A Framework for Data Regression of Heat Transfer Data Using Machine Learning

Jose Loyola-Fuentes[1], Nima Nazemzadeh[1], Emilio Diaz-Bejarano[1], Simone Mancin[2,3] and Francesco Coletti[1,3*]

[1]Hexxcell Ltd., Foundry Building, 77 Fulham Palace Road, W6 2AF, London, United Kingdom
[2]University of Padova, Department of Management and Engineering, S.lla San Nicola, 3, Vicenza, 36100, Italy
[3]Department of Chemical Engineering, Brunel University London, Kingston Lane, Uxbridge, UB8 3PH, United Kingdom

[*]Corresponding author: f.coletti@hexxcell.com

Abstract

Machine Learning (ML) algorithms are emerging in various industries as a powerful complement/alternative to traditional data regression methods. A major reason is that, unlike deterministic models, they can be used even in the absence of detailed phenomenological knowledge. Not surprisingly, the use of ML algorithms is being explored also in heat transfer applications. It is of particular interest in systems dealing with complex geometries and underlying phenomena (*e.g.* fluid phase change, multi-phase flow, heavy fouling build-up). However, heat transfer systems present specific challenges that need addressing, such as the scarcity of high-quality data, the inconsistencies across published data sources, the complex (and often correlated) influence of inputs, the split of data between training and testing sets, and the limited extrapolation capabilities to unseen conditions. In an attempt to help overcome some of these challenges and, more importantly, to provide a systematic approach, this article reviews and analyses past efforts in the application of ML algorithms to heat transfer applications, and proposes a regression framework for their deployment to estimate key quantities (*e.g.* heat transfer coefficient), to be used for improved design and operation of heat exchangers. The framework consists of six steps: i) data pre-treatment, ii) feature selection, iii) data splitting philosophy, iv) training and testing, v) tuning of hyperparameters, and vi) performance assessment with specific indicators, to support the choice of accurate and robust models. A relevant case study involving the estimation of the condensation heat transfer coefficient in microfin tubes is used to illustrate the proposed framework. Two data-driven algorithms, Deep Neural Networks and Random Forest, are tested and compared in terms of their estimation and extrapolation capabilities. The results show that ML algorithms are generally more accurate in predicting the heat transfer coefficient than a well-known semi-empirical correlation proposed in past studies, where the mean absolute error of the most suitable ML model is 535 $[Wm^2K^{-1}]$, compared to the error using the correlation of 1061 $[Wm^2K^{-1}]$. In terms of extrapolation, the selected ML model has a mean absolute error of 1819 $[Wm^2K^{-1}]$, while for the correlation is 1111 $[Wm^2K^{-1}]$, indicating a disadvantage of the use of semi-empirical models, although the comparison was not entirely suitable, given that the correlation was used as is and no training was done. In addition, feature selection enables simpler models that depend only on features that are potentially most related to the target variable. Special attention is needed however, as overfitting and limited extrapolation capabilities are common difficulties that are encountered when deploying these models.

1. Introduction

The thermal performance of heat exchangers is predominantly dependent on the value of the heat transfer coefficient (HTC), which determines the rate of heat exchanged for a given area. Its value is the result of a series of complex interactions between the thermo-physical properties of fluids, the geometry of the system, operating conditions and several other factors (*e.g.* metallurgy, surface conditions, fouling, etc.) [1, 2]. Accurate estimation of the heat transfer coefficient is important to better design and monitor heat transfer equipment. The efficiency of such equipment plays a crucial role in energy systems in terms of safety, profitability, and impact on the environment. Given that thermal energy usage represents a substantial proportion of the total energy consumption in developed countries, efficient heat transfer is critical to increase the sustainability of industries across multiple sectors, such as chemical, biochemical, petrochemical, oil and gas, food, electronic, solar energy, pulp and paper, etc [3, 4]. As an example, thermal energy usage in the United Kingdom corresponds to approximately 50% of the total energy consumption and the related processes are responsible for around a third of the CO_2 emissions in the entire country [5]. In this context, there is a need for more accurate prediction of the rate of heat transfer in traditional industrial heat transfer equipment, but, more critically, to develop new and accurate tools to estimate these quantities for novel technologies, to enable transfer to and adoption by the process industries. Many of the novel technologies arising tend to involve complex geometries (*e.g.* for heat transfer enhancement) and underlying phenomena (*e.g.* fluid phase change, multi-phase flow), with multiple interacting variables and non-intuitive effects. The use of ML algorithms emerges as a promising alternative to traditional correlation approaches to identify key variables and capture their impact on the rate of heat transfer. While the use of ML algorithms to heat transfer applications has been explored in the past, as will be reviewed in the following, there is no systematic approach to apply these tools while taken into consideration the specific challenges encountered in the field. The objective of this work is to develop a framework to provide to help in better analysing data in complex heat transfer applications and easily train models to estimate relevant quantities, even with limited understanding of the underlying phenomena involved.

A great deal of research efforts was put in the 1960s and 1970s to devise design methods for single-phase, two-phase and multi-phase systems which are still used today [6, 7, 8]. These traditional methods for estimating local HTCs consisted in performing experiments to calculate specific dimensionless numbers and establishing correlations among them. For example, in turbulent single-phase flows, the Chilton-Colburn [9] and the Dittus-Boelter [10] correlations, where the Nusselt number is a power function of both Reynolds and Prandtl numbers, have a typical error range of 10% - 30% [11, 12]. However, over the last decades, researchers have tried to improve the accuracy of these correlations by using linear or nonlinear regression techniques. In two-phase flows, the complexity of the phenomena involved increases exponentially and many more system characteristics must be accounted for. For example, small changes in operating conditions may lead to a transition from one flow pattern to another, which profoundly affects heat transfer [13, 14, 15]. For this reason, efforts have focused on developing robust correlations to estimate the two-phase HTCs, particularly in the area of cooling of electronics [16, 17, 18], refrigeration and air conditioning [19, 20, 21, 22]. This is not a trivial task, as a successful attempt would need large amounts of experimental data and a deep understanding of the underlying heat and mass transfer mechanisms. The inherent complexity of these systems, whose behaviour is determined by several, interacting phenomena, leads to a lack of fundamental understanding of the correlations among variables. As a result, appropriate mathematical models capable of estimating and predicting target variables (*e.g.* a two-phase HTC) as a function of relevant inputs are challenging to develop and validate, and very often computationally expensive to solve. An example is large-scale heat transfer problems solved with computational fluid dynamics, which need to solve a set of extremely large numbers of equations and suffer from convergence problems [23, 24].

An alternative approach to traditional regression methods is using data-driven methods that have been receiving great attention in various areas of science and engineering. This is due to the rapid development in computational power, the lower cost of sensors, and storage capacity, which lead to wider availability of data and more efficient algorithms such as artificial intelligence, and more specifically machine learning (ML). These algorithms and data-driven approaches do not necessarily require a deep phenomenological understanding of the systems and can provide a solution to problems that may be too challenging to solve with deterministic models [25]. ML and heat transfer has not been an exception.

In this context, the optimal mapping of specific inputs to targeted outputs, using previously measured data, is the main goal of Supervised Learning (SL), which is a widely studied branch of Machine Learning [26]. In cases where input and output variables are continuous, these methods are referred to as Regression, which is commonly used in heat transfer applications, particularly linear and nonlinear regression models. The deployment of regression algorithms varies depending on the characteristics of the desired model. This range covers simple implementation approaches such as a linear model, to more abstract and complex methods such as Deep Neural Networks (DNNs) or Random Forest (RF). These methods present the advantage of exploiting the availability and interactions of measured or calculated data efficiently, as well as being able to be trained with simplicity [27]. All these characteristics enable predictive capabilities that facilitate the design of predictive analytics or decision-making criteria. However, there are very well-known challenges in the application of the ML algorithms namely, the size and quality of the collected data and model performance assessment. As discussed below, heat transfer problems are not an exception.

This work aims to (i) review state-of-the-art in the application of ML to heat transfer problems; (ii) identify benefits and limitations of ML-based regression approaches applied to heat transfer systems; (iii) provide a framework for the systematic use of ML algorithms to regress heat transfer data, as an alternative to classic correlations; and (iv) illustrate the approach with a relevant Case Study. The work is structured as follows. Section 2 reviews past efforts in the application of ML algorithms to regress heat transfer data and analyses the main challenges encountered. Section 3 describes in detail regression models from the ML perspective and compares the approach against classic methods. Section 4 provides a step-by-step description of the framework developed in this work. In Section 5, the case study used to illustrate the application of the framework is introduced: the estimation of the condensation heat transfer coefficient in microfin tube systems. Section 6 provides the specific method used in the case study, where two regression algorithms are tested and compared with a validated and widely used deterministic correlation for the same system [16]. The use of different input features, hyperparameter tuning on each algorithm, and their sensitivity to specific features are assessed to identify those scenarios where ML can provide benefits over the use of empirical correlations, and those where the latter option seems to be more desirable. The results and outcomes of the Case Study are shown and discussed in Section 7 and the conclusions are presented in Section 8.

2. Application of ML Algorithms to Heat Transfer Problems: Literature Review

ML algorithms applied to heat transfer problems require large amounts of data over a wide range of operating conditions, geometrical parameters and working fluids. Obtaining these large datasets is time-consuming. Moreover, as multivariate regression (*e.g.* more than one input variable) is used in most heat transfer applications, it is of great significance that the each input variable is selected properly.

Data quality is another challenge that very much depends upon the class of heat transfer problems (these classes are defined next) and the scale of the heat transfer system. For instance, in industrial scale problems, several factors may affect the quality of the measurements including large measurement errors, missing measurements or unexpected events. The quality of the collected data has a significant effect on the ML model performance since these models fundamentally rely on the features of the data. Obtaining high-quality data across various sources is a major challenge in the heat transfer area due to potential inconsistencies across these sources.

Among the challenges described above, the latter becomes important when assessing a models' prediction capabilities, including accuracy and extrapolability. The model accuracy refers to the error between the ML outputs and the real data. Extrapolability refers to the model prediction capabilities on the data that does not lie in the search space. This is of great relevance in heat transfer. The performance of the models can also be assessed against the accuracy of existing models in the literature. Knowing these challenges allow for a systematic classification of types of heat transfer problems that are suitable for ML deployment.

Heat transfer problems where an ML approach can be applied are broadly classified into i) design and synthesis of heat transfer systems, ii) operation of heat transfer systems, iii) calculation of thermophysical properties and iv) fundamentals of heat transfer phenomena. ML algorithms have been applied to the synthesis of heat exchanger networks with the aid of genetic algorithms [28, 29] and to optimise the design of heat exchanger geometry [30, 31, 32]. The challenge here is to ensure that ML algorithms incorporate manufacturability considerations and meet generally accepted industry standards for construction. From an operational point of view, ML algorithms have been applied to the monitoring of heat exchangers and the prediction of fouling [33, 34, 35, 36, 37], which affect a large portion of the equipment in the process industry [38]. The complex system dynamics, the general lack of key measurements and often poor measurements available that suffer from bias, drift and other errors [39, 40, 41, 42] constitute the biggest barriers to the application of pure machine learning algorithms to this class of problems. Successful industrial applications have been shown when ML models are combined with deterministic models [43]. There are other studies that have incorporated ML algorithms to study fouling rather from a fundamental point of view to analyse the fouling behaviour against various operating conditions excluding the time variable [44, 45]. However, the amount of data available severely limits the extrapolability of these results. Other examples of the deployment of ML algorithms is the optimisation and control of indoor temperature and energy consumption in buildings [46] and the use of state-of-the-art ML models to control variable speed compressor in chillers [47]. On the topic of the calculations of thermophysical properties, ML models have been developed to estimate the viscosity, density, and thermal conductivity of thermal fluids [48, 49, 50, 51]. While all of these areas are of interest, this paper focuses on the latter class of problems: the application of ML to the prediction of fundamental heat transfer problems.

Table 1 provides an overview of some of the recent applications of ML algorithms in heat transfer fundamentals and by no means is intended to be a comprehensive review of ML applications in this field. The first area reported in Table 1 is heat transfer in solids. While this phenomenon is better understood than systems involving convection or radiation, there are still challenges related to sensor limitations and low model accuracy (*e.g.* the estimation of the heat transfer across the interface of two solid materials in cooling of electronics [23]). Consequently, the application of ML algorithms has been explored by various authors as an alternative approach to overcome these limitations. Wu *et. al* [52] developed several ML-based models using least-squares boosting (LSBoost), support vector machine (SVM), and Gaussian process regression (GPR) to model the temperature discontinuity between two dissimilar materials using approximately 80,000 material systems, which is described as interfacial thermal resistance (ITR) [52]. The main motivation for applying ML algorithms was driven by the limitations of the first-principles approaches commonly used for this problem, including the large discrepancies between experimental and predicted data, when measured via the acoustic mismatch model (AMM) and diffuse mismatch model (DMM), and computational limitations for more accurate models such as molecular dynamics (MD) simulations. The results of the study showed a significant improvement in prediction accuracy when using these algorithms, compared to its deterministic counterparts. Peng *et al.* [53] applied a convolutional neural network (CNN) model to predict steady-state heat conduction with a random geometry in a 2D space and illustrated that the model is capable of predicting the temperature distribution quite accurately, while is 3 to 4 times more computationally efficient, compared to numerical solutions provided by OpenFOAM. Another example of ML application in solid-state heat transfer is the work of Szénási *et al.* [54], which developed an artificial neural network (ANN) model to predict the conduction heat transfer coefficient according to the temperature signals recorded during the heat treatment process.

Another area in heat transfer where ML algorithms have been applied is single-phase fluid flow. Systems in this area can be modelled using computational fluid dynamics (CFD) simulations, which are known to be computationally demanding and often come with convergence issues [23]. An example of the ML application in this area is the work of Kwon *et al.* [55] who analysed the performance of a random forest (RF) regressor against a finite volume model in predicting the local convective HTC of cooling channels with 243 different channel geometries. The study showed that the RF regressor has similar predictive abilities compared to the results from a CFD simulation with high accuracy and more versatility in terms of expanding the range of parameters and refining the resolution. Cai *et al.* [56] developed a physics-informed neural network (PINN) model for simulating the temperature, pressure, and velocity of internal and external flow-forced convection and compared the results with CFD simulations. PINN is a type of neural network model that combines deterministic models with the ML algorithm as an extra constraint to guide the ML algorithm for more accurate results. Moreover, Souayeh *et al.* [57] applied an ANN model to predict the thermal energy transport coefficient, pressure penalty, and thermohydraulic efficiency of a circular channel with corrugated spring tape inserts and the results lead to predictions with approx. 97% of accuracy on an unseen data set. Another example in single-phase flow area is the study of Pai and Weibel [58], which shows the application of ML-based surrogate models to optimise the flow cross-section shapes in constant cross-section channels. The study demonstrates that the model performs good prediction accuracy for shapes that have been used to train the model but rather has poor prediction capabilities on unseen shapes.

In two-phase fluid flow, ML algorithms have been deployed for two main classes of problems, namely flow pattern classification and prediction of key quantities (*i.e.* HTC and pressure drops). ML models are very suitable for classification tasks, so it is not surprising that one class of problem that has been explored is flow pattern classification. Mask *et al.* [59] developed several ML models using tree-based algorithms to classify two-phase flow regimes starting from temperature and heat flux data (Tree-based algorithms are a type of ML model that is deployed to solve classification and regression problems by using tree-like structures for predictions) [26]. They assessed the performance of several models and reported the same accuracy of approx. 93% for all the models developed. Hobold and da Silva [60] proposed a framework using a combination of machine learning models to classify different boiling regimes using only images, this was an interesting approach as it does not rely on measurements but rather visual observations. Loyola-Fuentes *et al.* [61] incorporated different algorithms such as random forest, K-nearest neighbours, and multilayer perceptron to classify two-phase flow patterns in pulsating heat pipes. The area of flow pattern recognition has been receiving increasing attention recently and researchers have conducted a lot of research on the topic using machine learning models, a large portion of which can be found in the review article carried out by Yang *et al*. [62]. The second class of problems where ML algorithms have been applied is the prediction of heat transfer coefficients and pressure drops in two-phase flows. Hughes *et al.* [63] also showed the superiority of machine learning algorithms over conventional correlations for predicting condensation frictional pressure drop and heat transfer coefficient. Another example of the prediction of pressure drops in a two-phase heat transfer system is the study by Khosravi *et al.* [64]. They developed models using ANN and SVM algorithms to model the pressure drop during the evaporation of R407C and illustrated that ANN outperformed the SVM with an R^2 = 0.99865 [23, 64]. Hobolt and da Silva [65] developed a framework for a specific application of ML algorithms in pool boiling problems to quantify the heat flux of nucleate boiling by using different types of ML models and reached an error in predictions of less than 7%. The authors trained their algorithms based on visualisation data to infer the heat flux from nucleate boiling experiments. Longo *et al.* [66, 67, 68] demonstrated the interesting capabilities of the use of ML algorithms to predict both the condensation and boiling heat transfer coefficients and the two-phase pressure drops inside brazed plate heat exchangers. Several other applications also exist, which can be found in the review article by Hughes *et al.* [23].

As seen from the review above, it is of great importance to understand the inherent differences between traditional and ML-based regression methods to appropriately deploy these techniques. These differences are analysed in detail in the following section, from a modelling and conceptual point of view.

