trained steady state neural controller is found to be 98%. We deploy this trained steady state neural controller in tandem with a proportional controller and study its tracking performance characteristics in the following section.

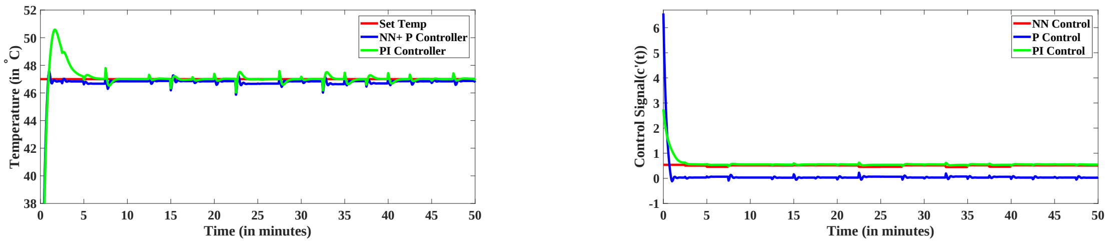
2.5.3 Controller Performance Characteristics

The hybrid controller model's performance characteristics are evaluated by subjecting it to two test set-point air-outlet temperature signals over a 50 minute duration- (1) a constant temperature of $47°C$ and (2) 40-sample periodic square pulse signal switching between $40°C - 45°C$ every 20 steps (200s period). The same set of test set-point temperature signals is used for evaluating the $n-$ step advance neural controller.

This forms the basis for comparing the characteristics of the proposed control algorithms with conventional PI control. Here, we train the neural controller to predict the steady state output of PI controller with $k_p = 0.075$, $k_i = 0.001$. We present the control characteristics of this trained neural controller which is augmented with a proportional controller whose gain(k_p) is selected from the set $k_p = 0.1, 0.15, 0.20$. Later, we develop an approach for optimally tuning this proportional controller gain (k_p) for the entire operating range of air-outlet temperatures for this heater system.

Prop. Gain (K_p)	**Rise-Time (s)**	**Max. Overshoot (%)**	**Steady State Error ($^\circ C$)**	**Variance ($(^\circ C)^2$)**
PI (No NN control)	60s	7.63%	$0.78^\circ C$	1.13
0.1	45s	2.24%	$0.33^\circ C$	0.03
0.15	45s	1.98%	$0.27^\circ C$	0.02
0.2	45s	1.96%	$0.22^\circ C$	0.02

Table 2.5: **Performance characterization of hybrid controller with different proportional gains for constant temperature set-point signal of 47°C**



(a) **Temperature Tracking**

(b) **Control Effort**

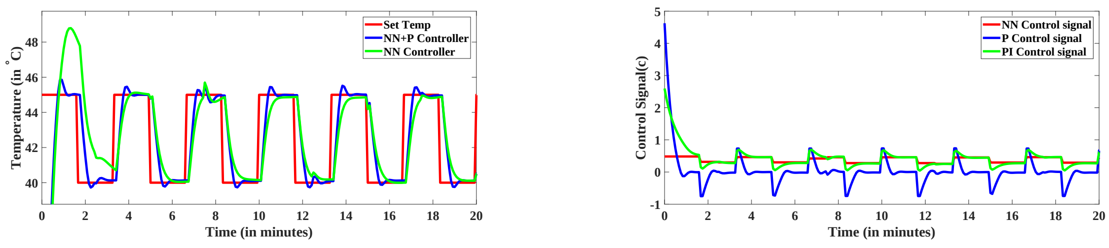
Figure 2.17: **Tracking Performance of P+NN controller with** $k_p = 0.2$ **vs PI controller for constant set point temperature signal of** $47^\circ C$ **(2.17(a)), Control effort comparison of P+NN controller vs PI (2.17(b))**

Based on our observations from Tables 2.5, 2.6 and figs (2.17, 2.18), we observe that the key metrics of evaluation of control response characteristics such as rise-time, maximum overshoot and steady state error all decrease with the increase in proportional gain (k_p) of the hybrid controller module from 0.1 to 0.2. Further, all the three considered proportional gains of the hybrid controller outperform PI control in tracking both the air-outlet set-point temperature signals. We also see from the control effort plots in Figs 2.17(b) and Fig 2.18(b) that the neural controller operates in a slow time-scale while the proportional controller operates in real-time to correct for deviance from the set-point temperature at that time instant. The control effort plots also show that the effort of the proportional controller is much smaller than that of a PI controller as the neural controller provides a fraction of the control response that it has learnt from the input flow-rate and temperature variables. Thus, this hybrid control module has the potential to save energy consumption from heat-exchanger control operations.

The control effort plots in Figs 2.17(b) and Fig 2.18(b) also give us an understanding of the working of the hybrid controller module. The neural controller module reduces the order of the water-air heater system around its local operating points. Thus, the proportional controller then operates on a reduced order model of the heat-exchanger which improves its robustness of operation.

Prop. Gain (K_p)	**Rise-Time (s)**	**Max. Overshoot (%)**	**Steady State Error ($°C$)**
PI (No NN control)	36.25s	3.05%	$0.12°C$
0.1	23.12s	2.72%	$0.08°C$
0.15	22.5s	2.67%	$0.07°C$
0.20	22.5s	2.33%	$0.06°C$

Table 2.6: **Performance characterization of hybrid controller with different proportional gains for square pulse temperature set-point signal between $40°C$ and $45°C$**



(a) **Temperature Tracking**

(b) **Control Effort**

Figure 2.18: **Tracking Performance of P+NN controller with $k_p = 0.15$ vs PI controller for periodic square pulse set point temperature signal between $40°C - 45°C$ (2.18(a))(zoomed over first 20 minutes of operation), Control effort comparison of P+NN controller vs PI (2.18(b))**

We explore the possibility of replacing the proportional controller with PI controller in our hybrid control algorithm. We see that the addition of an integral gain to the proportional control in our hybrid control module worsens the set-point temperature tracking characteristics of the controller (Table 2.7). This is because of the integral wind-up problem observed in PI controllers. The neural controller causes an offset in the operating set-point of the PI controller. This frequent change in the PI controller's set-point causes its integral term to rise-up and creates large overshoots as we observe. Further, as the PI controller corrects for this large error, there is another offset in the PI controller's set-point caused by the neural controller's response. As a result, the integral control component never really settles down. This results in sub-optimal control performance.

Prop. Gain (K_p)	**Integral Gain (k_i)**	**Max. Overshoot (%)**	**Steady State Error ($^\circ C$)**
0.15	0	2.67%	$0.07^\circ C$
0.15	0.001	3.87%	$0.28^\circ C$
0.15	0.005	7.93%	$0.33^\circ C$
0.15	0.01	10%	$1.22^\circ C$

Table 2.7: **Performance characterization of hybrid controller with different proportional gains and integral gains for square pulse temperature set-point signal between $40^\circ C$ and $45^\circ C$**

Thus, we eliminated the need for an integral component in our hybrid control module design. This is particularly advantageous since we have effectively reduced the parametric space from which our controller gain has to be chosen. We had earlier seen that the tuning of PI control gains is challenging for industrial heat-exchangers with large unmodeled dynamics. With our steady state neural network + proportional controller design, we have to optimize only the proportional gains for the specific heat-exchanger.

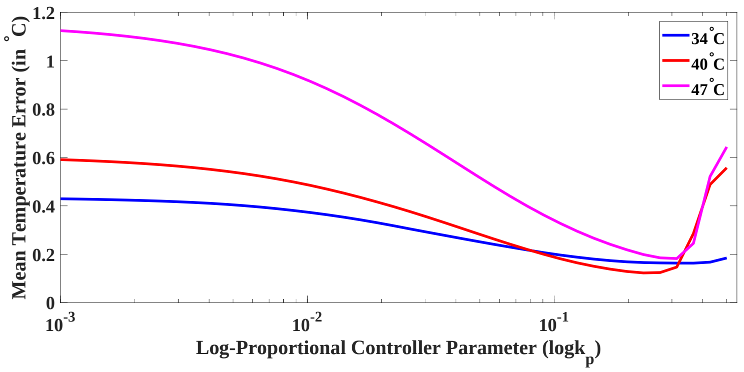


Figure 2.19: **Effect of proportional controller gain (k_p) of the hybrid control module in tracking constant set-point air outlet temperatures**

We present an approach for optimizing the proportional gain in our hybrid controller. We create a geometric series of possible proportional gain (K_p) for our hybrid control algorithm from a closed set of gain values between 0.001 and 1 (The lower and upper range are defined by the specific heat exchanger's process gains). We evaluate the steady state tracking error for our hybrid control scheme for each of these proportional gains on 6 test signals for a 50 minute duration- (1) Constant set-point air outlet temperature signals of $34^\circ C, 40^\circ C, 47^\circ C$ (2) Square pulse signals of 200s-period with a range of $35^\circ C - 40^\circ C$, $40^\circ C - 45^\circ C$, $42^\circ C - 47^\circ C$. Figs 2.19,2.20 show that the steady state temperature tracking error curves for all these set-point signals flatten out for proportional gains (k_p) between [0.1,0.16]. Thus, choosing a proportional gain (k_p) for our hybrid controller from this range would ensure excellent air-outlet set-point temperature signal tracking characteristics for this water-air heater system. Thus, this tuning approach of our hybrid control module overcomes some of the tracking range restrictions and tuned gain's high sensitivity issues faced by a conventional PI controller.

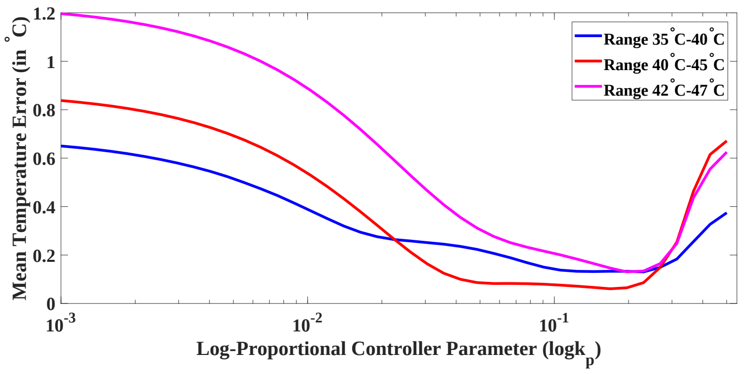


Figure 2.20: **Effect of proportional controller gain (k_p) of the hybrid control module in tracking periodic square pulse set-point air outlet temperatures**

2.6 Comparing Proposed Control Designs with PI and Open loop

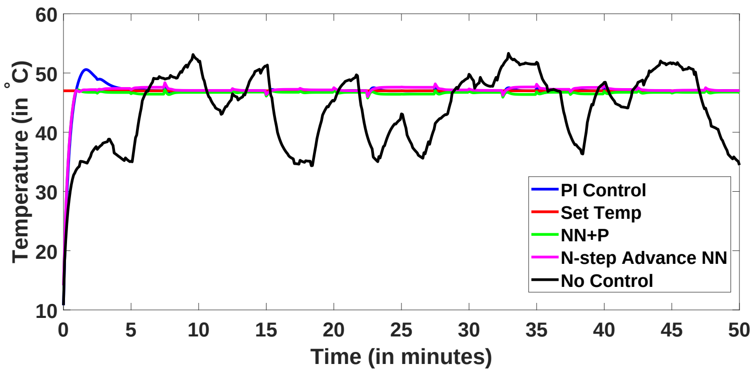


Figure 2.21: **Tracking performance of different control designs for constant temperature set-point signal of 47°C**

Control Type	Max. Overshoot (%)	Steady State Error ($^\circ C$)	Variance ($(^\circ C)^2$)
Open Loop	13.46%	6.57$^\circ C$	16.93
PI	7.63%	0.78$^\circ C$	1.13

5-Step Advance NN	1.96%	$0.11°C$	0.03
NN+ P Control ($k_p = 0.15$)	1.98%	$0.27°C$	0.02

Table 2.8: **Tracking performance metrics of different control designs for constant temperature set-point signal of 47°C**

In this section, we quantify the tracking performance improvements obtained through the proposed $n-$ steps advance neural controller and hybrid controller module over PI control and no control (open loop) cases. We compare the best performing $n-$ steps advance neural controller and hybrid neural + proportional controller with optimized gains with PI control and open loop by evaluating their tracking characteristics on two test set-point temperature signals- (1) a constant temperature of $47°C$ and (2) 40-sample periodic square pulse signal switching between $40°C - 45°C$ every 20 steps (200s period). These tracking characteristic tests are consistent with our controller evaluation schemes in Sections (2.4) and (2.5). We observe that both the proposed control approaches, based on $n-$ steps advance neural controller and hybrid steady state neural network+ proportional controller, outperform conventional PI controller by over 70% in the case of both the test set-point temperature signals (Tables 2.8,2.9, Figs 2.21,2.22). While the $n-$ step advance neural network controller has the lowest mean set-point temperature tracking error, the hybrid control scheme has the quickest response to a given set-point temperture signal. The hybrid steady state neural network + proportional controller with a gain of 0.15 is the best control choice for this water-air heater system as it has excellent set-point temperature tracking characteristics and guaranteed bounded input-bounded output (BIBO) stability on its deployment in real-time.

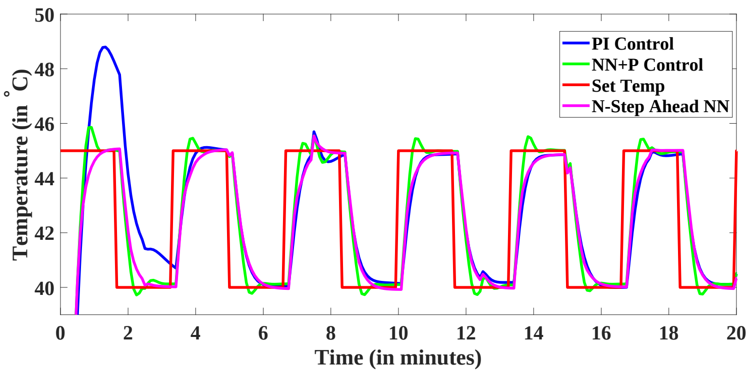


Figure 2.22: **Tracking performance of different control designs for periodic square pulse set-point temperature signal between** $40°C$ **and** $45°C$

Control Type	Rise-Time (s)	Max. Overshoot (%)	Steady State Error ($°C$)

PI	36.25s	3.05%	$0.12°C$
5-Step Advance NN	35.6s	1.99%	$0.03°C$
NN+ P Control ($k_p = 0.15$)	22.5s	2.67%	$0.07°C$

Table 2.9: **Performance characterization of hybrid controller with different proportional gains for square pulse temperature set-point signal between** $40°C$ **and** $45°C$

2.7 Conclusions and Future Work

. The developed control approaches based on $n-$ step advance neural networks and hybrid controller based on neural network + proportional controller delivered superior tracking characteristics compared to conventional PI controller on simulated water-air heater system. The 5-step advance neural controller had the least steady state air outlet temperature tracking error and the least maximum overshoot compared to all other control algorithms. The hybrid control algorithm based on a steady-state neural network with proportional controller had lesser steady state error compared to PI controller in tracking both constant and high-frequency set-point air-outlet temperature signals. Further, we developed a mechanism for optimal tuning of proportional gain in the hybrid controller module. We also ensured the stable operation of the hybrid controller with our network design choices. The hybrid controller module is favorable for real-time deployment over the $n-$ step ahead neural controller which offers no theoretical guarantees for stability. Further, the hybrid control module's gain is robust to small disturbance and operates across the entire range of heat-exchanger's operating points. The broad direction of future work will be to study various ways of combining other learning algorithms with conventional control architectures. For instance, we could augment the output of a PI controller with the output from a reinforcement learning agent for obtaining better tracking characteristics.

