Nonlinear Process Modelling and Optimization Control Using Computational Intelligence Techniques

Changhao Zhu

A thesis submitted for the degree of Doctor of Philosophy

School of Engineering
Newcastle University
United Kingdom

June 2023

Abstract

With the ever-increasing global competition and customer requirement, chemical industrial processes face increasing pressure of maintaining high product quality and reducing production costs. To reduce the production cost and limit the amount of off-specification products production, close control of polymer product quality is required. The melt index (MI) of polymer is an important product quality indictor that needs to be closely monitored and controlled. However, it is difficult to measure MI in real-time. To overcome these issues, a reliable data-driven soft sensor for MI using deep belief network (DBN) is developed in this study. The training of DBN involves two phases: initial unsupervised training using input data only and supervised fine-tuning. It can capture profuse information from latent operational inputs. It is shown that DBN gives better performance in estimating MI than conventional neural networks. To further enhance the accuracy and robustness of DBN models, a bootstrap aggregated deep belief network (BAGDBN) is developed in this study. Several DBN models are developed from bootstrap resampling replications of the original modelling data and are combined to form a BAGDBN model. The effectiveness of the proposed model is demonstrated on two application examples, inferential estimation of polymer MI in an industrial polypropylene polymerization process and dynamic modelling of water level in a conical water tank. It is shown that BAGDBN has a much better generalisation capability than a single DBN model.

A reliable optimal control strategy for a batch process is developed using BAGDBN model. To enhance the reliability of the optimal control policies, the model prediction confidence bound is incorporated into the optimisation objective function. Wide model prediction confidence bounds are penalised in order to enhance the reliability of the obtained optimal control policy. Application to a batch reactor demonstrates the effectiveness of the proposed reliable optimisation control strategy.

Acknowledgement

In the past four years, I spent most of my time living and studying in the UK. This valuable experience will not be forgotten and will inspire me in the future of my life. At this time, I would like to express my greatest appreciation and warm thanks to the supervisors, colleagues, and friends.

I would like to give my best appreciation to my supervisor, Dr Jie Zhang. He gives me very kind and patient guidance for my study and every question I asked him received valuable feedback. When I have problems with either research or life, I will have a talk with Dr Jie Zhang. His suggestions and encouragement help me get through a very hard time. Many thanks.

I would like to thank everyone who gave me help during the Covid-19 lockdown. Everyone suffered a hard time and finally, we made it through. Wish you stay healthy and happy.

I will give my special thanks to my colleagues, Ruosen Qi, Kai Liu and Xuhua Yan. They gave much help in the past four years. We had a great time. Wish you all the best in the future.

At last, I wish to thank my parents for their support and love to me.

Contents

List of Figures

List of Tables

Notations

Cp_i	molar heat capacity of component i ($kJ \cdot kmol^{-1}{}^{\circ}C^{-1}$)
F_j	flowrate ($m^3 \cdot min^{-1}$)
Hi	heat of reaction of reaction i ($kJ \cdot kmol^{-1}$)
J	objective function
k_i	rate constant for reaction i
k_i^1	rate constant 1 for reaction i
k_i^2	rate constant 2 for reaction i
Mi	number of moles of component i (kmol)
Q_r	heat released from reactions ($kJ \cdot min^{-1}$)
R	reactor
Sp	set point
T	reactor temperature (°C)
U	heat transfer coefficient ($kJ \cdot min^{-1} m^{2^{-1}}{}^{\circ}C^{-1}$)
V	reactor volume (m^3)
W	reactor content (kg)
x	independent value
y	dependent value
$\hat{Y}$	predictions

Greek Letters

α	intercept
β	model slope
ε	model residual
$\hat{\theta}$	model weights
ρ_j	density ($kg \cdot m^{-3}$)

Abbreviations

AI	artificial intelligence
ADALINE	adaptive linear network
ANFIS	adaptive neural fuzzy inference system
ANN	artificial neural networks
ARMA	auto-regressive moving average
BANN	bootstrap aggregated neural network
BACT-ANN	bootstrap aggregated classification tree neural network
BAGDBN	bootstrap aggregated deep belief network
Bi-LSTM	bi-directional LSTM
CD	contrastive divergence
CNN	convolutional neural network
CSTR	continuous stirred tank reactor
DBN	deep belief network
DRNN	deep recurrent neural network
FBR	fluidized-bed reactors
GNN	grouped neural network
IMC	internal model control
LSTM	long-short-term-memory model
MI	melt index of polymer
MSE	mean-squared error
MWCNT	multi-walled carbon nanotubes
NARMA	nonlinear auto-regressive moving average
NARX	nonlinear auto-regressive exogenous
NMPC	general nonlinear model predictive control
PCA	principal component analysis
PLS	partial least squares
RBM	restricted Boltzmann machine
RBPN	recurrent backpropagation neural network
STA	spatiotemporal attention

SQP	sequential quadratic programming
SSE	sum of squared errors
SVM	support vector machines
TE	Tennessee Eastman
XOR	exclusive-or (XOR) problems

Chapter 1. Introduction

1.1 Background

Modern chemical industries face great pressure of the increase of energy cost and strict environmental regulation. To improve the competitiveness of industries among the world-wide competitors, advanced process control technology becomes more and more important during industrial production. Industrial chemical process operability, cost reduction, energy efficiency, control performance and waste minimisation are all the key parts to be concerned in the past decades. With the great development of computing techniques, many efficient modelling techniques and successful applications of modelling are carried out for complex chemical processes.