3. Traditional versus Machine Learning Regression

Regression can be defined as the process of finding a mathematical representation of a set of output or target variables (Y_i), contained in $\boldsymbol{Y}$, as a function of a set of input or explanatory variables ($\boldsymbol{x}_i$), contained in $\boldsymbol{X}$. In addition, both $\boldsymbol{X}$ and $\boldsymbol{Y}$ are continuous. When $\boldsymbol{X}$ has only one input feature (*i.e.* a vector of N observations), the regression is said to be univariate, whereas when $\boldsymbol{X}$ has more than one input feature (*i.e.* a matrix of observations of multiple features M) the regression is named multivariate [26]. A general formulation of this representation is shown in Equation (1), where $\boldsymbol{\Theta}$ contains adjustable parameters that are estimated during the regression. Note that $\boldsymbol{\Theta}$ depends on whether $\boldsymbol{X}$ is univariate or multivariate and the type of regression. Note that notations with bold capital letters for variables and parameters (*e.g.* $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\Theta}$) represents matrices, bold small letters (*e.g.* $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}$) are vectors, and light letters are the representation of their elements.

$$\boldsymbol{Y} = f(\boldsymbol{X}, \boldsymbol{\Theta}) \tag{1}$$

Finding $f(\boldsymbol{X}, \boldsymbol{\Theta})$ can be done in numerous ways. Perhaps the most common method is linear regression, where the machine learns a linear representation of the target variables, described by Equation (2).

$$\boldsymbol{Y} = \boldsymbol{\theta}_1 + \boldsymbol{X}\,\boldsymbol{\theta}_2 \tag{2}$$

In this case, $\boldsymbol{\Theta}$ is the parameter matrix comprising the intercept $\boldsymbol{\theta}_1$ and the slope $\boldsymbol{\theta}_2$. These parameters are normally estimated using training data via the minimisation of the sum of squared residuals (SSR). Once the parameters in $\boldsymbol{\Theta}$ have been optimally estimated, model-based predictions are enabled, and new values of $\boldsymbol{X}$ may be utilised to obtain a predicted estimation of $\boldsymbol{Y}$.

Linear regression offers great interpretability, as the interactions among input and output variables are clear and no transformations are carried out in between, and owing to its simplicity has been applied to

Table 1: Relevant studies on the application of ML algorithms in the fundamentals of heat transfer phenomenon

	Reference	Year	Application	ML models
Heat transfer in solids	Wu *et al.* [52]	2019	Modelling interfacial thermal resistance (ITR)	LSBoost, SVM, and GPR
	Peng *et al.* [53]	2020	Modelling heat conduction in complex geometries	CNN
	Szénási *et al.* [54]	2020	Inverse heat conduction problem	ANN
Heat transfer in single-phase fluid flow	Kwon *et al.* [55]	2020	Modelling local convective HTC of a cooling channel	RF regressor
	Cai *et al.* [56]	2021	Modelling temperature, velocity, and pressure of flow-forced convection	PINN
	Souayeh *et al.* [57]	2021	Modelling friction factor and Nusselt number of flow in a circular tube	ANN
	Pai and Weibel [58]	2022	Optimization of internal flow channel cross-sections	ANN
Heat transfer in two-phase fluid flow	Mask *et al.* [59]	2019	Gas-liquid flow pattern prediction	RF, Boosted bagging, and Extreme gradient boosting
	Hobold and da Silva [60]	2018	Classification of boiling regimes	PCA, SVM, and ANN
	Loyola-Fuentes *et al*. [61]	2022	Flow pattern classification in heat pipes	KNN, RF, and MLP
	Hughes *et al*. [63]	2021	Modelling condensation frictional pressure drop and heat transfer coefficient	SVR, ANN, and RF
	Khosravi *et al.* [64]	2018	Modelling pressure drop during evaporation of R407C	ANN and SVR
	Hobold and da Silva [65]	2019	Visualisation-based quantification of nucleate boiling heat flux	PCA, CNN, and MLP
	Longo *et al.* (a) [66]	2020	Modelling of condensation HTC inside brazed plate heat exchangers	ANN
	Longo *et al*. (b) [67]	2020	Modelling of boiling HTC inside brazed plate heat exchangers	ANN
	Longo *et al*. (c) [68]	2020	Modelling refrigerant two-phase frictional pressure inside brazed plate heat exchangers	GBM model
	Calati *et al*. [69]	2020	Modelling of nucleate boiling inside porous media	ANN
	Das and Akpinar [70]	2018	Modelling convective HTC of air-heated solar collector	SVM

many problems. However, it is not capable of capturing non-trivial and hidden interactions and dealing with larger numbers of observations and input variables thus it can be used only in simple problems. In complex systems, the correlations among system variables tend to be non-linear rather than linear. For example, the design of heat exchangers needs the calculation of heat transfer coefficients, which depends on a wide range of input variables, such as geometry, fluid thermo-physical properties and operating conditions, whose interactions are known to be non-linear. For these types of systems, the use of non-linear regression models, which define $f(\boldsymbol{X}, \boldsymbol{\Theta})$ in Equation (1) as a non-linear function, is required. This makes the regression analysis more difficult to solve and requires more efficient algorithms, which could be impractical in various cases. Hence, more efficient models and algorithms are needed to tackle these complex problems.

The two methods described above are referred to as *traditional regression* models, in which the assumptions and rules are introduced in the form of a regressed function (hand-crafted function) and constraints. These methods require *a priori* knowledge of the functional form of the equation to be regressed and use statistical techniques to fit the model into the collected data. On the contrary, *ML-based regression* does not require prior knowledge and the functional form of the equation to be regressed is not imposed and it can help bridge the gap when the detailed physics of the system is not known. Figure 1 illustrates a broad overview of the classical- and ML-based algorithms for regression problems. The main difference between these two approaches lies in the model training phase. ML-based regression algorithms utilise a rather different approach to solve these problems: instead of using a pre-defined model, a model configuration is set up by defining a set of hyperparameters. These hyperparameters shall not be confused with the model parameters. A model hyperparameter is a parameter that is external to the model and its value controls the learning process and cannot be affected by the data [71]. Afterwards, part of the data is used as a training set to identify settings for the ML model parameters using statistical techniques. Several training algorithms exist that are similar to the ones used in classical regression models. When training the ML model, a set of rules for the model parameters will be found that provides the best realistic prediction (prediction algorithm) [72]. After the training step, a new input is fed to the model to assess the predictive capabilities of the built model. Table 2 provides a summary of the pros and cons of both modelling techniques. In science and engineering, traditional regression methods are also referred to as mechanistic (or white-box) modelling since they require an understanding of the underlying physics of the problem, while the ML counterpart is referred to as black-box modelling. This is because the phenomenological understanding is missing in the latter modelling approach. In most complex systems, developing a mechanistic model can be challenging and time-consuming since the correlations between all variables are unknown or difficult to measure [74]. Hence, ML-based models can become useful to extract those correlations by using only the features of the data. It is evident that since the only source of information for these models is the collected data set, they require a larger amount of data compared to traditional approaches [75]. Moreover, it is well-known that ML-based models have strong interpolation properties and poor extrapolation properties. As a result, the predictive capabilities are limited to the validation space.

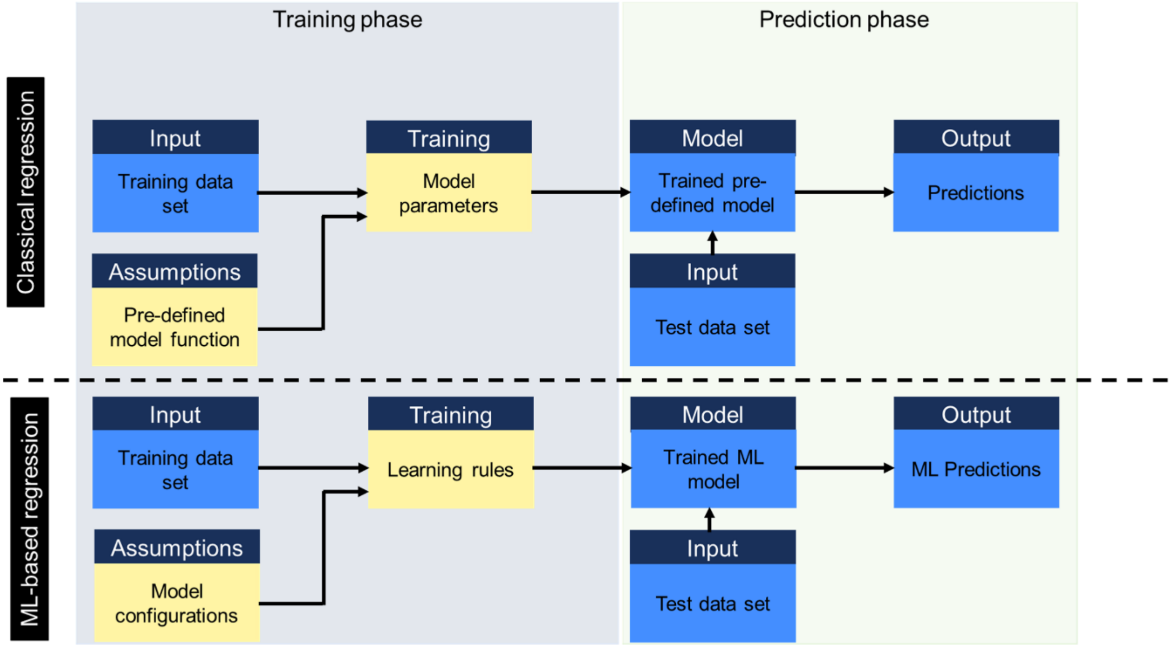


Figure 1: Overview of classical (top) and ML-based (bottom) regression models workflow [72]

Table 2: Comparison between traditional and ML-based regression models [73]

Regression technique	Pros	Cons
Traditional	• A lower amount of data required • Better extrapolation capabilities in general • Physical interpretability	• High phenomenological understanding required • Computationally expensive for complex problems • Large prediction errors • Deterministic
ML-based	• Fewer assumptions required • Less *a priori* knowledge required • Lower prediction errors • Possibly probabilistic	• Poor extrapolation capabilities • A large amount of data required • No physical interpretation

A common and widely used ML-based modelling option to capture complex interactions is the use of Artificial Neural Networks (ANNs). ANNs employ universal approximator functions (activation functions) to perform complex regression tasks and unhide interactions between input and output variables that would be otherwise not identified [76]. They consist of individual processing units named neurons, which process the input data with an activation function to the next neuron. These units can be fully connected, present loops or skip connections. These different configurations give rise to different architectures and types of ANNs (*e.g.* Feed-forward, Convolutional, Recurrent, Graph, etc.) [77, 78, 79]. The basic structure of an ANN is shown in Figure 2, where neurons are grouped in different levels: (i) an input layer, which takes as arguments the data from selected input features, (ii) a hidden layer that builds the representation of the outputs based on the inputs, and (iii) an output layer that holds the prediction of the output variables based on the calculations in previous layers. When the number of hidden layers increases and the structure of the ANN becomes more complex, these are referred to as Deep Neural Networks (DNNs), which enable establishing the representation of more complex systems in a wide range of areas such as image, text, and sequential data processing.

A set of hyperparameters must be defined for these models prior to training like other machine learning models, which may have a significant effect on the model performance. That includes (i) the number of neurons or processing units, (ii) the activation functions for each layer to approximate the output of each neuron, (iii) the number of epochs indicating the number of iterations that the learning algorithm works on the entire training dataset, (iv) loss function as the objective function for the optimisation problem (*e.g.* mean absolute error, mean squared error, etc.), (v) optimisation algorithm to define the training method for fitting the model to the training data (*e.g.* gradient-based such as "Adam" or "SGD" and gradient-free such as particle swarm optimisation) [80, 81], (vi) batch size as the number of data samples that are processed before the model parameters are updated at each epoch, (vii) weight and bias initialisation that initially assigns random values to the neural network parameters for training. Other hyperparameters also exist to configure a neural network model such as regularisation, learning rate, and dropout, the details of which can be found in [82]. Current computational tools such as Python and R allow for simple and straightforward options to build, save and load ML-based models. This is done to avoid building a pre-trained model from scratch, which represents a disadvantage versus equation-based models that have been trained beforehand.

Other popular methods for ML-based regression are those based on decision trees, in particular the Random Forest (RF) algorithm [83]. For regression tasks, this algorithm builds a fixed number of decision trees, which are fed from independent subsets of data. Each tree uses a random selection of input features and grows the trees to obtain a prediction. Once all trees are finished, the mean predicted value is estimated and output as the resulting predicted value. Using this approach is advantageous as no bias is introduced in each tree, and the contribution of each input feature to the target value can be estimated. The overall structure of the RF algorithm is depicted in Figure 3. Similar to ANNs, a set of hyperparameters have to be set for the RF models. However, the nature of these hyperparameters would be different from ANN models due to the difference in the implemented algorithms. The RF hyperparameters include (i) the number of decision trees that have to be built by the model to average out at the end, (ii) the split criterion that states the condition of a decision node, (iii) the use of bootstrapping that divides the data set into different subsets and train the model on those subsets that are randomly chosen with replacements, (iv) the minimum sample size to split that defines the minimum number of samples used to split an internal node. There are other hyperparameters such as the maximum depth of a tree, maximum number of nodes, and maximum features, the details of which can be found in [84].

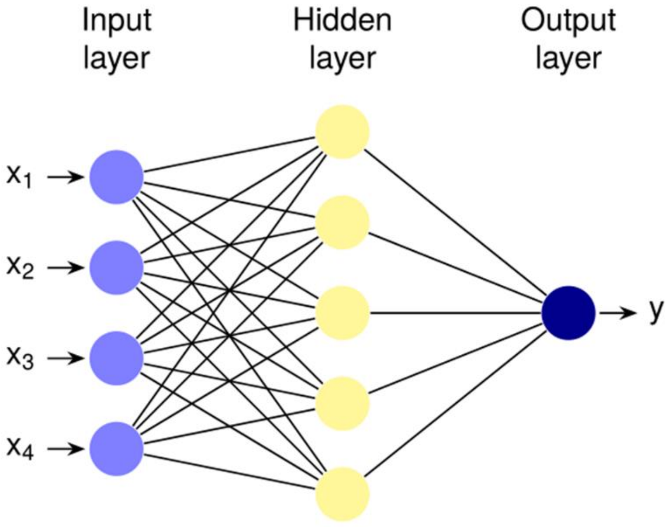


Figure 2: Basic structure of an ANN

The benefits of the use of these more refined methods (*i.e.* flexibility in their use, relatively high accuracy and realisation of non-trivial correlations between input and output data) are mainly limited by data-driven issues (*i.e.* quantity and quality) and by their own interpretability. On one hand, the number of available observations plays a major role in regression models, provided that such data cover a varied or expected set of experiments or settings. The number of input features is also important, as redundant variables could mask underlying correlations among input features. The quality of the data should also be considered, as faulty measurements are mostly present in real-life systems. On the other hand, methods such as DNNs and RFs tend to lose interpretability (in a physical fashion) as they increase the number of parameters to be estimated during training (vector $\boldsymbol{\Theta}$).

Similar to other machine-learning algorithms, ANNs and RFs must undergo training/validation, and testing phases based on the collected dataset. Thereafter, models can be compared concerning their prediction accuracy and generalisation capabilities. A few techniques are currently practised for training and validation of the data-driven models as listed in the following: (i) holdout, (ii) k-fold cross-validation, and (iii) ensemble method. In the first validation technique, the dataset is split randomly into training and test set. After training, the model is tested against an unseen dataset. Hence, a single estimate of the performance will be provided, which in most cases is not sufficient to evaluate the model performance. There are two main disadvantages associated with this method. One is that the performance of the model on a different random split will be unknown or, in other words, the model sensitivity to the data split will not be determined. To overcome this issue, it is suggested to employ the k-fold cross-validation technique, in which the data is split into different training and testing folds and the model is trained for each [85]. This approach will give a more realistic performance of the model over the entire dataset. The other drawback to the random split method rises from the fact that the optimisation algorithms for training these models heavily depend upon the initialisation of the model parameters. Since the model parameters are randomly initialised, the optimised parameters will be different every time the model is trained. Therefore, by employing the first validation approach, the sensitivity of the model performance against the initialisation of the parameters will be unknown. An approach to resolve this issue is to fix the initialisation of the model or use an ensemble method to train the model with various initialisations and provide a more robust model performance against the data. In this work, the first two validation techniques (*i.e.* holdout and k-fold cross-validation) are utilised separately and in combination to more accurately estimate the model performance on an unseen dataset [86].

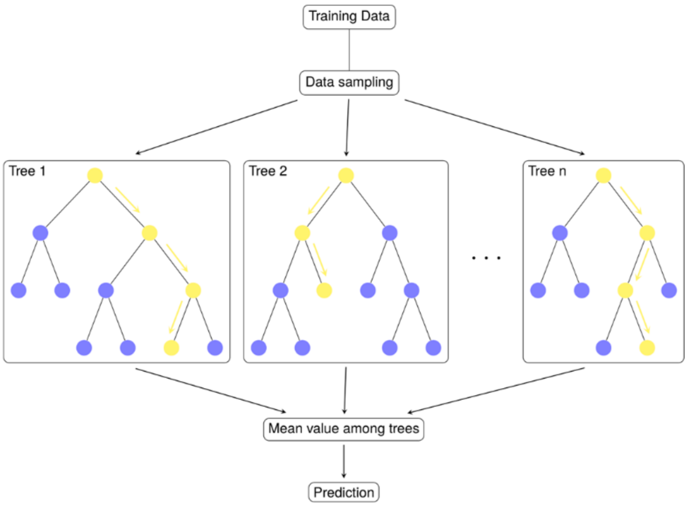


Figure 3: Basic structure of a Random Forest

The model complexity in terms of hyperparameters plays an important role in determining the generalisation capabilities of the model. This falls under the bias-variance trade-off analysis of models with various complexities. A schematic representation of the concept of bias-variance trade-off is illustrated in Figure 5. An oversimplified model will lead to underfitting the training data, meaning that the correlation among the variables will not be learned by the model properly. Hence, the model will have a high bias in predictions and consequently a low variance. On the other hand, overfitting occurs when the model is too complex. In this case, the training data is memorised by the model and hence, the model prediction will have extremely low bias and extremely high variance. However, an optimum model will provide a relatively effective trade-off between bias and variance of the predictions as shown in Figure 5. The bias-variance trade-off is assessed throughout analysing model generalisation error or the so-called out-of-sample error, which evaluates the model performance on a previously unseen data set (test data set). The theory behind generalisation error can be found in [87].