Chapter 3

Fouling Resistance and Prediction Module

3.1 Problem Set-up

The data generation scheme seeks to model the fouling resistance of the heat exchanger from its operating conditions. First, we provide a description of the geometry of the simulated flue-gas to water heat exchanger as well as its range of operating flow-rates and temperatures. We propose modifications to existing empirical correlations for estimating flue-gas side and water side heat transfer coefficients of a cross-flow heat exchanger to account for fouling. Fouling impacts the overall heat transfer coefficient of the heat exchanger by causing a reduction in fluid flow-rates (hydrodynamical effects) and by increasing the wall thermal resistance to thermal transport (thermal effects). The effectiveness-NTU ($epsilon - NTU$) scheme [61, 62] on fouling incorporated thermal models of the heat-exchanger generates the dataset used for training the prediction algorithm. This dataset captures the deterioration in the simulated heat exchanger's performance under progressively increasing fouling conditions.

3.1.1 Simulated Heat Exchanger Model Description

The modeled cross-flow heat exchanger consists of banks of water carrying tubes placed in cross-flow to a flue gas stream flowing in a stack. A series of water tubes, each with outer diameter D_o, thickness t, length L, wall thermal conductivity k_{tube}, is placed in an inline square arrangement with a equal spacing s, along lateral and flow directions. Corbels are attached to the stack walls to regulate the flow of flue-gas towards the tubes bank (Fig. 3.1). The hot flue-gas enters the duct with temperature T_{fi}, flow rate $\dot{M}_{flue}$ and reaches a peak velocity of V_{max} as it flows over the tubes bank. The hot flue-gas loses heat to cold water flowing in the tubes to ultimately exit the duct at a lower temperature T_{fo}. Heat that is lost from the flue-gas is gained by water flowing in the tubes at a flow-rate $\dot{M}_w$. This increases its temperature from T_{wi} at the pipe inlet to T_{wo} at its exit.

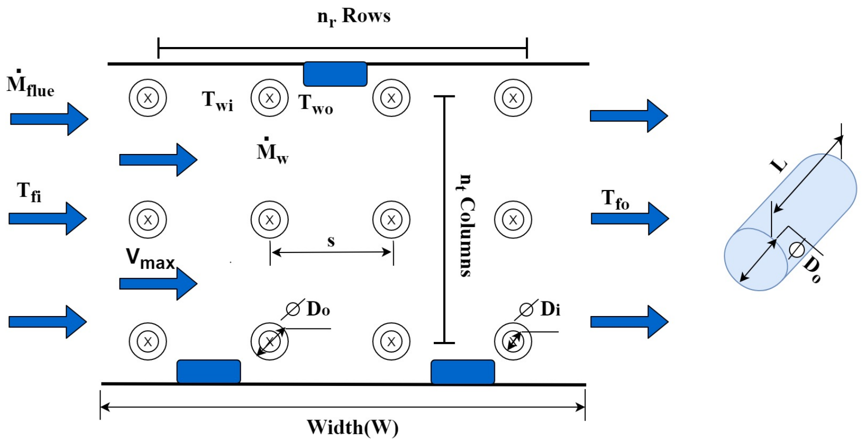


Figure 3.1: **Geometry of cross-flow tube arrays: The numerical values of heat-exchanger parameters used in simulations are: Number of tubes, $N = 150$, tube outer diameter $D_o = 3cm$, thickness $t = 1cm$, length $L = 1.5m$, wall thermal conductivity $k_{tube} = 10Wm^{-1}K^{-1}$, spacing $s = 1.5cm$, flue duct width and height $W, H = 0.5m$, duct length $L = 1.5m$, number of tubes in flow direction $n_r = 15$, number of tubes normal to flow direction $n_t = 10$**

Range of Input Operating conditions The inlet temperatures and flow-rates for the heat exchanger are chosen for a typical waste heat recovery application. A range of input parameter values for the cross-flow heat exchanger is chosen for a maximal rating of about 0.5MW (Table 3.1). Further, the maximum values of fouling factor values have been fixed assuming the operating fluid to be coal flue gas (mixed stream) and river water (unmixed stream) [7]. The choice of parameter values based on Table 3.1 ensures that the training data-set for the proposed fouling model covers a wide-range of operating points and the corresponding predictions from the fouling model can be generalized across the entire operating region of the heat exchanger.

Heat Exchanger Variable	**Operating Range**
Flue gas Flow rate($\dot{M}_{flue}$)	$4kgs^{-1} \leq \dot{M}_{flue} \leq 8.5kgs^{-1}$
Water Flow rate($\dot{M}_w$)	$0.75kgs^{-1} \leq \dot{M}_w \leq 10.65kgs^{-1}$
Flue inlet Temperature(T_{fi})	$170°C \leq T_{fi} \leq 250°C$
Water inlet Temperature(T_{wi})	$20°C \leq T_{wi} \leq 30°C$
Flue-gas side Fouling Factor(f_1)	$0(m^2.K/W) \leq f_1 \leq 0.00175(m^2KW^{-1})$
Water side Fouling Factor(f_2)	$0(m^2.K/W) \leq f_2 \leq 0.00053(m^2KW^{-1})$

Table 3.1: Range of input flow-rate and temperature parameter values in simulation motivated by [1, 2]

3.1.2 Fouling Models for Flue-gas and Water Side Heat Transfer Coefficient

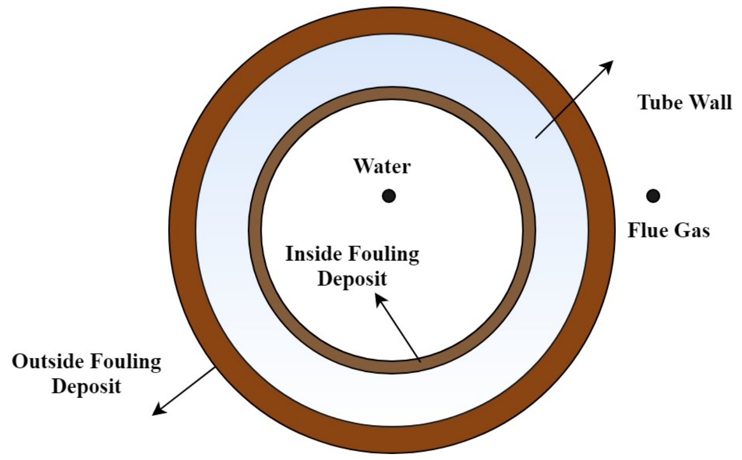


Figure 3.2: **Schematic of fouling layer deposits on heat exchanger pipe walls [3]. We show a cross section with fouling assumed on either sides of the pipe wall**

Water-Side Fouling and Heat-Transfer Model The fouling deposit on interior of the pipes (Fig 3.2) can occur due to crystallization of unwanted salts present in treated river-water that flows through these pipes. The frictional pressure drop in the tube for a single-phase flow is given by [63]:

$$\triangle p = 4f\left(\frac{L}{D_i}\right)\frac{\rho U_m^2}{2}, \tag{3.1.1}$$

where f, L, D_i, ρ and U_m in eq. 3.1.1 respectively represent the fanning friction factor, length, pipe inner diameter, density and mean flow velocity of water. The fouling layer causes the inner diameter of the pipe to reduce (Fig 3.2) while increasing the roughness of the pipe surface. This causes an increase in the pressure drop according to eq. 3.1.1.

$$\frac{\triangle p_f}{\triangle p_c} = \frac{f_f}{f_c}\left(\frac{D_i}{D_{f,i}}\right)\left(\frac{U_f}{U}\right)^2 \tag{3.1.2}$$

where subscripts f and c represent properties under fouled and clean (unfouled) flow conditions. Further, it is reasonable to assume that the Fanning friction factor under fouled and clean conditions do not differ greatly [3]. If the pressure drop under fouled and un-fouled conditions is held constant under fouled and clean conditions (assuming the pump head to be the same under both operating circumstances), the ratio of fouled inner pipe diameter($D_{f,i}$) to the clean inner pipe diameter(D_i) is related to the ratio of the fouled water flow velocity(U_f) to the water flow velocity under clean conditions (U) as [3]:

$$\frac{D_i}{D_{f,i}} = \left(\frac{U}{U_f}\right)^2, \tag{3.1.3}$$

where D_i and $D_{f,i}$ are also related through the fouling layer thermal conductivity(k_f) and the water-side fouling factor(f_2) as:

$$ln\left(\frac{D_i}{D_{f,i}}\right) = \frac{2k_f f_2}{D_i} \qquad \text{that is} \qquad D_{f,i} = D_i e^{-\frac{2k_f f_2}{D_i}}. \tag{3.1.4}$$

Equation 3.1.4 brings out how the pipe diameter decreases exponentially with an increase in fouling conductivity(k_f) and water-side fouling resistance(f_2). The fouling thermal conductivity(k_f) depends on the nature of material deposit on the pipe walls. In our simulations, we chose $k_f = 2.3Wm^{-1}K^{-1}$, to reflect deposits due to calcium and magnesium salts that are widely present in river water [3]. For fouled conditions, the Reynolds number and mass flow rate, based on the fouled inner pipe diameter($D_{f,i}$) and the fouled mass flow rate($\dot{M}_{w,f}$), are given by:

$$Re_{w,f} = \frac{U_f D_{f,i}}{\nu} \qquad \text{and} \qquad \dot{M}_{w,f} = \rho\left(\frac{\pi D_{f,i}^2}{4}\right)U_f, \tag{3.1.5}$$

which can be used to estimate the friction factor [64, 65], Nusselt number [66] and in turn the water-side heat transfer coefficient(h_w) [63, 67] from empirical correlations given by:

$$f(Re_{w,f}) = \begin{cases} (1.58.ln(Re_{w,f} - 3.28)^{-2} & \text{if } Re_{w,f} > 2300 \\ \frac{16}{Re_{w,f}} & \text{otherwise} \end{cases} \tag{3.1.6}$$

and

$$Nu_w(D_{f,i}, L, Re_{w,f}, Pr) = \begin{cases} \left(\frac{f(Re_{w,f})}{2}\right).\frac{(Re_{w,f}-100).Pr}{1+12.7.\left(\frac{f(Re_{w,f})}{2}\right)^{0.5}.(Pr^{\frac{2}{3}}-1)} & \text{if } Re_{w,f} > 2300 \\ 1.86.\left(\frac{D_{f,i}.Re_{w,f}.Pr}{L}\right)^{\frac{1}{3}} & \text{otherwise} \end{cases} \tag{3.1.7}$$

Flue gas-Side Fouling and Heat-Transfer Model A fouling layer develops on the outside of pipes due to contaminants in flue-gas exhaust streams. The rate of fouling depends on the composition and the phase of waste heat streams [39]. The extent and rate of fouling tends to be more severe on the flue-gas side as seen in Table 3.1. For flue gas side fouling, we adopt an approach similar to the water side fouling model. The mean pressure drop across a single row of pipes ($\triangle\overline{p}_{row}$) in a tube bank is often expressed in terms of an Euler number(Eu) [68] as:

$$\triangle\overline{p}_{row} = Eu.\frac{\rho V_{max}^2}{2} \qquad \text{and} \qquad V_{max} = \frac{\dot{M}_{flue}}{\rho.n_t.(s - D_o)L} \tag{3.1.8}$$

where V_{max} is the maximum flue gas flow-velocity calculated for the minimum flow area between tubes normal to the local flow direction [69]. An analytical expression for Eu is obtained empirically from approximation of several Euler number(Eu) Vs Reynolds number(Re) experimental curves [70, 71] for heat-exchangers with different parameters and flow configurations. This facilitates to calculate pressure drop under clean and fouled tube conditions as given by:

$$\frac{\triangle p_{flue,f}}{\triangle p_{flue,c}} = \left(\frac{Eu}{Eu_f}\right)\left(\frac{V_{max}}{V_{max,f}}\right)^2 \tag{3.1.9}$$

where $Eu = \sum_{i=0}^{i=4} \frac{c_i}{Re_{flue}^i}$ and the coefficients c_i are obtained from Ref. [71] for the chosen cross-flow configuration. Therefore eq. 3.1.9 is rewritten as:

$$c_0x^4 + \left(\frac{c_1\nu_{flue}}{D_{f,o}}\right)x^3 + \left[\frac{c_2\nu_{flue}^2}{D_{f,o}^2} - \left(c_0 + \frac{c_1}{Re_{flue}} + \frac{c_2}{Re_{flue}^2} + \frac{c_3}{Re_{flue}^3} + \frac{c_4}{Re_{flue}^4}\right)\right]x^2 + \left(\frac{c_3\nu_{flue}^3}{D_{f,o}^3}\right)x + \frac{c_4\nu_{flue}^4}{D_{f,o}^4} = 0 \tag{3.1.10}$$

where,

$$Re_{flue} = \frac{V_{max}D_o}{\nu_{flue}} \qquad \text{and} \qquad D_{f,o} = D_o e^{\frac{2k_f f_1}{D_o}} \tag{3.1.11}$$

Further, an analytical expression (eq. 3.1.11) similar to eq. 3.1.4 correlating external pipe diameter(D_o) to fouled external pipe diameter($D_{f,o}$) in terms of fouling layer thermal conductivity(k_f), is obtained to create a bi-quadratic equation (eq. 3.1.10). One of the positive roots of the bi-quadratic equation, whose magnitude is greater than maximum flow velocity(V_{max}) is the fouled maximum flow velocity($V_{max,f}$). The flue gas-side Reynolds number($Re_{flue,f}$) and mass flow rate($\dot{M}_{flue,f}$) under fouled conditions, are evaluated based on $V_{max,f}$ (eq. 3.1.12). They are then used in suitable empirical correlations (eq. 3.1.13) for determining the heat transfer coefficient on the flue gas side(h_{flue}) [72, 73, 74, 7].

$$Re_{flue,f} = \frac{V_{max,f} D_{f,o}}{\nu_{flue}}, \qquad \dot{M}_{flue,f} = \rho . n_t .(s - D_{f,o}).L.V_{max,f} \qquad \text{and}$$

$$Nu_{flue}(D_{f,o}, Re_{flue,f}, L, Pr_{flue}) = \frac{h_{flue}.D_{f,o}}{k_{flue}} = 0.211(Re_{flue,f})^{0.651}(Pr_{flue})^{0.34}. \tag{3.1.12}$$

