Unsupervised-Learning Assisted Artificial Neural Network for Optimization

Varun Kote
Old Dominion University

UNSUPERVISED- LEARNING ASSISTED ARTIFICIAL NEURAL NETWORK FOR OPTIMIZATION

by

Varun Kote
B.E. June 2016,
SIR M Visvesvarya Institute of Technology

A Thesis Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

MECHANICAL ENGINEERING

OLD DOMINION UNIVERSITY
August 2019

ABSTRACT

UNSUPERVISED-LEARNING ASSISTED ARTIFICIAL NEURAL NETWORK FOR OPTIMIZATION

Varun Kote
Old Dominion University, 2019
Director: Dr. Oktay Baysal

Innovations in computer technology made way for Computational Fluid Dynamics (CFD) into engineering, which supported the development of new designs by reducing the cost and time by lowering the dependency on experimentation. There is a further need to make the process of development more efficient. One such technology is Artificial Intelligence. In this thesis, we explore the application of Artificial Intelligence (AI) in CFD and how it can improve the process of development.

AI is used as a buzz word for the mechanism which can learn by itself and make the decision accordingly. Machine learning (ML) is a subset of AI which learns any method without the need for any explicit algorithm. Deep Learning is another subset of ML, which is different in its composition. Deep Learning, or Neural Networks (NN), is made up of nodes like the neurons and works on the principle of the human brain. NN can be exploited for any problem without the need for any explicit algorithm for the task. It can be achieved by analyzing and inferring from the observations. Artificial Neural Network (ANN) is used for data analysis and Convolutional Neural Networks (CNN) for image analysis. Our area of interest herein is ANN and its application for a medical equipment called Convective Polymerase Chain Reaction (cPCR) device.

Many have relied on engineering experimentation to develop an optimized PCR device, which requires high cost and time. That makes the use of PCR devices less cost-effective as a commonplace for healthcare. We optimize a convective PCR reactor using a high-fidelity CFD-based surrogate model to find an economical and performance-effective one. We plan numerous possible design combinations, evaluating DNA doubling time. Based on these results, an accurate surrogate model is developed for optimization using Deep Learning. We produce two kinds of surrogate models using ANN; one by directly employing ANN and another by using unsupervised learning called, k-Means-Clustering-Assisted ANN, and then compare the results from these two methods. For developing a suitable model of ANN to fit our data, we carry out the analysis of model accuracy and obtain the best design by using a differential evolution method. The best designs obtained by the two methods are verified with the corresponding result obtained from CFD. This shows an effective way of designing an optimized device by reducing the number of CFD simulations required for the development. Consequently, the computational results demonstrate that the convective PCR device can be efficiently developed using our proposed methodology, making it viable for point-of-care applications.

ACKNOWLEDGMENTS

I thank Dr. Oktay Baysal for his excellent guidance during this dissertation. He has been a source of constant support and encouragement throughout the process, especially during the hard times. Without his advice, this thesis would not be possible.

Jung Shu has been of tremendous help during this work. Many people have contributed to the successful completion of this dissertation. I extend many, many thanks to my committee members for their patience and hours of guidance on my research and editing of this manuscript.

Also, this research would be impossible without the support and blessings from my family and friends who stood by me throughout this journey.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

Figure **Page**

INTRODUCTION

Smart technologies are proliferating in all streams. Smart technologies or smart products have become synonyms to innovation. Every product is becoming smarter and smarter over time. These intelligence systems are the mandatory standards for all products. Artificial Intelligence (AI) being the latest definition of smart technologies is seeing its use in all sectors. The whole process from conceptualization to delivering the final product to the consumer is enabled with Artificial Intelligence. And every industry wants a bite of this latest big innovation.

Mechanical and Aerospace industries are experiencing the demand for intelligent systems integrated across all levels of product development to the user end in improving the utilization of the products. The saturation in aerospace, aviation, automotive, and many other mechanical technologies has made Artificial Intelligence (AI) a necessity for the next breakthrough in innovation. All companies are placing their big bets on this innovation in all capacities. It is being used by scientists to help overcome their present limitations; this can be seen in the most advanced form of research happening in the world, i.e., nuclear fusion technology. Scientists are using AI in identifying the key points where the complex system is failing and also in methods to circumvent through those hurdles using AI. New alloys for their best performance in various applications are created by using Artificial Intelligence. Self-driving cars and self-flying drones are all designed with Artificial Intelligence as the core technology. Artificial Intelligence in medical diagnosis is the most anticipated breakthrough as it overcomes the difficulties faced in diagnosing chronic diseases like cancer at an early stage. The deployment of these technologies to remote locations gives accessibility of advanced medical facilities to people who are deprived of better medical treatments in third world countries. Every industry experiencing saturation or a slow-down in its development is looking at Artificial Intelligence for future innovation.

1.1 Purpose

Computational Fluid Dynamics (CFD) is another industry which is facing a lot of hurdles recently in its advancement. The very nature of approximations in its design and with the limited computing power available, there is very little progress made in its development made in the last decade. Computational Fluid Dynamics (CFD) had experienced the most advancement with the emergence of the computer age. As the computing power increased exponentially following the Moor's Law, the Computational Industries blossomed alongside. The Reynolds-Averaged Navier-

Stokes (RANS) equation model became very popular with growing the computational power and soon became the industry standard. Due to the ever-increasing industry demand in developing more and more high performing and sophisticated technologies, more accurate models are to be modeled to achieve it. More accurate models have seen its application recently, namely Large Eddy Simulation (LES) and Direct Numerical Simulation (DNS). The most accurate simulation of turbulent flow by solving the Navier-Stokes equation is achieved by DNS method where a wide range of time and length scales are resolved. Spalart estimated in the year 1999 that it would take until 2080 for Direct Numerical Simulation (DNS) to apply to industrial flows [27]. Large Eddy Simulation (LES) is predicted to see its industrial application around 2045. From the emergence of the Moor’s Law end era, which was predicted to close before any of that timeline, the computational capabilities will never reach the level for DNS and LES to be feasible. For the industry to further its advancement well into the future, we need to look for other technologies in supporting the development in CFD. The accuracy of flow simulations using between RANS and LES methods is shown in figure 1.

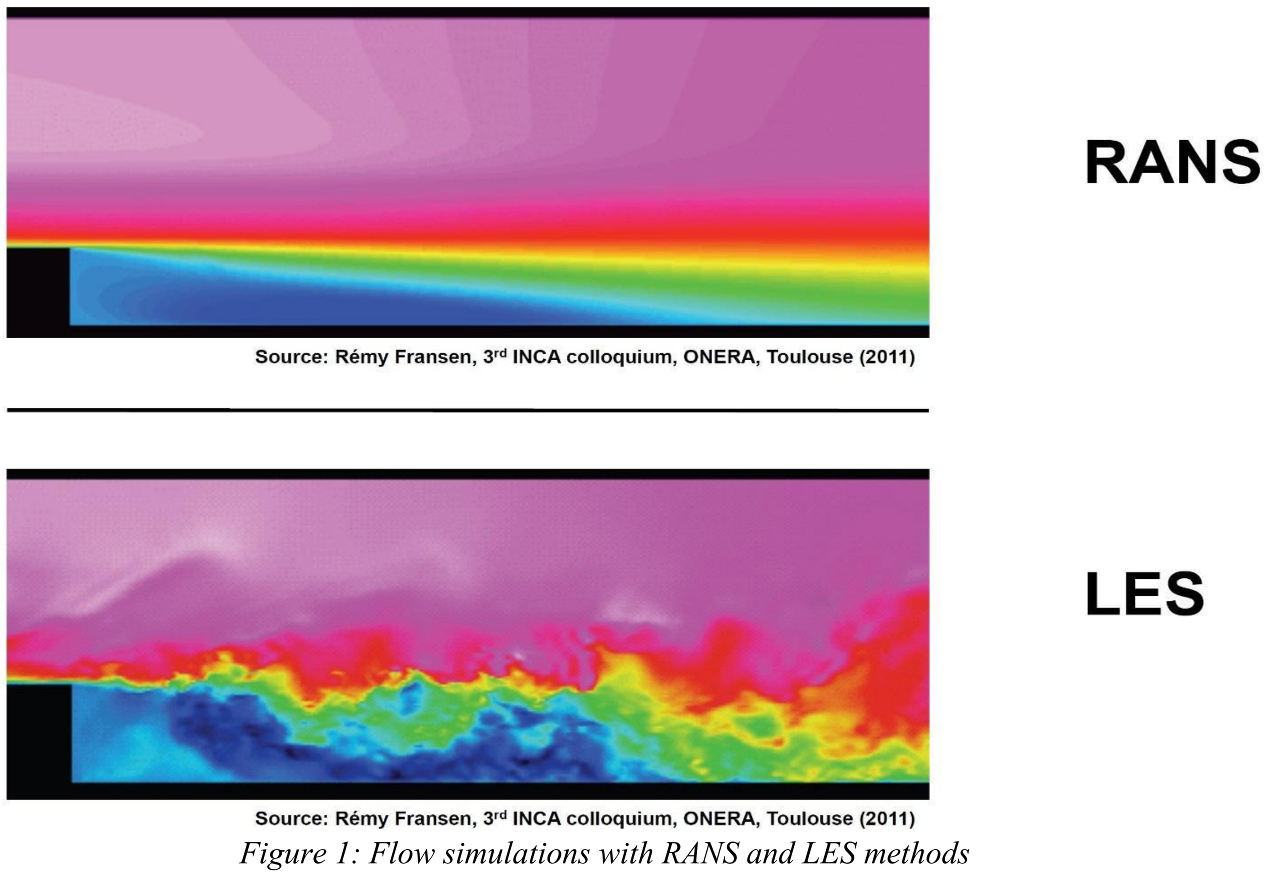

RANS
Source: Rémy Fransen, 3rd INCA colloquium, ONERA, Toulouse (2011)
LES
Source: Rémy Fransen, 3rd INCA colloquium, ONERA, Toulouse (2011)

Figure 1: Flow simulations with RANS and LES methods

Artificial Intelligence is the best technology available for Computational Fluid Dynamics to advance. I say this is the best technology because of the niche of AI being data-hungry, and CFD generates tons of data that is not made use of at any later stage in improving the CFD. Many scholars and researchers are exploring the paradigms of putting AI into CFD. This testing is happening at various levels, as discussed further. Direct Numerical Simulation is computationally expensive today, and by Moor's law, its feasibility in the future is also uncertain. Artificial Intelligence can be the alternative to DNS in achieving the accuracy in flow resolution.

The ultimate goal of Computational Fluid Dynamics in the far future is to automate the entire process chain. The engineers and researchers should be able to only define the boundary, and initial conditions on the CAD (Computer-Aided Design) model obtained from Computer-Aided Engineering (CAE) and generate accurate flow results without worrying about the appropriate models to choose. The modern developments in CFD are headed in this field of automation.

1.2 Scope

Currently, the development of more intelligent solvers that can automatically choose the appropriate wall boundary flow and turbulence models is being studied by researchers.
Artificial Intelligence can be applied to Computational Fluid Dynamics in two ways. First, in acquiring the data generated and using this pool of knowledge for solving complex problems more efficiently in the future. Second, in the process of applying the available knowledge to tractable aerodynamic design and development problems, it can reduce the time required for their numerical solutions. This report on "*Computational Aerodynamics and Artificial Intelligence"* [28] speculates on the role of Artificial Intelligence in computational aerodynamics.

National Aerospace and Space Administration (NASA) has put out a few key goals to achieve in Computational Fluid Dynamics by the year 2030 [29]. Two areas which NASA mentions in its vision 2030 for CFD are very relatable with Artificial Intelligence in CFD. First, is the ability to predict viscous turbulent flows with possible boundary layer transition and flow separation present. Secondly, to make use of the enormous CFD data that has been already generated in efficiently solving problems in the future. Application of Artificial Intelligence can solve the difficulties in the above mentioned two areas. Predicting adequate turbulent flows for design and development requires solving through Direct Numerical Simulation (DNS). Unlike in the Reynolds-Averaged Navier-Stokes (RANS) model and Large Eddy Simulation (LES), DNS resolves the flow equation completely without any turbulence models to predict accurate flow results, as shown in figure 2. The knowledge-based application of Artificial Intelligence can learn how to predict accurate turbulent flows from DNS for simple flows and replicating it for more complex problems. A research paper on Data-driven DNN for simple flows [30] models the error produced in every flow simulation. This modeled error is used to generate accurate results from the results obtained by coarsely modeled problems. Computational Fluid Dynamics is not just about generating data. The data generated requires interpretation to gain insight. Applying Artificial Intelligence in data interpretation decisions will make problem-solving much faster and effective. As more and more data is added to the pool of knowledge, more robust Artificial Intelligence emerges out of it.

Artificial Intelligence is a buzz word for smarter and intelligent systems. It has few well-defined subsets, which are getting more traction in its applications, as shown in figure 3. Machine learning is used in identifying and modeling of patterns in the data by using algorithms for clustering, classification, regression, etc. Algorithms like Support Vector modeling (SVM), K-means, k-nearest neighbors (k-NN), to name a few algorithms and statistical models, are used to perform a specific task effectively. Deep learning can be classified as another subset within machine learning, where it makes use of neural networks rather than statistical models and algorithms to achieve a task. The unique feature of deep learning is that it need not be programmed specifically to perform a task; rather, the neural networks use multiple layers to progressively extract higher-level features from the raw input data [47]. Deep learning architectures such as Artificial Neural Network (ANN), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, bioinformatics, drug design, medical image analysis, material inspection and board game program, where the produced results are in comparable to and in some cases far superior to human experts.

Machine learning algorithms and in particularly deep neural networks thrives in situations where a structural relation between input and output is presumably present but unknown [30]. Many training samples and the computing power to train and deploy these algorithms is available. The emergence of these technologies has given rise to an extraordinary interest in these algorithms and their applications. An overview of deep learning is given in the review article, LeCun et al. (2015) [31] and Scmidhuber (2015) [32].