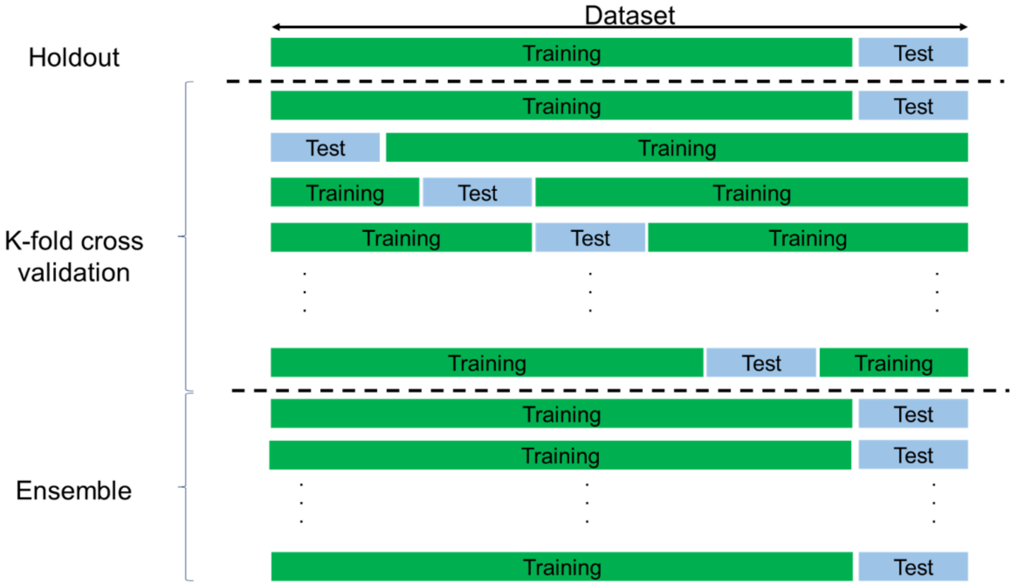


Figure 4: Schematic representation of validation methods: 1) Holdout (Top), 2) K-fold cross-validation (Middle), and 3) Ensemble method (Bottom)

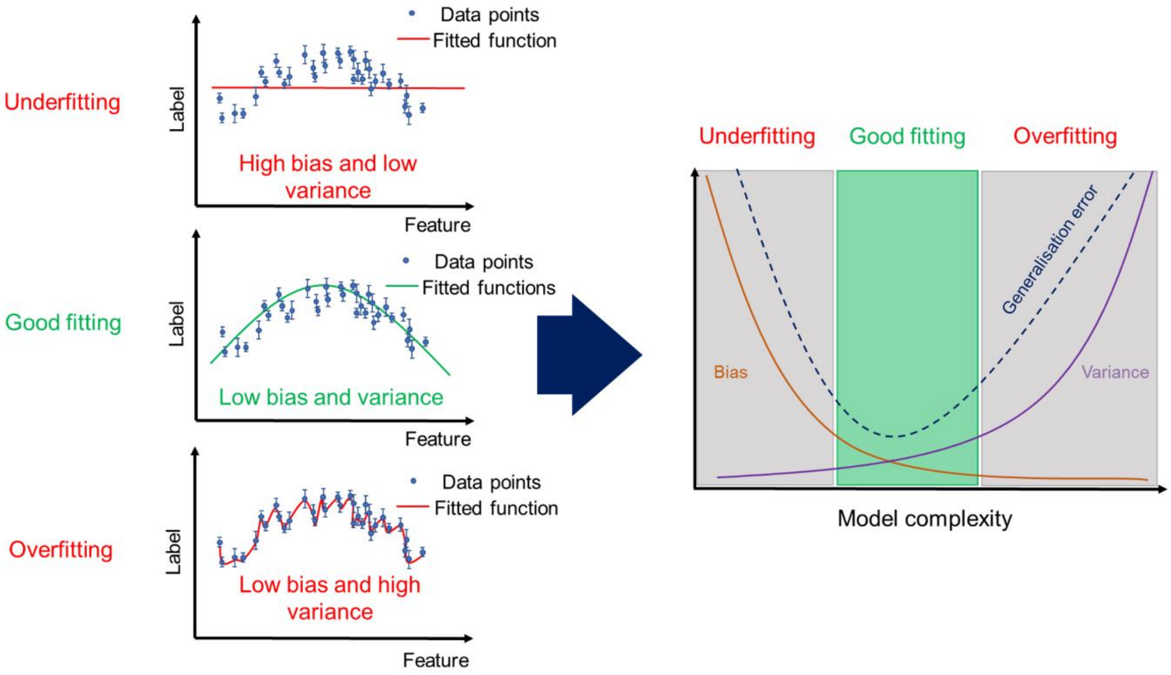


Figure 5: Schematic representation of bias-variance trade-off versus model complexity

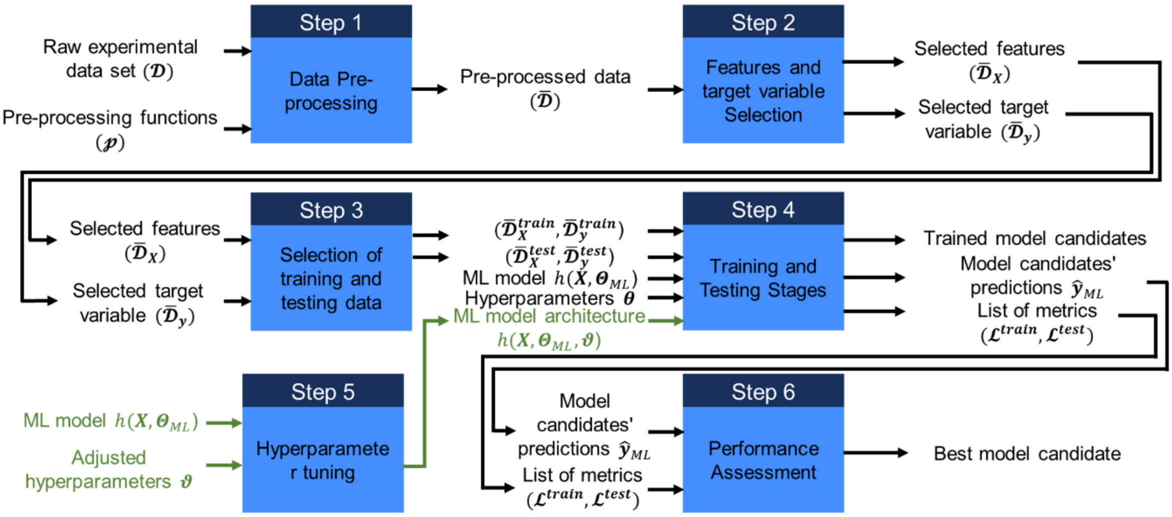

Raw experimental data set ($\mathcal{D}$)
Pre-processing functions ($\mathcal{p}$)
Step 1
Data Pre-processing
Pre-processed data ($\bar{\mathcal{D}}$)
Step 2
Features and target variable Selection
Selected features ($\bar{\mathcal{D}}_X$)
Selected target variable ($\bar{\mathcal{D}}_y$)
Selected features ($\bar{\mathcal{D}}_X$)
Selected target variable ($\bar{\mathcal{D}}_y$)
Step 3
Selection of training and testing data
($\bar{\mathcal{D}}_X^{train}, \bar{\mathcal{D}}_y^{train}$)
($\bar{\mathcal{D}}_X^{test}, \bar{\mathcal{D}}_y^{test}$)
ML model $h(\boldsymbol{X}, \boldsymbol{\Theta}_{ML})$
Hyperparameters $\boldsymbol{\theta}$
ML model architecture $h(\boldsymbol{X}, \boldsymbol{\Theta}_{ML}, \boldsymbol{\vartheta})$
Step 4
Training and Testing Stages
Trained model candidates
Model candidates' predictions $\hat{\boldsymbol{y}}_{ML}$
List of metrics ($\mathcal{L}^{train}, \mathcal{L}^{test}$)
ML model $h(\boldsymbol{X}, \boldsymbol{\Theta}_{ML})$
Adjusted hyperparameters $\boldsymbol{\vartheta}$
Step 5
Hyperparamete r tuning
Model candidates' predictions $\hat{\boldsymbol{y}}_{ML}$
List of metrics ($\mathcal{L}^{train}, \mathcal{L}^{test}$)
Step 6
Performance Assessment
Best model candidate

Figure 6: Overview of the proposed ML-assisted framework

4. Systematic Regression Framework for Heat Transfer Data

In this section, a systematic ML-assisted framework, specifically designed for heat transfer problems, is proposed. The framework is intended as a modelling guideline to model specific output variables (*e.g.* an HTC) based on acquired experimental data and to select the best model candidate among the ones tested in the search space. It should be noted that the framework is agnostic to the ML model and other machine learning techniques may be used in the boundary of the framework that outperform the ones suggested in this study. However, a more rigorous analysis is required by using other ML models to confirm. Figure 6 depicts the overview of the proposed framework in six steps. The tasks and actions required to follow the framework are provided in the rest of this section.

4.1. Step 1: Raw data pre-processing and use of baseline

The first step is to perform pre-processing of the acquired data using pre-processing functions suitable for the collected data set (from the literature or laboratory experiments). Moreover, a baseline correlation (or a model to compare the ML algorithms against) shall be selected as a basis of comparison. This is not a requirement for the procedure; however, it is useful to compare the performance of the developed models to the existing models in the literature. The algorithm corresponding to this step consists of the following sequence of actions:

1. Collect or acquire experimental data ($\boldsymbol{\mathcal{D}}$).
2. Define the baseline correlation and determine the correlation predictions using the collected data set to compare the ML model performance with (optional):

$$\hat{\boldsymbol{y}} = g(\boldsymbol{X}, \boldsymbol{\theta})$$

   Where g is the correlation function, $\boldsymbol{X}$ represents the matrix of M independent variables with N observations, $\boldsymbol{\theta}$ is the vector of correlation parameters, and $\hat{\boldsymbol{y}}$ represents the estimated output variable of that specific correlation.

3. According to the collected data set and the predictions of the correlation function, apply the desired pre-processing functions ($\boldsymbol{\mathcal{p}}$):

   a) Remove the potential outlier data points with data observation ($\boldsymbol{\mathcal{p}_{outlier}}$):

$$\boldsymbol{\mathcal{D}_{flitered}} = \boldsymbol{\mathcal{p}_{outlier}}(\boldsymbol{\mathcal{D}})$$

   b) Define lower and upper bounds for the target variable (y) on the new data set ($\boldsymbol{\mathcal{D}}_{out}$) and remove the data that lies beyond those bounds ($\boldsymbol{\mathcal{p}_{bounds}}$):

$$\bar{\boldsymbol{\mathcal{D}}} = \boldsymbol{\wp}_{\boldsymbol{bounds}}(\boldsymbol{\mathcal{D}}_{out})$$

Note: Other pre-processing functions may also be applied here to prepare a proper data set for developing an ML algorithm. More specific functions shall be defined by data observation.

4.2. Step 2: Features and Target Variable Selection

After pre-processing the data, the features (independent variables) and the target variables shall be selected. This step is crucial as it further provides the input and output variables of the machine learning model. A feature selection technique is used in this study to analyse the relevance of the features for the output variable and assess the model performance based on the most important features. The method implemented in this study is based on the recursive feature elimination technique, however, it must be noted that other techniques such as principal component analysis (PCA) can be employed for feature selection. The algorithm of Step 2 allows selecting the features and the target variable as follows:

1. For all variables in the data set:
   a) If the variable is to be modelled, categorise it as the target variable ($\boldsymbol{\mathcal{D}}_{\boldsymbol{y}}$)
   b) If the variable is an independent variable, categorise it as an input variable ($\boldsymbol{\mathcal{D}}_{\boldsymbol{X}}$)

2. Establish the importance of features by using an independent estimator: (employing the recursive feature elimination (RFE) method):
   a) If the independent estimator is the RF with recursive feature elimination, follow the steps below:

      i. If it is desired to reduce the number of features for the model go to Step 2.b. Else skip the following steps and move to Step 3.
      ii. Specify the objective function for the RF model (*e.g.* MAE, MSE, etc.):

$$J(h_{RF}(\boldsymbol{X}, \boldsymbol{\Theta}_{RF}), \boldsymbol{y})$$

      iii. Train and validate the model according to the objective function and a validation method described in the previous section:

$$\boldsymbol{\Theta}_{RF}^{opt} = arg\ \min_{\boldsymbol{\Theta_{RF}}} J(\boldsymbol{\Theta_{RF}})$$

$$\widehat{\boldsymbol{y}}_{RF} = h_{RF}(\boldsymbol{X}, \boldsymbol{\Theta}_{RF}^{opt})$$

      Note: It is recommended to use k-fold cross-validation technique since it provides a more realistic estimate of the model training error for feature selection.

      iv. Rank the importance of each variable based on a user-defined estimator (an indicator function).

$$\boldsymbol{x}^{sorted} = sort(\boldsymbol{x})$$

      v. Eliminate the least important variable and create a new set of features with the variables left.

$$X_{ij}^{new} = X_{ij}^{sorted} \text{ for } 1 \le i \le M-1 \text{ and } 1 \le j \le N$$

      vi. Replace $\boldsymbol{X}$ with $\boldsymbol{X}^{new}$ and M with M^{new} as the number of variables in the new set of variables.
      vii. Repeat steps b to f until an optimal number of input variables are selected.
   b) If other methods are used such as PCA, etc., the steps of feature selection or elimination must be followed accordingly.

4.3. Step 3: Selection of Training and Testing Data

In this step, the training and testing data should be split by either existing strategies or a user-defined approach according to the specific evaluation criteria of the model. It is important to note that the data split at this step must be carried out as such to prevent any data leakage from either of the data sets into another. Data leakage usually takes place when part of the test data set has been present in the training set. Hence, it must be avoided to prevent any misleading information for model performance assessment (Step 6). The following actions should be followed to prepare the two splits:

1. Given the input and target variables determined in the previous step, split the data into training and test sets. Decide the ratio of the data set that has to be put into training and testing sets. As a common practice in literature, a split of training/testing of 70%/30% or 80%/20% is recommended according to the Pareto principle [88]. The split shall be done randomly.

   Note: One may split the data systematically into training and testing sets and not randomly to assess the model extrapolation capabilities. In this study, a scenario is shown by following a systematic data split to illustrate the model performance according to this splitting strategy.

2. Now that the training and test sets have been selected, represent the splits as the following:

$$\bar{\boldsymbol{\mathcal{D}}}^{\boldsymbol{train}}: (\bar{\boldsymbol{\mathcal{D}}}_{\boldsymbol{X}}^{\boldsymbol{train}}, \bar{\boldsymbol{\mathcal{D}}}_{\boldsymbol{y}}^{\boldsymbol{train}})$$
$$\bar{\boldsymbol{\mathcal{D}}}^{\boldsymbol{test}}: (\bar{\boldsymbol{\mathcal{D}}}_{\boldsymbol{X}}^{\boldsymbol{test}}, \bar{\boldsymbol{\mathcal{D}}}_{\boldsymbol{y}}^{\boldsymbol{test}})$$

4.4. Step 4: Training and Testing Stages

As the training and testing data has been prepared by following the previous steps, now the model of interest with its properties has to be built and to be fitted to the corresponding data. To this aim, the model of interest has to be selected first and then the model hyperparameters have to be selected accordingly. Afterwards, the training and testing stages can be carried out as described below:

1. Choose the ML model of interest, $h(\boldsymbol{X}, \boldsymbol{\Theta}_{ML})$.

   Where h is the function representing the ML model and $\boldsymbol{\Theta}_{ML}$ represents the model parameters upon its development.

2. Specify the set of model hyperparameters ($\boldsymbol{\vartheta}$):

   a) If the selected model is a neural network, select the following hyperparameters and leave the rest as the default values in the implementation:

      i. The number of processing units in the input layer is equal to the number of inputs:

$$Input\ layer\ size = \sum_{i}^{input\ variables} n_i$$

      ii. The number of processing units in the output layer:

$$Output\ layer\ size = \sum_{k}^{target\ variables} n_k$$

      iii. The number of hidden layers
      iv. The number of processing units at each hidden layer
      v. The activation function for each layer's processing units
      vi. The number of epochs for model training
      vii. The batch size for training epochs
      viii. The objective/loss function (*e.g.* L1 or L2 norm, MAE, MSE, etc.)

$$J_{DNN}(\hat{\boldsymbol{y}}_{\boldsymbol{DNN}}, \boldsymbol{y}), \text{ where } \hat{\boldsymbol{y}}_{\boldsymbol{DNN}} = h_{DNN}(\boldsymbol{X}, \boldsymbol{\Theta}_{\boldsymbol{DNN}}, \boldsymbol{\vartheta}_{\boldsymbol{DNN}})$$

ix. Optimisation algorithm
x. The initialisation of the model weights and biases

b) If the selected model is Random Forest, select the following hyperparameters and leave the rest as the default values in the implementation:
   i. The number of estimators (trees)
   ii. The split criterion
   iii. Use of bootstrapping
   iv. Minimum sample size to split
c) Else, if other ML models have been selected (*e.g.* SVM, GPR, etc.), the hyperparameters must be selected according to that model configuration.