As an important part of industrial process control, soft sensors attract much attention. Soft sensors are predictive models based on historical industrial chemical process operational data. Most of the quality data of products predicted by soft sensors are responsible for the product safety and these indispensable variables cannot be measured by the hardware measurements. Some important variables which affect the process are difficult to be measured online and the cost of hardware measuring instruments is too high (Tham et al, 1991). For example, the polypropylene polymerization process is a typical industrial process. Polypropylene is one of the most widely used products in the world. The characteristics of polypropylene products are tough. The products include packaging, films, healthcare, pipes, textiles, fibres, automotive and electrical applications. The products of polypropylene have hundreds of grades and almost ten thousand types. In such a highly nonlinear process, the polypropylene polymerization process is difficult to be modelled by using mechanistic modelling approaches. And the quality of products is sensitive to the operating conditions. The temperature of reactors, the conditions of feeding and mixing affect much in the process of production. In a single manufacturing process, many different grades of products need to be produced. Therefore, the grades transition needs to change the operating conditions. During the time of transition, many undesirable products can be produced. Therefore, the key quality variable melt index (MI) of the polymer needs to be measured to reduce the production of undesirable products. In practice, the MI of the polymer can be measured by the MI measuring device in the laboratory when the products are already finished. However, it cannot be monitoring online. This problem increases the cost of the whole industrial process. To overcome this issue, the approach of soft-sensor is a good choice to estimate the MI of polymer products. Modern chemical processes are highly automated due to the advanced process control systems. The research of chemical processes is

focused on the computational modelling, scale-up and optimisation of real chemical manufacturing processes. The operational process variables which affect the key quality attributes are measured and monitored to achieve the objective of quality control. Advanced control and supervision of industrial processes require accurate process models. Among the various types of process models, data-driven models are the most widely used for process monitoring and control applications.

Process models can typically be classified into three types, mechanistic model, data-driven empirical model and hybrid model. Mechanistic models are also known as first-principle models. The development of mechanistic models requires solid physical and chemistry knowledge of the specific chemical process, in the form of differential and algebraic equations. The advantage of mechanistic models is that they are valid for a wide range. However, the development of a detailed mechanistic model for a complex industrial chemical process, such as the polypropylene polymerization process, requires hundreds of equations need to be used. The computation with mechanistic models is also time demanding and they are usually not suitable for online optimisation even though the computing technique has been developed greatly in recent decades. Compared with mechanistic models, data-driven empirical models are easier to be developed. They are based on process operational data and experimental data. Their development does not need detailed physical and chemistry knowledge. Their independence on a priori knowledge makes data-driven empirical models much more popular than before for the industrial processes (Kadlec, Gabrys & Strandt, 2009; Kano & Ogawa, 2010). They are less computationally demanding than mechanistic models. Therefore, they are suitable for the online optimisation of industrial chemical processes. The third type of process model is the hybrid model. A hybrid model combines mechanistic model and data-driven model in one model. Hybrid models intends to combine the advantages of both mechanistic models and data-driven models. The development of hybrid models is often specific to the modelled process and depends on the available first principle knowledge and the process operational data.

Since the last century, numerous successful studies have been conducted on process modeling using multivariate statistical techniques. Principal component analysis (PCA) was proposed by Karl Pearson (Pearson, 1901) and was further developed by Harold Hotelling in the 1930s (Hotelling, 1933; Hotelling, 1992). Based on PCA, principal component regression (PCR) and partial least squares (PLS) have emerged as useful modelling methods to address the problem of co-linearity among the input variables. Herman O. A. Wold first introduced PLS and Svante Wold further developed it (Wold, Sjöström & Eriksson, 2001). As an improvement of PCR, PLS regression can model both the process data and quality data at the same time. There were

many applications based on the PLS technique in processes modelling. Data-driven soft sensors based on PCR can be developed by using principal components as the predictor variables (Zhang, 2001; 2006). One limitation of PLS and PCR is that they are both linear techniques. They are not very effective when applied to nonlinear process modelling.

With the development of machine learning, many studies on developing soft sensors based on machine learning techniques have been reported in the past few years. There are many successful process modelling techniques based on machine learning, such as support vector machines (SVM) and artificial neural networks (ANN). W. Pitts and W. McCulloch proposed the original neural network, called perceptron, in the 1940s (McCulloch & Pitts, 1943). After 20 years, with the vast improvement of computer capability, the neural network became a popular research topic. The back-propagation algorithm was applied to ANN by Werbos (Werbos, 1974). The advantage of ANN is that it can be used to approximate any nonlinear functions. ANN gives very good performance on estimation and prediction of quality data. The original perceptron contains just one layer and cannot solve the exclusive-or (XOR) problems due to it cannot deal with nonlinear separable problems. Multi-layer perceptron with hidden layer was proposed later. Trained by the backpropagation algorithm, a multi-layer perceptron neural network with at least one hidden layer can solve the XOR problem. The hidden layers in neural networks improve the network representation capability. However, conventional ANN suffers from problems of local optima and lack of generalisation capability. SVM can achieve accessible optima of training even if the amount of training data is small (Desai, Badhe, Tambe & Kulkarni, 2006). However, when apply SVM to processes with large amount of modelling data, the pressure of computation will increase. In 2006, Hinton first introduced deep learning method (Hinton, Osindero & The, 2006). deep belief network (DBN) is one kind of the most well-known data-driven modelling techniques based on deep learning. It shows strong generalisation capability in modelling highly nonlinear processes. This model is established with a deep architecture. Deep learning has many applications in speech recognition and images classification (Mnih & Hinton, 2009). It has shown significant performance in many other applications (Li et al, 2018; Yao, Bi & Chen, 2018).