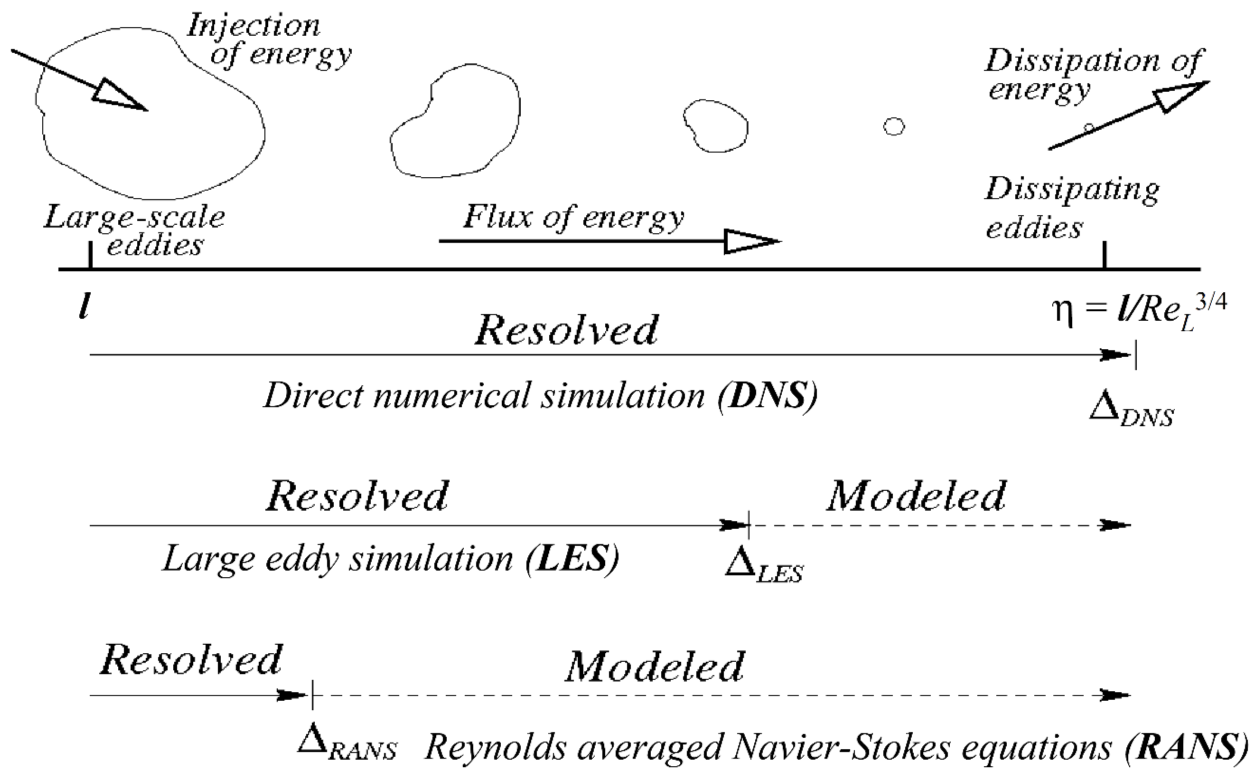


Figure 2: The figure shows resolution of different numerical methods (André Bakker) [http://www.bakker.org/dartmouth06/engs150/10-rans.pdf]

Deep learning neural networks working is similar to the neurons to the biological neurons in the brain; this makes it very flexible in its application to a variety of fields with different forms of data inputs. Deep neural networks can be applied in fields which has to work across different verticals like building a sophisticated model based on computer vision data, statistical data, and acoustic data, and producing a conclusive result from it. This kind of technology has seen its application in non-destructive testing of materials, which has produced results far more superior to existing technology. Applying deep neural networks on raw data and extracting higher-level features for data interpretation and making concise decisions from it is the most popular form of its application. The same structure of deep neural networks can be applied to computational fluid dynamics data and utilizing this pool of knowledge to improve its effectiveness.

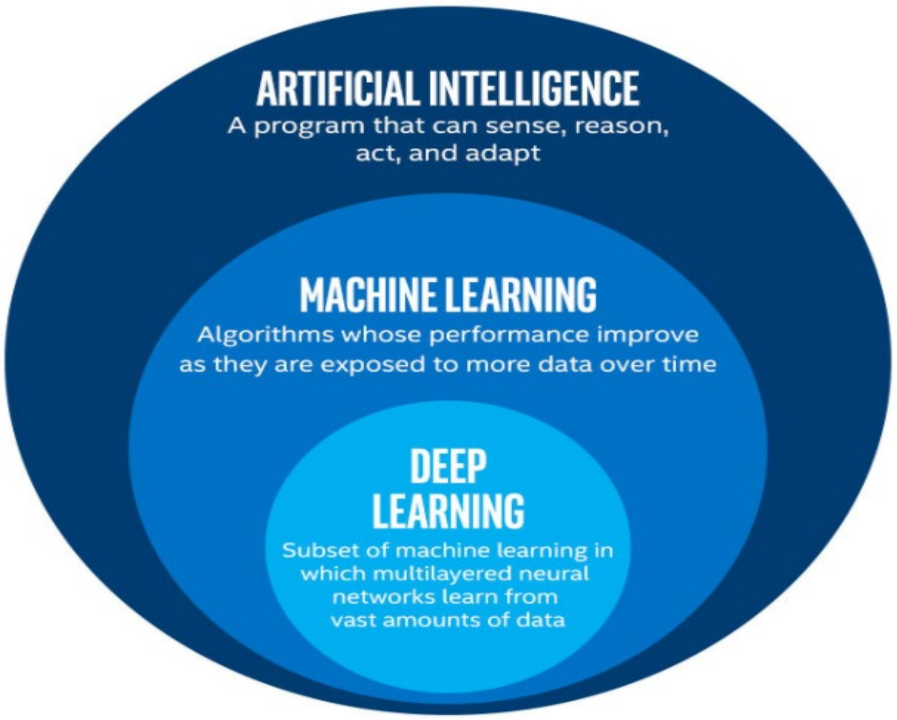

ARTIFICIAL INTELLIGENCE
A program that can sense, reason, act, and adapt
MACHINE LEARNING
Algorithms whose performance improve as they are exposed to more data over time
DEEP LEARNING
Subset of machine learning in which multilayered neural networks learn from vast amounts of data

Figure 3: Hierarchy of artificial intelligence, machine learning and deep learning

1.3 Problem

The number of CFD cases that we run is exactly proportional to the processing power available. The number of cases that we need not run because of a trained neural network can vastly exceed the number of cases required to run. We can train a neural network using 1000 cases to avoid running a million simulations. This trained neural network becomes more reliable by adding the test simulations results generated back to its pool of knowledge in training. Though the longer vision of artificial intelligence is to provide an alternative to the solvers itself, the present technologies available can be used to strengthen deep neural networks in CFD model deduction. Neural networks can be used as an interpretation method; the main advantage of this in the CFD field is set up the interpretation problem is a problem in itself due to the high nonlinearity nature of turbulence results.

Artificial Neural Networks is identical in its structure to the biological neural network, where it can learn tasks by considering examples without being explicitly programmed to perform a task. ANN is based on a collection of nodes called artificial neurons. The weights in the neuron are adjusted to match the task output to its raw input data. Building an effective neural network for raw data is a challenge in itself. The number of nodes and layers in a neural network should have an optimum value for each case, depending on the scale of data. Weights in the neuron are adjusted from the algorithms used, like backpropagation, feedforward networks, and other learning algorithms. Choosing the right mix of these parameters will produce an effective artificial neural network which can be trained to produce accurate results.

A combination of different types of machine learning and deep neural network techniques can be used to build more robust models. The process of applying these models on the raw data has a huge impact on its effectiveness. A novel methodology in applying a combination of different techniques and through a specific process for optimization problems is proposed in this thesis. Data from designing a convective polymerase chain reaction [Shu] is used as raw data to build the neural network and test the new methodology proposed.

BACKGROUND

2.1 Convective Polymerase Chain Reaction (cPCR)

Polymerase chain reaction (PCR) is a novel technique to amplify a few copies of DNAs to a detectable level [1, 2]. PCR has already become common in biomedical research, criminal forensics, molecular archaeology, and so on [2]. The mechanism of DNA amplification using PCR is based on the three distinct temperature cycling processes. In general (figure 4), the hot temperature is maintained in the vicinity of the bottom of the capillary tube, separating a double-stranded DNA into two single-stranded DNA during the denaturation process (95-97°C) [2]. In the top region where the fluid is cooled down at approximately 50-60°C (Annealing step), primers bind themselves to the ends of the two single-stranded DNAs [2]. In the mid-section of the tube, enzymes react to DNA synthesis at 72°C [2]. Thus, the two single-stranded DNAs turn into two double-stranded DNAs [2].

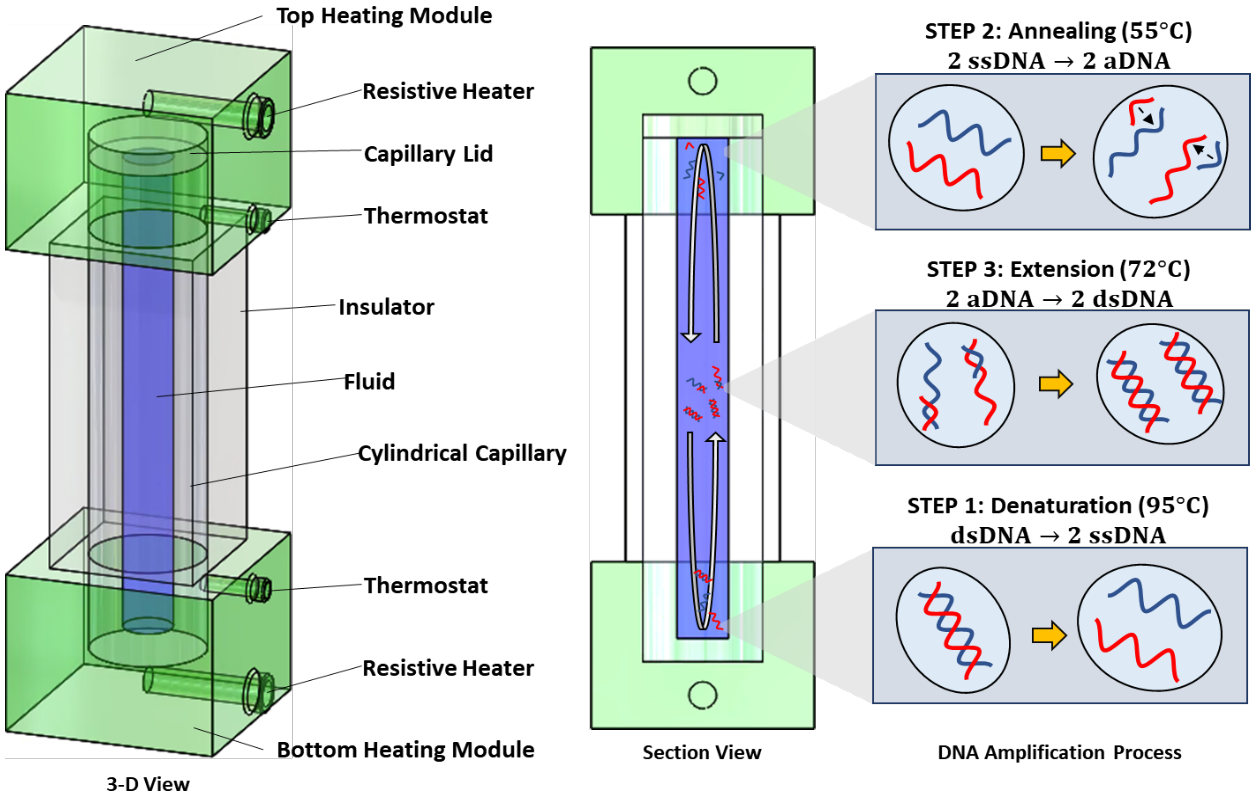


Figure 4: Convective Polymerase Chain Reaction (cPCR) device and DNA amplification process [35]

Many have attempted to develop PCR devices in numerous types for the purpose of the lab-on-chip (LOC) or point-of-care (POC). The use of PCR devices as POC testing is advantageous for the public health improvement by expanding the range of medical tests, and by providing rapid testing for immediate intervention for patients [3, 4]. To develop PCR devices used for POC testing, the price, size, and difficulty of operating the device should be lower, whereas the performance is guaranteed [3]. Generally, for the PCR devices, the high development cost results from adopting the methods which completely depend on experimentations. Computational methods are necessary to lower the dependency on the experimentations [2]. They can find the optimal design that assures the best performance without numerous experimentations.

2.2 Unsupervised Learning

Unsupervised learning is a type of "Hebbian Learning" where previously unknown patterns in data sets without pre-existing labels are identified [33]. Unsupervised learning draws inferences from the data for exploratory analysis, such as finding hidden patterns or grouping similar samples or separating distinct samples. Unsupervised, supervised, and reinforcement learning is the main three categories of machine learning. Unsupervised learning consists of Principal component analysis (PCA) and cluster analysis. Clustering analysis is widely used for exploratory data analysis to find hidden patterns or grouping in the data. Cluster analysis is further divided into a partition and hierarchical clustering. Hierarchical clusters is a set of nested clusters organized as a multilevel hierarchical tree. Partition clustering is dividing the samples in a dataset into non-overlapping subsets or clusters such that each sample is exclusively in one subset or cluster. K-means algorithm is an example of partition clustering where the data is split into *'k'* distinct clusters based on the measure of similarity, which is defined by the euclidean or probabilistic distance of a data sample to the centroid of a cluster. This k-means algorithm is used in this thesis for unsupervised learning application.

The general procedure for k-means clustering is as follows:

- *'K'* number of points are selected at random starting points as the initial centroids.
- *'K'* clusters are formed by assigning all points to the closest centroid.
- Centroid of each cluster is recomputed.
- This process is repeated until the centroids don't change.

Unsupervised learning is ambiguous, ie., it gives different ouputs for every run. Thus k-means clustering cannot give the best clustering, its only option is to keep track of these clusters, and their total variance, and do the whole thing over again with different starting points.