3.1.3 Outlet Temperature Computation using ϵ-NTU

1. Randomly select inlet flow-rates (clean conditions), inlet temperatures and fouling factors for flue-gas and water from multi-variate random distribution within working range (Table 3.1)
2. Guess a feasible solution for flue-gas exit temperature ($T_{fo,guess} \leq T_{fi}$) and initialize another variable T_{fo} to be equal to T_{fi}
3. Repeat (a)-(g) till $|T_{fo,guess} - T_{fo}| > 10^{-6}$
   - (a) Evaluate the flue gas fluid properties at the flue bulk mean-temperature
   - (b) Compute flue-side fouling variables $D_{f,o}, V_{max,f}, \dot{M}_{flue,f}$ and h_{flue} using equations (10),(11),(12),(13)
   - (c) Evaluate $T_{wo,guess}$ from energy balance and evaluate water properties at water bulk-mean temperature
   - (d) Determine water-side fouling variables $D_{f,i}, U_f, \dot{M}_{w,f}$ and h_w using equations (3.1.4),(3.1.5),(3.1.6),(3.1.7)
   - (e) Calculate Overall heat transfer co-efficient($U = f(h_{flue}, h_w, R_1, R_2)$), dimensionless number of transfer units ($NTU = f(U, (\dot{M}C_p)_{min})$) and heat exchanger efficiency ($\epsilon = f(U, NTU, C_r)$)
   - (f) Obtain T_{fo} and T_{wo} from ϵ computations in previous step
   - (g) Update $T_{fo,guess}$ as $\frac{T_{fo,guess}+T_{fo}}{2}$ and $T_{wo,guess}$ as $\frac{T_{wo,guess}+T_{wo}}{2}$
4. Record the generated heat-exchanger operating data sample along with their corresponding fouling resistances (R_1,R_2)

Table 3.2: Algorithmic Overview for data-generation scheme

We use the ϵ-NTU method (Table 3.2) for computing the outlet fluid temperatures on the flue-gas and water side. The analytical expression for overall heat transfer co-efficient of the cross-flow heat exchanger (eq. 3.1.14) incorporates the thermal effects of fouling through the addition of both the thermal resistance terms (R_1, R_2) while the hydrodynamic fouling effects is accounted for in the heat transfer coefficient designs on the flue gas and water sides (h_{flue}, h_w). The overall heat transfer coefficient is then used to calculate the number of transfer units (NTU) and the effectiveness (ϵ) of the heat exchanger. The iterative scheme used for generating heat exchanger operation data is summarized in Table 3.2.

$$\frac{1}{UA} = \frac{1}{h_w * A_{f,i}} + \frac{log(\frac{D_o}{D_i})}{2 * \pi * L * k_{tube}} + \frac{1}{h_{flue} * A_{f,o}} + R_1 + R_2 \tag{3.1.13}$$

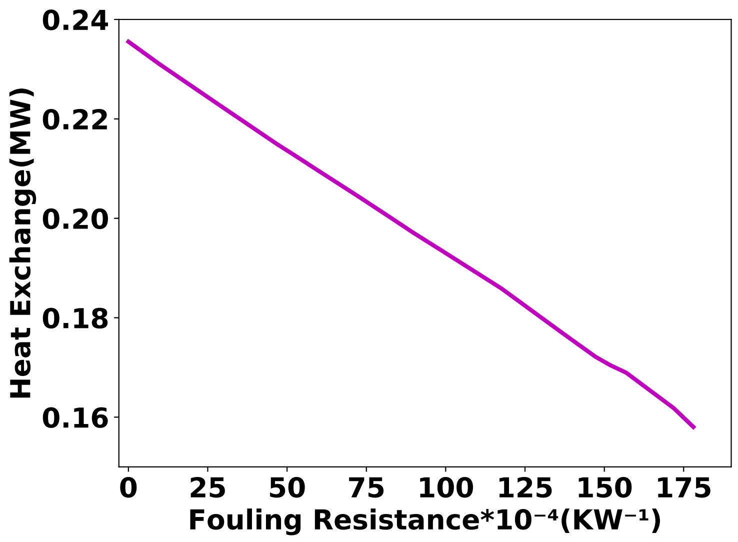
$$NTU = \frac{UA}{C_{min}} \qquad \text{and} \qquad \epsilon = 1 - e^{-\frac{(1-e^{NTU.C_r})}{C_r}} \tag{3.1.14}$$
$$\text{where } C_r = \frac{C_{min}}{C_{max}} = \frac{min(\dot{M}_{flue,f}C_{p,flue}, \dot{M}_{w,f}C_{p,w})}{max(\dot{M}_{flue,f}C_{p,flue}, \dot{M}_{w,f}C_{p,w})}$$

3.1.4 Model Convergence and Fouling Effects

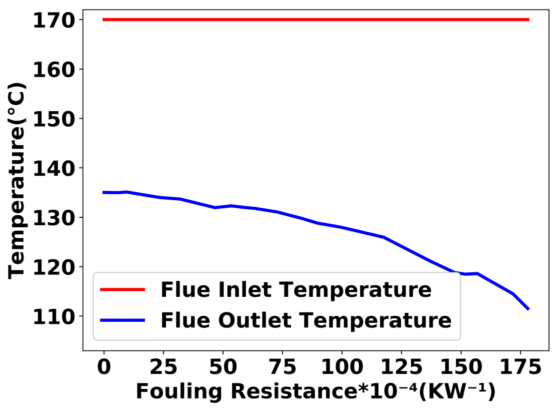
$\mathbf{\dot{M}_{flue,f}}$	$\mathbf{T_{fi}}$	$\mathbf{\dot{M}_{w,f}}$	$\mathbf{T_{wi}}$	$\mathbf{f_1 \times 10^5}$	$\mathbf{f_2 \times 10^5}$	$\mathbf{T_{fo}}$	$\mathbf{T_{wo}}$	$\mathbf{e_{temp} \times 10^{-15}}$	$\mathbf{e_{\dot{Q}} \times 10^{-11}}$
4.23	198.08	4.69	25.13	8.69	1.32	136.20	39.66	0	5.82
4.87	242.22	5.53	29.45	68.2	2.47	165.23	47.25	7.11	11.6
3.70	189.34	1.65	24.04	61.8	5.48	142.03	51.69	0	0
3.99	235.7	1.09	20.15	51.9	12.8	182.35	71.46	14.2	0
2.63	229.58	6.66	21.89	135	21.7	126.59	32.54	14.2	5.82

Table 3.3: Input-Output Data from Fouling Data generation algorithm: The units of inlet and outlet temperature variables and flow rates are $^\circ C$ and kgs^{-1} respectively. The error in convergence of the temperature variables ($\mathbf{e_{temp}}$) is defined as: $e_{temp} = max(|T_{fo,guess} - T_{fo}|, |T_{wo,guess} - T_{wo}|)$. The variable $\mathbf{e_{\dot{Q}}}$ measures the difference in energies of the flue gas side and water side which can be defined as: $e_{\dot{Q}} = \left|[\dot{M}_{flue,f}C_{p,flue}(T_{fo} - T_{fi})] - [\dot{M}_{w,f}C_{p,w}(T_{wo} - T_{wi})]\right|$. The data-generation scheme satisfies both energy balance and temperature convergence criteria as illustrated by their negligible error magnitudes. The errors are even lower than the system's minimum floating point standard which results in their magnitudes to be zero in certain cases. Once we ascertain the temperature convergence and energy balance, the heat exchanger rating ($\mathbf{\dot{Q}}$) is computed as: $\dot{Q} = \dot{M}_{flue,f}C_{p,flue}(T_{fo} - T_{fi})$.

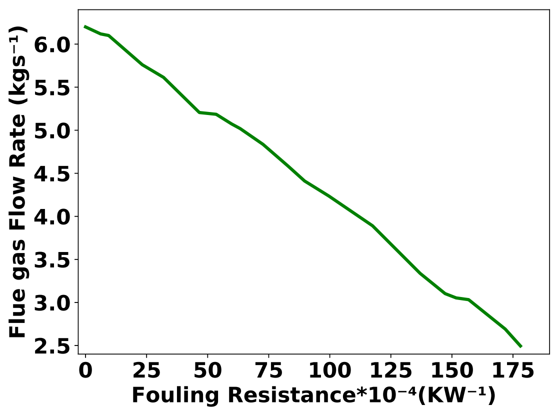
The convergence of iterations in the effectiveness-NTU method was verified through 15000 test runs (at every data sample corresponding to the size of the dataset used for training the prediction algorithm subsequently explained) and ensuring that the guess temperatures converged to the actual exit temperatures. The method was further verified by checking the energy balance - that is, ensuring that the energy lost by the flue gas is equal to the energy gained by the water. This 2-step approach ensured the accuracy and consistency of the fouling data generation scheme. The data-sample quality evaluation results are briefly summarized in the Table 3.3 for randomly selected 5 data samples.



(a)



(b)



(c)

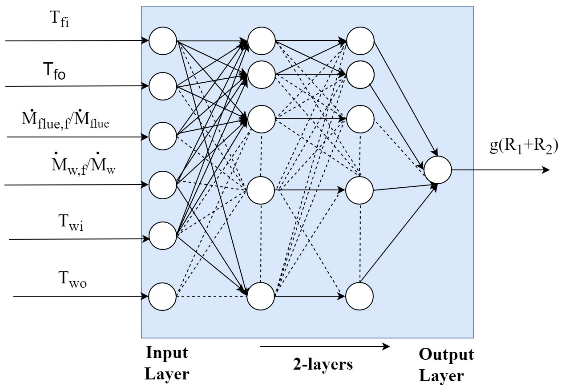
Figure 3.3: **Effect of increasing overall fouling resistance ($\mathbf{R_1 + R_2}$) on heat exchanged($\dot{Q}$) (a), flue gas outlet temperature ($\mathbf{T_{fo}}$)(b) and flue gas flow rate ($\mathbf{\dot{M}_{flue,f}}$)(c)**

Table 3.3 demonstrates that our data generation approach is accurate and robust. Therefore, our $\epsilon - NTU$ model with fouling incorporated is fairly representative of a cross-flow heat exchanger used in waste heat recovery systems. The graphs in Fig. 3.3 show the variation in flue gas outlet temperature and the quantity of heat exchanged for $T_{fi} = 170°C, \dot{M}_{flue} = 6.2kg/s, T_{wi} = 25°C, \dot{M}_w = 3.54kg/s$. While the exit temperature of the flue gas decreases from about $135°C$ to $111.5°C$ (Fig. 3.3(b)), the flue gas flow rate was greatly reduced from $6.2kgs^{-1}$ to $2.5kgs^{-1}$ (Fig. 3.3(c)) due to high fouling effects. Consequently, there is a reduction in the quantity of heat exchanged which drops by about $0.07MW$ (Fig. 3.3(a)). These inefficiencies in heat exchanger's operations and its reduced energy performance induced by high fouling rates, present a strong case for monitoring fouling in real-time. This objective can be achieved through an algorithm that makes real-time fouling resistance predictions based on operating conditions of the heat exchanger. Such an algorithm can be used to substantially reduce operating costs through timely scheduling of heat-exchanger cleaning and maintenance operations while benefiting environment through reduced energy and material wastage.

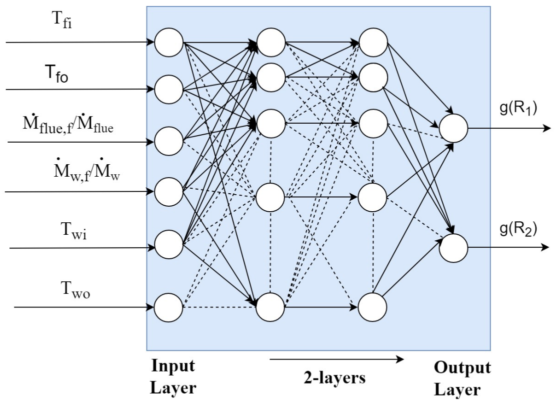
3.2 Solution Approach

Core Algorithmic Idea Modeling fouling based on first principles is challenging due to the numerous physical, chemical, or biological interactions between surfaces. Here, we develop an algorithmic framework to identify and map functional relationships between heat exchanger process variables and fouling. This approach requires sample data instances that are representative of relationships between process variables and extent of fouling at any given instant of time. We have adopted the data generation scheme, explained in section 3.1, for empirically obtaining these data samples in the absence of real industrial operation data. This can also be achieved through high fidelity numerical simulations using conjugate heat transfer and fluid flow solvers for most heat exchanger applications, with the availability of sufficient computational resources.

The key component of our algorithm is a neural network architecture that takes in operating conditions data of cross-flow heat exchanger as its input and predicts fouling resistances. This general approach can be applied to any heat-exchanger module. The inputs to the network are inlet fluid temperatures(T_{fi}, T_{wi}), ratio of fouled fluid flow-rates to flow-rates under clean circumstances ($\frac{\dot{M}_{w,f}}{\dot{M}_w}, \frac{\dot{M}_{flue,f}}{\dot{M}_{flue}}$) and outlet fluid temperatures (T_{fo}, T_{wo}). All of these predictors that have been considered for model development are easy to measure in an industrial setting through fluid flow-rate and temperature sensors at the inlet and exit of the duct and pipes respectively. Alternatively, pressure drop data can be used instead of fluid fouling flow ratio if the heat exchanger is operated under constant flow-rate conditions.



(a) Target: $g(R_1 + R_2)$



(b) Target: $g(R_1) + g(R_2)$

Figure 3.4: **Neural Network Architecture for predicting a) Overall Fouling Resistance($\mathbf{R_1 + R_2}$) b) Flue gas-side Fouling Resistance($\mathbf{R_1}$) & Water side fouling Resistance($\mathbf{R_2}$)**

We use techniques from deep-learning [38, 44] for training the neural network architecture of the fouling resistance module. In essence, a deep neural net is a specifically structured and parameterized function, often with millions of parameters, which can approximate complex nonlinear maps by determining appropriate parameters. These parameters are determined or *learned* by using *training data*; that is, the function parameters or *weights* are identified by fitting the observed values of function arguments - the *predictor variables* (e.g. operating conditions data such as measured temperatures, flow rates) and the corresponding function values - the target variables (e.g. fouling resistance). Such a fitting to learn fouling requires a large training set - a large dataset comprising the operating conditions variables with known values of fouling resistances on flue-gas and water sides. This data-set was generated using the scheme discussed in Table 3.2.