Note: According to the choice of the user, different hyperparameters can be set up for the models that are not listed above.

3. Select the validation method for the models based on the following criteria and the hyperparameters selected above:
   a) holdout
   b) k-fold cross-validation
   c) Ensemble

4. Train the model on the training data set ($\bar{\boldsymbol{\mathcal{D}}}^{\boldsymbol{train}}$):
   a) If the holdout method has been chosen as the validation method, save the optimised model parameters and training loss value and go to step 5 of this algorithm.
   b) Else, If k-fold cross-validation has been selected, split the training data into k folds and then train the model for each, take the average training loss of all folds as the representative loss value of the model
   c) Else, if an ensemble method has been chosen, save all the model parameters and training loss values

5. Perform prediction on the test data set for the trained models at the previous step of this algorithm and save the predictions and model data to be compared later on.

4.5. Step 5: Hyperparameter Tuning based on Grid Search

In this step, the hyperparameters of the ML models are to be tuned for assessing the performance of the models against these hyperparameters. Since the search space to tune the hyperparameters is quite large, a grid search approach is recommended to tune the hyperparameters based upon using the following algorithm:

1. Select the desired hyperparameters ($\boldsymbol{\vartheta}$) to be tuned based on the ML model (h) selected in Step 4.

2. Define an array of various choices for each selected variable (*i.e.* a grid of different values)

3. Create a list of hyperparameters for each model to be trained for and according to their combinations.

4. Go to Step 4 and train each model with adjusted hyperparameters.

4.6. Step 6: Performance Assessment

Now that all the models have been trained and the predictions have been carried out, models can be compared based on their prediction performance and learning capabilities. It must be noted that the performance assessment is only valid within the studied search space of the hyperparameters and there may be models that outperform the ones tested in this study. Hence, the selected model at this step by no means is the global optimum model. The following algorithm represents the step for performance assessment:

Assess the models' performance according to the following criteria:

a) If the CV scores are significantly different, choose the model with the lowest CV score
b) Else, if the evaluation metric of the test set is significantly different from one another, choose the model with the lowest metric value
c) Else, select the model with the minimum difference between training and test metrics values
d) Optional: comparison against existing models. If existing first-principle or semi-empirical models exit, then it is useful to assess the performance of the predictions of the ML-regression against these models. The mean absolute error (MAE), shown in Equation (3) is selected as a performance indicator:

$$MAE = \frac{\sum_{i=1}^{N} |y_i - f(x_i, \boldsymbol{\Theta})|}{N} \tag{3}$$

This value indicates the main discrepancy between the predictions of the ML models and what is considered ground truth (in this case, the experimental HTC). The selected ML models are assessed using this indicator during the training and testing stages. Additionally, to provide potential prediction improvements over traditional methods, the MAE values of the chosen ML models are compared to those estimated by the semi-empirical correlation developed by Cavallini *et al.* [16], which has proven to output robust estimations within varied experimental ranges (with ±20% of error), and more accurate than previous attempts for such estimation. Please refer to the corresponding reference to review more details.

Furthermore, the percentage of data points outside a fixed threshold is estimated using Equation (4), where the critical value τ_c is selected as 20% of experimental values. This value is selected to maintain consistency with the estimation error of the baseline correlation. The integrated use of this indicator and the MAE for each regression model is sufficient to understand and assess regression performance in a discretised (by counting those data points outside the error threshold) and quantitative (by measuring how far each estimation is from its corresponding experimental value) manner.

e)

$$\text{\% points outside } = \frac{|number\ of\ points| \geq \tau_c}{N} \tag{4}$$

5. Case Study: Estimation of the Condensation Heat Transfer Coefficient in Microfin tubes

5.1. Background

In the last forty years, it has become a common practice to use extended surfaces to enhance condensation heat transfer inside horizontal tubes. Thus, microfin tubes, as a design technology that improves heat transfer, are widely used in heat exchangers for numerous HVAC and refrigeration systems. In fact, since they were invented by Fujie *et al.* [89], microfin tubes have received significant attention for their ability to provide large heat transfer enhancements (80-180%) with a relatively small increase in pressure drop (20-80%) as compared to an equivalent smooth tube under the same operating conditions. They provide enhanced heat transfer by (i) increasing the effective exchange area; (ii) inducing increased turbulence in the liquid film (iii) exploiting the surface tension effect to facilitate the condensate drainage [16]. Common microfin tubes are characterised by fin tip diameter (d_f), number of fins (n_f), helix angle (β), apex angle (α) and fin height (h_f). These features are depicted in Figure 7, which shows a basic design of a microfin tube. In general, microfin tubes have an inside diameter from 3 to 15 mm, a single set of 40-70 fins with helix angle (β) from 0° to 30°, fin height (h) from 0.1 to 0.25 mm, triangular or trapezoidal fin shapes with an apex angle (γ) from 25 to 90°. The heat transfer coefficient (HTC) for condensation in microfin tubes relies on complex interactions among the systems' characteristics (*e.g.* geometry, operating conditions, flow pattern, etc.) and heat transfer mechanisms (*e.g.* forced convection, temperature-driven, etc.).

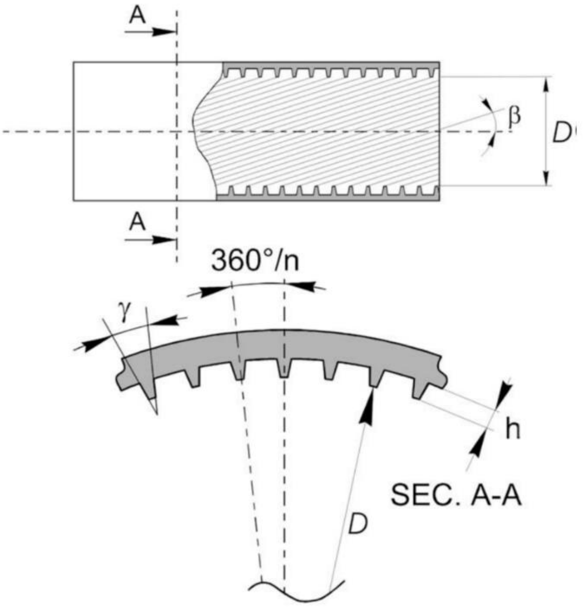


Figure 7: Basic design of a microfin tube reprinted with permission from [16]. © 2009 Elsevier

5.2. Dataset

The dataset used for the Case Study consists of 4,333 experimental data points of quasi-local and local heat transfer coefficients collected from several independent laboratories inside 53 different tubes as a function of 21 input features. These observations correspond to experimental runs carried out by multiple authors over the last three decades (see [16] for a detailed list of authors). The experimental (calculated) HTC for each observation is considered the target or output variable. A summary of the experimental inputs can be found in Table 3: Experimental input features. The data set is put through the proposed framework where outliers are filtered, training and testing data sets are selected and the ML models are set up and deployed correspondingly.

5.3. Approach

Cavallini *et al*. [16] proposed a correlation, which is widely used and considered one of the most accurate methods available to estimate the condensation heat transfer coefficient inside microfin tubes. This was developed based on traditional regression methods using the dataset described above. The prediction accuracy of this semi-empirical correlation is taken here as a reference to evaluate the performance of the ML approach.

Given that the development of semi-empirical correlations usually suffers from the limitations mentioned above, this study attempts to showcase the proposed regression workflow to compare, select from, and assess the performance of two Machine Learning models, to regress the value of the HTC using as inputs the experimental data. The models are compared against each other and the benchmark semi-empirical correlation proposed by Cavallini *et al.* [16].

Two important issues are addressed in this Case Study (for all ML models): i) the selection of data to be used during training, and ii) the selection of input features and their relevance concerning the output or target variable. The first issue is related to the amount and quality of data to use in the training stage, as the accuracy and robustness of any regression model greatly depend upon whether the training data include a wide range of system conditions. The second issue is related to the number of input features to the regression model, and their direct effect on the output variable, as it is often the case where a larger number of input features only add disturbances to the model outputs, decreasing its prediction performance. On the other hand, a smaller number of input features could not suffice for explaining the target variables.

Table 3: Experimental input features

Variable	Symbol	Units	Range	Variable	Symbol	Units	Range
Vapour quality	x	–	[0.02 – 1.0]	Heat of vaporisation	h_{LV}	J kg^{-1}	[8,50E04 – 3.0E05]
Refrigerant mass flux	G	kg m^{-2}s^{-1}	[79 – 919]	Reduced pressure	P_{red}	–	[2.80E-02 – 6.70E-01]
Liquid density	ρ_L	kg m^{-3}	[884 – 1481]	Surface temperature	T_s	K	[-25.0 – 74.0]
Liquid thermal cond.	λ_L	W m^{-1}K^{-1}	[0.05 – 0.14]	Surface to wall temp. difference	$T_s - T_w$	K	[0.23 – 18.0]
Liquid heat capacity	cp_L	kg m^{-3}	[881 – 2263]	Fin tip diameter	d_f	m	[6.0E-03 – 1.60E-02]
Liquid viscosity	μ_L	Pa s	[8.30E-05 – 4.30E-04]	Fin height	h_f	m	[1.20E-04 – 6.30E-04]
Liquid surface tension	σ_L	N m^{-1}	[1.50E-03 – 1.80E-03]	Helix angle	β	°	[0.0 – 30.0]
Vapour density	ρ_V	kg m^{-3}	[7.0 – 167]	Apex angle	γ	°	[0.0 – 90.0]
Vapour thermal cond.	λ_V	W m^{-1}K^{-1}	[8.60E-3 – 2.60E-2]	Number of fins per tube	n_f	–	[21.0 – 82.0]
Vapour heat capacity	cp_V	kg m^{-3}	[616 – 2588]	Tube length	L_f	m	[0.15 – 6.40]
Vapour viscosity	μ_V	Pa s	[1.0E-05 – 1.80E-05]				

To consider both issues systematically, this work studies two separate Scenarios and three Sub-scenarios each as summarised in Table 4. In Scenario 1, the data splitting for the training and testing stages is done randomly, where 70% of the data set is used for training, while the rest (30%) is used as a testing set. A consequence of this splitting is that the testing set might consist of some observations that are within the ranges considered in the training set, meaning that the test dataset might contain datapoints that are within the range of training data. On the contrary, Scenario 2 considers the training of the models using a specific set of data, which is selected manually to take only into account certain types of microfin tubes. This way, the testing set is completely out of the range of training data to assess the extrapolation capabilities of the developed model. The data have been pre-processed and categorised according to the type of microfin tube, resulting in 53 different tubes. The fin tip diameter is selected as the key variable in this scenario to split the data into training and testing sets. Note that even though each tube is different, they have been put through similar conditions across different experiments, in other words, there are still similar operating ranges in both training and testing data.

Furthermore, to address the selection of input features, the two Scenarios include three Sub-Scenarios, where different sets of input features are used to train and deploy the models. In the first Sub-Scenario (a) all input features in Table 4 are used, in the second Sub-Scenario (b) a reduced set is used, which is selected using feature importance estimations, and in the third Sub-Scenario (c) traditional dimensionless numbers, detailed in Table 5, are used.

Note that the baseline correlation was not retrained during this study, that is, the accuracy of estimation between the baseline correlation and the ML models of neither scenario presented here can be entirely compared, as only the ML-based methods were trained with different sets of data. Nevertheless, Scenario 1 is the closest to the training of the baseline correlation, as both original trainings use the

Table 4: Summary of the scenarios considered in the Case Study

Scenario number	Description
1a	Random data split, training with all input features
1b	Random data split, training with a set of reduced input features
1c	Random data split, training with dimensionless numbers
2a	Systematic data split, training with all input features
2b	Systematic data split, training with a set of reduced input features
2c	Systematic data split, training with dimensionless numbers

Table 5: Set of traditional dimensionless numbers

Variable	Units	Description	Definition
$\mathrm{J_V}$	$[-]$	Vapour velocity	$xG/\left[gd_f\rho_V(\rho_L-\rho_V)\right]^{0.5}$
$\Delta\mathrm{T_{dimless}}$	$[-]$	Temp. difference	T_s-T_w/T_s
Bo	$[-]$	Bond number	$g\rho_L h_f \pi d_f/8\sigma_L n_f$
$\mathrm{R_x}$	$[-]$	Geometrical enhancement factor	$\frac{1}{cos\,(\beta)}\left\{1+\frac{2h_f n_f[1-sin\,(\gamma/2)]}{\pi d_f cos\,(\gamma/2)}\right\}$
$\mathrm{Pr_L}$	$[-]$	Liquid Prandtl number	$\mu_L cp_L/\lambda_L$
$\mathrm{Re_L}$	$[-]$	Liquid Reynolds number	Gd_f/μ_L

entire set of data used in this study. In Scenario 2, it is expected to have large differences between the baseline correlation and the ML-based methods, as the training set in this scenario is much more reduced in the latter than that of the former.

6. Application of the Regression Framework to the Case Study

In this section, the steps of the framework reported in Section 4 are followed for the prediction of the condensation HTC in microfin tubes (case study reported in Section 5).

6.1. Data Pre-processing

As proposed in the first step of the framework, the experimental data are inspected to find and discard data points that are either unexpected or inconsistent. First, a screening through the input features allow the detection of outliers coming from the data samples originated in the study of Vollrath *et. al* [90] (a subset of the selected data set), where several experimental runs at lower mass fluxes were performed, and the helix angle β was set to zero. As a result, when estimating the HTC using the correlation developed by Cavallini *et. al* [16], extreme values are calculated. Additionally, as a further filtering procedure, those HTC values lower than 24,000 W m^{-2} K^{-1} were used. This is done to maintain consistency with the results reported in the work used as a baseline [16]. This filtering procedure reduced the total number of experimental points from 4,333 to 4,115.

6.2. Features and Target Variables Selection

Once the data have been properly pre-processed, the feature selection should take place. The easiest selection in this case is the one for the output or target feature, which is the value of the HTC. For the input features, this selection depends on the different Scenarios described in Table 4. In Scenarios 1a/2a, and 1c/2c the selection translates into the use of the features shown in Table 3 and Table 5 respectively.

In Scenarios 1b/2b, the selection is much more elaborated. In this work, relevant input features are selected using a recursive elimination approach, where an optimal set of input features is selected based on their relative importance to the output variable. An external data-driven estimator, which is able to estimate these importance values, is deployed. This procedure is repeated k times with different portions of the training data and an average score value is estimated to determine the most relevant features. This procedure is known as Recursive Feature Elimination with Cross-validation (RFECV) and is available on the Python library Scikit-learn.

6.3. Selection of Training and Testing Data

Training and testing data are selected differently depending on whether Scenario 1 or Scenario 2 is carried out. For the first one, a random split of 70/30% is used for training and testing sets, respectively. This creates a training data set consisting of 2,880 data points and a testing set of 1,235 data points. Given that this selection was done randomly, there is no conscious bias towards specific experimental ranges. This practice usually yields more general training sets, provided that the original data allow for it.

For Scenario 2, each data observation was assigned to a specific microfin identification label, which encodes the tube's geometrical main characteristics. There is a total of 53 types of microfin tubes. The training set is then selected to: (i) maintain consistency with the number of data points in Scenario 1 (*i.e.* around 70% of data) and (ii) consider tubes with similar characteristics and leave different ones for the testing set. The main selection criterion for Scenario 2 is the fin tip diameter d_f. The values for this selection are shown in Table 6. This practice allows for a more detailed study of the extrapolation of the ML models when facing an entirely unseen dataset. It is expected that the results from Scenarios 1 and 2 differ significantly, given their fundamental designs. Furthermore, differences in accuracy of estimation between the baseline correlation and the ML-based models are expected to be larger in Scenario 2. This is due to what was mentioned above about the use of the baseline correlation without retraining.

6.4. Training and Testing Stages

The ML models in this work are trained using the two training sets defined for Scenario 1 and Scenario 2. Additionally, cross-validation is deployed during training in all cases to obtain a more general assessment of the performance of the regression. Given that the k-fold cross-validation outputs multiple scores for each fold in the training data, a mean score value can be obtained, providing a wider scope into such performance.

To deploy the training stage, the ML model needs to be built up first. To do this, both the RF and the DNN algorithms are initialised using fixed parameters that depend on the type of model. When building the RF algorithm, default values from the Sci-kit learn library are used. These values are summarised in Table 7, for RF and DNN respectively. Note that the DNN building, training, and deployment are done using the TensorFlow [91] library in Python. Furthermore, to explore the effect of more complex DNNs on regression performance, three different DNN architectures are used. The details of these distinct architectures are provided in Table 8, where DNN 1 is the most complex neural network model used in this study, followed by DNN 2 and DNN 3, which is the simplest.