2.3 Elbow Method

Clustering consists of grouping objects in sets, such that objects within a cluster are as similar as possible, whereas objects from different clusters are as dissimilar as possible [41]. The idea is that one should choose several clusters so that adding another cluster doesn't give much better modeling of the data [40]. Thus, the optimal clustering is subjective and dependent on the characteristic used for determining similarities within-cluster and differences in partitions. For cluster analysis, the clusters are derived from Euclidian distance between points to determine inter and intra-cluster similarity. The following measure represents the squared sum of intra-cluster distances between points in a given cluster C_k containing n_k points:

$$D_k = \sum_{x_i \in C_k} \sum_{x_J \in C_k} \left\| x_i - x_J \right\|^2 = 2n_k \sum_{x_J \in C_k} \| x_i - \mu_k \|^2 \quad (1)$$

Adding the normalized intra-cluster sums of squares, also referred to as the within-cluster sum of squares (WCSS), gives a measure of the compactness of our clustering: $W_k = \sum_{k=1}^{k} \frac{1}{2n_k} D_k$.....(2)

This variance W_k is the basis of a naïve procedure to determine the optimal number of clusters: the elbow method. The optimal number of clusters can be defined as follows [42]:

1. Compute k-means clustering algorithm for different values of k, by varying k from 1-10 cluster.
2. For each k, calculate the total within-cluster sum of square (*W_k / WCSS*).
3. Plot the curve of WCSS according to the number of clusters k, as shown in figure 5.
4. The location of a bend (elbow) in the plot is considered as an indicator of the appropriate number of clusters.

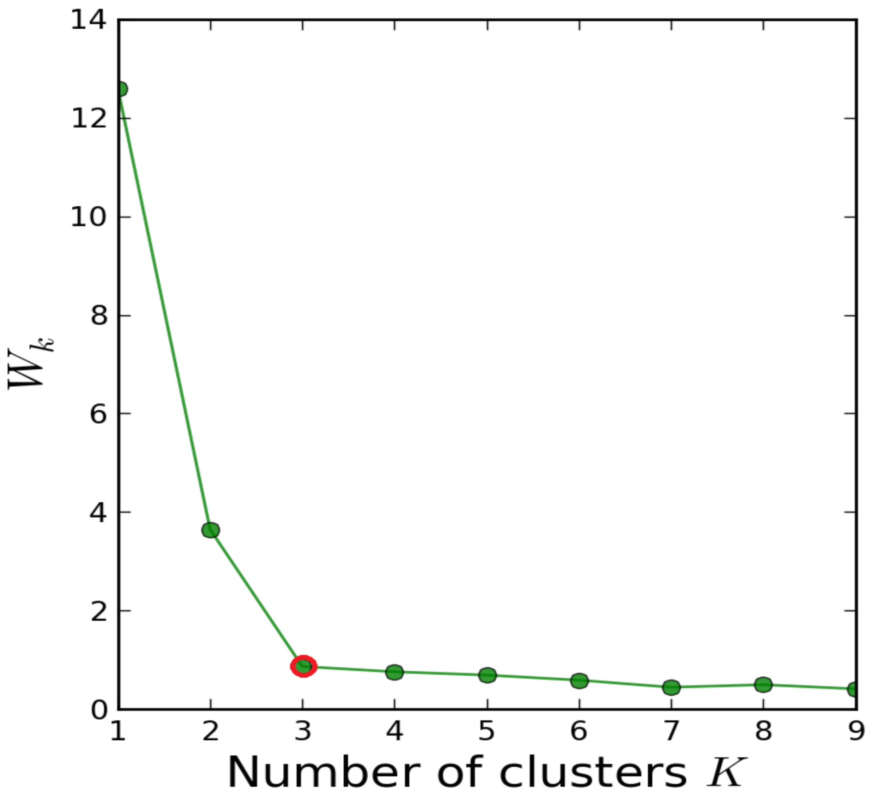


Figure 5: Optimum number of cluster by elbow method is indicated by the red circle.

2.4 Artificial Neural Network (ANN)

Deep-learning studies how to simulate the functions of a biological neural network; has produced intelligence, cognition, and responses. Artificial neural networks (ANN) are computing systems that are inspired by the biological neural networks that constitute animal brains. ANN simulates the connectivity in the computer neural systems through signals to mimic the massive parallel operations of the human brain. Artificial neural networks learn to perform a task by considering examples, without being exclusively programmed with any task-specific rules. It identifies characteristic features from the learning examples that they process and recognize these features on new data just like the neurological systems in the human brain [36].

The artificial neural network system is a collection of connected artificial neurons called nodes, which loosely model the neurons in a biological brain. Like the electrical impulses called synapses generated in biological brains to transmit information, the artificial neural network sends a signal from one node to another to process information and to signal additional nodes connected to it.

The signal through the nodes in the artificial neural network is a real number, and the output of each node is computed by a non-linear function of the sum of its inputs. These connections between nodes are called 'edges.' The nodes and edges in an artificial neural network typically have weights that adjust as learning proceeds. The weights increase or decrease the strength of the signal at a node. Nodes can have a threshold such that the signal is sent only when the threshold is crossed. Nodes are aggregated into layers to form a network; different layers may perform different kinds of transformations on their input signals. Signals from nodes are generated at the input layer and travel through multiple layers to the final output layer. This traversing through layers occurs multiple times in a feedback propagation mechanism of learning.

Multi-layer perceptron (MLP) is a type of feed-forward neural network. They are a class of models that are formed from layered nodes with activation function *(f)*, such as sigmoidal, rectifier, etc. These activation functions can realize any logical function based on the samples it is trained on. MLP are commonly trained using gradient descent on a mean squared error performance function, using a technique known as error back-propagation to compute the gradients. Multi-layer perceptron is the commonly used artificial neural network to make predictions and classification on numerical data. MLP is used on the the numerical data from cPCR design in this thesis, it is also referred as artificial neural network.

The inception of an artificial neural network was to solve problems in a generalized fashion like a human brain. However, today application of neural networks has moved to perform specific tasks, unlike what it was meant to perform. The artificial neural network has been used in a variety of specialized tasks like computer vision, speech recognition, machine translation, social filtering, playing games, and medical diagnosis. Though the endeavor to achieve a human intelligence level of the neural network is still vivid, and far from reality, ANN has found its application in performing specialized tasks.

2.5 Cross-Validation

Cross-Validation is a resampling procedure used to evaluate machine learning models on a limited data sample. It allows us to compare different machine learning methods and get a sense of how well they will work in practice. While training the machine learning models on training and testing datasets, the testing dataset may not be random, and a particular group could be left out while training. To overcome this possibility cross-validation method is used to estimate the skill of a machine learning model on unseen data. This method results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test splitting of data [38].

The general procedure of cross-validation is as follows:

1. Shuffle the dataset randomly.
2. Split the training dataset into k groups, hence the name k-fold cross-validation.
3. Set aside a one group as a validation set.
4. Train and fit a model on the remaining dataset or the training dataset.
5. Evaluate the model on the validation set.
6. Record the evaluation score on its accuracy over the training and validation dataset.
7. Repeat this by setting aside different groups as validation dataset.

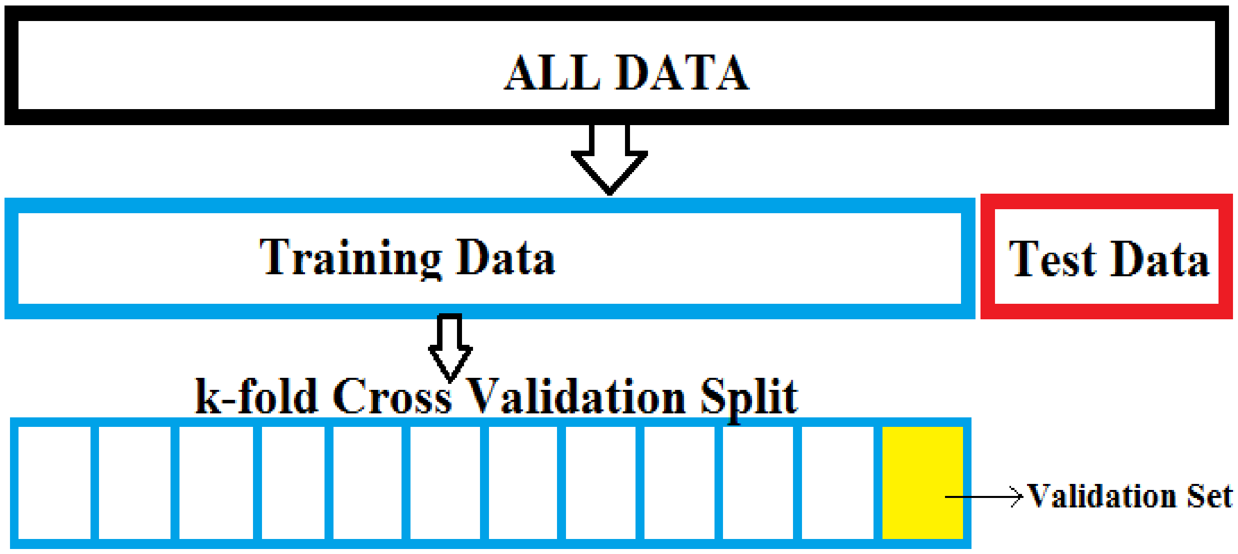


Figure 6: Visual representation of training, testing and validation split

The test data, which is usually around 10% -20% is kept aside and not exposed to the machine learning model. This set is only used for testing the machine learning model with unseen data to verify its accuracy. The training dataset is further split into a training set and validation set, as shown in figure 6. In k-fold cross validation data is split into k different subsets (or folds). Data is trained on k-1 subsets (training data) and is tested on last subset (validation data), this process is repeated by considering different subsets as validation data in each cycle or epoch [39] as shown in figure 7. The model is averaged against each of the folds and then finalized. After this, the model is tested against the test data, which was kept aside in the initial stage for model accuracy on unseen data.

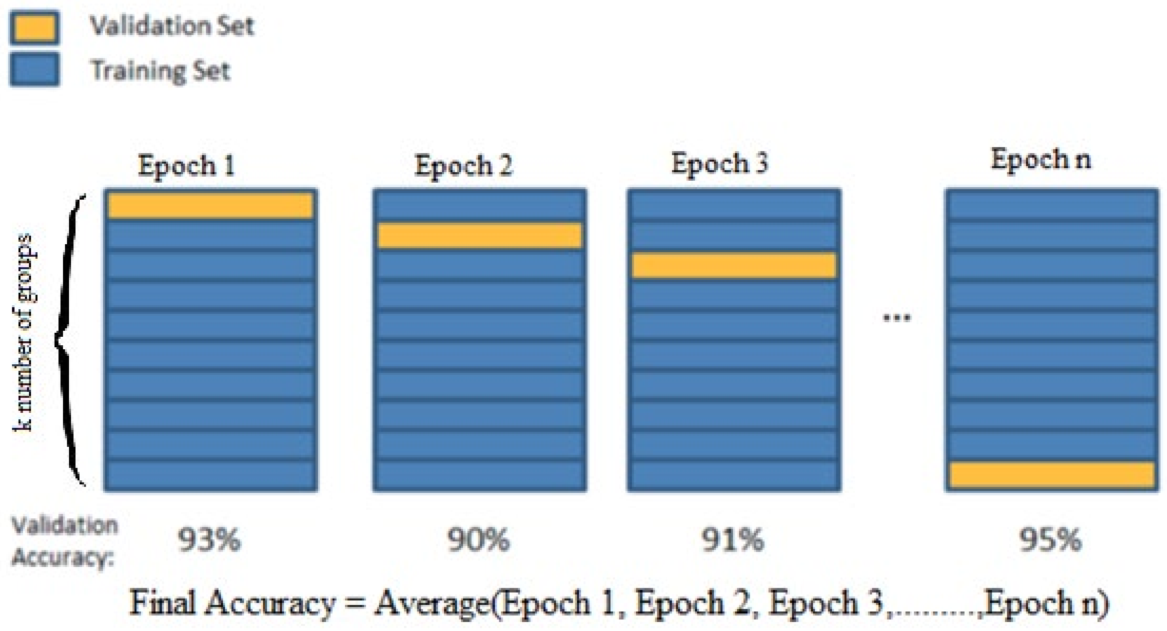


Figure 7: Visual representation of k-fold cross-validation

2.6 Grid Search for Tuning Hyperparameters

Hyperparameters are characteristics of a model that is external to the model and whose value cannot be estimated from data. The values of hyperparameters have to be set before the learning process begins. For example, k in k-means, the number of nodes in neural networks are hyperparameters. In contrast, a parameter is an internal characteristic of the model, and its value can be estimated from data, like the coefficients in linear and logistic regression. Grid-search is used to find the optimal hyperparameters of a model which results in the most accurate predictions. It builds a model for every combination of hyperparameters specified and evaluated each model. A more efficient technique for hyperparameters tuning is the sklearn's "GridSearchCV" [43], here the grid of hyperparameters to search over is defined and then algorithm is run to optimize the hyperparameters.

2.7 Differential Evolution

Differential evolution (DE) is a very simple population-based stochastic function minimizer, which is very powerful at the same time. DE is the best genetic type of algorithm for solving the real-valued test function suite of the First International Contest on Evolutionary Computation (1stICEO) in 1996. The algorithm used here is by Rainer Storm and Kenneth Price [45], it is a very powerful algorithm for black-box optimization (also called derivative-free optimization). Black-box optimization is about finding the minimum of a function where we don't know its analytical form, and therefore no derivatives can be computed to minimize it [46]. DE works better in those problems where other techniques like gradient descent cannot be used. Figure 8 shows how the DE algorithm approximates the minimum of a function. DE optimizes a problem by maintaining a population of candidate solutions and creating new candidate solutions by combining existing ones, and then keeping whichever candidate solution has the best fitness for optimization. In this method, the problem is treated as a black-box that merely provides a measure of quality given a candidate solution, and the gradient is therefore not needed. Neural networks create black-box like models, which makes differential evolution the best-suited optimization method.