Despite the availability of advanced models proposed, there are still many issues of development and applications for the industrial chemical processes that need to be researched and studied.

1.2 Motivation

For the SVM and conventional neural networks, they can approximate nonlinear processes. However, they can lack of generalisation capability due to the problems of overfitting and under-fitting (e.g. training terminated in a local optima). To enhance the stability and accuracy of the model, an advanced data-driven model, DBN, is carried out for the predictions of quality variables of highly nonlinear processes in this study. When modelling a specific industrial process, how to construct a reliable structure of a DBN model? How to balance the relationship between different amounts of input variables and output variables? How to improve the robustness of the empirical model? Can the DBN model be used for the optimisation of batch processes and how to optimise the model?

This thesis concentrates on the development of advanced data-driven empirical models based on deep learning techniques for the modelling and optimisation of industrial chemical processes. The novel data-driven models are developed and their applications are the focuses of this research.

1.3 Aim and Objectives

The aim of this research is to develop accurate and reliable computational intelligence-based data-driven models for nonlinear process modelling and optimal control. In order to achieve this aim, the following objectives are set:

1 Developing reliable data-driven models using deep learning methods, such as deep belief network (DBN) models, for chemical processes.

2 Investigation of methods for improving model reliability and generalisation capability through the aggregation of multiple DBNs (i.e. BAGDBN).

3 Integration of the above models with nonlinear process optimisation and control for chemical processes.

1.4 Contributions

This thesis contributes to developing data-driven models based on deep learning techniques and optimisation for chemical processes. Empirical models based on machine learning have had

many successful applications in the past few decades. This study focuses on developing reliable and efficient data-driven models for chemical processes. A DBN model has been developed and applied to an industrial polypropylene polymerization process. The performance of the DBN model has been compared with a conventional neural network. It demonstrates that DBN with a deep structure has better performance than a single hidden layer neural network.

A bootstrap aggregated deep belief network (BAGDBN) has been developed and applied to an industrial polypropylene polymerization process and a batch reactor process. The approach of bootstrapping improves the robustness and accuracy of the DBN model significantly. Significant optimal control performance improvement has been achieved through the incorporation of model prediction confidence bounds into a reliable optimal control system.

1.5 Structure of the Thesis

Chapter 2 presents a literature review of relevant techniques of empirical data-driven models, relevant techniques of learning algorithms and techniques of machine learning. Related knowledge gaps are also highlighted.

In Chapter 3, inferential estimation of the polypropylene polymerization process using DBN is introduced. A DBN model with a deep structure is developed and applied to the industrial polypropylene polymerization process of a real plant in China. Both feedforward neural network and DBN models are developed and compared. Compared with the conventional neural network, more latent information of process variables can be discovered by the DBN model. A reliable structure of DBN model is discussed.

Chapter 4 is focused on developing a bootstrap aggregated deep belief network (BAGDBN) model for the inferential estimation of MI of the polypropylene polymerization process and multi-step ahead prediction of the level of a water tank. Several DBN models are developed from bootstrap re-sampling replications of the process training data and are combined. The accuracy and reliability of DBN are enhanced. The results of both applications are compared and discussed. BAGDBN achieves reliable and robust estimation and prediction on both applications.

Chapter 5 is focused on developing a reliable data-driven modelling and optimisation strategy for a batch reactor using BAGDBN. BAGDBN improves the generalisation capability than that of a DBN model for the prediction of product quality variable. For the optimisation of a batch reactor, model prediction confidence bounds calculated from individual DBN predictions are

involved to achieve a reliable optimal control policy. Wide model prediction confidence bound is penalised to enhance the reliability of optimisation. The results of the application indicate the advantages of the proposed BAGDBN model.

Chapter 6 presents the conclusions from the research and recommendations for future works.

1.6 Publications

Published journal papers

1. Zhu C, Zhang J. Developing Soft Sensors for Polymer Melt Index in an Industrial Polymerization Process using Deep Belief Networks. *International Journal of Automation and Computing*, 2020, 17(1), 44-54.
2. Zhu C, Zhang J. Developing Robust Nonlinear Models through Bootstrap Aggregated Deep Belief Networks. *AIMS Electronic and Electrical Engineering* 2020, 4(3), 287–302.

Conference papers

1. Zhu C, Zhang J. Inferential Estimation of Polymer Melt Index Using Deep Belief Networks. *In: 24th International Conference on Automation and Computing (ICAC 2018)*. 2018, Newcastle upon Tyne, UK: IEEE.
2. Zhu C, Zhang J. Developing Robust Nonlinear Models through Bootstrap Aggregated Deep Belief Networks. *In: 25th International Conference on Automation and Computing (ICAC'19)*. 2019, Lancaster University, Lancaster, UK: IEEE.
3. C. Zhu and J. Zhang, "Reliable data-drive modelling and optimisation of a batch reactor using bootstrap aggregated deep belief networks", *Book of Short Papers SIS2021, the 50th Meeting of the Italian Statistical Society (SIS 2021)*, 21-25 June 2021, Pisa, Italy, pp. 94-99.

Chapter 2. Literature Review

2.1 Artificial Intelligence Technology Relevant to Process Control

In the past almost thirty years, artificial intelligence (AI) drew much attention in the research areas of chemical engineering, in particular process systems engineering. Many research works relevant to process control based on AI technology have been carried out and achieved many successful applications for the control strategies in chemical and biochemical engineering. The characteristic of AI technique has its advantages of 'understanding and learning' the information from external data and using that knowledge to adapt the varying conditions of task and achieve the goal. This notable feature of systems based on AI has many potential application areas within process control. An optimised process control policy ensures the processes run optimally despite upsets.