Table 6: Selection criteria for Scenario 2

Parameter	Value	Units	Selected tubes	Number of points
Training range for d_f	(7.75 – 8.51)	[mm]	13	2729 (66%)
Testing range for d_f	(5.95 – 7.69)/(8.52 – 15.8)	[mm]	40	1386 (34%)

Table 7: Default parameters for RF and DNN models

Parameter: RF	Initial value	Parameter: DNN	Initial value
Number of estimators (trees)	100	Number of epochs	50
Split criterion	“squared error”	Number of batches	5
Minimum sample size to split	2	Loss function	“mean absolute error”
Use of bootstrapping	True	Optimisation algorithm	“Adam”

Table 8: DNN architectures used in Cases 1 and 2

DNN	Number of hidden layers	Number of neurons per hidden layer
1	4	(256, 128, 64, 32)
2	3	(128, 64, 32)
3	2	(64, 32)

6.5. Hyperparameter Tuning

When evaluating the accuracy score of an ML model, this value is directly related to the parameters used in such a model. Depending on the input and output data (*i.e.* number of observations, number of input features, nonlinearities between inputs and outputs, etc.), more complexity should be added or discarded. To address this, a grid search is carried out to assess and compare the regression performance of each ML model, when using different sets of parameters. Grid search performs cross-validation using the selected model and estimates different accuracy scores for all combinations of the selected set of parameters. Model selection is carried out based on the mean cross-validation score of each model tested, where the model with the best accuracy score (lowest in the case of MAE) is further used to deploy estimations.

In this work, the parameters and values used during the grid search are summarised in Table 9. For consistency, the same parameters shown in Table 7, used in the initialisation of each model, are also used in the grid search stage.

6.6. Performance Assessment

Model selection takes place considering two levels of comparison, that is, regression accuracy is assessed within single regressors (default versus grid search parameters), and among regressors (RF versus DNNs). Each comparison is based on both the mean cross-validation MAE at each instance of the models, and those MAE from the single fitting of the models to the selected training data.

As a baseline, the correlation developed by Cavallini *et. al* [16] is used to calculate the HTC with the data of the selected training and testing sets, and their corresponding MAE values are also estimated. This allows for performing comparisons across the models and selecting those that exhibit regression improvements over the baseline correlation. In addition, the percentage of data points outside a 20% of experimental error (see Equation (4)) is calculated for all regression methods (including the baseline correlation) to add an extra layer of interpretation of the results.

7. Results

7.1. Feature Importance Analysis

As explained above, this step can assist in identifying the relevant features to the output variable and help simplify the model by only considering the most relevant features. Figure 8(b) and Figure 8(d) show the feature importance for Scenario 1a and Scenario 2a, respectively.

Table 9: Parameter grid for the grid search

Parameter: RF	Initial value	Parameter: DNN	Initial value
Number of estimators (trees)	[100, 300, 500]	Number of epochs	[200, 400, 600]
Split criterion	[“squared error”, “absolute error”]	Number of batches	[5, 10, 20]
Minimum sample size to split	[2, 3, 5]	Loss function	[“mean absolute error”, “mean squared error”]
Use of bootstrapping	[True, False]	Optimisation algorithm	[“Adam”, “SGD”]

In both cases, not surprisingly, the vapour fraction x represents the most relevant input feature, followed (in different order) by the mass flux G, the liquid-phase thermal conductivity λ_L, the temperature difference ΔT, and the heat of vaporisation h_{LV}. The remaining input features vary in their relative importance and their corresponding order. The reason for this is the set of data used for training: in Scenario 1, the training data contain information from most (if not all) microfin tubes, increasing the importance of different sets of geometrical characteristics such as the angles β and γ, whereas in Scenario 2, the selected microfin tubes are similar in this manner, and as a consequence, the refrigerant characteristics such as viscosity (μ_L), density (ρ_L) and surface tension (σ_L) become more relevant. In addition, the vapour-phase physical properties exhibited a lower relevance in both cases. This already suggests that the data split in Scenario 2 could mask the relative importance of certain features, in favour of others, leading to an unsuccessful data splitting.

When reducing the number of input features from Scenarios 1a and 2a to Scenarios 1b and 2b, the relevant features of the former are shared with the latter and the liquid-phase heat capacity cp_L) along with the apex angle (γ) are included. This extra set of features accounts for both, a geometrical and fluid-related parameter, which balances out the contribution of these two types of input features into the overall feature relevance. Note that for each case, the reduced set of features was deployed iteratively, until a unique subset of input features was obtained. The number of iterations for each case differs, but the final results are the ones shown in this work.

Finally, when comparing the feature relevance in Scenarios 1c and 2c, it is clear that most of the features and their relative hierarchy are the same in both cases, where the vapour fraction, fluid’s mass flux and temperature difference are still the most relevant. The only difference between these two scenarios is the presence of the liquid-phase Reynolds number in Scenario 2c. As the fluid’s physical properties are more relevant in Scenario 2, the relevance of the liquid-phase surface tension is increased.

This feature importance analysis is crucial for the construction and deployment of an ML-based regression model, as the correct selection of input features not only decreases computational time but also delivers a qualitative understanding of the input data and their correlation with the target variable. These insights can then be validated via expert knowledge and/or further experiments. Overall, the comparison between Scenario 1 and 2 resulted in consistent outputs, where the slight difference between the two scenarios can be explained by the fact that two different strategies have been selected for preparing training and testing datasets (*i.e.* random versus systematic).

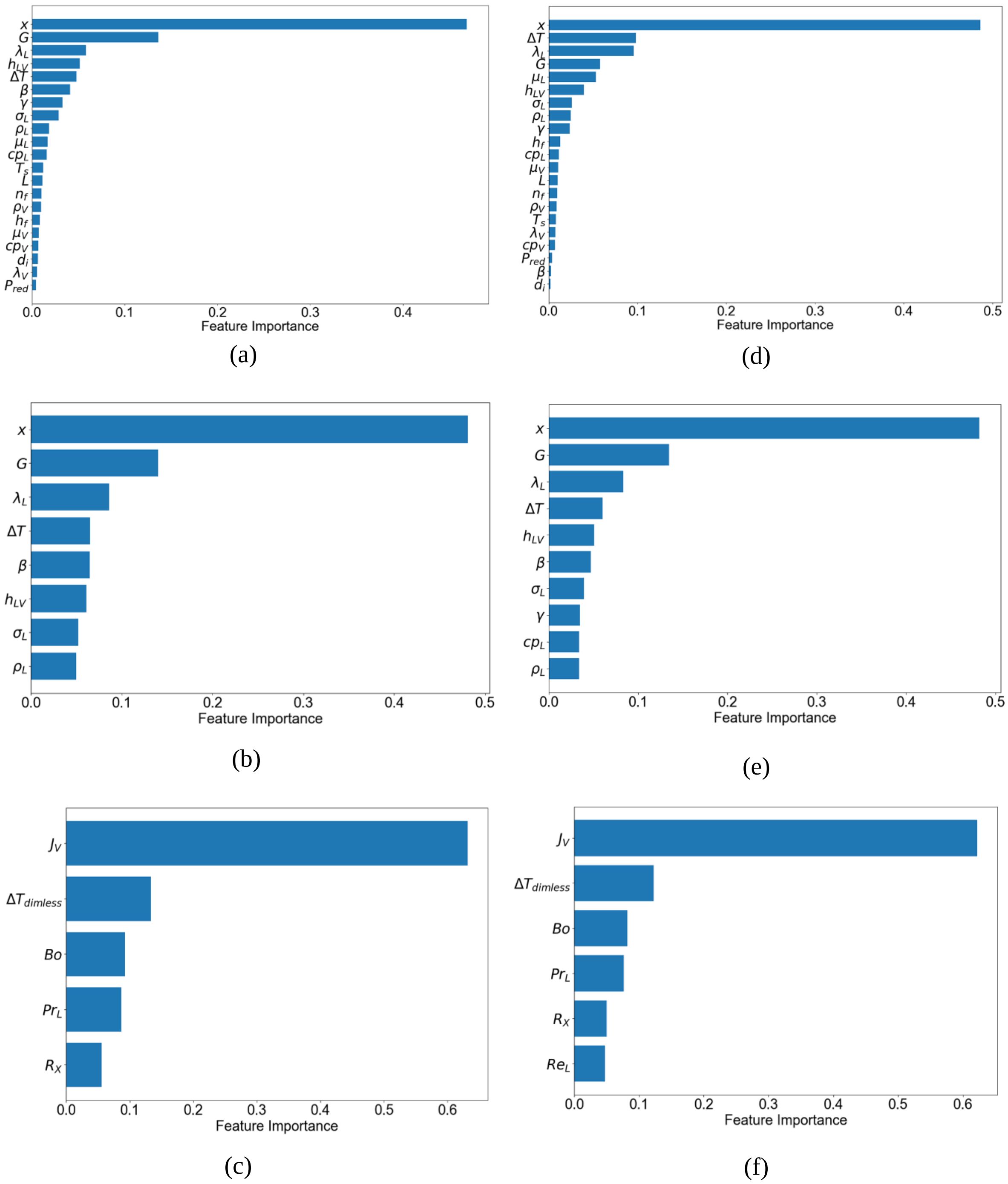


Figure 8: Feature importance for (a) Scenario 1a, (b) Scenario 1b, (c) Scenario 1c, (d) Scenario 2a, (e) Scenario 2b, and (f) Scenario 2c

7.2. Regression Performance

The performance indicators for the regressions carried out in Scenarios 1 and 2 are shown in Table 10 for Scenario 1, and Table 11 for Scenario 2. Both tables show the MAE and the out-of-range values for training and testing sets, along with the cross-validation scores for each algorithm. These values are reported for the use of default and grid-search-based hyperparameters. Note that to compare these results with the baseline correlation, the performance indicators were also calculated using such correlation and the selected training and testing sets. Note that the ML-based models were trained separately for Scenarios 1 and 2 (as data sets are different), whereas the baseline correlation is used without retraining, that is, the parameters of the original correlation have not been calculated.

In Scenario 1, all MAEs and out-of-range percentages are lower than those of the baseline correlation. This suggests that the use of any of these ML-based regression models is capable of estimating the condensation HTC with more accuracy than the baseline correlation. In Scenario 1, the lowest cross-validation score belongs to DNN 1 with the use of grid search. Furthermore, this model has the lowest MAE in the testing set. In terms of training MAE, the RF model provides the lowest training in all cases; however,, there is a larger gap between these values and testing MAEs (compared to the rest of the models). This means that this model presents a low bias and a high variance, which is an indication that there is memorisation rather than learning (or overfitting). With this in mind, it can be said for now that the use of DNN 1 plus grid search provides for a more suitable estimation model for Scenarios 1a, 1b, and 1c, however, as more results are shown below, the importance of avoiding overfitting will become more relevant.. The differences in estimations between the baseline correlation and DNN 3 with grid search for both sets of data are shown in Figure 9.

The effect of feature selection can also be assessed with these results. In all three sub-scenarios, the estimation accuracy tends to decrease when the number of input features decreases. This behaviour is shared across all ML-based models, and independent of the set of hyperparameters. However, this decrease seem to be negligible among models. For example, when using the MAE during testing of DNN 1, the difference between Scenario 1a (use of all input features) and Scenario 1b (use of relevant features) is of 105 $[Wm^2K^{-1}]$, which is equivalent to around 1.7% of the mean HTC in the testing set (6132 $[Wm^2K^{-1}]$,). When comparing Scenarios 1a and 1c (use of dimensionless numbers), the difference in accuracy is of 155 $[Wm^2K^{-1}]$, or 2.5% of the mean HTC in the testing set. Two observations can be made from this: i) the differences in MAE among subscenarios is negligible, as both modelling approaches provide similar accuracy (because of the use of relevant features Scenario 1b and physical knowledge in Scenario 1.c), and ii) the use of feature importance is able to provide similar degree of accuracy as both the use of individual experimental features and of dimensionless numbers (which contain a physical interpretation of the phenomenon under study).

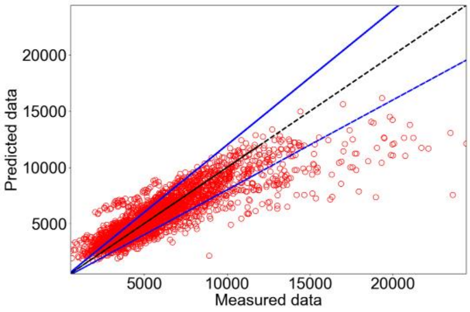
In Scenario 2, the use of the baseline correlation provides more accurate results than those of the different ML-based models. The main reason for this is the selection of training data. According to the authors who developed the baseline correlation [16], this was trained using a variety of experimental points, which cover more types of microfin tubes compared to those selected in Scenario 2. This leads to a rather unfair comparison between the baseline correlation (which has been trained with a richer data set) and the ML-based models (which have only been trained with subsets of data).The reader should be aware of this and (provided the information is available), to guarantee a more fair comparison, the retraining of the baseline correlation should be considered.

Given the above, when comparing the MAE and out-of-range percentages results between Scenario 1 and Scenario 2, it is seen that the estimations of the former are more accurate than those of the latter. This comparison clearly shows that the appropriate selection of training data, which covers a large variety of microfin tubes, is able to deploy accurate models that overperform the semi-empirical correlation used as a baseline Note that this is also true for the training of the baseline correlation. The regression performance results for Scenario 2 are summarised in Table 11.

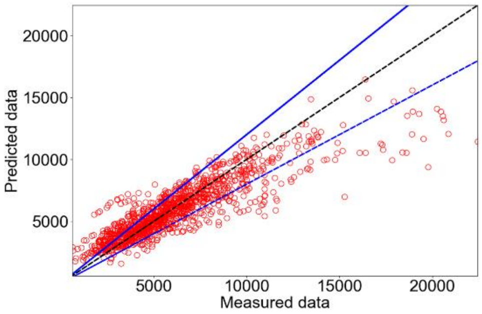
A comparison among the ML-based models still provides significant insights. Note that since Scenario 2 is tested with a completely different set of microfin tubes (based on diameter), and the cross-validation is done using the training set, the MAE values for the former set have more priority than the MAE of the latter, given that the testing set is entirely unknown (from a fin tip diameter perspective). Because of this, the most accurate model among the ML algorithm is the one presenting the lowest MAE in the testing set, which in this case is the RF method.

Table 10: Regression results for all models in Scenario 1

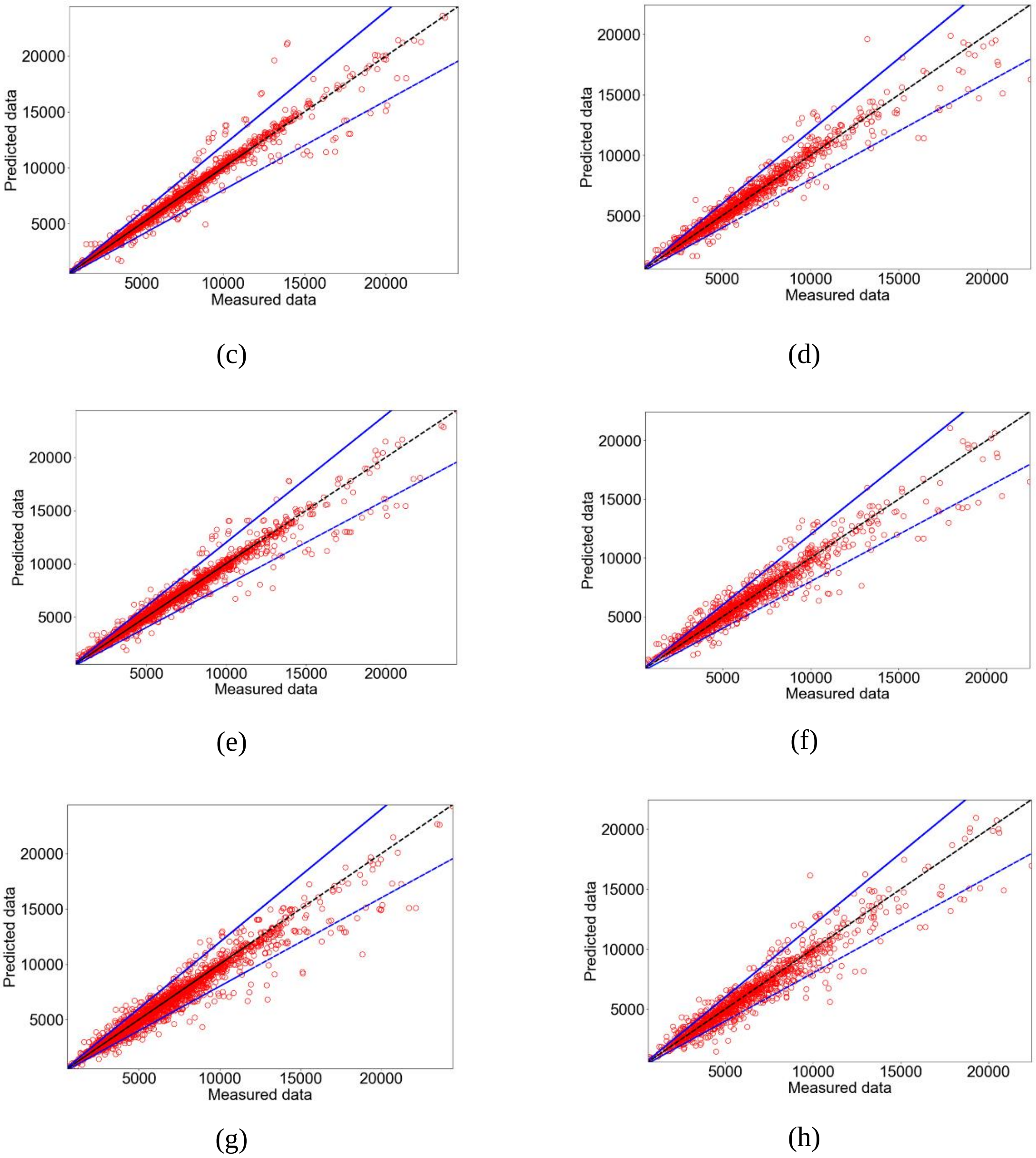
	Method	MAE training	MAE testing	CV score	% out-of-range	MAE training – GS	MAE testing – GS	CV score – GS	% out-of-range – GS
Scenario 1a	Cavallini *et. al* [16]	1039	1061	--	(33, 34)	1039	1061	--	(33, 34)
	Random Forest	198	516	572	(0.7, 9)	195	511	569	(0.8, 9)
	DNN 1	598	651	664	(12, 14)	236	426	480	(2, 7)
	DNN 2	679	718	769	(15, 18)	307	465	532	(3, 7)
	DNN 3	845	870	897	(22, 25)	429	535	600	(6, 9)
Scenario 1b	Cavallini *et. al* [16]	1039	1061	--	(33, 34)	1039	1061	--	(33, 34)
	Random Forest	222	562	639	(1.2, 11)	220	553	637	(1, 11)
	DNN 1	705	731	776	(17, 18)	308	531	578	(4, 11)
	DNN 2	763	788	864	(18, 19)	386	544	637	(7, 12)
	DNN 3	895	914	976	(23, 26)	550	647	680	(11, 14)
Scenario 1c	Cavallini *et. al* [16]	1039	1061	--	(33, 34)	1039	1061	--	(33, 34)
	Random Forest	231	583	657	(0.9, 12)	231	581	649	(0.9, 12)
	DNN 1	855	863	874	(23, 25)	398	581	676	(6, 13)
	DNN 2	936	932	956	(27, 28)	638	719	765	(14, 18)
	DNN 3	970	988	1037	(28, 28)	657	685	812	(16, 18)