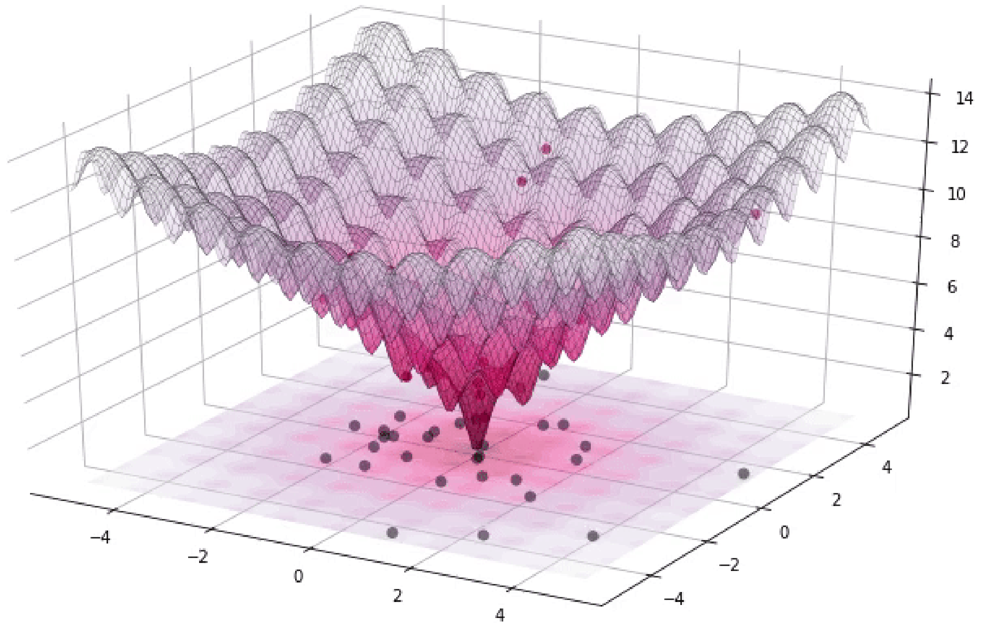


Figure 8: Example of differential evolution algorithm approximating the minimum of a function in successive steps [46]

CFD DATA COLLECTION

3.1 Introduction

The computational method used here in the design and development of Convective Polymerase Chain Reaction device is computational fluid dynamics (CFD). Krishnan et al. (2002), and Chen et al. (2004) utilized CFD to provide an insight into the buoyancy-driven flow, including velocity and temperature fields [5, 6]. Li et al. (2016) studied the flow conditions of several geometries for PCR capillary reactor design using CFD, and Qiu et al. (2019) computationally analyzed the flow changes inside the PCR reactor in the vertical and horizontal positions [1]. Yariv et al. (2005) developed a mathematical model for DNA amplification, applied it to a very simple nondimensional geometry, and showed its potential to be used for the estimation of the PCR performance [7]. Allen et al. (2009), Muddu et al. (2011), and Shu et al. (2019) adopted Yariv's mathematical model and applied it to more practical problems for DNA amplification [2, 8, 9]. These studies made attempts to partially or fully replace parts of the PCR experimentation with the computational methods using CFD, showing exceptional results in point of analysis.

Computational-based design perspective studies are demanded to fulfill PCR devices for POC testing. An optimal PCR reactor design can be found using combined methodologies of CFD with optimization. A direct method of CFD-based design optimization may find an optimal design with high accuracy. However, it is computationally expensive since the optimizer calls a CFD-based function multiple times at every iteration to estimate the direction of the optimum point [10]. The surrogate-based optimization approach is effective for computationally expensive problems [11]. Approximation techniques are used to reduce the order of magnitude of a set of data, so surrogate models are cheaper to run [10, 11].

In this research, we computationally develop a convective PCR (cPCR) reactor in which the convection is induced by buoyancy-driven force. A convective PCR reactor we develop is in contact with double heaters on the top and the bottom, as depicted in Fig. 9. A resistive heater is inserted into a heating module made of aluminum, transferring heat over the entire cylindrical tube. The module targets to maintain $55°C$ on the top and $95°C$ on the bottom using the thermostats attached to the sides. The cylindrical tube is enclosed by an insulator to restrain it from heat loss. The tube is filled with PCR reagents, amplifying a few copies of DNA samples according to the DNA amplification process.

3.2 Computational Method for Convective PCR Reactors

We consider the 3-D conjugate heat and momentum transfer equations for the fluid and solid domains. We assumed that the PCR reagents used for the numerical simulation are a type of fluid (pure water) of which the attributes are Newtonian, steady-state, incompressible, and laminar. The fluid is governed by continuity, momentum, and energy equations, as seen in [2, 15]. We use the thermal properties in a polynomial form, a function of temperature, to produce flow driven by buoyancy force. The heat transfer in the solid domains is solved by the conduction equations, as shown in [2, 15]. Since the cylindrical tube is made of polymethyl methacrylate (PMMA), we use the constant thermal properties of PMMA. An unsteady concentration species model (also known as convection-diffusion-reaction equations) is utilized to evaluate the performance of a convective PCR reactor as denoted in [2, 7, 8].

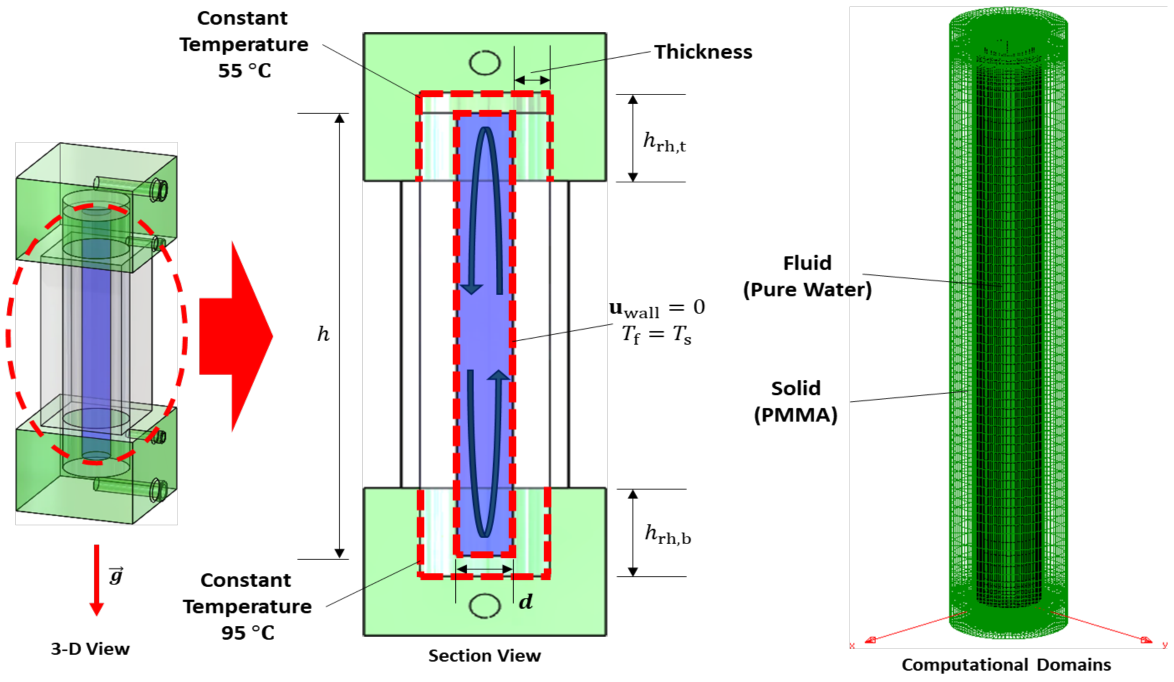


Figure 9: Boundary conditions and computational domains a cPCR [35]

The boundary conditions are given, as shown in figure 9 and described as follows:

1) No-slip flow velocity condition at the wall, $u_{wall} = 0$;
2) Isothermal condition at the interface between the fluid and solid domains, $T_f = T_s$;
3) Constant temperature $T = 95°C$ and $T = 55°C$ on the bottom and top of the tube, respectively;
4) Zero temperature gradient on the outer surface of the tube due to the insulator, $\nabla T = 0$;

5) The concentrations of the double-stranded, single-stranded, and annealed DNAs are initially given as 100, 0, and 0, respectively.

Once the nonlinear, partial derivative equations are solved for the species concentrations, the doubling time can be computed as follows:

$$t_d = \frac{ln\ ln\ (2)\ \Delta t}{ln\ ln\ \left(\frac{c_{ds,t=t_f}}{c_{ds,t=0}}\right)} \tag{3}$$

where, Δt is the duration, and $c_{ds,t=t_f}$ and $c_{ds,t=0}$ denote the concentrations for the double-stranded DNA at the final and initial time, respectively. The DNA doubling time is utilized to evaluate the performance of convective PCR reactors. The simulation time of the unsteady mathematical model is set to 30 minutes, so that, the doubling time should neither exceed it nor yield negatively.

3.3 Computational Fluid Dynamics (CFD) Simulations

The design space is established by simulating a total of 625 design candidates via CFD based on the mathematical models. The velocity and temperature fields are obtained via the simulation, utilized for the unsteady species transport model. The doubling time of the individual designs is revealed, as shown in figure 10.

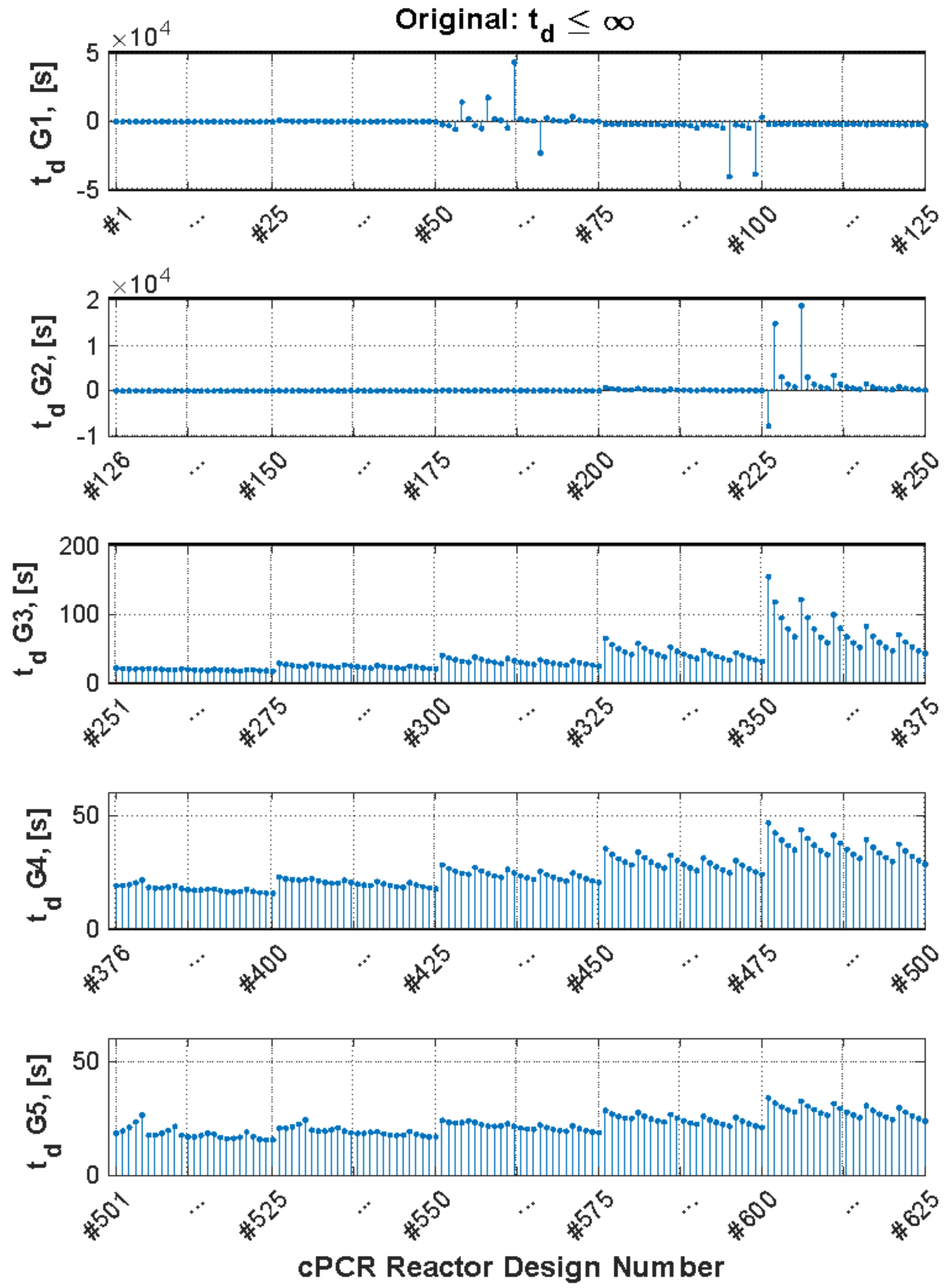


Figure 10: Contour plots of velocity and temperature for the randomly selected cases [35]

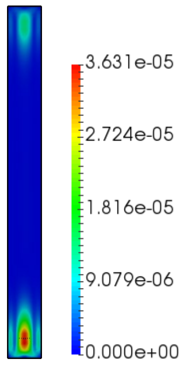
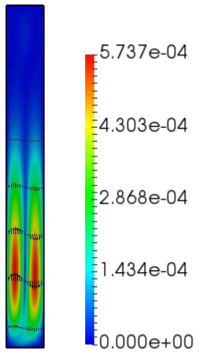
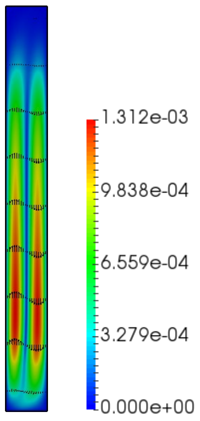
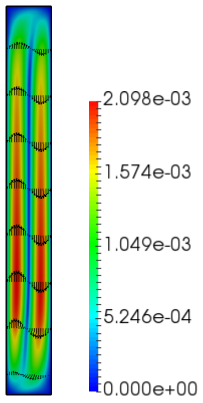
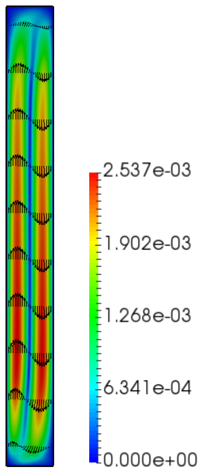
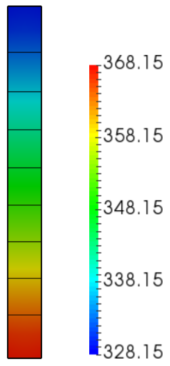
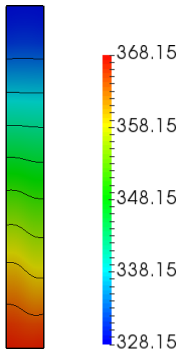
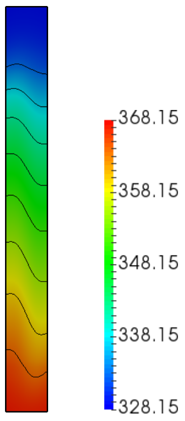
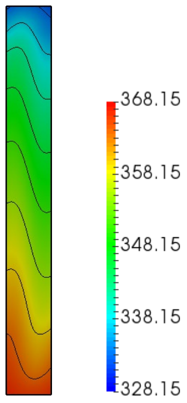
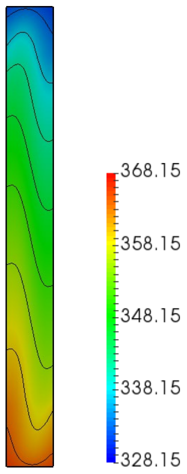
Design	**#101**	**#211**	**#373**	**#455**	**#614**
Group	**G1**	**G2**	**G3**	**G4**	**G5**
***d* (mm)**	1.3	1.4	1.5	1.6	1.7
h (mm)	13.0	12.6	15.0	14.4	17.0
$h_{rh,t}$ (%)	7.0	11.0	15.0	7.0	11.0
$h_{rh,b}$(%)	7.0	7.0	15.0	15.0	13.0
Velocity					
Temperature					

Figure 11: Contour plots of velocity and temperature fields for the randomly selected cases [35]

Most design candidates are deemed to be normal in performance based on their obtained doubling times. Presented in figure 11 are the contour plots for the velocity and temperature fields of a randomly selected case of each respective group. For example, Design #101 is one of the representative cases of which the performance is not satisfactory. Its velocity contour shows no flow convection inside the cPCR reactor.