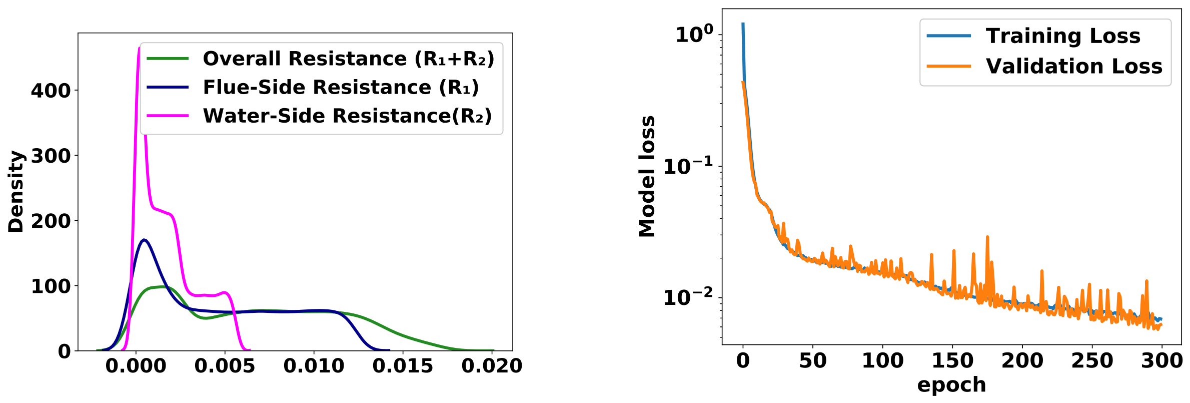
More specifically, a neural network is structured as a weighted directed graph where nodes are arranged in layers as shown in Fig. 3.4. Each node, a *neuron*, is a computing unit that outputs $z = f(\sum w_i x_i + b_i)$, where $\sum w_i x_i$ is a weighted sum of its inputs, b_i is the bias and f is a specific *activation* function. That is, all the edges going out from a neuron have this output, each of which is an input to a neuron of the next layer. The first layer termed as in the input layer (Fig. 3.4) takes in the inputs to the network and the number of neurons in this layer corresponds to size of functional inputs, e.g., the number of operating conditions in our case. The main task of *learning* or *training* a neural network is to determine the *weights* w_i for each neuron in the network. During the training process, the weight parameters of the neural network (mathematical function) are initially randomly initialized. For every data sample in the dataset, the predicted value of target variable (fouling resistance) is compared with the observed value and the weights are updated to minimize the prediction error (defined as $(y - \hat{y})^2$ where $\hat{y}$ is the network prediction and y is the observed value) on that data-sample. This process is repeated for every data sample in the training dataset several times until the network *learns* to accurately predict the fouling resistance for all data samples in the training dataset. With the right training data-set encompassing the entire range of operation of the heat-exchanger, the neural network can now be used to predict fouling resistance for new data samples. In our work, we segmented the dataset into training and validation datasets, where we trained the neural network on the training dataset and verified the performance of the trained network on the validation dataset. We describe this process below in detail.

Problem Formulation The proposed fouling prediction module has been set-up to solve two prediction problems specifically. The objective of the first is to predict the overall-fouling resistance ($R_1 + R_2$) as a function of heat-exchanger operating conditions (Fig. 3.4(a)), while the second problem requires predicting each of the individual fouling resistances (R_1 & R_2) using the same set of predictor variables (Fig. 3.4(b)). The proposed approach is very flexible, that is the same network structure can be used just by replacing the output neuron (for $R_1 + R_2$) with two neurons on the output layer (for R_1 and R_2). The $R_1 + R_2$ prediction quantifies the cumulative fouling effects, while predicting individually the resistances R_1 and R_2 gives a finer scale information of relative fouling levels on the inside and outside sides of the tubes. This approach can be used to get more specific (finer) information by increasing the number of neurons in the output layer. For instance, if data is available from different sections of a heat-exchanger, fouling resistances at each of those sections can be predicted. The accuracy of prediction in all these cases will depend on the data size and quality.

Fouling Prediction Network Architecture The fouling resistance architecture used in this study (Fig. 3.4) is a feed-forward neural network with six neurons in the input layer (corresponding to input data-size), two hidden layers with twelve neurons and eight neurons respectively and an output layer with one neuron and two neurons pertaining to prediction of overall fouling resistance ($R_1 + R_2$) and each of the individual fouling resistance (R_1 & R_2) respectively (Fig. 3.4(a),3.4(b)). We have used a activation function, Rectified Linear Unit (ReLU) function given by

$$\textbf{RELU: } f(x) = max(0, x). \tag{3.2.1}$$

RELU (eq. 3.2.1) is easily implementable by thresholding a matrix of activations at zero which results in creation of a sparse matrix. The alternative popular activation functions - sigmoid $\frac{1}{1+e^{-x}}$ and hyperbolic tangent $\frac{2}{1+e^{-2x}} - 1$ functions involve expensive matrix exponential computations and often result in dense matrices. RELU greatly accelerates the convergence of optimization algorithm (by a factor of six or more in certain cases) used for tuning the parameters of the neural network [54]. A linear activation function was used at the output layer of the network.



(a) (b)

Figure 3.5: **5(a) Fouling Resistance Distribution Plot- This plot depicts the density of fouling resistance values present in the training dataset and 5(b) Training-Validation Model loss curve- The loss curve shows the decline in fouling resistance prediction error as the weights of the network are tuned based on the training samples**

Data Pre-processing Input-output data to the network is suitably pre-processed to speed-up the training process and achieve faster convergence. Specifically, the inputs to the network are subjected to unity normalization (range:$[0, 1]$) for achieving faster training rates [75]. The target variables (outputs from the network) is transformed to the logarithmic domain. The advantages of training the network to predict logarithmic fouling resistances in neural architectures shown in Fig. 3.4(a) and 3.4(b) are multifold and they follow. The kernel density estimation plots [76] (Fig. 3.5(a)) show that the probability distribution of overall fouling, flue-gas side and water-side fouling resistances($R_1 + R_2$,R_1,R_2) in the training dataset are left-skewed with a long right tail. Further, the orders of magnitude of fouling resistances that have to be estimated are miniscule. Hence, direct estimation of fouling resistances using multi-layered neural network is challenging because the magnitudes of the neural network parameters and their computed gradients would be extremely small, potentially slowing down the neural training process, which often results in floating point underflow issues [77, 78]. While logarithmic transformation on outputs has been traditionally used for transforming a highly skewed-variable into a more symmetric distribution [79, 80], it is observed that logarithmic transformation of fouling resistance variables increases the fouling prediction module's sensitivity, avoids floating point precision error and speeds up the training process in our problem-setting.

Module Training and Evaluation The objective of training the fouling prediction module is to learn the optimal network weights based on minimization of the prediction error. This optimization requires a variable that quantifies the model performance i.e, an error measure based on the predicted value $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{X}; \mathbf{W})$ and the true observation $\mathbf{y}$. Formally, this is captured through a functional representation of the fouling resistance prediction error called the *loss function* $\mathscr{L} : \mathbb{Y}^*\mathbb{Y} \to \mathbb{R}$. The structure of the loss function depends on the desired prediction objective. Hence, we have used a mean squared error as the loss function in the fouling prediction module. Thus, the problem of tuning of weights in our network reduces to minimizing the loss function $\mathscr{L}(y, \hat{y})$ over a set of n training samples which yields the following optimization problem:

$$W^* = argmin_{W \in \mathbb{W}} \sum_i L(y_i, \hat{y_i}) \qquad \text{with} \qquad L(y_i, \hat{y_i}) = (y_i - \hat{y_i})^2, \tag{3.2.2}$$

where $\mathbb{W}$ is the parameter space of the prediction network. This minimization problem (eq. 3.2.2) in our fouling prediction module is difficult to solve as the loss function is non-convex. Deterministic approaches such as the gradient descent algorithm (Appendix eq. 31) are sensitive to initialization, computationally intensive and offer no theoretical performance guarantees for convergence to global minima [81]. We overcame this challenge through the use of distributed stochastic mini-batch gradient descent algorithm [56, 57, 58] with Nesterov momentum [59] for minimizing the loss function of our fouling prediction network (Chapter 1 eq. 1.3.20-1.3.21). The mini-batch gradient is an unbiased stochastic estimator of the full gradient and is defined as:

$$\textbf{Mini-batch Stochastic Gradient: } \nabla_W L(W; x^{(i:i+n)}; y^{(i:i+n)}) = \frac{1}{n} \sum_{i=1}^{i=n} \nabla_W L(y_i, f(x_i; W)), \tag{3.2.3a}$$

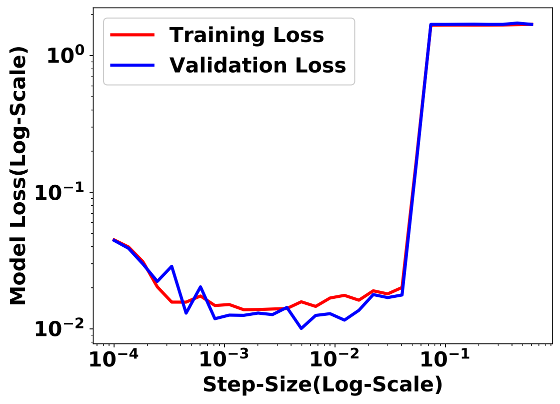
$$\textbf{(Full) Gradient Descent: } \nabla_W L(W; X; Y)) = \frac{1}{N} \sum_{i=1}^{i=N} \nabla_W L(y_i, f(x_i; W)) \tag{3.2.3b}$$

where n samples are chosen randomly without replacement from a dataset of size N ($n \leq N$). The gradient of the loss function in eq. 3.2.3(a), with respect to the parameters of the prediction network is efficiently computed using back-propogation algorithm [55]. Back-propagation calculates the gradient of the network parameters using chain rule, starting from the final layer of neurons and iteratively progressing backwards to the intermediate layers of the network to avoid redundant computations (Chapter 1 eq. (1.3.6-1.3.16)). The expression obtained for the loss function gradient with respect to the parameters of each layer is a function of prediction network's output ($\hat{y}$) (Chapter 1 eq. (1.3.12-1.3.16)). This creates the need for a forward pass (computation) to determine the network output $\hat{y}$ prior to the backward computation of loss function gradient for every sample (x_i, y_i).

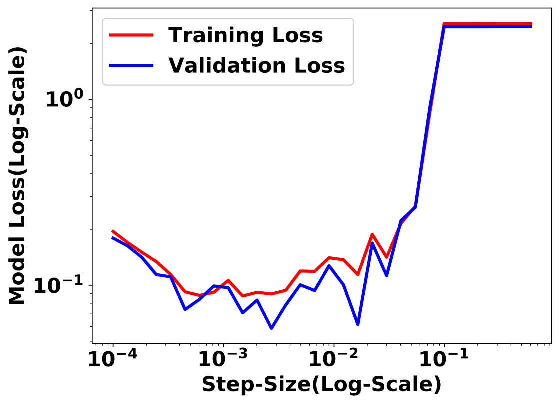
The iterative training procedure is summarized as follows. The fouling dataset generated through the algorithm in Table 3.2 is split into two parts before the commencement of training: (1) training data-set: About 80% of the data-set is randomly selected to train the network parameters (2) Validation data-set: About 20 % of data is used to assess model performance, generalization apart from aiding the selection of network hyper-parameters such as optimization algorithm's step-size, batch-size and stopping criterion, explained subsequently. The network parameters of the fouling prediction module are randomly initialized at the start of the first training epoch. An epoch is completed when the forward and backward pass computations through the prediction happen once for every sample in the training dataset. An epoch is structured into several iterations where the parameters of the network are adaptively tuned based on an optimization update rule (Appendix eq.(35-37)). This expression (Chapter 1 eq. 1.3.20,1.3.21) involves three variables: the *magnitude* of local stochastic gradient estimated from a batch of n randomly chosen training samples using back-propagation, *size* of the update step controlled through a learning rate parameter(α) and *rate* of gradient descent regulated by momentum factor(γ). These hyper-parameters are model characteristics that cannot be estimated directly from training data-set but play a prominent role in training rate and model accuracy. With an optimal selection of these hyper-parameters and network structure, the fouling resistance prediction loss's magnitude decreases as training progresses through several epochs, as illustrated in Fig. 3.5(b). As the parameters of the network are updated through several iterations, the *size* of the gradient update needs to be reduced periodically to achieve convergence to the loss function's local minima. We incorporate a time-based step-size (α) decay scheduler for this purpose (eq. 3.2.4). The training process of the fouling resistance prediction network is stopped around 100 epochs when it is observed that the validation loss fails to improve on prolonged training. Training the prediction module beyond this limit leads to model over-fitting when the network learns to conforms to the *noise* in the training dataset at the expense of constant or marginal increase in the validation loss. An overfit network fails to accurately learn the critical functional maps between the heat-exchanger operating variables and fouling resistance. Consequently, their performance on *unseen* data samples is sub-optimal.

$$\alpha_{t+1} = \frac{\alpha_t}{1 + \beta \times t}, \tag{3.2.4}$$

where t is the iteration number and β is the decay rate parameter.



(a)



(b)

Figure 3.6: **Hyper-parameter tuning for 6(a) Overall fouling Resistance Prediction($R_1 + R_2$), 6(b) Flue-gas and Water Side Fouling Resistance(R_1 & R_2). We used a step-size of 0.001 with a batchsize of 32 for network in Fig. 4(a) and a step size of 0.002 with a batchsize of 32 for network in Fig. 4(b)**. Further, the momentum factor (γ) and the learning rate decay parameter (β) is taken to be 0.9 and 10^{-6} respectively for both these networks.

The two major hyper-parameters of the optimization update rule (Chapter 1 eq. 1.3.20,1.3.21), the learning rate (α) and the step-size(n), are tuned using grid-search algorithm [60] over a specified subset of the hyper-parameter space. This subset is generated through possible combinations of two independently generated geometric series of learning rates and batch-size. The network is then trained for every tuple in this subset with every training run totaling 50 epochs. Model loss is computed after each run and the set of values of step-size and batch-size that gave low training and validation loss are taken to be the optimal hyper-parameter choice for subsequent algorithmic studies (Fig. 3.6(a),3.6(b)). Finally, an ensemble of 5 neural networks [82], each trained on randomly selected samples amounting to about 80% of the training data(bagging process) was used for both types of prediction networks and the final predictions is computed by averaging over the predictions of each network in the ensemble. The predictive performance of the ensemble network trained through bagging process(bagged network) is compared against the performance of an individual network in the subsequent section.