(a)



(b)



(c)

(d)

(e)

(f)

(g)

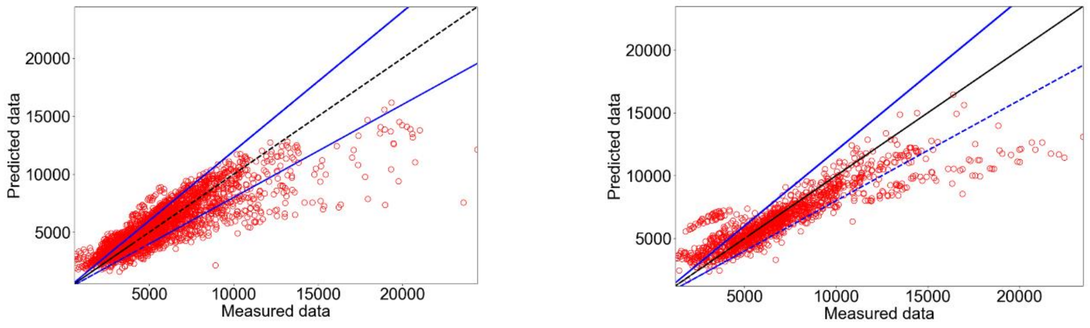
(h)

Figure 9: Parity plots for the HTC in $[Wm^2 \ K^{-1}]$ for (a) the baseline correlation in the training set, (b) the baseline correlation in the testing set, (c) DNN 1 with GS in the training set for Scenario 1a, (d) DNN 1 with GS in the testing set for Scenario 1a, (e) DNN 1 with GS in training set for Scenario 1b, (f) DNN 1 with GS in the testing set for Scenario 1b, (g) DNN 1 with GS in training set for Scenario 1c, (h) DNN 1 with GS in the testing set for Scenario 1c. Sidelines indicate $\pm$20% of error

The use of DNNs (with or without GS) is less appropriate, as they exhibit the highest MAE and out-of-range percentage values. Overall, GS increases the accuracy of all ML-based models, except in Scenario 2a when using the RF algorithm. The selection of input features is more important here than in Scenario 1, where the use of dimensionless numbers (namely Scenario 2c) presents higher accuracy. This suggests that including expert knowledge (and studying the relevance of each input feature) in the construction of ML-based models results in improved accuracy, especially when the training set is reduced in variability. A comparison of the estimations between the baseline correlation and the RF model is shown in Figure 10. As with Scenario 1, the differences in MAE among the use of sets of input features (Scenarios 2a, 2b and 2c) are relatively small, where a similar degree of accuracy is exhibited between the use of all input features (Scenario 2a) and the use of relevant features (Scenario 2b).

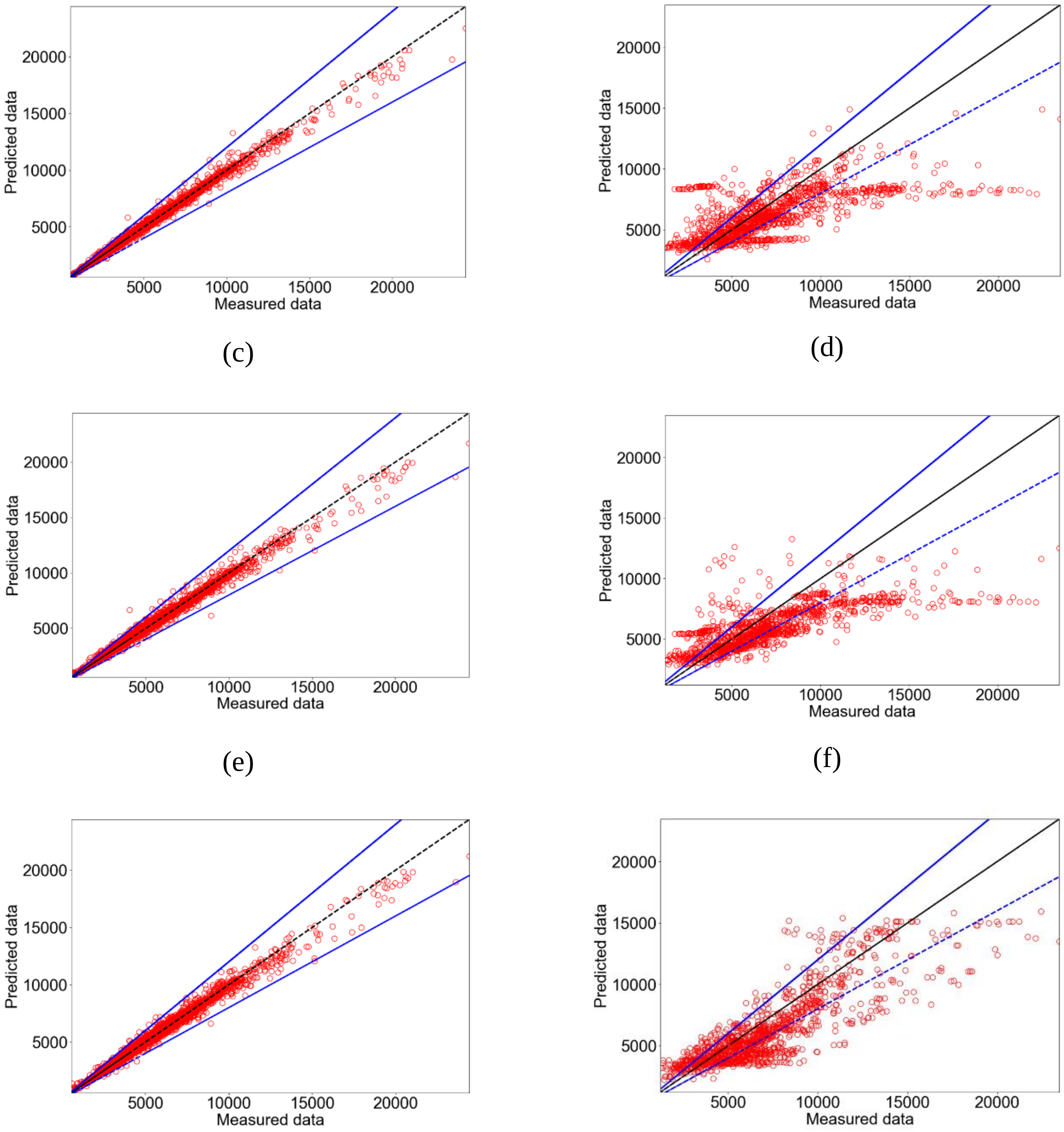
Table 11: Regression results for all models in Scenario 2

	Method	MAE training	MAE testing	CV score	% out-of-range	MAE training – GS	MAE testing – GS	CV score – GS	% out-of-range – GS
Scenario 2a	Cavallini *et. al* [16]	1026	1111	--	(38, 24)	1026	1111	--	(38, 24)
	Random Forest	167	1817	1405	(0.7, 46)	207	1828	1358	(1.3, 46)
	DNN 1	461	14337	2853	(9, 87)	247	23764	1568	(2, 89)
	DNN 2	568	14902	2429	(12, 93)	304	31176	2347	(3, 91)
	DNN 3	716	10037	1873	(17, 88)	450	12426	1560	(7, 68)
Scenario 2b	Cavallini *et. al* [16]	1026	1111	--	(38, 24)	1026	1111	--	(38, 24)
	Random Forest	170	1898	1318	(0.7, 51)	214	1827	1286	(1,4, 50)
	DNN 1	533	2666	3071	(10, 63)	251	3210	2186	(1.4, 71)
	DNN 2	690	2359	2189	(18, 53)	358	11047	2472	(5, 83)
	DNN 3	853	3206	1557	(26, 59)	533	4501	1629	(12, 74)
Scenario 2c	Cavallini *et. al* [16]	1026	1111	--	(38, 24)	1026	1111	--	(38, 24)
	Random Forest	181	1503	1487	(0.7, 45)	232	1465	1383	(1.2, 45)
	DNN 1	645	2085	1439	(16, 54)	343	5250	1380	(5, 72)
	DNN 2	681	1693	1133	(17, 50)	502	2844	1362	(10, 57)
	DNN 3	832	1798	1175	(23, 56)	637	1819	1076	(15, 52)



(a)

(b)



(c)

(d)

(e)

(f)

(g)

(h)

Figure 10: Parity plots for the HTC in $[Wm^2\ K^{-1}]$ for (a) the baseline correlation in the training set, (b) the baseline correlation in the testing set, (c) RF without GS in the training set for Scenario 2a, (d) RF without GS in the testing set for Scenario 2a, (e) RF with GS in training set for Scenario 2b, (f) RF with GS in the testing set for Scenario 2b, (g) RF with GS in training set for Scenario 2c, (h) RF with GS in the testing set for Scenario 2c. Sidelines indicate ±20% of error

Note that this portion of the study has not included the different levels of uncertainty in the experimental data, as the overall data set comes from a series of independent experiments, by several authors. This could play an important role, as larger uncertainties could be propagated to the estimations of the output variable. Given that such uncertainties were not available as part of the experimental data set, the results shown in this section are not entirely decoupled from these uncertainties. An alternative to estimate the uncertainty in the measurements is the use of Bootstrapping. In this method, an initial deterministic model is fitted, and several bootstrapped samples are generated that combine the reference model output with errors sampled with replacement. This leads to create “N” synthetic dataset and for each dataset, a new realisation of the model is obtained reflecting the impact of measurement errors on the uncertainty of the model parameters. Although this method assumes a normal distribution of errors and an equal probability of realisation, it can be useful in the absence of measurement errors from the experiments.

However, the objectives of this work are to explore the use of pure machine learning methods thus this is outside the scope of the present work.

7.3. Feature Sensitivity

A further insight into selecting an appropriate model is assessing the trend of the target variable with respect to specific relevant features. As shown in Figure 5, a generalised model has a balanced bias-variance trade-off, which can be visualised with a sensitivity study. For this study, HTC estimations were carried out with respect to variations in vapour quality x_v for a fixed mass flux of the refrigerant G. The estimations are done for eight different values of G (in Table 12) and each microfin tube. As a reference for expected behaviour, the HTC estimations using the baseline correlation, for a single microfin tube with the characteristics given in Table 12, are illustrated in Figure 11. As shown, each HTC trend increases with increasing vapour quality and mass flux. This behaviour is consistent with experimental observations. The smooth rate at which the estimations increase indicates that the baseline correlation is relatively robust to changes in these two relevant features. It is then expected that the selected ML-based models provide similar trends under the same conditions.

In Scenario 1, the most accurate model so far is DNN 1 with GS. The sensitivity results for this model with each set of input features (*i.e.* Scenario 1a, Scenario 1b, Scenario 1c) are shown in Figure 12(a) to Figure 12(c). The trend in each model is generally similar to that shown in Figure 11; however, such trends are not as smooth as those given by the baseline correlation. This discrepancy suggests that this DNN model tends to be biased towards the training data. A possible cause of this phenomenon is the higher complexity of the model (given its number of hidden layers and neurons per hidden layer) compared to the other DNNs used in this Scenario. The HTC estimations for the same conditions, using DNN 3 with GS are depicted in Figure 12(d) to Figure 12(c). These estimations seem to be more consistent with those of the baseline correlation, as well as being more accurate when trained and tested. In particular, the use of input features as Scenario 1a appears to be more realistic, indicating the DNN model benefits more from the use of a higher number of input features. In light of these insights, the selection of an appropriate model should be updated to DNN 3 with GS, as even though a higher MAE is expected (*i.e.* higher bias), it shows smoother trends than DNN 1 (*i.e.* lower variance), leading to a more balanced and therefore more suitable model, which can predict accurately and respond robustly to different sets of values.

Table 12: Characteristics of single microfin tube and feature ranges for sensitivity study

Variable [units]	**Value**	**Variable [units]**	**Value**
Fluid	R134a	d_f [mm]	7.70
T_s [°C]	40.2	n_f $[-]$	60
$T_s - T_w$ [°C]	7.40	h_f [mm]	0.23
γ [°]	43	β [°]	13
Range G $[\mathrm{kg\ m^{-2}s^{-1}}]$	[100, 800]	Range x_v $[-]$	[0.1, 0.9]

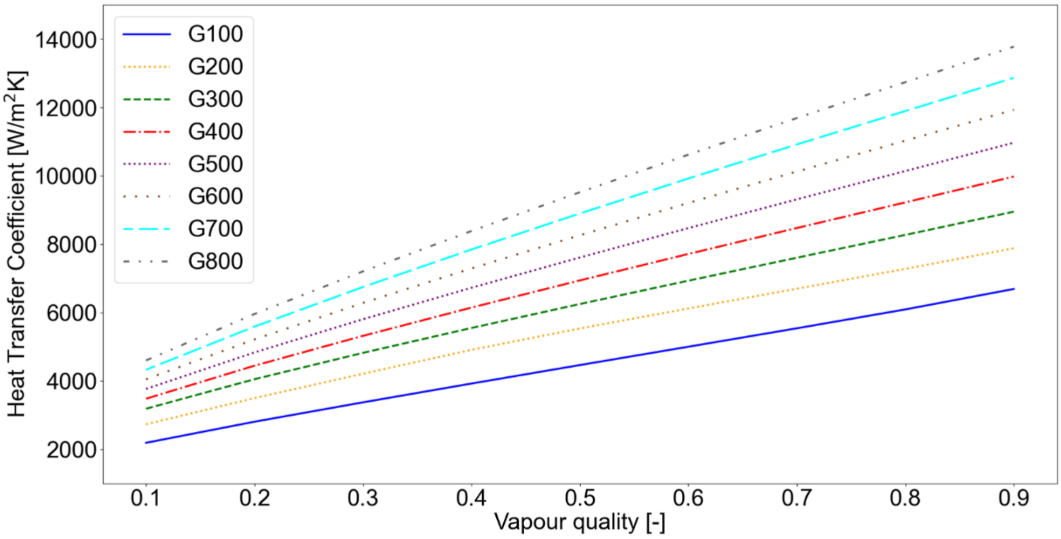
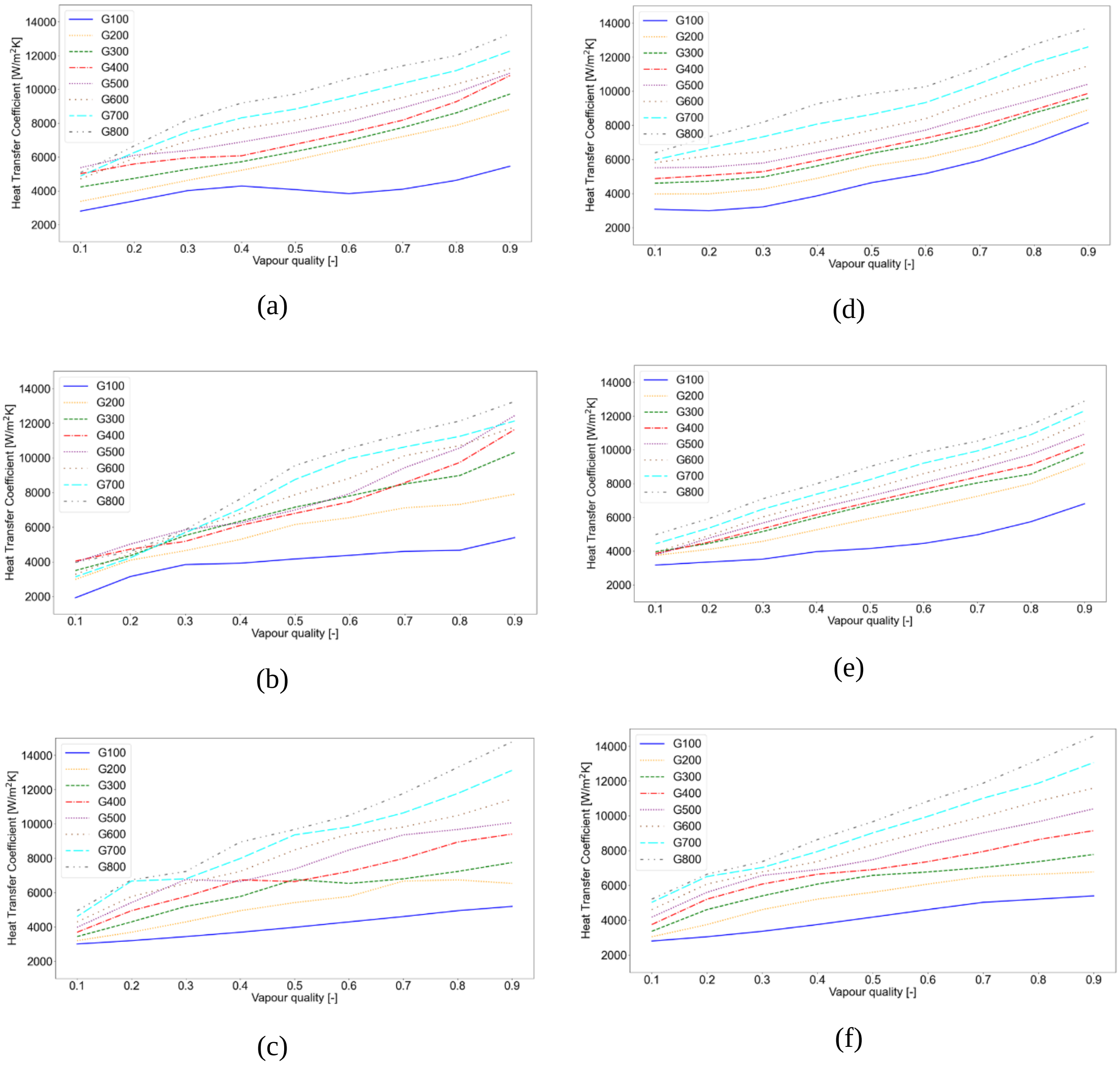