The temperature contours suggest that the heat is transferred in the form of conduction, even though the temperature is appropriately-distributed over the entire area, proper for the DNA thermal cycling process. This results in no DNA amplification; and that's the reason that the doubling time is very high, 1,937.95 sec. For Design #211, a small, weak single convection loop is generated, affected by existing forms of heat transfer: convection and conduction. The convection is observed in the vicinity of the bottom, while the heat transfer is in the form of conduction above the convection loop. The performance is determined to be very low as the doubling time is 377.54 sec. Designs #373, #455, and #614 have a fully developed single convection loop, and their temperature gradients are observed to be sufficient to induce the required flow convection. The doubling times are computed to be 52.49 sec, 28.35 sec, and 26.65 sec, respectively. The candidates with acceptable performance display similar velocity and temperature contours to these candidates. It appears that the diameter is one of the factors that majorly determine the reactor performance since the majority of the candidates in G1 and G2 as seen in Fig.10 have the abnormal performance, i.e., negative or very large values of the doubling time.

The maximum temperatures are confirmed to be 92.15 °C, and 92.85 °C, respectively. The minimum temperatures are measured to be 58.05 °C, and 56.85 °C, respectively. They are deviated a little bit from the ideal temperature for PCR thermal cycling processes due to the nature of the conduction process in the solid domains.

3.4 CFD Cases Before ANN

A total of 625 data sets is generated for design parameters diameter, aspect ratio, top heater height, bottom heater height as input data, and time as output data. This data needs to be analyzed, and an underlying relation is to be modeled for optimization. Using machine learning for this purpose makes the job easy, and the data utilized here can be carried forwarded for future development also.

METHODOLOGY

4.1 Introduction

Enormous numerical data generated in designing of Convective Polymerase Chain Reaction (cPCR) device is utilized in demonstrating the new methodology proposed in this thesis. Deep Learning comes very handily in analyzing a large amount of data. Neural networks are made to learn from the raw data without explicitly being programmed to do the task. This generalized approach neural networks of training from large data of a complex problem and building a model on the phenomena make it viable for its application in a large variety of streams. Neural networks can be applied to any scale of data form a million to a few tens by preprocessing the data effectively and changing the structure of neural networks to the data size. Artificial Neural Networks (ANN) also known as Deep Neural Network (DNN) is a type of deep learning which can work on numerical data to find a correct mathematical manipulation to turn the input into the output. The nature of the data we have generated in cPCR design and development numerical data that makes ANN the correct type of deep learning to demonstrate our methodology. Artificial Neural Network (ANN) can be the best option for the reduction of an order [12]. It works as a neuron, creates and modifies new connections in the network depending on the task it is used for.

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses [34]. Unsupervised learning is used on exploratory data analysis to find hidden patterns or grouping in the data. Clustering or cluster analysis is a commonly used unsupervised learning technique. The task of grouping a set of data in such a way that data in the same group, called cluster, are more similar to each other than those in a different cluster. Use of clustering is a key aspect of the proposed new methodology.

Clustering techniques are used to achieve higher accuracies of ANN-based surrogate models [13]. In general, two kinds of clustering algorithms are widely used: (1) Hierarchical clustering and (2) k-means clustering. The former uses a complicated data division method, so it is slower in computation. It is challenging that the appropriate number of clusters is precisely determined using the hierarchical method. On the other hands, the k-Means method utilizes the Elbow method to quantify the clusters needed for the surrogate model [13, 14]. Technically, the Elbow method is needed where interpretational consistent cluster analysis is required to find the appropriate number of clusters for the given dataset [14]. Through the comparison of the number of clusters against the computational cost, the best cluster count can be determined as feed for the neural network. Even though it may not be precise, the number of clusters can be determined with low computational efforts. This novel technique is capable of adapting into any complex phenomena, including various principles of fluid dynamics. Especially for CFD simulations, the cost of computation efforts can be dramatically reduced by employing ANN as well.

Based on the aforementioned model, we produce a total of 625 design candidates for a cPCR reactor, simulate them using a high-fidelity CFD analysis tool, develop surrogate models using the artificial neural network (ANN), and optimize it for the best cPCR reactor design. Then, we validate the best result with CFD data. By doing this, we avoid running multiple CFD processes. This reduces the cost and time in the development of the refined design of the device. We also use a novel method of deep learning to make sure the surrogate model is accurately modeled. This new method involves using unsupervised machine learning techniques in building a surrogate model through ANN.

4.2 Design Space Investigation

We consider various design candidates to construct the design space. The investigation is carried out using CFD simulations for a total of 625 design candidates, which are obtained by combining sets of values of the selected parameters. These are the diameter, aspect ratio (height divided by diameter), heater heights (top and bottom) (Table 1).

From the literature, commonly used sizes of the parameters for a cPCR capillary tube is selected. All of the design candidates are grouped by the diameter into five groups (g1 to g5), consisting 125 candidates per group sorted by the design parameters in the order of the bottom heater height, the top heater height, and the aspect ratio. The design number is assigned to all of the candidates in the same manner. Once the design space is established by the CFD simulations, the performance of each of the individual design is evaluated; then the data is utilized to develop an ANN-based surrogate model that can replace CFD simulations.

Design Parameters	Variable	Unit	Lower Limit	Upper Limit	Step-Size
Diameter	d	Mm	1.3	1.7	0.1
Device Height	h	Mm	7.8	17.0	0.1
Top Heater Height	$h_{rh,t}$	%	7	15	2.0
Bottom Heater Height	$h_{rh,b}$	%	7	15	2.0

Table 1. Various parameters in design variables of cPCR.

4.3 Modeling Neural Network Architecture

Surrogate models are developed by customizing the algorithms from TensorFlow [16] to ensure higher accuracy of the approximation. We use a single neuron model consisting of three different functions: synaptic weight, summing junction, and transfer function. Initially, the model takes the inputs and weighs them in the synaptic weight level.

Then, the weighted inputs are summed up with a bias; the product is called 'net inputs' (the summing junction level). In the final level, the transfer function takes the net inputs, calculating the output of the single neuron. This can be mathematically expressed as follows [17, 18]:

$$y(\vec{p}) = f\left(\sum_{i=1}^{R} w_i p_i + b\right) \tag{4}$$

Where p is the vector of inputs with R elements, w is the weighted value, b is the bias, and f is the transfer function. Combinations of the neurons creates a neural network.

A wider neural network [24] is designed, with more nodes and fewer layers using 'Keras' library [22], this was chosen due to limited four input parameters. A preferred number of nodes was determined by using a "GridSearchCV" algorithm from sklearn to determine the appropriate number of nodes, optimizer, epochs, and loss function for higher accuracy of the model. The hyperparameters tuned for the optimized model are epochs, optimizer, batches, number of nodes. A common practice of a number of nodes used is (2*n) +1, where n is the number of input parameters, results in 9 nodes. Thus we give a range of 9-20 nodes for grid-search. For an accurate model for non-linear problems, more number of hidden layers are used in the neural network design [44], we repeat this grid-search process with 3,4 & 5 hidden layers. Table 2 shows the grid of hyperparameters to search for tuning. A Neural network with an input layer of shape 4 (for input parameters) and three hidden layers of nodes 17, one output Node, epoch 1000, batches 5 and Adam as the optimization algorithm was obtained from the grid-search algorithm with mean squared error loss function as -375.916 for the combination over 625 datasets. Later during the training of the neural network, dropout regularization is used to avoid over-fitting the data. In the dropout technique, a randomly selected neuron is temporarily removed on the forward pass, and no weights are updated during back-propagation [21].

Hyperparameter	**Range**
Epochs	100, 500, 1000
Optimizer	adam, rmsprop
Batches	5, 10, 15, 20, 25
Number of nodes	8 – 20

Table 2: Range of hyperparameters for grid-search

Shown in figure 12 is the neuron architecture customized for this problem, which consists of one input layer, three hidden layers, and one output layer [19]. The input layer takes the cPCR shape parameters, such as inputs, diameter, aspect ratio, bottom heater height, and top heater height. For a nonlinear model, all of the single-neuron models in the hidden layers utilize the 'Rectified Linear Unit' activation function defined as follows [20]:

$$y = max(x, 0) \tag{5}$$

The output layer has one single neuron with a linear transfer function to estimate an output parameter, the doubling time. The neural network is trained by updating the weights of each neuron after each cycle by 'method of backpropagation' [21].

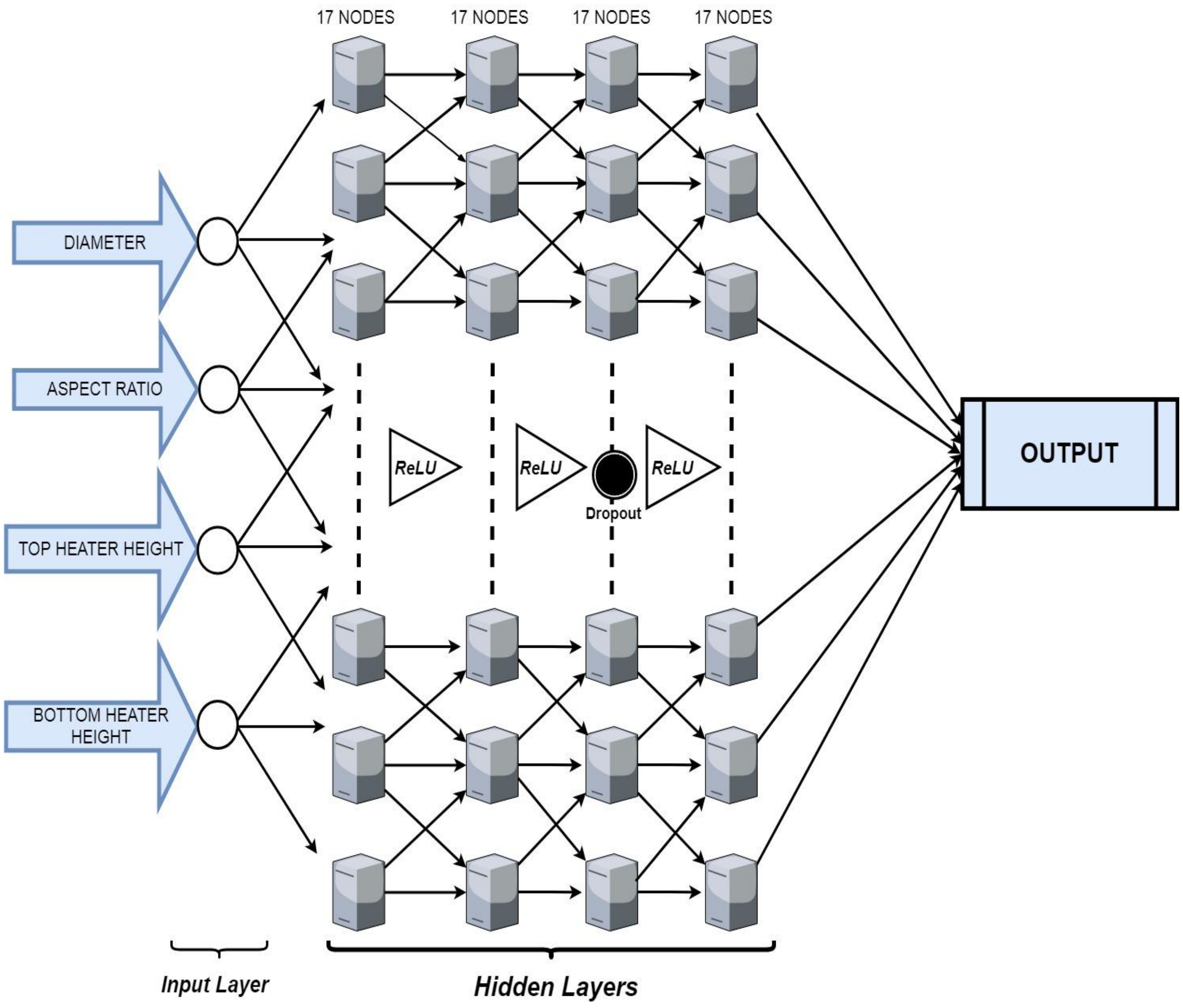


Figure 12: Customized feed-forward network for this problem with four hidden layers