The fouling prediction module for each of the two problem setting (Fig. 3.4(a),3.4(b)) returns an output corresponding to the logarithm of fouling resistance. The performance of our trained fouling resistance prediction module is evaluated using absolute prediction errors (eq. 3.2.5(b)) and the co-efficient of determination(R^2) (eq. 3.5.2(f)) [53]. Let $y_1, y_2, y_3, ...y_n$ be the n values of target variable in the dataset and $\hat{y}_1, \hat{y}_2, \hat{y}_3, ...\hat{y}_n$ be the corresponding model predictions, we define the following quantities:

$$\textbf{Residuals}: \; e_i = y_i - \hat{y}_i \tag{3.2.5a}$$

$$\textbf{Absolute prediction error}: \; |e_i| = |y_i - \hat{y}_i| \tag{3.2.5b}$$

$$\textbf{Sample Mean}: \; \bar{y} = \frac{1}{n}\sum_{i=1}^{i=n} y_i \tag{3.2.5c}$$

$$\textbf{Total Sum of squares}: \; TSS = \sum_{i=1}^{i=n}(y_i - \bar{y})^2 \text{ (Proportional to data variance)} \tag{3.2.5d}$$

$$\textbf{Residual Sum of squares:} \; RSS = \sum_{i=1}^{i=n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{i=n} e_i^2 \tag{3.2.5e}$$

$$\textbf{Coefficient of Determination}(R^2) \; = 1 - \frac{RSS}{TSS} = 1 - \text{Fraction of Explained Variance(FVU)} \tag{3.2.5f}$$

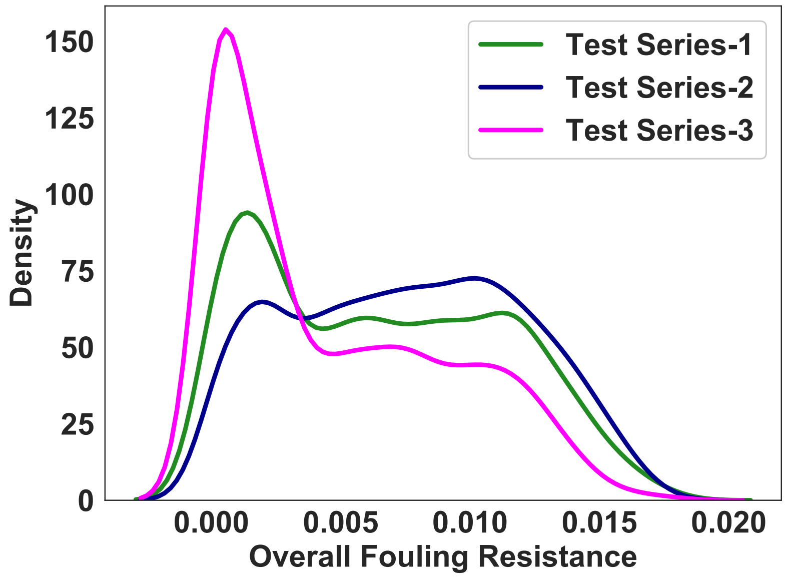


Figure 3.7: **Sample Distribution of Overall Fouling Resistance($R_1 + R_2$) in 3 Test Data-sets**

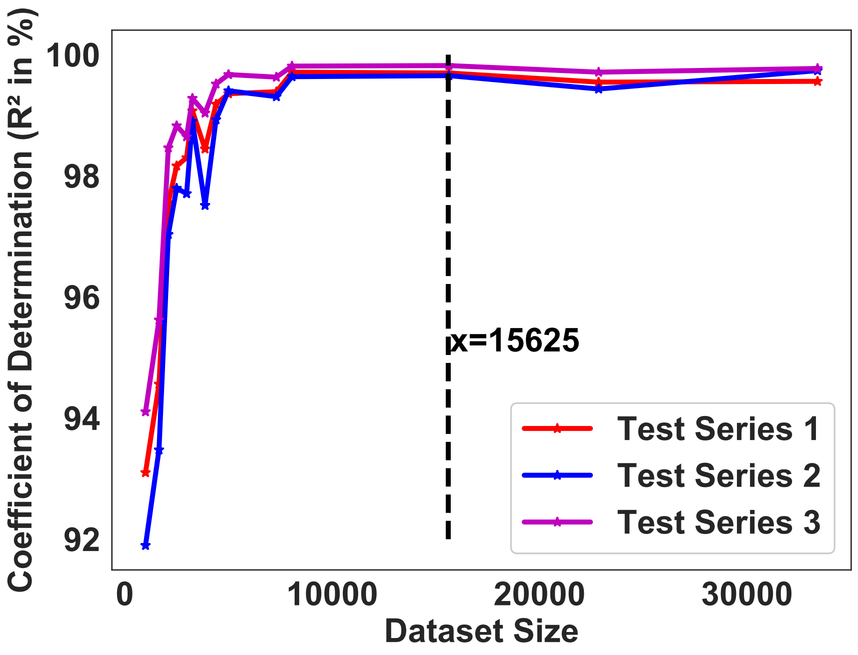
The co-efficient of determination(R^2) (eq. 3.5.2(f)) is defined as the proportion of variance in the dependent variable(fouling resistance) that is explained by its predictor variables (heat-exchanger operation data) or the reduction in error over a mean-only model. A higher model R^2 is indicative of high model prediction power as the model is able to explain a significant fraction of variation observed in fouling resistance values present in the data-set. R^2 values typically range between $[0, 1]$ though it may be negative in certain cases when the learning model performs worse than mean-only model. We have quantified the performance of the prediction networks by computing the coefficient of determination(R^2) of our log-transformed model and the actual predictions, obtained by taking the exponential of the outputs returned by the log-transformed model. It is essential that the coefficient of determination(R^2) is evaluated for both these predictions as a higher coefficient of determination(R^2) of log-transformed model does not necessarily imply a high coefficient for actual predictions [83, 84]. Further, we proceed to evaluate the performance of both single and ensemble prediction networks on three test data-sets comprising of 2500 data samples each. A test data-set is one that is generated by our scheme(Table 3.2) which the prediction model has not been trained on. The performance of the fouling prediction module on these 3 test data-sets each with its signature sample distribution (Fig. 3.7) is indicative of prediction algorithm's generalization and utility. Additional studies have been performed to quantify the effects of data-set size and measurement noise on fouling prediction model's performance.

3.3 Results

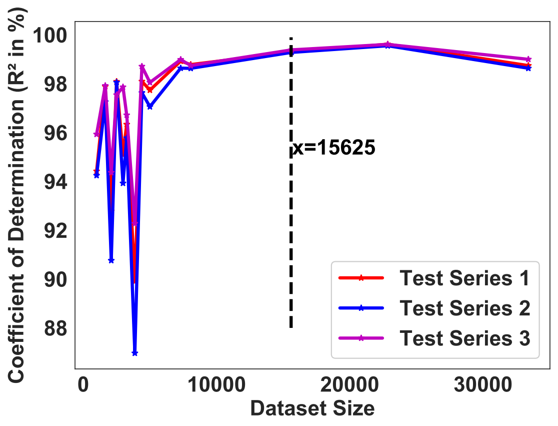
Effect of Training Data-set Size on Model Performance The model was trained on various datasets to test the impact of training data-size on fouling prediction model's performance. The neural network based model was trained on 5 different training data-sets with their sample sizes varying between 1000 and 30000. The model accuracy was then evaluated on three independent test data-sets with different sample distributions (Fig. 3.7). The model predictive accuracy for overall fouling resistance($R_1 + R_2$) increases with the sample size and saturates after about a sample-size of 10000 (Fig. 3.8(a)). The accuracy metric plots for flue-gas side and water-side fouling resistance (Fig. 3.8(b),3.8(c)) seem to saturate at a training data-set size of about 15000 data-set samples.

Data-set Size	**Overall Fouling($\mathbf{R_1 + R_2}$)**			**Flue-Gas& Water Side Fouling($\mathbf{R_1 \& R_2}$)**					
	Series 1	**Series 2**	**Series 3**	**Series 1**		**Series 2**		**Series 3**	
5000	99.36%	99.40%	99.67%	97.73%	97.66%	97.05%	97.50%	98.05%	98.61%
7310	99.39%	99.31%	99.63%	98.93%	98.43%	98.64%	98.41%	98.98%	98.98%
8060	99.71%	99.63%	99.81%	98.78%	96.46%	98.63%	95.84%	98.74%	97.46%
15625	99.70%	99.65%	99.82%	99.29%	98.77%	99.28%	98.57%	99.39%	99.27%
22845	99.55%	99.43%	99.71%	99.62%	98.37%	99.55%	98.10%	99.61%	98.89%
33400	99.55%	99.74%	99.77%	98.74%	98.98%	98.63%	98.77%	98.99%	99.25%

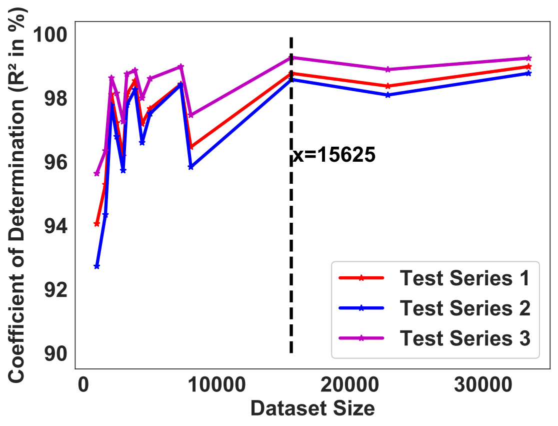
Table 3.4: **Tabulation of Model Predictive Accuracies(R^2) of Figures 8(a),8(b),8(c)**



(a)



(b)

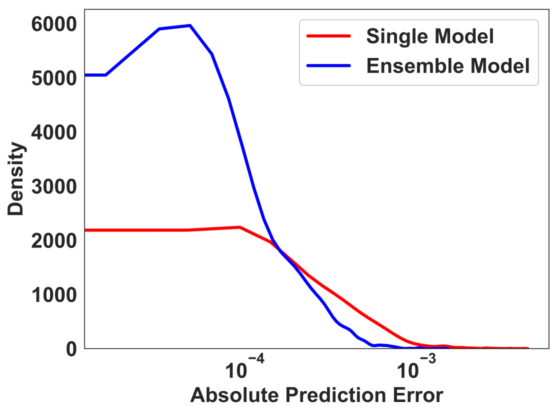


(c)

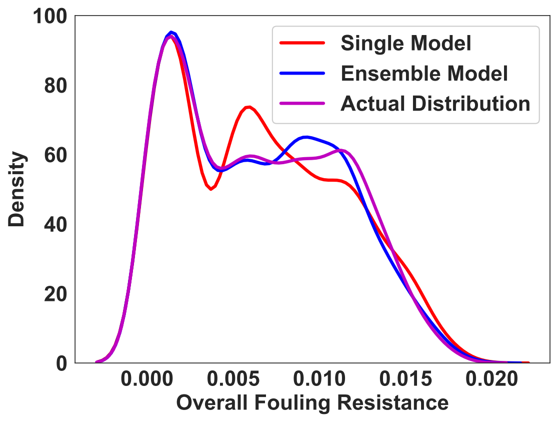
Figure 3.8: **Data-set Size Influence on 8(a) Overall fouling Resistance($R_1 + R_2$) Model 8(b) Flue-gas Fouling Resistance(R_1) 8(c) Water Fouling Resistance Model(R_2)**

From Table 3.4 and Figs. 3.8(a),3.8(b) & 3.8(c), we infer that the test data-set accuracies are over 97% for training data-set size of over 5000 points. It is also observed that the fouling resistance predictions on the flue-gas side is nominally better than the corresponding water-side fouling resistance predictions on each of the test data-sets for same training data-set sample size. This can perhaps be attributed to the fact that the water-side fouling resistances are smaller than flue-gas side fouling resistances which makes their accurate predictions more challenging. It should also be noted that the model predictive accuracies are significantly high due to lack of measurement noise in test data-set. Overall, the predicted fouling resistance values are in good agreement with the actual fouling resistance.

Single and Ensemble Model Prediction Comparison We compare the model prediction accuracy metrics for a trained single network and an ensemble(bagged model) of 5 neural networks on each of the three data-sets. These studies were performed for both the prediction problems- Overall Fouling Resistance (R_1+R_2), Flue gas and Water-side Fouling Resistances (R_1, R_2) (Network Architecture in Fig. 3.4(a),3.4(b)).



(a)



(b)

Figure 3.9: **Gaussian Kernel Density estimation plots of absolute model prediction errors (a), predicted and actual values of overall fouling resistance($\mathbf{R_1 + R_2}$)($\mathbf{KW^{-1}}$) (b) in test dataset-1 (single and ensemble networks)**

The predictions obtained from single and ensemble prediction networks for overall fouling resistance ($R_1 + R_2$) have been compared with the actual values using Gaussian kernel-density estimation plots for test data-set-1 (Fig. 3.9(b)). The kernel-density estimation plot for actual and predicted values of overall fouling resistance ($R_1 + R_2$) estimates the probability density function of these respective variables in a non-parametric manner based on their data-samples. It shows that a significant proportion of prediction errors is less than 10^{-4} for both single network and ensemble network based prediction models (Fig. 3.9(a)). However, the presence of higher densities in lower error range for the ensemble network curve coupled with the presence of long right tail for single-network error curve make a solid case for the predictions of ensemble model of five networks to be more robust and accurate in comparison to a single-network fouling resistance model.

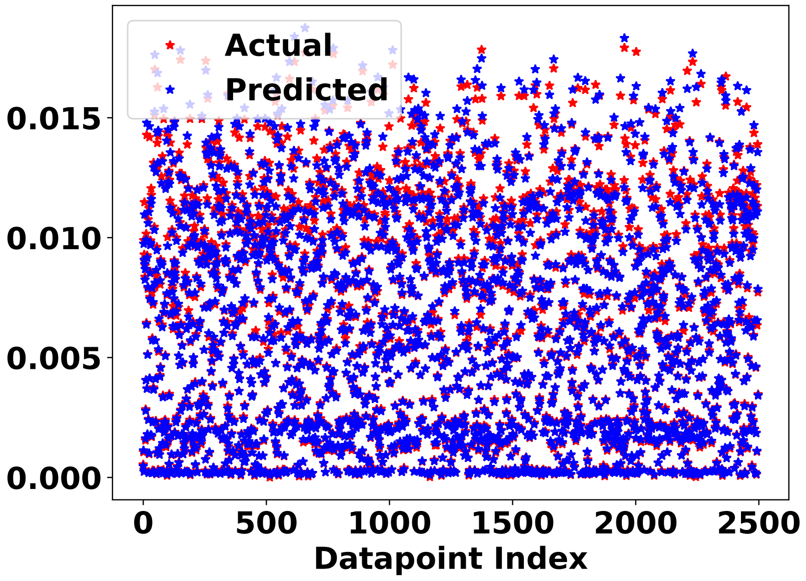
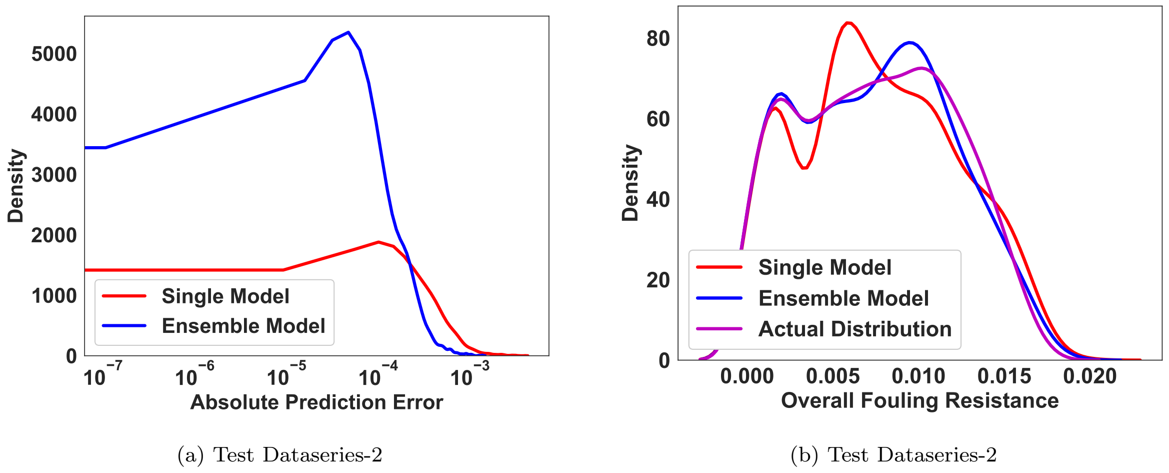


Figure 3.10: **Scatter plot of predicted and actual values of overall fouling resistance($\mathbf{R_1 + R_2}$)($\mathbf{KW^{-1}}$)- This plot depicts the ensemble-network predictions and the corresponding "ground-truth" values of overall fouling resistance for each of the 2500 samples in test dataset-1**

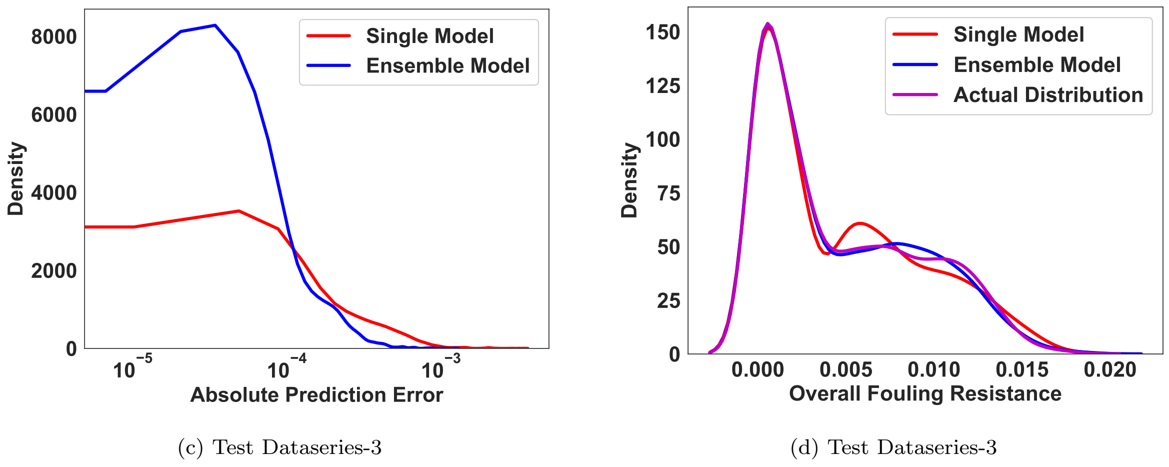
This finding is further reinforced through the kernel density plots shown in Fig. 3.9(b). The kernel-density curve for ensemble model predictions seem to be in close agreement with the actual distribution of overall fouling resistance values($R_1 + R_2$) seen in test dataset-1. The predictions from a single network model, however, appears to be slightly discordant with the actual fouling resistance distribution in the range of $[0.005, 0.015]KW^{-1}$ which accounts for its higher absolute prediction error rates. The scatter plot for ensemble model predictions with actual fouling resistance values in test dataset-1 (Fig. 3.10) illustrates that while both these values largely overlap, there is a slight mismatch seen in the range of $[0.005, 0.015]KW^{-1}$. The model tends to marginally over-predict the fouling resistance values which fall in the mentioned range. These observations agree with the inferences from the distribution plots in Fig. 3.9.