Figure 11: HTC estimations for the single microfin tube described in Table 12



(a)

(d)

(b)

(e)

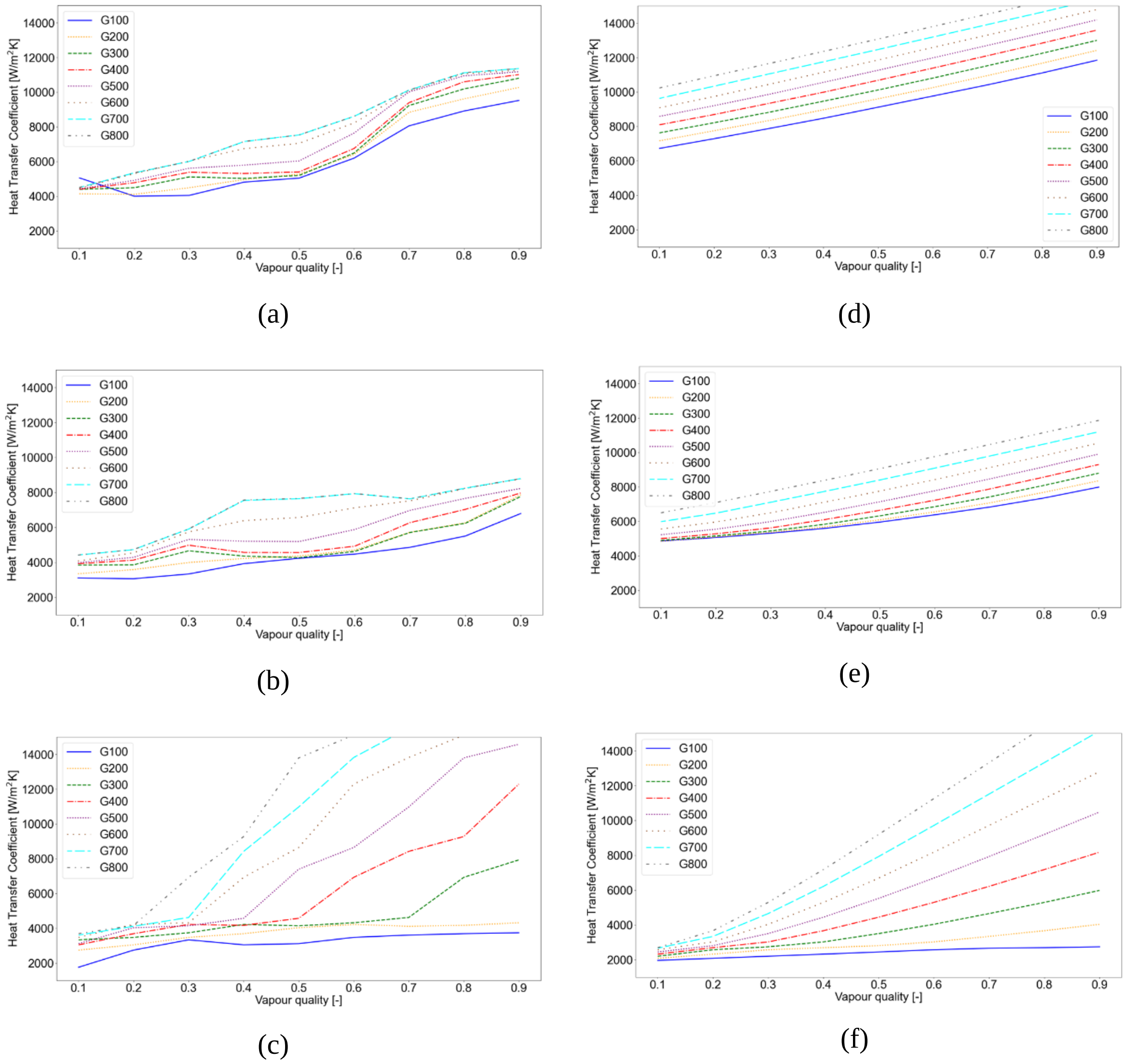
(c)

(f)

Figure 12: Estimations of the HTC in $[Wm^2\ K^{-1}]$ for the single tube in Table 12 using (a) DNN 1 with GS for Scenario 1a, (b) DNN 1 with GS for Scenario 1b, (c) DNN 1 with GS for Scenario 1c, (d) DNN 3 with GS for Scenario 1a, (e) DNN 3 with GS for Scenario 1b, (f) DNN 3 with GS for Scenario 1c

In Scenario 2, a similar phenomenon is observed. The estimations for the sensitivity study using the RF regressor without GS for each set of input features are shown in Figure 13(a) to Figure 13(c). The trends do not follow the expected behaviour and indicate that the RF model, considering its training data (*i.e.* low bias, see Figure 5), is too complex. It has already been pointed out that the selection of training data as in Scenario 2 does not provide a suitable regression model for the HTC. The results in Figure 13 support this claim, the model with the lowest MAE in this case not only provide less accurate estimations than the baseline correlation but also its response to changes in a pair of relevant features lacks interpretability and is not consistent with the selected baseline.

Similar to Scenario 1, the use of a simpler model, such as DNN 3 with GS, outputs more consistent trends than the initially selected model (*i.e.* RF without GS). These results are shown in Figure 13(d) to Figure 13(f) for each set of input features. Once again, the bias-variance trade-off shows in this case that from all the models studied, a model with a higher bias (or less complexity) is more desirable. Note that the set of input features with the lowest MAE (*i.e.* Scenario 2c) delivers a different behaviour with respect to the two remaining sets, which overestimate the HTC (when compared to the baseline correlation). However, the trends delivered by DNN 3 in Scenario 2c are still not as consistent with the baseline as the ones seen with DNN 3 in Scenario 1, as they tend to overestimate at higher values of x_v and G.



(a)

(d)

(b)

(e)

(c)

(f)

Figure 13: Estimations of the HTC in $[Wm^2\ K^{-1}]$ for the single tube in Table 12 using (a) RF without GS for Scenario 2a, (b) RF without GS for Scenario 2b, (c) RF without GS for Scenario 2c, (d) DNN 3 with GS for Scenario 2a, (e) DNN 3 with GS for Scenario 2b, (f) DNN 3 with GS for Scenario 2c

This parametric study is useful to understand the bias-variance trade-off of different models that can exhibit similarities in their accuracy scores. Note that if only the MAE and out-of-range percentage were used to select a model over the baseline correlation, those regressors with low biases would have been selected, and no further evaluation of the balance with its variance would have taken place, leading to an uninformed decision. Overall, a simpler model, with a lower number of parameters and well-balanced bias and variance is preferable. In addition, the use of input features including expert knowledge (as in Scenario 1c/2c) can bring improvements in estimation accuracy.

8. Conclusions

This work proposed a framework for the deployment of Machine Learning algorithms developed specifically for the regression of heat transfer data. The framework consists of several steps that include data pre-treatment, methods for feature selection and performance indicators to support the choice of accurate and robust models. The Case Study presented involved the estimation of the condensation heat transfer coefficient in microfin tubes using the proposed framework and two different data-driven algorithms. The dataset contained 4333 data points collected by several research studies over the last three decades with a sufficient range of variability in the independent variables.

The models developed in this study were assessed concerning their accuracy and extrapolation capabilities in two different scenarios. In Scenario 1, where data was split randomly, all developed models could predict the HTC quite accurately on the testing dataset (both the RF and the DNN models) with a MAE smaller than 988 W/m^2K. All DNN models perform more robustly than the RF algorithm, where the use of DNN 3 using grid search and all input features provides more suitable estimations with a MAE in the testing set of 685 $[Wm^2K^{-1}]$, compared to 1061 $[Wm^2K^{-1}]$ when using the baseline correlation. Moreover, differences in model accuracy when changing the number of input features are negligible, as similar accuracy is obtained when a proper set of input features is selected, either by feature importance analysis or use of expert knowledge via dimensionless numbers. In Scenario 2, where the data was split systematically, the model accuracy dropped orders of magnitude compared to Scenario 1. In addition, given the differences during training, the use of the baseline correlation is more accurate than any of the selected ML model (DNN 3 with grid search), where the former shows a testing MAE of 1111 $[Wm^2K^{-1}]$ and the latter, 1819 $[Wm^2K^{-1}]$. This scenario showcases the lack of robustness of ML-based methods when the training data are limited in terms of ranges of values to cover.

The feature sensitivity study allowed for a deeper understanding of the behaviour of the machine learning models. The use of random splitting to select training data also provided better trends than those from the systematic selection. The baseline correlation behaved more realistically; however, the selected DNN exhibited sufficiently close trends and at the same time, more accurate estimates during the training and testing stages.

The wide variety of scenarios tested in this study enables a deeper understanding of the way Machine Learning models perform in specific contexts (*e.g.* different sets of input features and different ranges of values for the training set). In future work, to improve the prediction capabilities of the Machine Learning models, a potential option is the use of hybrid ML algorithms that benefit from the features of both data-driven and physics-based models. In addition, the estimation of measurement uncertainty via Bootstrapping or other suitable methods will be accounted for.

Acknowledgements

The authors acknowledge Hexxcell Ltd. for providing the funding for the work presented here.

Nomenclature

Abbreviations		*h(.)*	Machine learning model

AMM	Acoustic mismatch model
ANN	Artificial neural network
Adam	Adaptive moment estimation method
CFD	Computational fluid dynamics
CNN	Convolutional neural network
CV	Cross-validation
DMM	Diffusive mismatch model
DNN	Deep neural network
GBM	Gradient boosting machine
GPR	Gaussian process regression
GS	Grid search
HTC	Heat transfer coefficient
HVAC	Heating, ventilation, and air conditioning
ITR	Interfacial thermal resistance
KNN	K-nearest neighbour
LSBoost	Least-squares boosting
MAE	Mean absolute error
MD	Molecular dynamics
ML	Machine learning
MLP	Multilayer perceptron
MSE	Mean squared error
PCA	Principal component analysis
PINN	Physics-informed neural network
RF	Random forest
RFE	Recursive feature elimination
RFECV	Recursive feature elimination with cross-validation
SGD	Stochastic gradient descent
SL	Supervised learning
SSR	Sum of squared error
SVM	Support vector machine
SVR	Support vector regression
Symbols	
Bo	Bond number $[-]$
cp	Heat capacity [J kg^{-1} K^{-1}]
$\boldsymbol{\mathcal{D}}$	Experimental data
h_f	Fin height $[m]$
h_{LV}	Heat of vaporisation [J kg^{-1}]
ℓ	Range limit of the data $[-]$
L_f	Tube length [m]
M	Number of independent variables $[-]$
N	Number of observations $[-]$
n_f	Number of fins per tube $[-]$
Nu	Nusselt number $[-]$
$\boldsymbol{\mathcal{p}(.)}$	Vector of pre-processing functions
P_{red}	Reduced pressure $[-]$
Pr	Prandtl number $[-]$
R	Range of data
R_x	Geometrical enhancement factor $[-]$
Re	Reynolds number $[-]$
T	Temperature [K]
$\Delta T_{dimless}$	Dimensionless temperature $[-]$
$\boldsymbol{X}$	Independent variables
x_v	Vapour quality $[-]$
$\boldsymbol{Y}$	Target variables
$\hat{\boldsymbol{Y}}$	Estimated target variables
Greeks	
β	Helix angle [°]
γ	Apex angle [°]
$\boldsymbol{\Theta}$	Regression parameters
$\boldsymbol{\vartheta}$	ML model hyperparameters
λ	Thermal conductivity [W m^{-1}K^{-1}]
μ	Viscosity [Pa s]
ρ	Density [kg m^{-3}]
σ	Liquid surface tension [N m^{-1}]
τ_c	Critical threshold $[-]$
Subscripts	
i, j, k	indexes
L	Liquid
S	Surface
V	Vapour

$\bar{\boldsymbol{\mathcal{D}}}$	Pre-processed data	W	Wall
d_f	Fin tip diameter [m]	***Superscripts***	
f(.)	Regression function	*opt*	optimised
$J(.)$	Objective function	*test*	Testing data set
J_V	Vapour velocity $[-]$	*train*	Training data set
$\mathcal{L}(.)$	Machine learning loss function		
G	Refrigerant mass flux $[\mathrm{kg\ m^{-2}s^{-1}}]$		
g(.)	Baseline correlation function		

References

[1] F. Coletti, Ed., Heat exchanger design handbook multimedia edition, New York: Begell House Inc., 2016.

[2] D. Taler, “Experimental determination of Correlations for Average Heat Transfer Coefficients in Heat Exchangers on Both Fluid Sides,” *Heat and Mass Transfer,* vol. 49, no. 8, pp. 1125-1139, 2013.

[3] D. Patil, R. Arakerimath and P. Walke, “Thermoelectric materials and heat exchangers for power generation–A review,” *Renewable and Sustainable Energy Reviews,* vol. 95, pp. 1-22, 2018.

[4] V. Pandiyarajan, M. Pandian, E. Malan, R. Velraj and R. Seeniraj, “Experimental investigation on heat recovery from diesel engine exhaust using finned shell and tube heat exchanger and thermal storage system,” *Applied Energy,* vol. 88, no. 1, pp. 77-87, 2011.

[5] S. Rashidi, F. Hormozi, B. Sundén and O. Mahian, “Energy saving in thermal energy systems using dimpled surface technology–A review on mechanisms and applications,” *Applied Energy,* vol. 250, pp. 1491-1547, 2019.

[6] D. Kern, Process heat transfer, New York: Mc Graw Hill, 1950.

[7] G. Hewitt, G. Shires and T. Bott, Process heat transfer, Begell House, 1994.

[8] G. Hewitt, Ed., Heat exchanger design handbook, New York: Begell House, 2008.

[9] D. T. Chin and K. L. Hsueh, “An Analysis Using the Chilton-Colburn Analogy for Mass Transfer to a Flat Surface From an Unsubmerged Impinging Jet,” *Electrochimica acta,* vol. 31, no. 5, pp. 561-564, 1986.

[10] F. W. Dittus and L. M. K. Boelter, “Heat Transfer in Automobile Radiators of the Tubular Type,” *International communications in heat and mass transfer,* vol. 12, no. 1, pp. 3-22, 1985.

[11] S. Aravinth, “Prediction of heat and mass transfer for fully developed turbulent fluid flow through tubes,” *International Journal of Heat and Mass Transfer,* vol. 43, no. 8, pp. 1399-1408, 2000.

[12] J. Ma, Y. Huang, J. Huang, Y. Wang and Q. Wang, “Experimental investigations on single-phase heat transfer enhancement with longitudinal vortices in narrow rectangular channel,” *Nuclear Engineering and Design,* vol. 240, no. 1, pp. 92-102, 2010.

[13] M. J. Vaze and J. Banerjee, “A Modified Heat Transfer Correlation for Two-phase Flow,” *Heat and mass transfer,* vol. 47, no. 9, pp. 1159-1170, 2011.

[14] T. Layssac, S. Lips and R. Revellin, “Effect of inclination on heat transfer coefficient during flow boiling in a mini-channel,” *International Journal of Heat and Mass Transfer,* vol. 132, pp. 508-518, 2019.

[15] J. Kim and A. Ghajar, "A general heat transfer correlation for non-boiling gas–liquid flow with different flow patterns in horizontal pipes," *International Journal of Multiphase Flow,* vol. 32, no. 4, pp. 447-465, 2006.