Figure 2.1 gives the classification of AI techniques which have been applied to process control applications for chemical, biochemical, and biomedical processes. The AI techniques have been divided into two categories, stand-alone and hybrid technologies (Ye et al., 2020). The first group contains expert systems whose development is based a solid knowledge base or rule base of the process. Fuzzy logic and expert systems suffer from high development costs/effort needed (Venkatasubramanian, 2019). ANN models based on historic process data have achieve satisfied results of regression modelling and fault classification. In general, the training of ANN is supervised training. This stand-alone group also includes reinforcement learning, and other techniques with supervised and unsupervised learning (Silva & Parpinelli, 2019).

The second group contains hybrid technologies, such as adaptive neuro-fuzzy inference system, hybrid neuro-fuzzy systems, etc. These techniques are a combination of two or more stand-alone techniques to overcome the limitations of a stand-alone AI technique (Ali et al., 2015). In AI based soft sensor development (Ali et al., 2015), the AI technologies use the supervised training, unsupervised training and reinforcement learning to find the relationship between process variables and quality variables. To estimate the quality variables, the model can be trained in a supervised way (Chen, 2019). The models trained in an unsupervised way can be utilised for the data clustering (Hinton & Sejnowski, 1999). Rewards and penalties are used by reinforcement learning to make the model learn adequate actions in environments with varying conditions (Sucar, 2011).

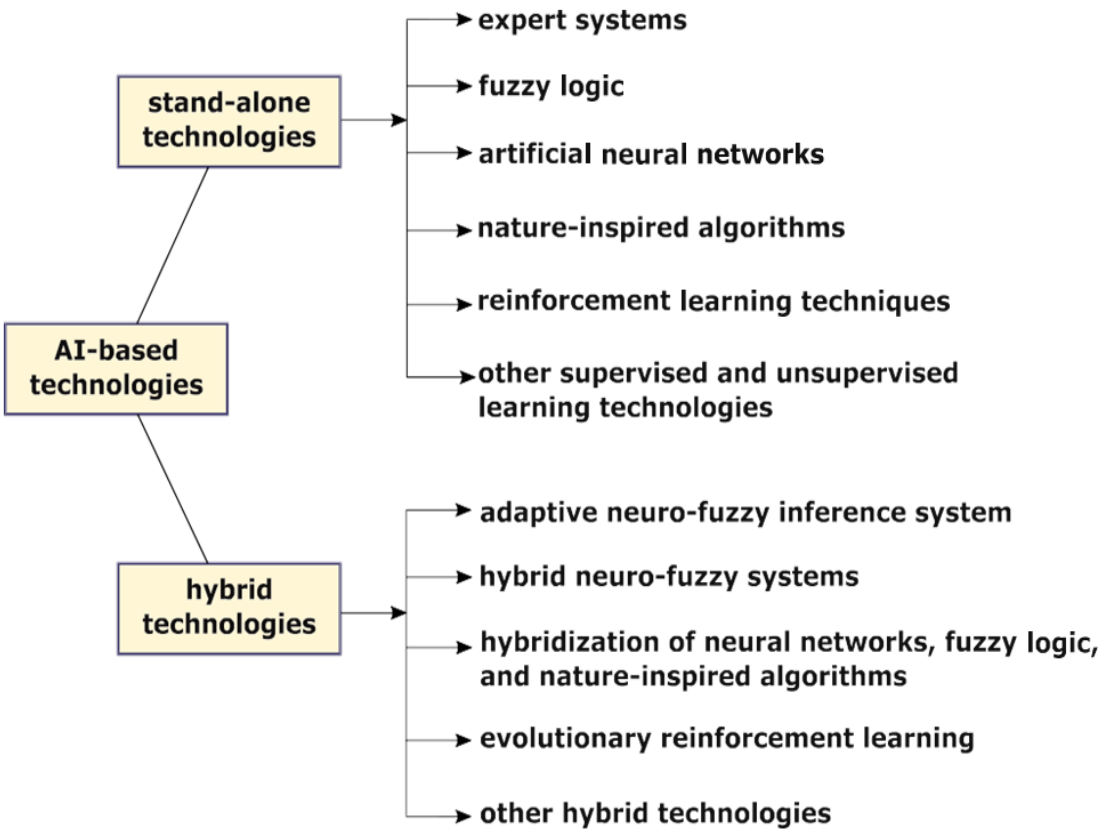

Figure 2.1 Classification of AI technologies utilised toward process control applications (Dutta and Upreti, 2021)

2.2 Regression Analysis

To estimating the relationship between variables, regression analysis can be carried out in statistical modelling. It finds the relationship between the independent variables and the dependent variables from a set of data. Actually, regression analysis is a set of statistical process for estimating the conditional expected value of the dependent variables when the independent value is fixed.

It is widely used in forecasting and prediction. By using regression analysis to find the relationship between independent and dependent variables, many techniques have been carried out, such as least squares, principal component regression and partial least squares regression, etc.

2.2.1 *Single Variable Linear Regression*

Single variable linear regression can be used to build a linear model between a dependent variable, *y*, and an independent variable, *x*. It establishes a relationship between the dependent variable and the independent variable by using a best fit line. The least squares method is the well-known method to perform linear regression.