The performance comparison kernel density estimation plots in test data-sets-2 and 3 also point to the superior performance of an ensemble network over single network in overall fouling resistance($R_1 + R_2$) predictions. The single network model's performance on test datasets 2 and 3 indicate a slight misfit between model predictions and actual sample distributions (Figs. 3.11(a),3.11(c)). The ensemble network model performs significantly better than a single network model in conforming to the actual distribution of overall fouling resistance samples present in test datasets 2 and 3. These observations are consistent with our earlier comments pertaining to single and ensemble network's predictive performance on test dataset-1. The trained neural network model seems to slightly over-predict the overall fouling resistance in the distribution's central zone(roughly between $0.01KW^{-1}$ & $0.015KW^{-1}$) and under-predicts the fouling resistance towards the distribution's tail. The model predictive performance of ensemble network has been tabulated in table 5. The ensemble model's predictive performance is the highest in test data-series 3 that has a high proportion of low overall fouling resistance values and least in test data-series 2 that has a high proportion of values in the distribution's central zone (for reasons previously discussed). The coefficient of determination(R^2) for ensemble network is very close to 1 for both overall fouling resistance and individual flue-gas and water side predictions. These are indicative of a great functional fit between operating conditions and fouling resistances in the absence of measurement sensor noise.



(a) Test Dataseries-2

(b) Test Dataseries-2



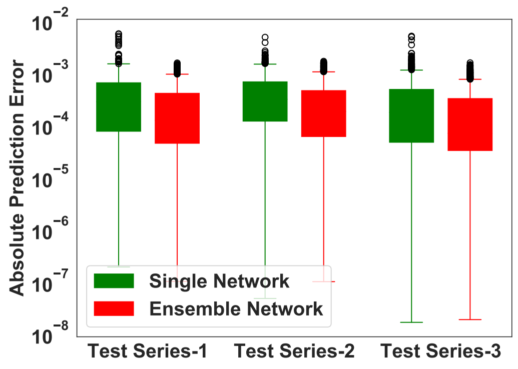
(c) Test Dataseries-3

(d) Test Dataseries-3

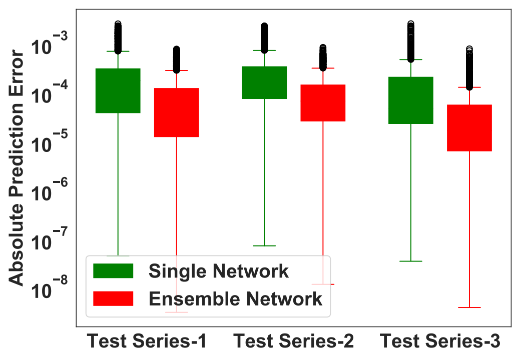
Figure 3.11: **Gaussian Kernel Density estimation plots of absolute model prediction errors, predicted Vs actual values of overall fouling resistance($\mathbf{R_1 + R_2}$)($\mathbf{KW^{-1}}$)- The error plots ((a),(c)) compare the sample distributions of absolute predictions errors of single and ensemble network prediction models on test datasets 2 & 3. The fouling resistance plots ((c),(d)) construct the distributions of "ground truth" values and network predictions based on observed samples of test datasets 2 & 3.**

Dataset	Overall Fouling($R_1 + R_2$)	Flue-Gas Side Fouling(R_1)	Water Side Fouling(R_2)
Test Dataset 1	99.86%	99.21%	99.20%
Test Dataset 2	99.80%	98.86%	99.05%
Test Dataset 3	99.92%	99.34%	99.47%

Table 3.5: **Tabulation of Ensemble Network's Model Predictive Accuracies(R^2)**



(a) Flue-Side Fouling Resistance



(b) Water-Side Fouling Resistance

Figure 3.12: **Ensemble and single network prediction error box plots- These plots graphically represent model absolute prediction errors on each of the three datasets through their quartiles. The box in these error plots correspond to the 2nd and 3rd quartiles (middle 50% of the sample population) with vertical line extensions within $1.5 * IQR$ on either side of the box. The black rings are the outliers that lie outside this statistical configuration. These box-plots help compare absolute prediction errors from different network configurations without making assumptions of the underlying statistical distribution**

The box plots (Fig. 3.12) depict the spread of absolute prediction errors for flue-side and water-side fouling resistances(R_1 & R_2) prediction model. We infer that the ensemble network prediction models for flue-side and water side fouling resistances have lesser mean absolute prediction error and lesser distribution outliers compared to a single network model(Fig. 3.12(a),3.12(b)). Subsequently, the ensemble network model has higher flue gas and water-side fouling resistance prediction accuracies(R^2) in all the test datasets compared to a single network model (Table 3.5). Further, the ensemble prediction model has fairly consistent and similar prediction accuracies for both flue-gas side and water side fouling resistances, unlike a single network model that has a marginally lower model fit on water-side fouling resistance series.

The analysis of single and ensemble network model for overall and individual resistance predictions indicate that while a single network's predictions may have some disagreements in conforming to actual test-dataset's distribution in certain zones, the ensemble model comprising of 5 networks flattens out these individual mispredictions to predict a distribution that fits the actual distribution better. A combination of outputs from several networks is favorable only if they disagree on some inputs to give varied predictions and this scenario occurs in this present problem setting. It can be shown mathematically that learning of continuous valued function using ensemble network gives improved accuracy and lesser prediction inconsistencies (variance) under the discussed scenario [85] and our observations are in line with the theoretically proven results.

3.4 Discussions

The previous section quantified the predictive performance of the fouling resistance module. In this section, we explore the suitability of the model and assess its predictive performance by subjecting it to datasets with input measurement noise. Further, we explain the working of the fouling resistance prediction network around select operating points through the technique of local interpretable model-agnostic explanations (LIME). Consequently, this section brings out the practical applicability of the proposed model for fouling prediction.

Quantifying Prediction Model's performance under input measurement noise The inputs to the fouling resistance prediction algorithm are heat exchanger fluid flow-rates, inlet and exit temperatures that are obtained through various sensor measurements in a real-life scenario. Thus, various forms of sensor noise, measurement errors and unmodeled physical phenomena creep into the input values that will be fed as fouling resistance prediction model inputs. The developed fouling resistance prediction model is practical only if we quantify the model performance in a mathematically robust test setting.

The developed module's performance is quantified by subjecting it to test datasets containing physical variables with varying degree of simulated noise. The noise in measurement variables is modeled as random Gaussian noise, whose probability density function is the standard normal distribution defined as:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} = N(\mu, \sigma^2) \tag{3.4.1a}$$

$$T(t) = \bar{T}(t) + \triangle T \tag{3.4.1b}$$

$$\triangle T \sim N(0, \sigma(t)^2) \qquad \text{where} \qquad 3\sigma(t) = x\% \text{ of } \bar{T}(t) \tag{3.4.1c}$$

The degree of noise in a test data-set is controlled by adjusting the $3-\sigma$ measurement noise levels. Noise is added to each "clean" value in these test datasets by augmenting it with a random value which is sampled from a normal distribution with a mean 0 and 3-standard deviation(σ) level corresponding to a certain percentage of the measured value (eq. 3.4.1(a-c)). For instance, all the flow-rate and temperature measurements in a test dataset with 5% $3-\sigma$ noise will be modified with a random value sampled from a normal distribution with 3-standard deviation(σ) level corresponding to 5% of the measured value.

The box plot in Fig. 3.13 shows the absolute prediction error spread under the influence of varying levels of measurement noise in input test dataset for overall fouling resistance($R_1 + R_2$) prediction models.

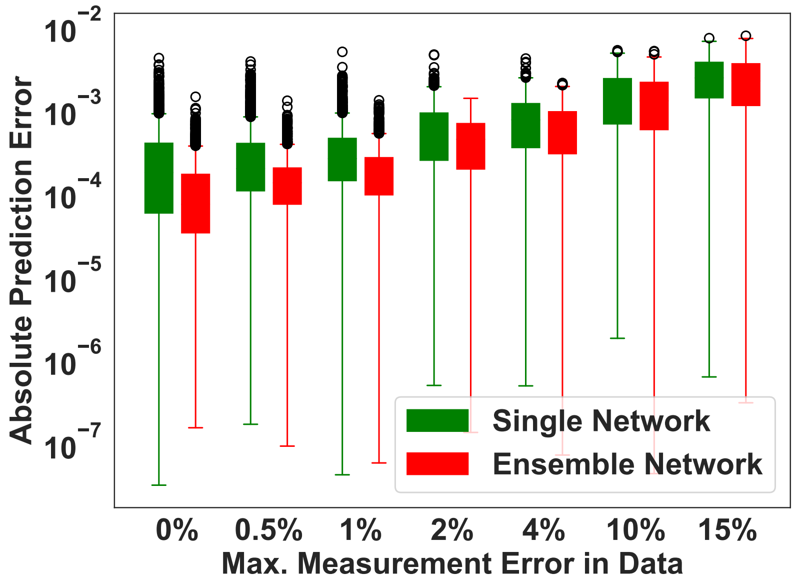


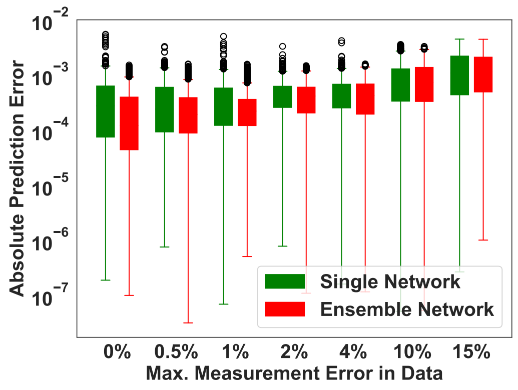
Figure 3.13: **Box plot of ensemble and single network model predictions of overall fouling resistance($\mathbf{R_1 + R_2}$)($\mathbf{KW^{-1}}$) in presence of varying levels of input measurement noise**

Test Dataset $3-\sigma$ Noise levels	Network Output($\log(\mathbf{R_1+R_2})$)	Overall Fouling($\mathbf{R_1+R_2}$)
0%	99.31%	99.85%
0.5%	97.63%	99.82%
1%	95%	99.73%
2%	79.2%	98.62%
4%	71.26%	97.26%
10%	43.13%	86.03%
15%	12.88%	61.47%

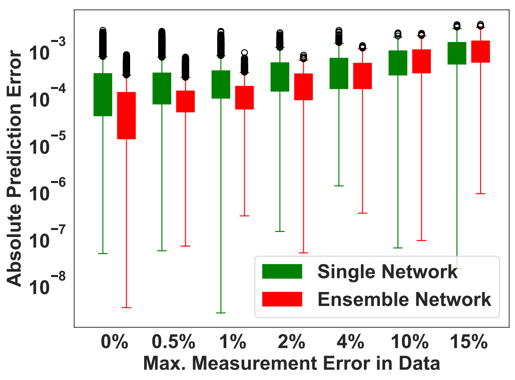
Table 3.6: **Ensemble model's log-transformed and actual predictive performance in the presence of measurement noise**

The error bars of both single network and ensemble prediction models (Fig. 3.13) progressively shift upwards with increase in $3-\sigma$ measurement noise levels. They are symptomatic of increasing mean absolute model prediction errors with increase in $3-\sigma$ measurement noise levels as one expects. The mean absolute prediction errors are considerably lower than 0.01 even at 15% $3-\sigma$ measurement noise levels. This demonstrates the stability of both single and ensemble network prediction module for overall fouling resistance at high measurement noise levels in its predictor variables. The plot (Fig. 3.13) also confirms that the absolute prediction error bands for ensemble network is lower than single network model with fewer distribution outliers. Thus, we infer that the ensemble network predictions are more accurate than that of a single network, for overall fouling resistance.

The ensemble model's predictive performance is summarized in Table 3.6. The key observations from the model predictive statistics is that the coefficient of determination(R^2) of log-transformed ensemble deteriorates rapidly with an increase in $3-\sigma$ measurement noise levels. The accuracy of ensemble predictions for overall fouling resistance (R_1+R_2) also decreases with an increase in measurement noise levels but the rate of decline is not as steep as that of the log-transformed model. The ensemble model's overall fouling resistance predictive accuracy is still as high as 86.03% at 10% measurement noise levels and 61.47% at 15% noise levels which brings out the practical utility of the proposed ensemble model.



(a) Flue-Side Fouling Resistance



(b) Water-Side Fouling Resistance

Figure 3.14: **Box plot of ensemble and single network model predictions of flue (14 (a)) and water-Side (14(b)) fouling resistance in presence of varying levels of input measurement noise**

The error bars for flue-gas side (R_1) and water-side(R_2) fouling prediction model(Fig. 3.14) also display similar trends to that of overall fouling resistance prediction model(R_1+R_2) where the mean absolute prediction errors in predicting individual fouling resistances increase steadily with an increase in $3-\sigma$ measurement noise levels in test dataset's inputs. Table 3.7 quantifies the ensemble model's flue gas-side and water-side fouling resistance predictive performance in the presence of measurement noise in its input variables. The rate of drop in model accuracies in the presence of input measurement noise is much faster in predictions of water-side fouling resistance compared to flue-side. Interestingly, it is seen that the co-efficient of determination(R^2) of log-transformed model becomes negative at 15% $3-\sigma$ measurement noise levels. Thus, the model starts behaving worse than a mean-only prediction model at around 15% input noise levels.