4.4 Overall Methodology

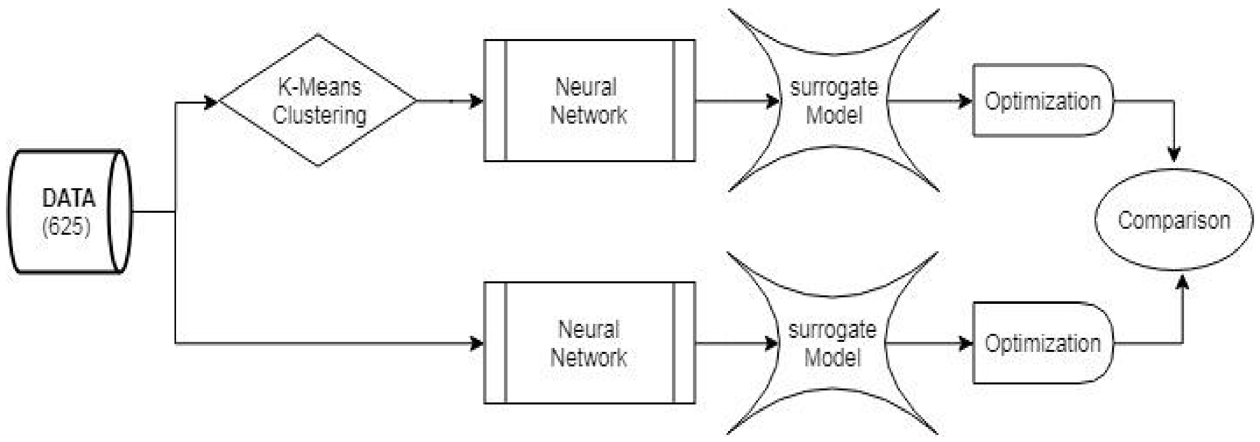


Figure 13: Process for high-accurate surrogate models using the k-Means clustering approach

A process for developing high-accurate surrogate models using the k-Means clustering method is established, as seen in figure 13. The process collects a total of 625 designs of the cPCR shape parameters and their doubling time obtained as solutions of the 3-D conjugate heat and momentum transfer model and the species transport model. The cPCR shape parameters and doubling time are set as the inputs and target variable for the neural network training, respectively. It would also take much time if all 625 data are processed for the neural network training; hence, we include the k-Means clustering in the process to reasonably eliminate some of the data unconducive to the accuracy of surrogate models.

4.5 Direct Neural Network

The data are given as inputs to the neural network in two different types. One is that inputs are provided for the neural network directly, and the second is, as aforementioned, to feed the appropriate cluster [26] to the neural network by performing the k-Means clustering. After a surrogate model is built by both ways, optimization is performed on the surrogate models, separately. The optimized results from both surrogate models are compared to draw conclusions.

4.6 Assisted Neural Network

The novel method proposed here is the method of clustering the data. Generally, for the k-Means clustering, the number of clusters is ambiguous, and there is no definite approach to concluding.

There are few methods of determining the number of clusters based on the computational cost of clustering called, Elbow method, here the cost of clustering increases as the quantity of clusters increases. This method does not consider the nature of the data considered or the task, which is to be achieved by clustering. This method also becomes limited as the quantity of data to be clustered where the computational cost is trivial. The proposed method gives a new guideline for clustering by throwing light on how the number of clusters affects the data and, by knowing the task to be achieved through clustering, we would be able to make a better choice of the cluster count, making it less ambiguous and more deterministic. This method enables the surrogate model to achieve a better fit with higher accuracy. The model shown in Fig. 14 & 16 is used in this step.

4.6.1 Clustering

We develop a surrogate model, which takes the inputs as diameter, aspect ratio, top heater height, and bottom heater height, and outputs the doubling time. In order to narrow our focus on the minimization, we ruled out the CFD-delivered data with negative or too high doubling time. Typical values for the doubling time is expected to be 30~60 sec.

4.6.2 Elbow Method

The process for data clustering, which groups data into smaller sets, is employed as a strategy for higher accuracy of a surrogate model. The CFD-derived dataset (total 625) is clustered using the aforementioned k-Means method. A "within-cluster-sum-of-squared-errors" (WCSS) curve is produced using the Elbow method as a function of the number of clusters, as shown in figure 14.

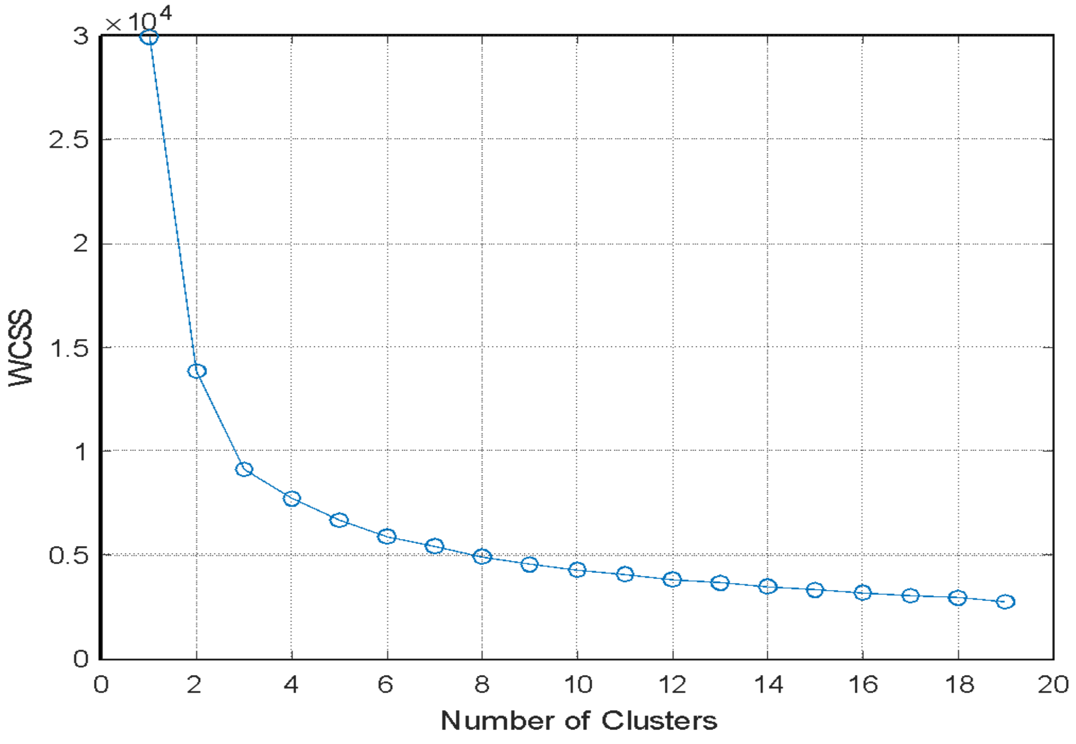


Figure 14: Relation between the number of clusters and WCSS (Within Cluster Sum of Squared Error) when the Elbow method is applied

4.6.3 Clustering for Present Methodology

As our interest is in minimization of the surrogate model for minimum doubling time, we consider the cluster with lower bound to fit the surrogate model from neural network. From the Elbow method, it can be observed that after the cluster quantity 3, the gain in the computational cost is marginal; this is usually chosen as the appropriate cluster quantity. We can further take this as the initial cluster quantity to proceed with the new technique of determining a more precise cluster count. This method can determine the suitable cluster quantity where the data can be divided into groups of similar characteristics. For minimization problems, the lower bound is fixed. Cluster analysis is performed for different cluster count until the upper bound within the desired cluster containing the minimum doubling time becomes constant. This process is visualized in figure 15, where the blue dots is the desired cluster containing the lower output value (minimum doubling time in our case). For example, if cluster count N = 6 to 9, the cluster with blue dots is unaltered. This signifies a distinctive characteristic of the cluster. This methodology is applied to our problem of cPCR device.

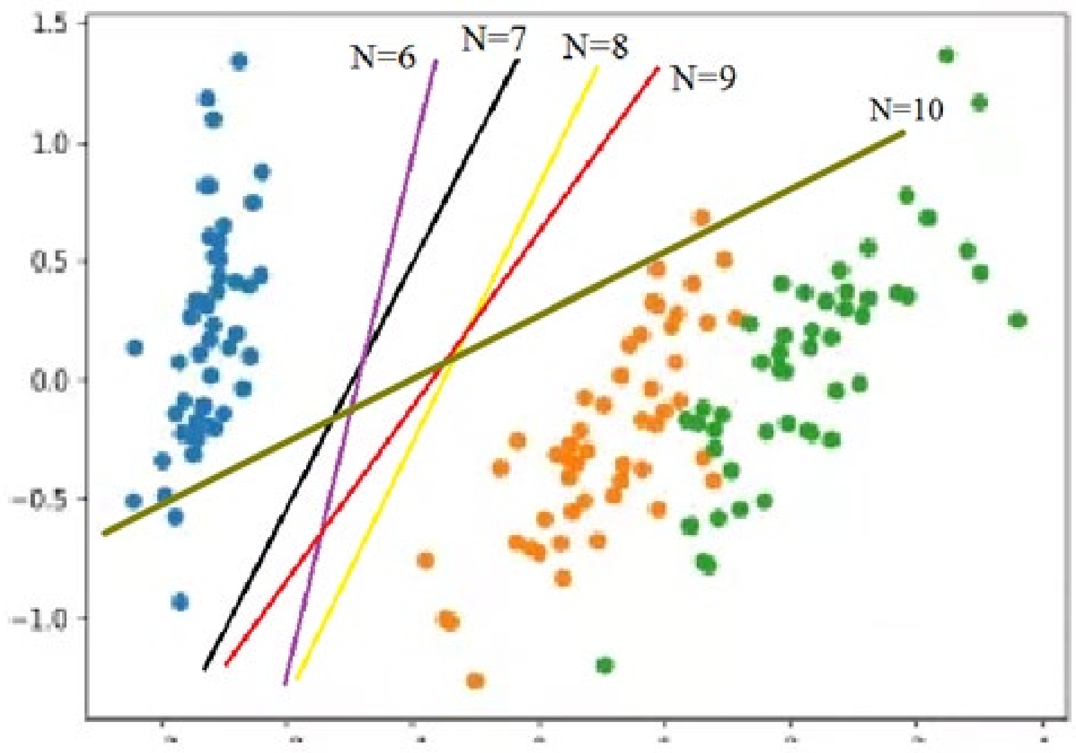


Figure 15: Visualizing clustering through the proposed methodology

The detailed analysis is made to ensure a suitable number of clusters. As presented in Table 3, we mainly investigate the relation between the upper bound within the cluster of interest and the number of the samples in that cluster as unique features in determining the optimum cluster count. Table 3 is visualized through figure 16 to determine the number of appropriate clusters. We find out that the variation of the maximum range within the cluster attained a constant value of 30.105 after the cluster quantity seven. The corresponding number of samples in the cluster is 56 when the quantity of clusters is seven; beyond this, the change in the number of samples changes marginally. As the lowest number of clusters is attained for all the distinct data, it is considered that the computational cost and the distinct nature of the group cannot be marginal beyond the Elbow method. This novel approach gives us the cluster quantity in a more reliable for the k-Means method. We select the cluster by the region of interest (minimum doubling time) to develop high accurate surrogate models. This method of clustering also enables us to avoid noise by outliers and from different characteristic set of data affecting the surrogate model.

Number of Clusters (k-Means)	Lower bound (Required cluster)	Upper bound (Required Cluster)	Number of Samples in the Cluster
3	22.851	34.37	162
4	22.851	32.65	108
5	22.851	32.467	84
6	22.851	30.26	70
7	**22.851**	**30.105**	**56**
8	22.851	30.105	51
9	22.851	30.105	50

Table 3. Number of clusters and the corresponding number of samples from CFD-derived data.

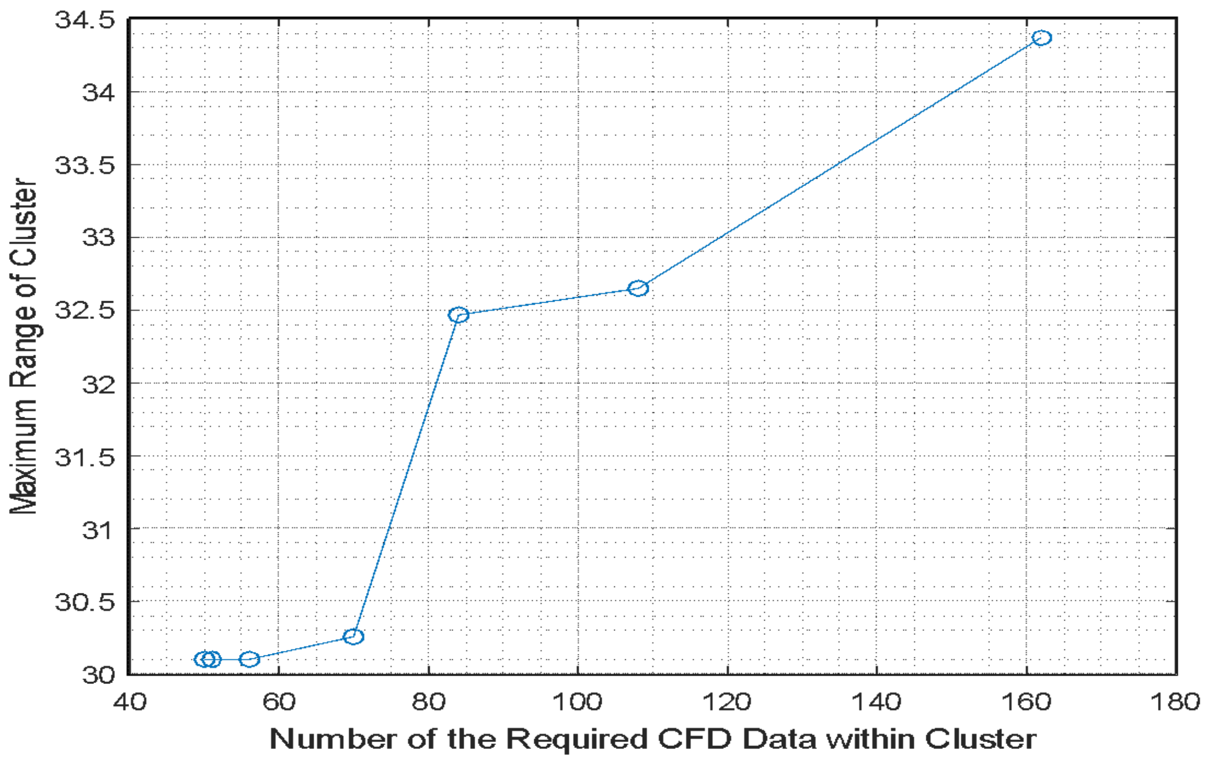


Figure 16: Variation in range within-cluster and the corresponding number of the required CFD data within-cluster at the different quantity of clusters

The 56 data samples in the cluster are used for grid-search in determining the optimized hyperparameters for the assisted neural network over the same range as done for the previous direct neural network. A neural network with similar hyperparameters was obtained with a mean squared error of -1.3245. This optimized value is ambiguous, which means a different value may be obtained if the process is repeated. But both neural networks are trained separately to get a good fit for each model.