Simple linear regression estimates the best fit line by minimizing the sum of the squares of the model prediction errors. When the errors are serially uncorrelated, the ordinary least squares regression is optimal in the class of linear unbiased estimators. The formula of model with one predictor variable and one response variable is given by,

$$y = \alpha + \beta x + \varepsilon \tag{2.1}$$

where x is the independent value, y is the dependent value, β is the slope of the model, α is the intercept, and ε represents the error term. Then the equation of prediction is given by,

$$\hat{y} = \hat{\alpha} + \hat{\beta} x \tag{2.2}$$

Where $\hat{y}$ is the model prediction, $\hat{\alpha}\ and\ \hat{\beta}$ are the estimated intercept and estimated slope respectively. The residuals between the prediction value $\hat{y}$ and actual response *y* can be formulated by,

$$r = y - \hat{y} = y - \hat{\alpha} - \hat{\beta} x \tag{2.3}$$

To estimate the model parameters, the sum of squared residuals should be minimized. The sum of squared residuals is calculated below,

$$J = \sum_{i=1}^{N} r_i^2 = \sum_{i=1}^{N} \left(y_i - \hat{\alpha} - \hat{\beta} x_i\right)^2 \tag{2.4}$$

Then *J* is differentiated with respect to $\hat{\alpha}\ and\ \hat{\beta}$,

$$\frac{\partial J}{\partial \hat{\alpha}} = 2N\hat{\alpha} - 2\sum_{i=1}^{N} y_i + 2\hat{\beta}\sum_{i=1}^{N} x_i \tag{2.5}$$

$$\frac{\partial J}{\partial \hat{\beta}} = -2\sum_{i=1}^{N} y_i x_i + 2\hat{\alpha}\sum_{i=1}^{N} x_i + 2\hat{\beta}\sum_{i=1}^{N} x_i^2 \tag{2.6}$$

At the best values of $\hat{\alpha}\ and\ \hat{\beta}$, the above gradients should be zero. Then we have,

$$\frac{\partial J}{\partial \hat{\alpha}} = 0, \frac{\partial J}{\partial \hat{\beta}} = 0 \tag{2.7}$$

$$\hat{\beta} = \frac{\sum_{i=1}^{N} y_i x_i - \frac{\sum_{i=1}^{N} y_i \sum_{i=1}^{N} x_i}{N}}{\sum_{i=1}^{N} x_i^2 - \frac{\left(\sum_{i=1}^{N} x_i\right)^2}{N}} \tag{2.8}$$

$$\hat{\alpha} = \frac{\sum_{i=1}^{N} y_i - \hat{\beta} \sum_{i=1}^{N} x_i}{N} \tag{2.9}$$

Therefore, the best fitting line is found. However, outliers affect the estimated model parameters. It means linear regression is very sensitive to outliers.

2.2.2 *Multiple Linear Regression*

The multiple linear regression can be used to find the relationship between two or more independent variables and one or more dependent variables by fitting a linear equation to the observed data. The formula of the model is given by,

$$y = \sum_{i=1}^{n} \theta_i x_i + \varepsilon \tag{2.10}$$

where y is the dependent value, $x_1, x_2, \dots, x_n$ are the independent values, θ_1 to θ_n are model parameters, and ε represents the error term. The prediction of y is shown below,

$$\hat{y} = \sum_{i=1}^{n} \widehat{\theta_i} \, x_i \tag{2.11}$$

where $\widehat{\theta_i}$ is the estimate of θ_i. Therefore, the residual between actual value and predicted value is,

$$e = y - \hat{y} = y - \sum_{i=1}^{n} \widehat{\theta_i} \, x_i \tag{2.12}$$

Then the matrix equation of the model is given by,

$$\widehat{\boldsymbol{Y}} = \boldsymbol{X}\widehat{\boldsymbol{\theta}} \tag{2.13}$$

$$\boldsymbol{E} = \boldsymbol{Y} - \widehat{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{X}\widehat{\boldsymbol{\theta}} \tag{2.14}$$

In the above equations, $\boldsymbol{Y}$ is a vector of observed variable values, $\widehat{\boldsymbol{Y}}$ is a vector of model predictions, $\boldsymbol{X}$ is a matrix of predictor values, $\boldsymbol{E}$ is a vector of model residuals, and $\widehat{\boldsymbol{\theta}}$ is a vector of model parameters.

To estimate the model parameters, the sum of squared residuals is minimized.

$$J = \boldsymbol{E}^T\boldsymbol{E} = \left(\boldsymbol{Y} - \boldsymbol{X}\widehat{\boldsymbol{\theta}}\right)^T\left(\boldsymbol{Y} - \boldsymbol{X}\widehat{\boldsymbol{\theta}}\right) = \boldsymbol{Y}^T\boldsymbol{Y} - 2\boldsymbol{Y}^T\boldsymbol{X}\widehat{\boldsymbol{\theta}} + \widehat{\boldsymbol{\theta}}^T\boldsymbol{X}^T\boldsymbol{X}\widehat{\boldsymbol{\theta}} \tag{2.15}$$

Then, $\frac{\partial J}{\partial \widehat{\theta}}$ is calculated as

$$\frac{\partial J}{\partial \widehat{\boldsymbol{\theta}}} = -2\boldsymbol{X}^T\boldsymbol{Y} + 2\boldsymbol{X}^T\boldsymbol{X}\widehat{\boldsymbol{\theta}} \tag{2.16}$$

To optimise the model parameters, $\frac{\partial J}{\partial \widehat{\boldsymbol{\theta}}}$ should be a zero vector. The solution is given as,

$$-2\boldsymbol{X}^T\boldsymbol{Y} + 2\boldsymbol{X}^T\boldsymbol{X}\widehat{\boldsymbol{\theta}} = \boldsymbol{0} \tag{2.17}$$

$$\widehat{\boldsymbol{\theta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y} \tag{2.18}$$

From Eq(2.18), $\widehat{\boldsymbol{\theta}}$ can be calculated. For the extension to model with multiple outputs $\boldsymbol{y}_1$, $\boldsymbol{y}_2$, ... , $\boldsymbol{y}_i$, $\boldsymbol{Y}$ will be a matrix of the observed responses.