$3-\sigma$ Noise levels	**Flue-gas side Fouling($\mathbf{R_1}$)**		**Water Side Fouling($\mathbf{R_2}$)**	
	Log-Output(log($\mathbf{R_1}$))	**Predicted(R_1)**	**Log-Output(log($\mathbf{R_2}$))**	**Predicted (R_2)**
0%	97.25%	99.20%	98.32%	99.19%
0.5%	94.1%	99.12%	91.93%	99.21%
1%	90.62%	99.15%	88.25%	99.12%
2%	70.39%	98.36%	69%	97.63%
4%	65.93%	97.86%	47.75%	93.28%
10%	39.75%	91.63%	15.08%	73.95%
15%	17.77%	80.86%	-16.93%	-

Table 3.7: **Ensemble Model's log-transformed and actual predictive performance in the presence of input measurement noise**

This study has helped us identify the stability threshold for predictions from the proposed fouling prediction network (Fig. 3.4(b)) for water-side and flue-side fouling resistances. We cap the maximum $3-\sigma$ measurement noise levels for ensemble model predictors at 10% beyond which the flue-gas side and water-side fouling resistance network loses its predictive capabilities and utility value. A likely explanation for water-side fouling resistance(R_2) predictions to lose stability relatively quickly, over that of flue-gas side(R_1) or overall fouling resistance($R_1 + R_2$) is its high-sensitivity to input parameter variations. This is due to its smaller range and magnitude of fouling factor. Overall, the choice of using an ensemble(bagged) network over a single network in our prediction model greatly increases the prediction accuracy and makes it robust to input measurement noise. The proposed fouling prediction model can operate effectively even at 10% input measurement noise levels which reinforces its suitability for practical applications.

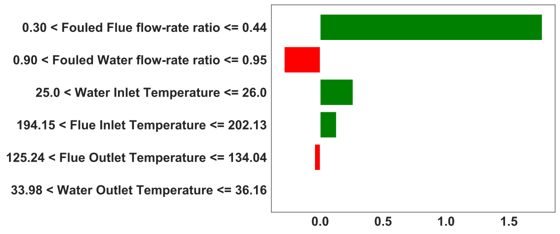
Interpreting Fouling Prediction Model Results The predictions made by any simplistic data-driven machine learning approach can be understood by analyzing the internal components and how they interact. However, it is very challenging to understand the working of deep learning based neural network model with all its internal components and non-linear nature of interactions between various neurons in the network. Lack of mathematical models that account for generalized and wholistic working of neural networks is what makes this entire exercise of trying to interpret the relationship between input features and output target a tedious exercise. Here, we attempt to explain the working of our fouling resistance prediction network not by model specific approaches but by using local interpretable model-agnostic explanations(LIME)[42].

The concept of locally interpretable model agnostic explanations is based on the theory of linearization of non-linear functions. The proposed neural network module for fouling resistance prediction can be thought of a non-linear curve fitting technique based on input-output predictions. Thus, the predictions of our fouling resistance module locally around an input sample can be understood heuristically by fitting weighted linear regression model(see Appendix A). The inputs to this weighted interpretable surrogate model for our fouling resistance network is obtained by perturbing the network around a point of interest and the corresponding outputs are the fouling resistance module predictions around the perturbed input. The weights for the generated new samples, close to the point of interest, were generated using proximity-measure(distance) evaluated between the sample and the point of interest. The weighted interpretable linear model is trained based on the generated input-output instances. This model acts as a local approximation of our network model around the point of interest. The feature coefficients of this local surrogate model is a measure of that particular input variable's importance in our fouling resistance module's local predictions.

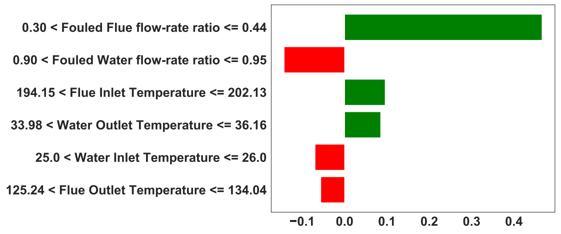
We have explained the local working of the proposed overall fouling resistance($R_1 + R_2$) prediction module by considering two test instances(model inputs, actual and predicted model outputs in Table 3.8 and Table 3.9)

Test Sample-1		**LIME Weights**	
Input Variable	**Variable Value**	**Single Network**	**Ensemble Network**
Fouled Flue Flow-rate ratio($\frac{\mathbf{\dot{M}_{flue,f}}}{\mathbf{\dot{M}_{flue}}}$)	0.34	0.47	1.77
Flue Inlet Temperature(T_{fi})	200.69 $°C$	0.1	0.13
Flue Outlet Temperature(T_{fo})	129.20 $°C$	-0.06	-0.04
Fouled Water Flow-rate ratio($\frac{\mathbf{\dot{M}_{w,f}}}{\mathbf{\dot{M}_{w}}}$)	0.93	-0.14	-0.29
Water Inlet Temperature(T_{wi})	25.81 $°C$	-0.07	0.26
Water Outlet Temperature(T_{wo})	35.99 $°C$	0.09	-0.004
Local Linear Model Accuracy		82.87%	97.16%

Table 3.8: **Input variable values and neural Network LIME model weights for test sample-1**



(a) Ensemble Network



(b) Single Network

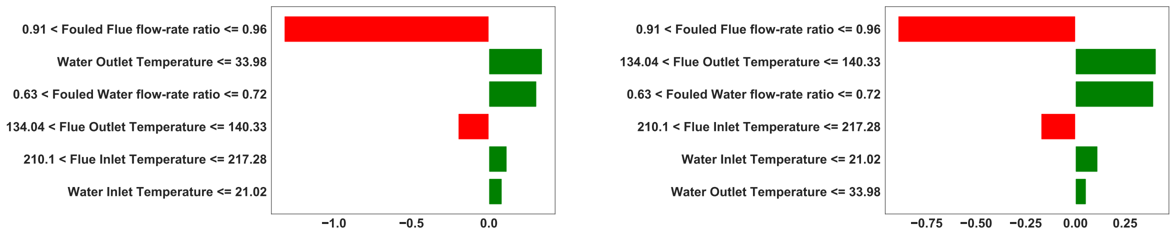
Figure 3.15: **Model Weights for LIME Models for single and ensemble networks for test sample-1: Each of the inputs of the training dataset are partitioned into groups of 10 (decile) based on their sample distribution. The local representative models for the networks were trained on 5000 samples in the neighborhood of the chosen operating point. Each of the sample inputs like the flow-rates and temperature variables were sampled from their respective deciles. The relevance of the generated sample to the chosen operating point was quantified using distance based similarity measure. This similarity measure was incorporated through the Gaussian Kernel (See LIME in Appendix) with a kernel width of 0.3 in the error function of local surrogate models.**

The LIME model weights shows how the neural network based module predicts the overall fouling resistance $(R_1 + R_2)$(Fig 3.4(a)) based on the input variables. The order of importance of heat-exchanger operation variables in local linear models for ensemble (Fig. 3.15(a)) and single networks (Fig. 3.15(b)) are largely similar. However, the two local linear models differ in how they weigh the individual predictor variables. Both the LIME models accord high positive weights to fouled flue flow ratio ($\frac{\dot{M}_{flue,f}}{\dot{M}_{flue}}$) that takes a value of 0.34 for the chosen data sample. The high fouled water flow ratio($\frac{\dot{M}_{w,f}}{\dot{M}_w}$) of 0.93 has a negative effect on overall fouling resistance predictions in the local models as expected. The flue-gas and water temperature variables have lower weights than their corresponding flow rate variables in the local representative model. The contributions of the temperature variables in the local models need to be understood in the larger context of their range in training data (Table 3.1). The flue gas inlet temperature for the chosen sample is $200.69°C$ which is between 30%-40% of the considered range in our data-generation scheme. This is seen as favorable conditions for fouling by the local surrogate models, however this influence is partly countered by the observed low flue outlet temperature of $129°C$ in this sample. The water inlet and outlet temperature variable weights of local surrogate models are seen to be negligible.

The validity of linear models as an approximation to the proposed single and ensemble fouling resistance prediction networks is assessed by evaluating their coefficient of determination in the vicinity of the chosen sample. Both of these LIME models to our prediction networks display high accuracies in the considered local region (Table 3.8). Thus, these linear models encapsulate the local working of our fouling resistance prediction networks and simplify their analysis.

Test Sample-2		**LIME Weights**	
Input Variable	**Variable Value**	**Single Network**	**Ensemble Network**
Fouled Flue Flow-rate ratio($\frac{\dot{\mathbf{M}}_{\mathbf{flue,f}}}{\dot{\mathbf{M}}_{\mathbf{flue}}}$)	0.92	-1.32	-0.89
Flue Inlet Temperature(T_{fi})	216.13	0.12	-0.17
Flue Outlet Temperature(T_{fo})	134.82	-0.2	0.41
Fouled Water Flow-rate ratio($\frac{\dot{\mathbf{M}}_{\mathbf{w,f}}}{\dot{\mathbf{M}}_{\mathbf{w}}}$)	0.65	0.31	0.39
Water Inlet Temperature(T_{wi})	21.01	0.08	0.11
Water Outlet Temperature(T_{wo})	30.16	0.34	0.05
Local Linear Model Accuracy		94.11%	97%

Table 3.9: **Input variable values and Neural Network LIME Model weights for test sample-2**



(a) Ensemble Network (b) Single Network

Figure 3.16: **Model Weights for LIME Models for single and ensemble networks for test sample-2**

The weight distribution of predictors in local surrogates models for single and ensemble networks is different in test sample-2 compared to test sample-1. The local models for both these networks accord large negative weights to fouled flue flow-rate ratio($\frac{\dot{\mathbf{M}}_{\mathbf{flue,f}}}{\dot{\mathbf{M}}_{\mathbf{flue}}}$) which lies between $(0.91, 0.96]$ for the chosen sample. The temperature variables, water outlet temperature (T_{wo}) and flue outlet temperature (T_{fo}), follow the flue flow-rate ratio variable in the precedence order with high positive weights in the linear surrogates of ensemble and single prediction networks respectively. These weight distributions for local representative models is different from that seen in the case of sample-1 (Fig. 3.15) where the fluid fouled flow-rate variables had a higher influence than the temperature variables. The local models around test sample-2 also differed in how they assessed the effect of flue temperature variables. The representative linear model for ensemble prediction network gave positive weights to flue-inlet temperature and negative weights to flue-outlet temperature with the opposite seen in the case of single network's local model. These local effects account for the differences in predictions of single and ensemble network models. These local-linear models have high co-efficient of determination in the vicinity of test-sample-2 (Table 3.9) with marginal prediction errors due to local non-linear effects between the input variables and overall fouling resistance.

The local exposition for proposed fouling network obtained through LIME model demonstrates the tendency of the network to afford higher weights to the hydrodynamic indicators like fouled flue-flow rate ratio($\frac{\dot{\mathbf{M}}_{\mathbf{flue,f}}}{\dot{\mathbf{M}}_{\mathbf{flue}}}$) and fouled water flow ($\frac{\dot{\mathbf{M}}_{\mathbf{w,f}}}{\dot{\mathbf{M}}_{\mathbf{w}}}$) rate ratio variables over thermal indicators like fluid inlet and outlet temperatures. This can be explained by the fact that outlet fluid temperatures(T_{fo}, T_{wo}) are inherently dependent on fouled fluid flow-rates($\dot{M}_{flue,f}, \dot{M}_{w,f}$) and fluid inlet temperatures(T_{fi}, T_{wi}). Hence, the hydrodynamic indicators like fouled flue-flow rate ratio and fouled water flow already account for some variations observed in thermal indicators like fluid outlet temperatures and the additional information gained from the thermal variables with respect to fouling resistance predictions are only marginal.

We had earlier seen about two major fouling effects- hydrodynamic and thermal effects that dominated the heat exchanger's fouling physics while developing a data generation scheme for training our prediction model. Fig. 3.3 showed the extent of decrease in heat-exchanged from 0.24 MW under un-fouled heat exchanger conditions to 0.16 MW under maximal heat-exchanger fouling for an input variable combination of $T_{fi} = 170°C, \dot{M}_{flue} = 6.2kg/s, T_{wi} = 25°C, \dot{M}_w = 3.54kg/s$. We proceeded to assess the contribution of hydrodynamic effects of fouling in quantity of heat-exchanged without the thermal fouling effects (Table 2) for theoretical purposes. Interestingly, the quantity of energy exchanged by the heat-exchanger dropped from about 0.24 MW under un-fouled conditions to about 0.18 MW under fully fouled conditions which is nearly 71% of the energy drop witnessed with both hydrodynamic and thermal effects incorporated.

This presents a strong evidence to suggest that the hydrodynamic fouling effects dominate the local heat transfer and fluid flow physics of the considered cross-flow waste heat recovery heat-exchanger. The proposed fouling resistance prediction network also seems to operate in line with local heat transfer and flow physics by giving higher weights to hydrodynamic fouling variables such as fouled flue-flow rate ratio($\frac{\dot{\mathbf{M}}_{\mathbf{flue,f}}}{\dot{\mathbf{M}}_{\mathbf{flue}}}$) and fouled water flow ($\frac{\dot{\mathbf{M}}_{\mathbf{w,f}}}{\dot{\mathbf{M}}_{\mathbf{w}}}$) over thermal variables in the local explanatory models for overall fouling resistance predictions. Thus, the local-linear surrogate model approach proposed for explaining the working of fouling prediction algorithm has successfully brought out the practical utility and physics of operation of the model.

3.5 Conclusion and Future Work

In conclusion, we have developed an accurate and generalized ensemble deep neural network framework capable of predicting overall fouling resistance and individual flue-gas side and water-side fouling resistances of a cross-flow heat exchanger used in waste heat recovery. The trained fouling resistance prediction module's performance was evaluated on 3 different test data-sets each with its own signature input sample distribution. The model predictions had an average co-efficient of determination(R^2) of 99.86%, 99.14% and 99.24% for overall fouling resistance, flue-gas side fouling resistance and water-side fouling resistance respectively on the 3 randomly generated test datasets. The ensemble based fouling prediction module was robust to input measurement noise with the model predictive accuracy(R^2) of 86.03%, 91.63% and 73.95% for overall fouling resistance, flue-gas side fouling resistance and water-side fouling resistance respectively at a 10% $3-\sigma$ sensor noise cap. The developed fouling resistance prediction model's generalization and ability to learn heat-exchanger's fouling flow and heat transfer physics was demonstrated through an analysis of a local surrogate model of the network on a couple of test samples. Hence, an adequately trained model of the proposed fouling resistance prediction module can be deployed to make real-time predictions in an industrial setting with the inputs to the module obtained from various flow-rate and temperature sensors installed at the fluid inlets and outlets of the heat-exchanger. The proposed ensemble network framework for fouling resistance prediction can also be appropriately adapted to any heat exchanger problem-setting by suitably modifying the number of neurons in the input, inter-mediate and output layers of the network while also adjusting the number of networks in the prediction ensemble. The modularity of the proposed solution also provides opportunities for feeding in region specific temperature and flow-rate data of any chosen heat exchanger module to the network to identify and ascertain regions of the heat-exchanger most affected by fouling at that instant of time. This information will be particularly useful for undertaking region-specific corrective actions in large-scale heat exchangers. Moreover, this ensemble network fouling resistance prediction module can also be recast into a recurrent neural network with long-short term memory(LSTM) framework for learning heat-exchanger fouling dynamics and optimizing fouling cleaning and maintenance schedules. These discussed approaches will broadly be the subject of our future work.