4.7 Neural Network Training:

As the number of clusters and the number of samples in a cluster are determined, a total of 56 CFD data of interest are utilized for training the neural network. Both the neural networks are trained on the same architecture. The data is split into training and testing datasets. The testing dataset is 10% and is kept aside unseen to the neural network; it is used to test the fully trained model to check its prediction accuracy. Initially, the epoch size of 100 is considered for training with 10% of it as the validation set. By plotting the graph of loss function of validation and training set during the training of the model as shown in figure 17, changes are made to the epoch size to avoid over-fitting or under fitting of the model. Finally, the epoch size of 50 is determined as the appropriate balance for the model between over-fitting and under-fitting.

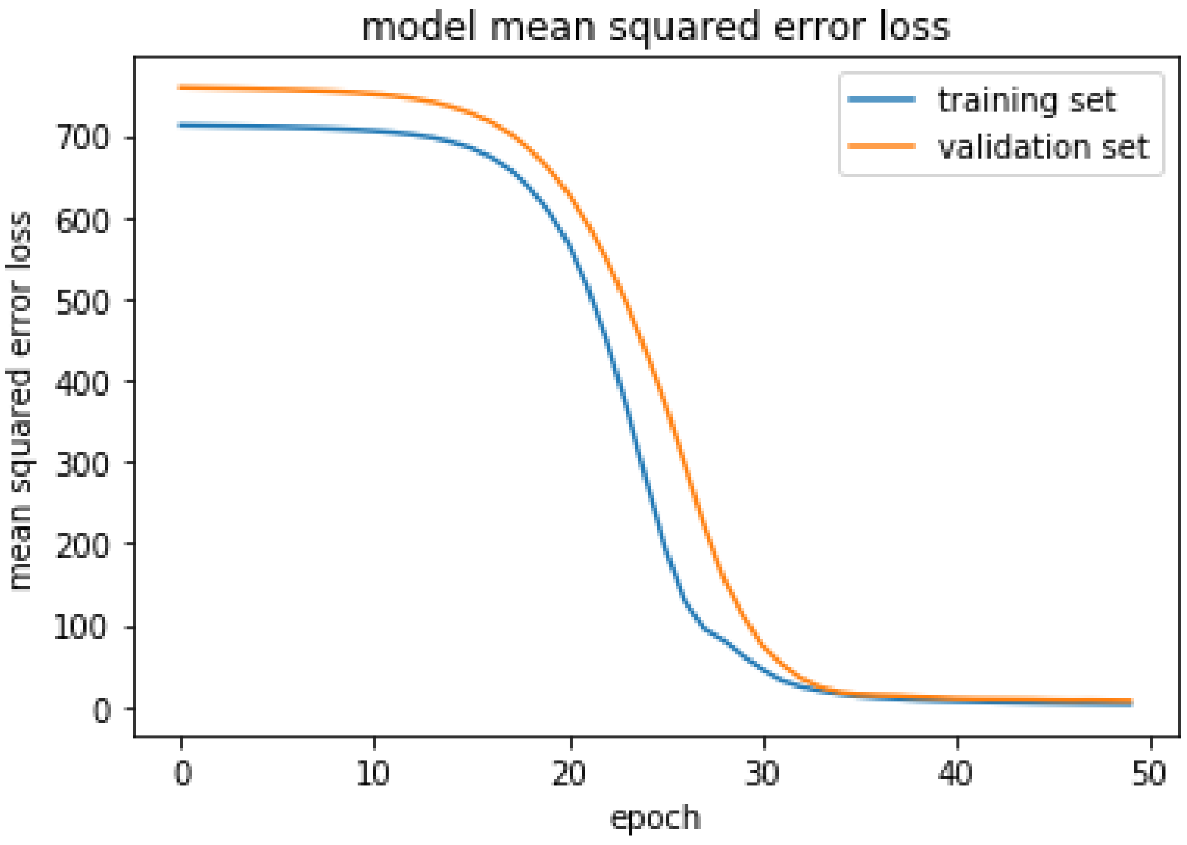


Figure 17: Neural network training history that tracks mean squared error with respect to epoch

The k-fold cross-validation approach [25] is used to predict the model with higher accuracy. Fig. 16 represents the history of the neural network training, which describes the variation of the mean-squared error (MSE) with respect to epoch. The best validation performance is determined to be MSE = 0.386 at the epoch 50 with by introducing a dropout function in the last hidden layer. The surrogate model is developed based on these parameters. This fully trained model is put to the test with an unseen testing dataset, as shown in figure 18. The doubling time of the 56 designs, which is predicted by the surrogate model, is compared with the doubling time of the corresponding design points from the CFD data used for the network test. The model accuracy relies on how well the model prediction data fits the CFD data used for the network test.

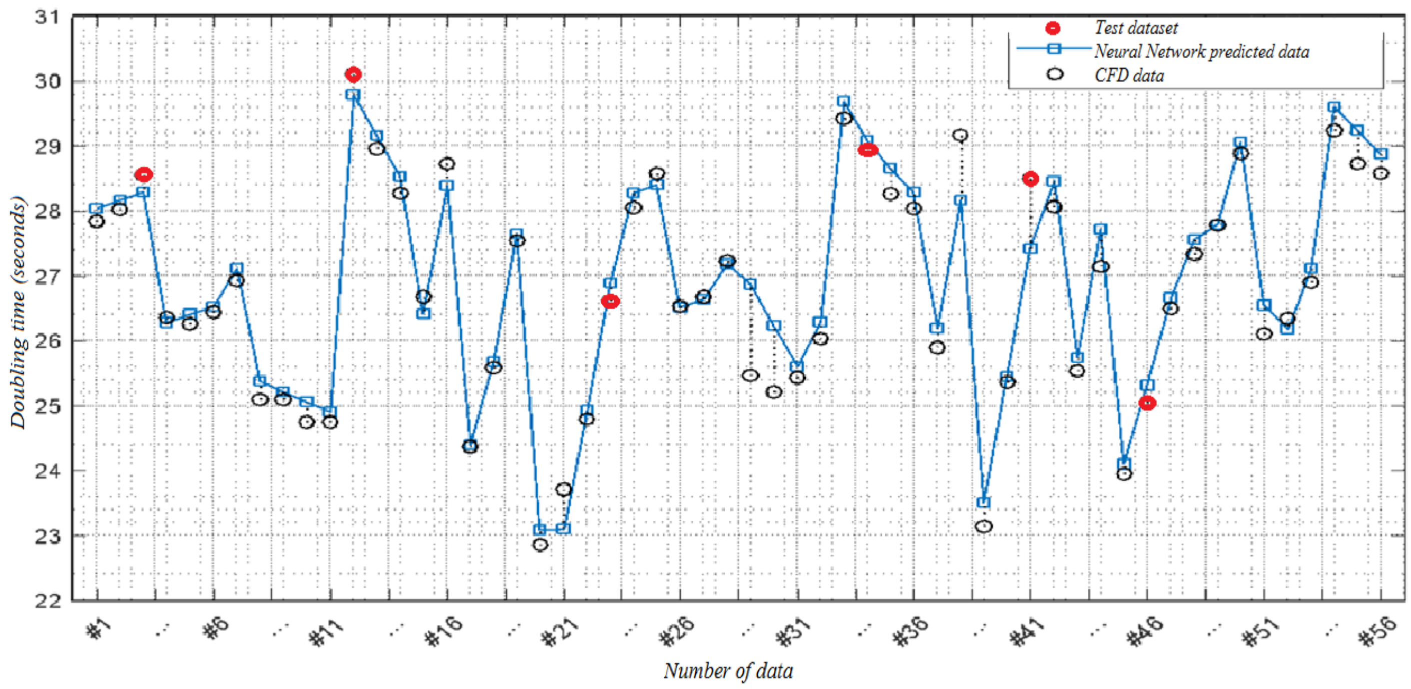


Figure 18: Comparison of model prediction (56 samples) with CFD data for neural network test

4.8 Optimization

Surrogate-Assisted Design Optimization and Verification

The surrogate models are connected with an optimization algorithm (differential evolution) to find the optimal shape of a cPCR reactor. The optimal results are compared to investigate the effect of the k-Means clustering method. One surrogate model is developed by k-Means, and the other is built based on the conventional direct ANN training process. A design optimization problem is formulated as follows:

Minimize t_d

Find $\vec{x} = \{d,\ h,\ h_{rh,t},\ h_{rh,b}\}$

Subject to

$$g_1 \ \rightarrow\ 1.3 \le d \le 1.7$$

$$g_2 \ \rightarrow\ 6.0 \le h \le 10.0$$

$$g_3 \ \rightarrow\ 7.0 \le h_{rh,t} \le 15.0$$

$$g_4 \ \rightarrow\ 7.0 \le h_{rh,b} \le 15.0$$

The objective function is to minimize the doubling time (t_d); it is one of the major factors related to the performance. The design variables are comprised of the shape parameters like the diameter (d), device height (h), top heater height ($h_{rh,t}$), and bottom heater height ($h_{rh,b}$). The box constraints (g_1, g_2, g_3, g_4) are used to ensure the optimal design to be found in the design space considered. The algorithm used here is the "differential evolution optimization" [23] that employs a heuristic method for global optimization in TensorFlow (Keras) [16, 22].

RESULTS AND DISCUSSION

Presented in Table 4 are the results from the optimization and verification of two different methods: (1) with clustering (assisted NN), (2) without clustering (Direct NN). The outcomes are compared with the best sample as a reference, which has been used to train the network for the surrogate models. The concentration species model estimates the doubling time of the cPCR designs obtained via the optimization of the surrogate models. For the design corresponding to the one from Surrogate Model 1, the CFD simulation estimates the doubling time to be 20.54 sec. The difference is 7.98%, which is deemed acceptable.

For the one obtained from Surrogate Model 2, the doubling time is estimated to be 24.22 sec. The difference is 13.96 %, which is larger than that of Surrogate Model 1 (the clustering method). Comparing the doubling time of three cases in Table 4, the clustering method even shows better performance than that of the best sample. The clustering method gives better results than the non-clustering method, and the CFD simulation assures the accuracy of the surrogate models. In this sense, the result from the surrogate model developed using the clustering method is considered as verified.

Parameters	**Unit**	**Surrogate Model 1 (Assisted NN)**	**Surrogate Model 2 (Direct NN)**	**Best sample in CFD data input**
Diameter	mm	1.636	1.68	1.6
Device Height	mm	9.816	13.692	9.6
Top Heater Height	%	14.86	14.94	15.0
Bottom Heater Height	%	7.09	7.22	7.0
Doubling Time (Surrogate Estimated)	seconds	22.18	27.60	-
Doubling Time (CFD Results)	seconds	**20.54**	**24.22**	**22.85**
Difference between surrogate estimated and actual CFD results	%	7.98	13.96	-

Table 4. CFD results for verification and comparisons

Surrogate Model 1 of which the clustering method is adopted, estimates the optimal doubling time to be 22.18 sec. For Surrogate Model 2 of which the clustering method is not applied, the optimal doubling time is estimated as 27.60 sec. The CFD simulations are carried out to validate the optimal doubling time estimated by Surrogate Models 1 and 2. The 3D conjugate heat and momentum transfer model confirms the existence of convection in the fluid domains for both cases, as seen in figure 19. The velocities of the cases are at most 1.635 mm/s, and 1.624 mm/s, respectively.

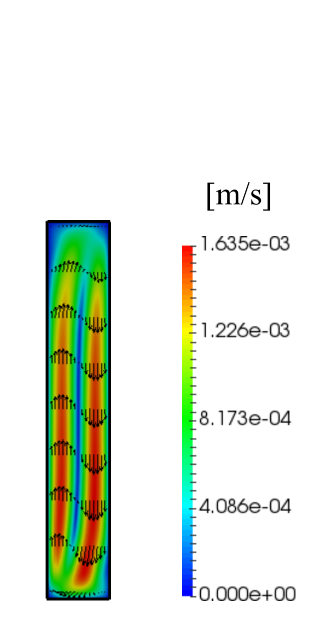
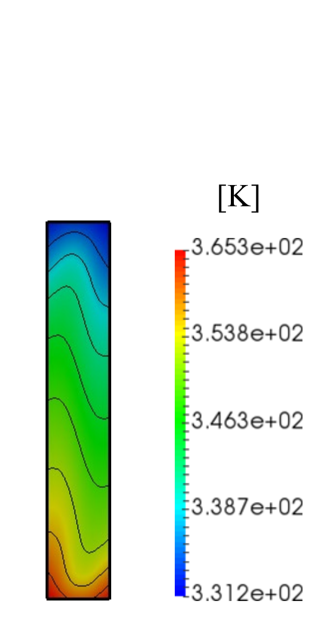
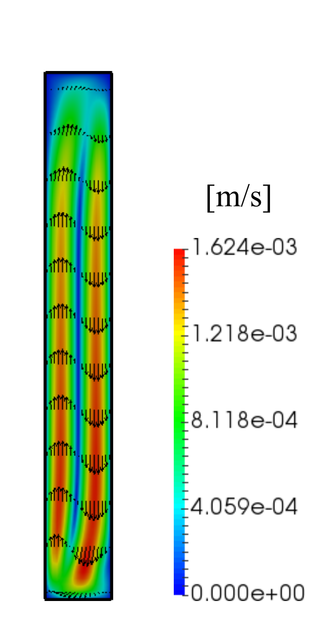
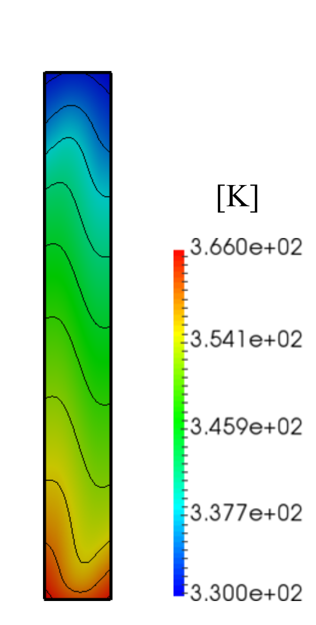
			
(a)	(b)	(c)	(d)

Figure 19: CFD results for validation of surrogate models: (a) velocity field of Model 1; (b) temperature field of Model 1; (c) velocity field of Model 2; (d) temperature field of Model 2

CONCLUSIONS

The doubling time is obtained via CFD simulations for all design candidates that feature various quantities in the design variables, which are in our case are the diameter, height, top heater height, and bottom heater height. The data is utilized to develop surrogate models for cPCR optimization. The flexibility and robustness of ANN have shown to be beneficial for fitting surrogate models for the analysis of the complicated problem that uses the CFD data. The k-Means clustering method is adopted to improve the surrogate model by training the neural network that utilizes a specific cluster of the CFD data. Our present methodology is a novel one that logically determines the appropriate cluster quantity required.