2.2.3 *Principal Component Analysis*

Principal component analysis (PCA) was first described by Karl Pearson in 1901 (Pearson, 1901). It is a statistical procedure that transform a set of correlated observations into a set of values of linearly uncorrelated variables. Harold Hotelling developed the principal component analysis independently in the 1930s (Hotelling, 1933, 1936).

The data samples are defined as a matrix $\boldsymbol{X}$, which is a $n \times m$ matrix. Here n represents the number of samples and m the number of variables. And let $\boldsymbol{t}_i \in \boldsymbol{R}^n$ be the ith score vector of $\boldsymbol{X}$ and $\boldsymbol{p}_i \in \boldsymbol{R}^m$ be the ith loading vector. Then, the formula of PCA is given by,

$$\boldsymbol{X} = \boldsymbol{t}_1\boldsymbol{p}_1^T + \boldsymbol{t}_2\boldsymbol{p}_2^T + \cdots + \boldsymbol{t}_m\boldsymbol{p}_m^T = \boldsymbol{T}\boldsymbol{P}^T \tag{2.19}$$

In the above equation, for any $i \; and \; j$, if $i \neq j$, $\boldsymbol{t}_i\boldsymbol{t}_j^T = 0$ and $\boldsymbol{p}_i\boldsymbol{p}_j^T = 0$. If $i = j$, $\boldsymbol{p}_i\boldsymbol{p}_j^T = 1$. It means that score vectors are mutually orthogonal. So as loading vectors. Then multiply $\boldsymbol{X}$ by $\boldsymbol{p}_i$, we have

$$\boldsymbol{X}\boldsymbol{p}_i = \boldsymbol{t}_1\boldsymbol{p}_1^T\boldsymbol{p}_i + \boldsymbol{t}_2\boldsymbol{p}_2^T\boldsymbol{p}_i + \cdots + \boldsymbol{t}_m\boldsymbol{p}_m^T\boldsymbol{p}_i \tag{2.20}$$

Then, the score vector $\boldsymbol{t}_i$ is given by,

$$\boldsymbol{t}_i = \boldsymbol{X}\boldsymbol{p}_i \tag{2.21}$$

The length of $\boldsymbol{t}_i$ reflects the variation of $\boldsymbol{X}$ in the direction of $\boldsymbol{p}_i$. The score vectors are arranged in descending order of their lengths as,

$$\|\boldsymbol{t}_1\| > \|\boldsymbol{t}_2\| > \cdots > \|\boldsymbol{t}_m\| \tag{2.22}$$

The first loading vector $\boldsymbol{p}_1$ represents the largest direction of variation in $\boldsymbol{X}$ and $\boldsymbol{p}_m$ represents the smallest direction of variation in $\boldsymbol{X}$. When the variables of $\boldsymbol{X}$ are correlated, the variation of $\boldsymbol{X}$ will be represented by the first a few score vectors (e.g. the first k score vectors),

$$\boldsymbol{X} = \boldsymbol{t}_1\boldsymbol{p}_1^T + \boldsymbol{t}_2\boldsymbol{p}_2^T + \cdots + \boldsymbol{t}_k\boldsymbol{p}_k^T + \boldsymbol{E} \tag{2.23}$$

where $\boldsymbol{E}$ is a matrix of the minor principal components which are mainly noise. The last a few score vectors does not affect the variation $\boldsymbol{X}$ much. The variation of $\boldsymbol{E}$ mainly reflects noise and

k is typically much smaller than m. By ignoring $\boldsymbol{E}$, the data set $\boldsymbol{X}$ can be approximated as,

$$\boldsymbol{X} \approx \boldsymbol{t}_1\boldsymbol{p}_1^T + \boldsymbol{t}_2\boldsymbol{p}_2^T + \cdots + \boldsymbol{t}_k\boldsymbol{p}_k^T \tag{2.24}$$

The PCA of $\boldsymbol{X}$ is equivalent to eigenvector analysis of the covariance of $\boldsymbol{X}, \boldsymbol{X}^T\boldsymbol{X}$ in that $\boldsymbol{p}_i$ is the ith eigenvector of $\boldsymbol{X}^T\boldsymbol{X}$, $and \lambda_1 > \lambda_2 > \cdots > \lambda_m$ are the eigenvalues of $\boldsymbol{X}^T\boldsymbol{X}$.

As mentioned before the variation of $\boldsymbol{X}$ can be represented by the first a few principal components when the variables of $\boldsymbol{X}$ are correlated. The PCA of $\boldsymbol{X}$ is used to transform a high dimensional matrix into several principal component plots. The structure of data can be observed from the plots of the first a few principal components. In practice, the data will be scaled to zero mean and unit variance before applying PCA.

2.2.4 *Principal Component Regression*

Principal component regression (PCR) is a regression approach which is based on the theory of PCA to overcome the problem with MLR when the predictor variables are correlated.