References

[1] A. Thekdi and S. U. Nimbalkar, "Industrial waste heat recovery-potential applications, available technologies and crosscutting r&d opportunities," Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), Tech. Rep., 2015.

[2] N. Zhou, X. Wang, Z. Chen, and Z. Wang, "Experimental study on organic rankine cycle for waste heat recovery from low-temperature flue gas," *Energy*, vol. 55, pp. 216–225, 2013.

[3] S. Kakaç, H. Liu, and A. Pramuanjaroenkij, *Heat exchangers: selection, rating, and thermal design.* CRC press, 2002.

[4] P. A. Gonzalez and J. M. Zamarreno, "Prediction of hourly energy consumption in buildings based on a feedback artificial neural network," *Energy and buildings*, vol. 37, no. 6, pp. 595–601, 2005.

[5] M. Kawashima, C. E. Dorgan, and J. W. Mitchell, "Optimizing system control with load prediction by neural networks for an ice-storage system," American Society of Heating, Refrigerating and Air-Conditioning Engineers , Tech. Rep., 1996.

[6] P. S. Curtiss, J. Kreider, and G. Shavit, "Neural networks applied to buildings–a tutorial and case studies in prediction and adaptive control," American Society of Heating, Refrigerating and Air-Conditioning Engineers , Tech. Rep., 1996.

[7] G. F. Hewitt, G. L. Shires, and T. R. Bott, *Process heat transfer.* CRC press Boca Raton, 1994, vol. 113.

[8] A. Pritchard, "The economics of fouling," in *Fouling Science and Technology.* Springer, 1988, pp. 31–45.

[9] B. I. Master, K. S. Chunangad, and V. Pushpanathan, "Fouling mitigation using helixchanger heat exchangers," 2003.

[10] H. Müller-Steinhagen, M. Malayeri, and A. Watkinson, "Fouling of heat exchangers-new approaches to solve an old problem," *Heat transfer engineering*, vol. 26, no. 1, pp. 1–4, 2005.

[11] T. R. Bott, *Fouling of heat exchangers.* Elsevier, 1995.

[12] A. Watkinson, "Critical review of organic fluid fouling," Argonne National Lab., IL (USA); British Columbia Univ., Vancouver, BC , Tech. Rep., 1988.

[13] A. P. Watkinson, "Chemical reaction fouling of organic fluids," *Chemical Engineering & Technology: Industrial Chemistry-Plant Equipment-Process Engineering-Biotechnology*, vol. 15, no. 2, pp. 82–90, 1992.

[14] Z. S. Saleh, R. Sheikholeslami, and A. Watkinson, "Fouling characteristics of a light australian crude oil," *Heat transfer engineering*, vol. 26, no. 1, pp. 15–22, 2005.

[15] D. Klaren, "end bailie re," *Hydrocarbon Processing*, vol. 3, p. 48, 1989.

[16] P. Fryer, W. Paterson, and N. Slater, "Robustness of fouling heat exchanger networks and its relation to resilience," *Chemical Engineering Research and Design*, vol. 65, pp. 267–71, 1987.

[17] F. Smaïli, V. Vassiliadis, and D. Wilson, "Mitigation of fouling in refinery heat exchanger networks by optimal management of cleaning," *Energy & fuels*, vol. 15, no. 5, pp. 1038–1056, 2001.

[18] M. Markowski and K. Urbaniec, "Optimal cleaning schedule for heat exchangers in a heat exchanger network," *Applied Thermal Engineering*, vol. 25, no. 7, pp. 1019–1032, 2005.

[19] P. Heggs, "Heat exchanger fouling-mitigation and cleaning technologies by hans muller-steinhager," *TCE CHEMICAL ENGINEER*, pp. 49–49, 2001.

[20] N. Epstein, "General thermal fouling models," in *Fouling Science and Technology.* Springer, 1988, pp. 15–30.

[21] R. B. Ritter, "Crystalline fouling studies," *Journal of Heat Transfer*, vol. 105, no. 2, pp. 374–378, 1983.

[22] B. Reitzer, "Rate of scale formation in tubular heat exchangers. mathematical analysis of factors influencing rate of decline of over-all heat transfer coefficients," *Industrial & Engineering Chemistry Process Design and Development*, vol. 3, no. 4, pp. 345–348, 1964.

[23] A. Watkinson and O. Martinez, "Scaling of heat exchanger tubes by calcium carbonate," *Journal of Heat Transfer*, vol. 97, no. 4, pp. 504–508, 1975.

[24] W. Augustin, *Fouling on heat transfer surfaces; Verkrustung (Fouling) von Waermeuebertragungsflaechen*, In institut fur Verfahrens- und kerntechnik., Technische Universitat Braunschweig: Germany, 1992.

[25] W. Ebert and C. Panchal, "Analysis of exxon crude-oil-slip stream coking data," Argonne National Lab., IL (United States), Tech. Rep., 1995.

[26] S. Zubair, A. Sheikh, M. Budair, and M. Badar, "A maintenance strategy for heat transfer equipment subject to fouling: A probabilistic approach," *Journal of heat transfer*, vol. 119, no. 3, pp. 575–580, 1997.

[27] S. Lalot, O. Palsson, G. Jonsson, and B. Desmet, "Comparison of neural networks and kalman filters performances for fouling detection in a heat exchanger," *International Journal of Heat Exchangers*, vol. 8, no. 1, p. 151, 2007.

[28] F. Delmotte, M. Dambrine, S. Delrot, and S. Lalot, "Fouling detection in a heat exchanger: A polynomial fuzzy observer approach," *Control Engineering Practice*, vol. 21, no. 10, pp. 1386–1395, 2013.

[29] G. Jonsson, "Parameter estimation in models of heat exchangers and geothermal reservoirs." Ph.D. dissertation, Department of Mathematical Statistics, Lund Institute of Technology, Sweden., 1990.

[30] L. Sun, Y. Zhang, X. Zheng, S. Yang, and Y. Qin, "Research on the fouling prediction of heat exchanger based on support vector machine," in *2008 International Conference on Intelligent Computation Technology and Automation (ICICTA)*, vol. 1. IEEE, 2008, pp. 240–244.

[31] T. Ardsomang, J. W. Hines, and B. R. Upadhyaya, "Heat exchanger fouling and estimation of remaining useful life," in *Annual Conference of Prognostics and Health Management Society.* Estados Unidos ˆ eKnoxville Knoxville, 2013, pp. 1–9.

[32] C. Riverol and V. Napolitano, "Estimation of fouling in a plate heat exchanger through the application of neural networks," *Journal of Chemical Technology & Biotechnology: International Research in Process, Environmental & Clean Technology*, vol. 80, no. 5, pp. 594–600, 2005.

[33] V. Radhakrishnan, M. Ramasamy, H. Zabiri, V. Do Thanh, N. Tahir, H. Mukhtar, M. Hamdi, and N. Ramli, "Heat exchanger fouling model and preventive maintenance scheduling tool," *Applied Thermal Engineering*, vol. 27, no. 17-18, pp. 2791–2802, 2007.

[34] J. Zhang, A. Morris, E. Martin, and C. Kiparissides, “Estimation of impurity and fouling in batch polymerisation reactors through the application of neural networks,” *Computers & chemical engineering*, vol. 23, no. 3, pp. 301–314, 1999.

[35] S. Lalot and S. Lecoeuche, “Online fouling detection in electrical circulation heaters using neural networks,” *International Journal of Heat and Mass Transfer*, vol. 46, no. 13, pp. 2445–2457, 2003.

[36] A. Pacheco-Vega, M. Sen, K. Yang, and R. L. McClain, “Neural network analysis of fin-tube refrigerating heat exchanger with limited experimental data,” *International Journal of Heat and Mass Transfer*, vol. 44, no. 4, pp. 763–770, 2001.

[37] S. S. Sablani, “A neural network approach for non-iterative calculation of heat transfer coefficient in fluid–particle systems,” *Chemical Engineering and Processing: Process Intensification*, vol. 40, no. 4, pp. 363–369, 2001.

[38] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, p. 436, 2015.

[39] I. Johnson, W. T. Choate, and A. Davidson, “Waste heat recovery. technology and opportunities in us industry,” BCS, Inc., Laurel, MD (United States), Tech. Rep., 2008.

[40] H. Jouhara, N. Khordehgah, S. Almahmoud, B. Delpech, A. Chauhan, and S. A. Tassou, “Waste heat recovery technologies and applications,” *Thermal Science and Engineering Progress*, vol. 6, pp. 268–289, 2018.

[41] S. M. Jeter, “Effectiveness and lmtd correction factor of the cross flow exchanger: a simplified and unified treatment,” in *ASEE Southeast Section Conference*. Citeseer, 2006, pp. 1–10.

[42] M. T. Ribeiro, S. Singh, and C. Guestrin, “Model-agnostic interpretability of machine learning,” *arXiv preprint arXiv:1606.05386*, 2016.

[43] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach.* Malaysia; Pearson Education Limited,, 2016.

[44] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural networks*, vol. 61, pp. 85–117, 2015.

[45] I. Goodfellow, Y. Bengio, and A. Courville, “Deep learning cambridge,” 2016.

[46] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, “The loss surfaces of multilayer networks,” in *Artificial Intelligence and Statistics*, 2015, pp. 192–204.

[47] G. Montavon, G. Orr, and K.-R. Müller, *Neural networks: tricks of the trade.* springer, 2012, vol. 7700.

[48] J. S. Judd, *Neural network design and the complexity of learning.* MIT press, 1990.

[49] R. Sutton, “Two problems with back propagation and other steepest descent learning procedures for networks,” in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society, 1986*, 1986, pp. 823–832.

[50] K. C. Kiwiel, “Convergence and efficiency of subgradient methods for quasiconvex minimization,” *Mathematical programming*, vol. 90, no. 1, pp. 1–25, 2001.

[51] D. M. Underwood and R. R. Crawford, *Dynamic nonlinear modeling of a hot-water-to-air heat exchanger for control applications.* American Society of Heating, Refrigerating and Air-Conditioning Engineers, 1991.

[52] R. W. Haines and D. C. Hittle, *Control systems for heating, ventilating, and air conditioning.* Springer Science & Business Media, 2006.

[53] N. J. Nagelkerke et al., "A note on a general definition of the coefficient of determination," *Biometrika*, vol. 78, no. 3, pp. 691–692, 1991.

[54] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[55] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception.* Elsevier, 1992, pp. 65–93.

[56] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010.* Springer, 2010, pp. 177–186.

[57] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[58] F. Chollet et al., "Keras," https://keras.io, 2015.

[59] Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence o (1/kˆ 2)," in *Doklady AN USSR*, vol. 269, 1983, pp. 543–547.

[60] R. Bardenet, M. Brendel, B. Kégl, and M. Sebag, "Collaborative hyperparameter tuning," in *International conference on machine learning*, 2013, pp. 199–207.

[61] B. Spang and W. Roetzel, "Approximate equations for the design of cross-flow heat exchangers," in *Design and Operation of Heat Exchangers.* Springer, 1992, pp. 125–134.

[62] B. Jackson and R. Troupe, "Plate heat exchanger design by epsilon-ntu method(heat transfer effectiveness-number of transfer unit relations determined, using differential equations to obtain temperature profile of each channel in exchanger)," in *CHEMICAL ENGINEERING PROGRESS, SYMPOSIUM SERIES*, no. 64, 1966, pp. 185–190.

[63] F. P. Incropera and D. P. DeWitt, *Fundamentals of heat and mass transfer.* Wiley New York, 1996, vol. 4.

[64] L. F. Moody, "Friction factors for pipe flow," *Trans. Asme*, vol. 66, pp. 671–684, 1944.

[65] G. Filonenko, "Hydraulic resistance in pipes. heat exchanger design handbook. teploenergetica," 1954.

[66] E. N. Sieder and G. E. Tate, "Heat transfer and pressure drop of liquids in tubes," *Industrial & Engineering Chemistry*, vol. 28, no. 12, pp. 1429–1435, 1936.

[67] W. M. Kays and A. L. London, "Compact heat exchangers," 1984.

[68] C. K. Batchelor and G. Batchelor, *An introduction to fluid dynamics.* Cambridge university press, 2000.

[69] A. Zukauskas and R. Ulinskas, "Heat transfer in tube banks in crossflow," 1988.

[70] J. P. Hartnett, W. M. Rohsenow, E. Ganic, and Y. Cho, *Handbook of heat transfer.* McGraw-Hill, 1973.

[71] E. U. Schlunder, "Heat exchanger design handbook," 1983.

[72] J. Biery, "Prediction of heat transfer coefficients in gas flow normal to finned and smooth tube banks," *Journal of Heat Transfer*, vol. 103, no. 4, pp. 705–714, 1981.

[73] A. P. Colburn and E. d. P. de, "Mean temperature difference and heat transfer coefficient in liquid heat exchangers," *Industrial & Engineering Chemistry*, vol. 25, no. 8, pp. 873–877, 1933.

[74] A. Zukauskas, "Convective heat transfer in cross flow," *Handbook of single-phase convective heat transfer*, 1987.

[75] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.

[76] V. A. Epanechnikov, “Non-parametric estimation of a multivariate probability density,” *Theory of Probability & Its Applications*, vol. 14, no. 1, pp. 153–158, 1969.

[77] Y. Bengio, P. Simard, P. Frasconi et al., “Learning long-term dependencies with gradient descent is difficult,” *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[78] J. W. Hines, “A logarithmic neural network architecture for unbounded non-linear function approximation,” in *Proceedings of International Conference on Neural Networks (ICNN’96)*, vol. 2. IEEE, 1996, pp. 1245–1250.

[79] J. Pek, O. Wong, and C. Wong, “Data transformations for inference with linear regression: Clarifications and recommendations.” *Practical Assessment, Research & Evaluation*, vol. 22, 2017.

[80] J. W. Osbourne, “Notes on the use of data transformation.” *Practical Assessment, Research & Evaluation*, vol. 8, no. 6, p. n6, 2002.

[81] I. J. Goodfellow, O. Vinyals, and A. M. Saxe, “Qualitatively characterizing neural network optimization problems,” *arXiv preprint arXiv:1412.6544*, 2014.

[82] L. Breiman, “Bagging predictors,” *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[83] T. O. Kvålseth, “Cautionary note about r 2,” *The American Statistician*, vol. 39, no. 4, pp. 279–285, 1985.

[84] A. Scott and C. Wild, “Transformations and r 2,” *The American Statistician*, vol. 45, no. 2, pp. 127–129, 1991.

[85] A. Krogh and J. Vedelsby, “Neural network ensembles, cross validation, and active learning,” in *Advances in neural information processing systems*, 1995, pp. 231–238.