Two separate differential evolution (DE) methods for optimization are implemented; one with and the other without clustering. Both sets of optimal design results are compared with the CFD benchmark, that is, the best performing one of the entire CFD data set. It is observed that the surrogate model with clustering gives better results. The best design from this clustered surrogate model produces an even better-performing design than those CFD-produced samples, which are used for the neural network training. In conclusion, it is demonstrated that the present methodology, an unsupervised-learning assisted ANN, using k-Means clustering, is efficient in optimizing designs for cPCR reactors. As future work, this methodology can be applied to design other medical devices and processes.

The proposed methodology proves to be efficient in design & development of any process or instrument using machine learning, cPCR reactors in this case. The suggested method has the potential to dramatically reduce the cost and time in design & development by lowering the dependency on the expensive developmental and operational experimentations. It has proven that the best performance can be found rapidly by adopting the unsupervised-learning assisted ANN.

REFERENCES

[1] Li, Z.Q., Zhao, Y., Zhang, D.W., Zhuang, S.L., and Yamaguchi, Y., *The development of a portable buoyancy-driven PCR system and its evaluation by capillary electrophoresis.* Sensors and Actuators B-Chemical, 2016. **230**: p. 779-784.

[2] Shu, J.-I., Baysal, O., Qian, S., Qiu, X., and Wang, F., *Performance of convective polymerase chain reaction by doubling time.* International Journal of Heat and Mass Transfer, 2019. **133**: p. 1230-1239.

[3] Sia, S.K. and Kricka, L.J., *Microfluidics and point-of-care testing.* Lab on a Chip, 2008. **8**(12): p. 1982-1983.

[4] Niemz, A., Ferguson, T.M., and Boyle, D.S., *Point-of-care nucleic acid testing for infectious diseases.* Trends in Biotechnology, 2011. **29**(5): p. 240-250.

[5] Chen, Z.Y., Qian, S.Z., Abrams, W.R., Malamud, D., and Bau, H.H., *Thermosiphon-based PCR reactor: Experiment and modeling.* Analytical Chemistry, 2004. **76**(13): p. 3707-3715.

[6] Krishnan, M., Ugaz, V.M., and Burns, M.A., *PCR in a Rayleigh-Benard convection cell.* Science, 2002. **298**(5594): p. 793-793.

[7] Yariv, E., Ben-Dov, G., and Dorfman, K.D., *Polymerase chain reaction in natural convection systems: A convection-diffusion-reaction model.* Europhysics Letters, 2005. **71**(6): p. 1008-1014.

[8] Allen, J.W., Kenward, M., and Dorfman, K.D., *Coupled flow and reaction during natural convection PCR.* Microfluidics and Nanofluidics, 2009. **6**(1): p. 121-130.

[9] Muddu, R., Hassan, Y.A., and Ugaz, V.M., *Chaotically Accelerated Polymerase Chain Reaction by Microscale Rayleigh-Benard Convection.* Angewandte Chemie-International Edition, 2011. **50**(13): p. 3048-3052.

[10] Mack, Y., Goel, T., Shyy, W., and Haftka, R., *Multiple Surrogates for the Shape Optimization of Bluff Body-Facilitated Mixing*, in *43rd AIAA Aerospace Sciences Meeting and Exhibit*. 2005, AIAA: Reno, Nevada.

[11] Hacioglu, A., *Fast evolutionary algorithm for airfoil design via neural network.* Aiaa Journal, 2007. **45**(9): p. 2196-2203.

[12] Alsmadi, O.M.K., Abo-Hammour, Z.S., and Al-Smadi, A.M., *Artificial neural network for discrete model order reduction with substructure preservation.* Applied Mathematical Modelling, 2011. **35**(9): p. 4620-4629.

[13] Solomatine, D.P. and Shrestha, D.L., *A novel method to estimate model uncertainty using machine learning techniques.* Water Resources Research, 2009. **45**.

[14] Kodinariya, T.M. and Makwana, P.R., *Review on determining number of cluster in k-Means clustering.* International Journal of Advanced Research in Computer Science and Management Studies, 2013. **1**(6): p. 90-95.

[15] Çengel, Y.A., Ghajar, A.J., and Kanoglu, M., *Heat and mass transfer : fundamentals and applications*. 5th ed. 2015, New York: McGraw Hill Higher Education. 968 pages.

[16] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Xiaoqiang, Z., *TensorFlow: A system for large-scale machine learning*, in *12th USENIX Symposium on Operating Systems Designing and Implementation*. 2016: Savannah, GA, USA. p. 265-283.

[17] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., *Dropout: A Simple Way to Prevent Neural Networks from Overfitting.* Journal of Machine Learning Research, 2014. **15**: p. 1929-1958.

[18] Dongale, T.D., Patil, K.P., Vanjare, S.R., Chavan, A.R., Gaikwad, P.K., and Kamat, R.K., *Modelling of nanostructured memristor device characteristics using Artificial Neural Network (ANN).* Journal of Computational Science, 2015. **11**: p. 82-90.

[19] Arce-Medina, E. and Paz-Paredes, J.I., *Artificial neural network modeling techniques applied to the hydrodesulfurization process.* Mathematical and Computer Modelling, 2009. **49**(1-2): p. 207-214.

[20] Dahl, G.E., Sainath, T.N., and Hinton, G.E., *Improving deep neural networks for LVCSR using rectified linear units and dropout*, in *2013 IEEE Internatioanl Conference on Acoustic, Speech and Signal Processing*. 2013: Vancouver, BC, Canada

[21] Karnin, E.D., *A simple procedure for pruning back-propagation trained neural networks.* IEEE Transactions on Neural Networks, 1990. **1**(2): p. 239-242.

[22] Zaccone, G. and Karim, M.R., *Deep learning with TensorFlow : explore neural networks and build intelligent systems with Python*. Second edition, fully revised and updated. ed. 1 online resource (1 volume).

[23] Storn, R. and Price, K., *Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces.* Journal of Global Optimization, 1997. **11**(4): p. 341-359.

[24] Gaurav Pandey, Ambedkar Dukkipati – *To go deep or wide in learning?,* 17th International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. JMLR: W&CP volume 33.

[25] Yoonsuh Jung & Jianhua Hu (2015): *A K-fold averaging cross-validation procedure*, Journal of Nonparametric Statistics, 27:2, p.167-179.

[26] Shuyue Wang, Gang Sun, Wanchun Chen, Yongjian Zhong - *Database self-expansion based on artificial neural network: An approach in aircraft design*, Aerospace Science and Technology, Volume 72, 2018, Pages 77-83, ISSN 1270-9638.

[27] Wagner C., Shishkin A., Shishkina O. (2011) *The use of Direct Numerical Simulations for solving industrial flow problems*. In: Kuerten H., Geurts B., Armenio V., Fröhlich J. (eds) Direct and Large-Eddy Simulation VIII. ERCOFTAC Series, vol 15. Springer, Dordrecht.

[28] Unmeel B. Mehta and Paul Kutler (1984), *Computational Aerodynamics and Artificial Intelligence,* NASA Technical Memorandum 85994, Ames Research Center, Moffet Field, California 94035

[29] Jeffrey Slotnick and Abdollah Khodadoust et al. (2014), *CFD Vision 2030 study: A path to Revolutionary Computational Aerosciences, prepared by* Langley Research Center under Contract NNL08AA16B.

[30] Andrea D. Beck, David G.Flad and Claus-Dieter Munz (2018), *Deep Neural Networks for Data-Driven Turbulence Models,* Numerics Research Group, Institute of Aerodynamics and Gas Dynamics, University Struugart, Germany, https://www.researchgate.net/publication/325737916.

[31] Lecun, Yann, Bengio, Yoshua & Hinton, Geofffrey (2015), *Deep Leaerning*, International journal of science - Nature 521, 436-444, doi:10.1038/nature14539

[32] Schmidhuber, Jrgen (2015), *Deep Learning in neural networks: An Overview,* Science Direct Neural Networks ,The Swiss AI Lab IDSIA, University of Lugano & SUPSI, Galleria 2, 6928, Switzerland, https://doi.org/10.1016/j.neunet.2014.09.003

[33] Wikipedia contributors. (2019, June 13). *Unsupervised learning*. In Wikipedia, The Free Encyclopedia. Retrieved 15:02, July 10, 2019, from https://en.wikipedia.org/w/index.php?title=Unsupervised_learning&oldid=901695593

[34] Unsupervised Learning - Matlab & Simulink - Mathworks. (n.d.). Retrieved from https://www.mathworks.com/discovery/unsupervised-learning.html

[35] Varun Kote, Jung Shu, Oktay Baysal (2019), Unsupervised-Assisted Artificial Neural Network for Convective PCR Device Optimization, Old Dominion University.

[36] Wikipedia contributors. (July 5, 2019). *Artificial neural network.* In Wikipedia, the Free Encyclopedia. Retrieved 23:13, July 11, 2019, from https://en.wikipedia.org/w/index.php?title=Artificial_neural_network&oldid=904965773.

[37] Minsky, Marvin; Papert, Seymour (1969). *Perceptrons: An Introduction to Computational Geometry*. MIT Press. ISBN 978-0-262-63022-1

[38] Jason Brownlee (May 23, 2018), *A Gentle Introduction to k-fold Cross Validation*, Statistical methods, Machine learning mastery.

[39] Gareth James, Daniela Witten, Robert Tibshirani (2013), *An Introduction to Statistical Learning: with Applications in R*, 1st edition 2013, 7th printing 2017 edition, Springer.

[40] Poornima Bolowalia, Arvind Kumar (2014), *EBK-Means: A Clustering Technique based on Elbow Method and K-Means in WSN,* International Journal of Computer Applications.

[41] Charrad, Malika, Nadia Ghazzali, Véronique Boiteau, and Azam Niknafs. 2014. *"NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set",* Journal of Statistical Software *61: 1–36. http://www.jstatsoft.org/v61/i06/paper.*

[42] Kaufman, Leonard, and Peter Rousseeuw. 1990. *Finding Groups in Data: An Introduction to Cluster Analysis*.

[43] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay (Oct 12, 2011), *Scikit-learn: Machine Learning in Python*, JMLR 12, pp. 2825-2830, 2011.

[44] Martin T. Hagan, Howard B. Demuth, Mark Hudson Beale, Orlando De Jesus (2014), *Neural Network Design* (2nd edition, chapter 22).

[45] Rainer Storm, Kenneth Price (1997), *Differential Evolution-A simple and Efficient Heuristic for global Optimization over Continous Spaces,* Journal of Global Optimization, Volume11.

[46] Pablo R. Mier (2018), *A tutorial on Differential Evolution with Python,* Web. 04 Aug. 2019 https://pablormier.github.io/2017/09/05/a-tutorial-on-differential-evolution-with-python.

[47] "*Guide To Deep Learning - Data Science Central".* N.p., n.d. Web. 04 Aug. 2019 https://www.datasciencecentral.com/profiles/blogs/guide-to-deep-learning.

VITA **VARUN KOTE**

EDUCATION

Master of Science in Mechanical Engineering, Mechanical and Aerospace department, Old Dominion University, Norfolk, Virginia, USA. Major Field: Mechanical Engineering. Research interests: Computational Fluid Dynamics (CFD), machine learning and its applications. Thesis: ***Unsupervised Learning Assisted Artificial Neural Networks for Optimization***. Chair: Dr. Oktay Baysal. August 2017 – August 2019.

Bachelor of Engineering, Mechanical Engineering, SIR M Visvesvaraya Institute of Technology, Bangalore, Karnataka, India. Academic Project: *Computational Fluid Dynamics Analysis of a Small Gas Turbine Nozzle*, under Principal Scientist Venkat Iyengar, National Aerospace Laboratories, Bangalore, India. August 2012 – June 2016.

RESEARCH EXPERIENCE

Graduate Research Assistant for National Institute of Aerospace and Embraer project, NIA, Hampton, Virginia, USA. Collaborated research project of computational fluid dynamics analysis of propellers using SU2 open-source software, January 2019 - August 2019.

COLLEGE TEACHING EXPERIENCE

Spring 2018	Engineering Design (MAE 332), Teaching Assistant under Dr. Elmustafa.
Fall 2018	Solid Mechanics (MAE 220), Teaching Assistant under Dr. R. Prabhakaran.
Spring2019	Fluid Dynamics (MAE 303), Teaching Assistant under Dr. Oktay Baysal.

PROFFESSIONAL DEVELOPMENT

CAE software

Ansys Fluent, Hypermesh, ANSA, Pointwise, COMSOL, SU2.

Program language

Python, C++.