In regression analysis, when the predictor variables are correlated, the matrix $\boldsymbol{X}^T\boldsymbol{X}$ will be singular. However, due to the existence of noise, $\boldsymbol{X}^T\boldsymbol{X}$ can still be of full rank but will be very close to singular. In such case, MLR will give significant errors in model parameter estimation. Principal component regression can solve this problem.

The major score vectors are represented by $\boldsymbol{T}$,

$$\boldsymbol{T} = \boldsymbol{X}\boldsymbol{P} \tag{2.25}$$

By using the major principal components as the predictor variables, the model becomes,

$$\boldsymbol{Y} = \boldsymbol{T}\boldsymbol{B} + \boldsymbol{E} \tag{2.26}$$

where $\boldsymbol{B}$ is a vector of model parameters associated with the major principal components, and $\boldsymbol{E}$ is a vector of model errors.

The estimation of $\boldsymbol{B}$ is given by,

$$\boldsymbol{B} = (\boldsymbol{T}^T\boldsymbol{T})^{-1}\boldsymbol{T}^T\boldsymbol{Y} \tag{2.27}$$

In PCR, $\boldsymbol{X}$ are replaced by new variables which are mutually orthogonal. Due to orthogonality of the score vectors, $\boldsymbol{T}^T\boldsymbol{T}$ will not be close to singular. To solve the problem of multicollinearity, the score vectors corresponding to small eigenvalues can be ignored. The

estimation of $\boldsymbol{B}$ is converted into $\widehat{\boldsymbol{\theta}}$,

$$\widehat{\boldsymbol{\theta}} = \boldsymbol{PB} = \boldsymbol{P}(\boldsymbol{T}^T\boldsymbol{T})^{-1}\boldsymbol{T}^T\boldsymbol{Y} \tag{2.28}$$

where $\boldsymbol{P}$ is a matrix of loading vectors. The appropriate number of principal components can be found by building a number of PCR models with different numbers of principal components and the model with the smallest error on the testing data will be selected. The data sets need be divided as training data and testing data.

The approach of principal component regression can solve the multicollinearity problem. When a set of explanatory variables is close to being collinear, PCR excludes some of the low-variance principal components in the regression (Dodge, 2003).

2.2.5 *Partial Least Squares Regression*

Herman O. A. Wold first described the partial least squares (PLS) and contributed this approach with Svante Wold (Wold, Sjöström & Eriksson, 2001).

The method of partial least square regression utilises the variation in $\boldsymbol{X}$ which is most predictive of $\boldsymbol{Y}$. By using this method, the input data is considered as a matrix $\boldsymbol{X}$ and is decomposed as,

$$\boldsymbol{X} = \boldsymbol{TP}^T + \boldsymbol{E} = \sum_{h=1}^{a} \boldsymbol{t}_h \boldsymbol{p}_h^T + \boldsymbol{E} \tag{2.29}$$

The output $\boldsymbol{Y}$ is decomposed as,

$$\boldsymbol{Y} = \boldsymbol{UQ}^T + \boldsymbol{F} = \sum_{h=1}^{a} \boldsymbol{u}_h \boldsymbol{q}_h^T + \boldsymbol{F} \tag{2.30}$$

If enough latent variables are used in the model, then $\boldsymbol{E}$ and $\boldsymbol{F}$ can be made small to be ignored. PLS is used to make $\|F\|$ as small as possible. The inner relationship can be given by,

$$\widehat{\boldsymbol{u}}_h = b_h \boldsymbol{t}_h \tag{2.31}$$

where $b_h = \frac{\boldsymbol{t}_h^T \boldsymbol{u}_h}{\boldsymbol{t}_h^T \boldsymbol{t}_h}$ is the inner model parameter. The objective of PLS is to explain $\boldsymbol{Y}$ by $\boldsymbol{X}$ maximally.

Partial least squares regression projects the dependent variables and the independent variables to a new space. Then a linear regression model will be found by partial least squares. Partial least squares regression has good performance when the independent variables have multicollinearity among them and there are fewer variables in observations than predictors. It has less restriction than other methods in multiple linear regression.

2.3 Artificial Neural Networks

For the past 3 decades, there have been many applications of ANN for industrial process control. The objective of ANN is fulfilled through training the network by adjusting the connection parameters between neurons. An ANN model is constructed by several layers of neurons. The information in process operational data is transferred through the neurons in each layer to the output layer. The inputs variables are multiplied by the parameter of connections and added to a bias to obtain the neuron output. It can be given as,

$$y = f(\sum_{i=1}^{n} w_i x_i + b) \tag{2.32}$$

where x_i represents the input variables, w_i is the weight between the neuron and the *i*th input, *b* is a bias need, f is an activation function (Bagheri, 2019; Jarmulak, 1997; Haykin, 2004), *n* is the number of inputs and *y* represents the output of the neuron.