[16] A. Cavallini, D. Del Col, S. Mancin and L. Rossetto, "Condensation of Pure and Near-azeotropic Refrigerants in Microfin Tubes: A New Computational Procedure," *International Journal of Refrigeration,* vol. 32, no. 1, pp. 162-174, 2009.

[17] S. Mehendale, "A new heat transfer coefficient correlation for pure refrigerants and near-azeotropic refrigerant mixtures flow boiling within horizontal microfin tubes.," *International Journal of Refrigeration,* vol. 86, pp. 292-311, 2018.

[18] X. Wu, C. Li, J. Yang, Y. Liu and X. Han, Theoretical and Experimental Research on the Microchannel Flow Boiling Heat Transfer for Igbt Modules, Available at SSRN: https://ssrn.com/abstract=4305605, 2022.

[19] S. Kim and I. Mudawar, "Universal approach to predicting heat transfer coefficient for condensing mini/micro-channel flow," *International Journal of Heat and Mass Transfer,* vol. 56, no. 1-2, pp. 238-250, 2013.

[20] J. El Hajal, J. Thome and A. Cavallini, "Condensation in horizontal tubes, part 1: two-phase flow pattern map," *International journal of heat and mass transfer,* vol. 46, no. 18, pp. 3349-3363, 2003.

[21] J. Thome, J. El Hajal and Cavallini, "Condensation in horizontal tubes, part 2: new heat transfer model based on flow regimes," *International journal of heat and mass transfer,* vol. 46, no. 18, pp. 3365-3387, 2003.

[22] A. Cavallini, D. Col, L. Doretti, M. Matkovic, L. Rossetto, C. Zilio and G. Censi, "Condensation in horizontal smooth tubes: a new heat transfer model for heat exchanger design," *Heat transfer engineering,* vol. 27, no. 8, pp. 31-38, 2006.

[23] M. Hughes, G. Kini and S. Garimella, "Status, challenges, and potential for machine learning in understanding and applying heat transfer phenomena," *Journal of Heat Transfer,* vol. 143, no. 12, 2021.

[24] H. Ma, X. Hu, Y. Zhang, N. Thuerey and O. Haidn, "A combined data-driven and physics-driven method for steady heat conduction prediction using deep convolutional neural networks," *arXiv,* p. 2005.08119, 2020.

[25] S. Hwangbo, R. Al and G. Sin, "An integrated framework for plant data-driven process modeling using deep-learning with Monte-Carlo simulations," *Computers & Chemical Engineering,* vol. 143, p. 107071, 2020.

[26] G. James, D. Witten, T. Hastie and R. Tibshirani, An Introduction to Statistical Learning, New York: Springer, 2013.

[27] V. S. Kadam, S. Kanhere and S. Mahindrakar, "Regression Techniques in Machine Learning & Applications: A review," *International Journal for Research in Applied Science and Engineering Technology,* vol. 8, no. 10, pp. 826-830, 2020.

[28] P. Hang, L. Zhao and G. Liu, "Optimal design of heat exchanger network considering the fouling throughout the operating cycle," *Energy,* vol. 241, p. p.122913, 2022.

[29] M. Ravagnani, A. Silva, P. Arroyo and A. Constantino, "Heat exchanger network synthesis and optimisation using genetic algorithm," *Applied Thermal Engineering,* vol. 25, no. 7, pp. 1003-1017, 2005.

[30] G. Xu, L. Zhuang, B. Dong, Q. Liu and J. Wen, "Optimization design with an advanced genetic algorithm for a compact air-air heat exchanger applied in aero engine," *International Journal of Heat and Mass Transfer,* vol. 158, p. p.119952, 2020.

[31] H. Peng and X. Ling, "Optimal design approach for the plate-fin heat exchangers using neural networks cooperated with genetic algorithms," *Applied thermal engineering,* vol. 28, no. 5-6, pp. 642-650, 2008.

[32] M. Tayal, Y. Fu and U. Diwekar, "Optimal design of heat exchangers: A genetic algorithm framework," *Industrial & engineering chemistry research,* vol. 38, no. 2, pp. 456-467, 1999.

[33] R. Madhu PK, J. Subbaiah and K. Krithivasan, "RF-LSTM-based method for prediction and diagnosis of fouling in heat exchanger," *Asia-Pacific Journal of Chemical Engineering,* vol. 16, no. 5, p. e.2684, 2021.

[34] A. Gupta, V. Jadhav, M. Patil, A. Deodhar and V. Runkana, "Forecasting of Fouling in Air Pre-Heaters Through Deep Learning," in *ASME Power Conference,* 2021.

[35] J. Aminian and S. Shahhosseini, "Evaluation of ANN modeling for prediction of crude oil fouling behavior," *Applied thermal Engineering,* vol. 28, no. 7, pp. 668-674, 2008.

[36] S. Lalot and H. Pálsson, "Detection of fouling in a cross-flow heat exchanger using a neural network based technique," *International Journal of Thermal Sciences,* vol. 49, no. 4, pp. 675-679, 2010.

[37] 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," *Applied Thermal Engineering,* vol. 27, no. 17-18, pp. 2791-2802, 2007.

[38] F. Coletti and G. Hewitt, Eds., Crude oil fouling: deposit characterization, measurements, and modeling, Gulf Professional Publishing, 2015.

[39] J. Loyola-Fuentes, C. Planelles, E. Diaze-Bejarano and F. Coletti, "Data reconciliation and gross error detection methods in industrial processes: state of the art and focus on heat transfer".

[40] M. Bagajewicz and S. Rollins, "Data Reconciliation and Software Methods for Bias Detection," in *Instrument Engineers' Handbook, Volume 3: Process Software and Digital Networks*, CRC Press, 2016, p. 371.

[41] M. Bagajewicz and D. Nguyen, "Stochastic-based accuracy of data reconciliation estimators for linear systems," *Computers & Chemical Engineering,* vol. 32, no. 6, pp. 1257-1269, 2008.

[42] M. Bagajewicz and Q. Jiang, "Comparison of steady state and integral dynamic data reconciliation," *Computers & Chemical Engineering,* vol. 24, no. 11, pp. 2367-2383, 2000.

[43] F. Coletti, "Hybrid AI for Industrial Thermal Systems," in *AIChE Spring Meeting and 19th Global Congress on Process Safety,* Houston, 2023.

[44] S. K. A. Hosseini, M. Chowdhury, M. Ayari, T. Rahman, M. Chowdhury and B. Vaferi, "Novel and robust machine learning approach for estimating the fouling factor in heat exchangers," *Energy Reports,* vol. 8, pp. 8767-8776, 2022.

[45] W. Hale, E. Safikou and G. Bollas, "Inference of faults through symbolic regression of system data," *Computers & Chemical Engineering,* vol. 157, p. p.107619, 2022.

[46] S. Brandi, M. Piscitelli, M. Martellacci and A. Capozzoli, "Deep reinforcement learning to optimise indoor temperature control and heating energy consumption in buildings," *Energy and Buildings,* vol. 224, p. 110225, 2020.

[47] O. Ekren, S. Sahin and Y. Isler, "Comparison of different controllers for variable speed compressor and electronic expansion valve," *International Journal of Refrigeration,* vol. 33, no. 6, pp. 1161-1168, 2010.

[48] A. Asadi, A. N. Bakhtiyari and I. M. Alarifi, "Predictability Evaluation of Support Vector Regression Methods for Thermophysical Properties, Heat Transfer Performance, and Pumping Power Estimation of MWCNT/ZnO–engine Oil Hybrid Nanofluid," *Engineering with Computers,* vol. 37, no. 4, pp. 3813-3823, 2021.

[49] S. Bagherzadeh, A. D'Orazio, A. Karimipour, M. Goodarzi and Q. Bach, "A novel sensitivity analysis model of EANN for F-MWCNTs–Fe3O4/EG nanofluid thermal conductivity: Outputs predicted analytically instead of numerically to more accuracy and less costs," *Physica A: Statistical Mechanics and its Applications,* vol. 521, pp. 406-415, 2019.

[50] A. G. M. Alrashed, M. Goodarzi, L. de Oliveira, M. Safaei and E. Bandarra Filho, "Effects on thermophysical properties of carbon based nanofluids: experimental data, modelling using regression, ANFIS and ANN," *International Journal of Heat and Mass Transfer,* vol. 125, pp. 920-932, 2018.

[51] A. Akhgar, D. Toghraie, N. Sina and M. Afrand, "Developing dissimilar artificial neural networks (ANNs) to prediction the thermal conductivity of MWCNT-TiO2/Water-ethylene glycol hybrid nanofluid," *Powder Technology,* vol. 355, pp. 602-610, 2019.

[52] Y. Wu, L. Fang and Y. Xu, "Predicting interfacial thermal resistance by machine learning," *npj Computational Materials,* vol. 5, no. 1, p. 56, 2019.

[53] J. Peng, X. Liu, N. Aubry, Z. Chen and W. Wu, "Data-driven modeling of geometry-adaptive steady heat transfer based on convolutional neural networks: heat conduction," *arXiv,* 2020.

[54] S. Szénási, Z. Fried and I. Felde, "Training of Artificial Neural Network to Solve the Inverse Heat Conduction Problem," in *IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI),* Herl'any, Slovakia, 2020.

[55] B. Kwon, F. Ejaz and L. Hwang, "Machine learning for heat transfer correlations," *International Communications in Heat and Mass Transfer,* vol. 116, p. 104694, 2020.

[56] S. Cai, Z. Wang, S. Wang, P. Perdikaris and G. Karniadakis, "Physics-informed neural networks for heat transfer problems," *Journal of Heat Transfer,* vol. 143, no. 6, 2021.

[57] B. Souayeh, S. Bhattacharyya, N. Hdhiri and M. Waqas Alam, "Heat and fluid flow analysis and ann-based prediction of a novel spring corrugated tape," *Sustainability,* vol. 13, no. 6, p. 3023, 2021.

[58] S. Pai and J. Weibel, "Machine-learning-aided design optimization of internal flow channel cross-sections," *International Journal of Heat and Mass Transfer,* vol. 195, p. 123118, 2022.

[59] G. Mask, X. Wu and K. Ling, "An improved model for gas-liquid flow pattern prediction based on machine learning," *Journal of Petroleum Science and Engineering,* vol. 183, p. 106370, 201.

[60] G. Hobold and A. da Silva, "Machine learning classification of boiling regimes with low speed, direct and indirect visualization," *International Journal of Heat and Mass Transfer,* vol. 125, pp. 1296-1309, 2018.

[61] J. Loyola-Fuentes, L. Pietrasanta, M. Marengo and F. Coletti, "Machine Learning Algorithms for Flow Pattern Classification in Pulsating Heat Pipes," *Energies,* vol. 15, no. 6, p. 1970, 2022.

[62] B. Yang, X. Zhu, B. Wei, M. Liu, Y. Li, Z. Lv and F. Wang, “Computer Vision and Machine Learning Methods for Heat Transfer and Fluid Flow in Complex Structural Microchannels: A Review,” *Energies,* vol. 16, no. 3, p. 1500, 2023.

[63] M. Hughes, B. Fronk and S. Garimella, “Universal condensation heat transfer and pressure drop model and the role of machine learning techniques to improve predictive capabilities,” *International Journal of Heat and Mass Transfer,* vol. 179, p. 121712, 2021.

[64] A. Khosravi, J. Pabon, R. Koury and L. Machado, “Using machine learning algorithms to predict the pressure drop during evaporation of R407C,” *Applied Thermal Engineering,* vol. 133, pp. 361-370, 2018.

[65] G. Hobold and A. da Silva, “Visualization-based nucleate boiling heat flux quantification using machine learning,” *International Journal of Heat and Mass Transfer,* vol. 134, pp. 511-520, 2019.

[66] G. Longo, G. Righetti, C. Zilio, L. Ortombina, M. Zigliotto and J. Brown, “Application of an Artificial Neural Network (ANN) for predicting low-GWP refrigerant condensation heat transfer inside herringbone-type Brazed Plate Heat Exchangers (BPHE),” *International Journal of Heat and Mass Transfer,* vol. 156, p. 119824, 2020.

[67] G. Longo, S. Mancin, G. Righetti, C. Zilio, L. Ortombina and M. Zigliotto, “Application of an Artificial Neural Network (ANN) for predicting low-GWP refrigerant boiling heat transfer inside Brazed Plate Heat Exchangers (BPHE),” *International Journal of Heat and Mass Transfer,* vol. 160, p. 120204, 2020.

[68] G. A. Longo, S. Mancin, G. Righetti, C. Zilio, R. Ceccato and L. Salmaso, “Machine Learning Approach for Predicting Refrigerant Two-phase Pressure Drop Inside Brazed Plate Heat Exchangers (BPHE),” *International Journal of Heat and Mass Transfer,* vol. 163, p. 120450, 2020.

[69] M. Calati, G. Righetti, L. Doretti, C. Zilio, G. Longo, K. Hooman and S. Mancin, “Water pool boiling in metal foams: From experimental results to a generalized model based on artificial neural network,” *International Journal of Heat and Mass Transfer,* vol. 176, p. 121451, 2021.

[70] M. Das and E. K. Akpinar, “Investigation of Pear Drying Performance by Different Methods and Regression of Convective Heat Transfer Coefficient with Support Vector Machine,” *Applied Sciences,* vol. 8, no. 2, p. 215, 2018.

[71] T. Agrawal, Hyperparameter optimization in machine learning: make your machine learning and deep learning models more efficient, New York, NY, USA: Apress, 2021.

[72] D. Rativa, B. Fernandes and A. Roque, “Height and weight estimation from anthropometric measurements using machine learning regressions,” *IEEE journal of translational engineering in health and medicine,* vol. 6, pp. 1-9, 2018.

[73] L. Rajulapati, S. Chinta, B. Shyamala and R. Rengaswamy, “Integration of machine learning and first principles models,” *AIChE Journal,* vol. 68, no. 6, p. e.17715, 2022.

[74] S. Nandi, Y. Badhe, J. Lonari, U. Sridevi, B. Rao, S. Tambe and B. Kulkarni, “Hybrid process modeling and optimization strategies integrating neural networks/support vector regression and genetic algorithms: study of benzene isopropylation on Hbeta catalyst,” *Chemical Engineering Journal,* vol. 97, no. 2-3, pp. 115-129, 2004.

[75] M. Von Stosch, R. Oliveira, J. Peres and S. de Azevedo, “Hybrid semi-parametric modeling in process systems engineering: Past, present and future,” *Computers & Chemical Engineering,* vol. 60, pp. 86-101, 2014.

[76] G. M. Maggiora, D. W. Elrod and R. G. Trenary, “Computational Neural Networks as Model-free Mapping Devices,” *Journal of Chemical Information and Computer Sciences,* vol. 32, no. 6, pp. 732-741, 1992.

[77] K. O'Shea and R. Nash, “An introduction to convolutional neural networks,” *arXiv,* 2015.

[78] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li and M. Sun, “Graph neural networks: A review of methods and applications,” *AI Open,* vol. 1, pp. 57-81, 2020.

[79] D. Svozil, V. Kvasnicka and J. Pospichal, “Introduction to multi-layer feed-forward neural networks,” *Chemometrics and intelligent laboratory systems,* vol. 39, no. 1, pp. 43-62, 1997.

[80] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv,* 2014.

[81] B. Garro and R. Vázquez, “Designing artificial neural networks using particle swarm optimization algorithms,” *Computational intelligence and neuroscience,* pp. 61-61, 2015.

[82] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International conference on machine learning*, 2015, pp. 448-456.

[83] L. Breiman, “Random Forests,” *Machine Learning,* vol. 45, pp. 5-32, 2001.

[84] P. Probst, M. Wright and A. Boulesteix, “Hyperparameters and tuning strategies for random forest,” *Wiley Interdisciplinary Reviews: data mining and knowledge discovery,,* vol. 9, no. 3, p. e1301, 2019.

[85] T. Fushiki, “Estimation of prediction error by using K-fold cross-validation,” *Statistics and Computing,* vol. 21, pp. 137-146, 2011.

[86] A. Aouichaoui, S. Mansouri, J. Abildskov and G. Sin, “Uncertainty estimation in deep learning-based property models: Graph neural networks applied to the critical properties,” *AIChE Journal,* vol. 68, no. 6, p. e.17696, 2022.

[87] P. Mehta, M. Bukov, C. Wang, A. Day, C. Richardson, C. Fisher and D. Schwab, “A high-bias, low-variance introduction to machine learning for physicists,” *Physics reports,* vol. 810, pp. 1-124, 2019.

[88] V. Ramesh, P. Baskaran, A. Krishnamoorthy, D. Damodaran and P. Sadasivam, “ Back propagation neural network based big data analytics for a stock market challenge,” *Communications in Statistics-Theory and Methods,* vol. 48, no. 14, pp. 3622-3642, 2019.

[89] K. Fujie, M. Itoh, T. Innami, H. Kimura, W. Nakayama and T. Yanagida, “Heat transfer pipe”. United States Patent US 4044797, 8 1977.

[90] J. E. Vollrath, P. S. Hrnjak and T. A. Newell, An Experimental Investigation of Pressure Drop and Heat Transfer in an In-tube Condensation System of Pure Ammonia, Air Conditioning and Refrigeration Center. College of Engineering. University of Illinois at Urbana-Champaign, 2003.

[91] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden , M. Wattenberg, M. Wicke, Y. Yu and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous distributed systems,” *arXiv preprint arXiv:1603.04467,* 2016.