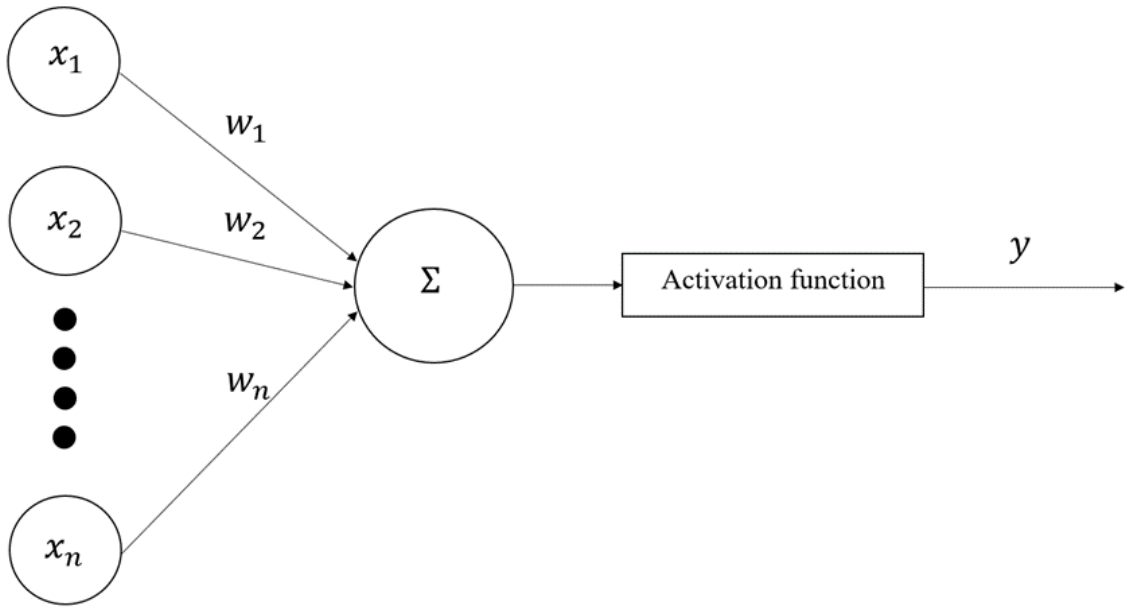

Figure 2.2 A simple perceptron invented by McCulloch and Pitts

McCulloch and Pitts (1943) invented the initial form of simple perceptron and used threshold function as the activation function in their model shown in Figure 2.2. The weighted sum of input variables is calculated and passed to an activation function to achieve the neuron output. McCulloch and Pitts (1943) proved universal computations can be performed by simple perceptron if weights are chosen appropriately. However, a lot of complicated systems cannot be represented by this method (Jain, 1996). Many other activation functions can be used, such as Heaviside step function, sigmoid function and Gaussian function (Honkela et al, 2011, Costarelli & Spigler, 2013, Gundogdu et al, 2016). These activation functions are sometimes also named as transfer function in ANN research. The most popular activation function is the

sigmoid function.

The characteristic of sigmoid function is that it is an 'S'-shaped curve as shown in Figure 2.3

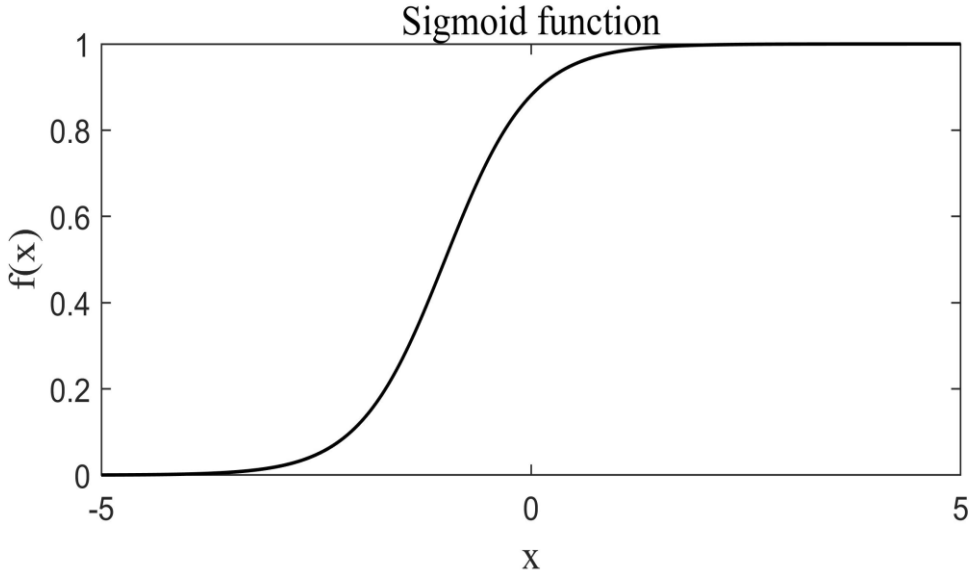

Figure 2.3 The S-shaped curve of sigmoid function

Sigmoid function maps its input values into a region from 0 to 1. In Figure 2.3, the output value approaches to 1 when x approaches to $+\infty$, whereas the output approaches to 0 when x approaches to $-\infty$. It has the appropriate asymptotic properties. The sigmoid function is given by (2.33),

$$S(x) = \frac{1}{1+e^{-\beta x}} \tag{2.33}$$

where x represents the sum of weighted input values and β is a slope parameter.

The most widely used ANN contains an input layer, a hidden layer and an output layer. Figure 2.4 shows an architecture of ANN. The weights and biases are initialized randomly at the beginning of training. During the procedure of neural network training, the key parameters, weights and biases between layers, will be adjusted to minimize the output error. The output error can be represented as sum of squares errors between true output values and the corresponding predictions. These errors represent the current status of the training of networks. The training uses the backpropagation algorithm and the weights and biases between neurons are calculated in a sequential way backward from the last layer (Haykin, 2004). These parameters are improved by a method base on gradient optimisation. Repeat the sequence of forward pass and improvement of weights and biases until the error is tolerable or achieve the set number of training iterations (Haykin, 2004).

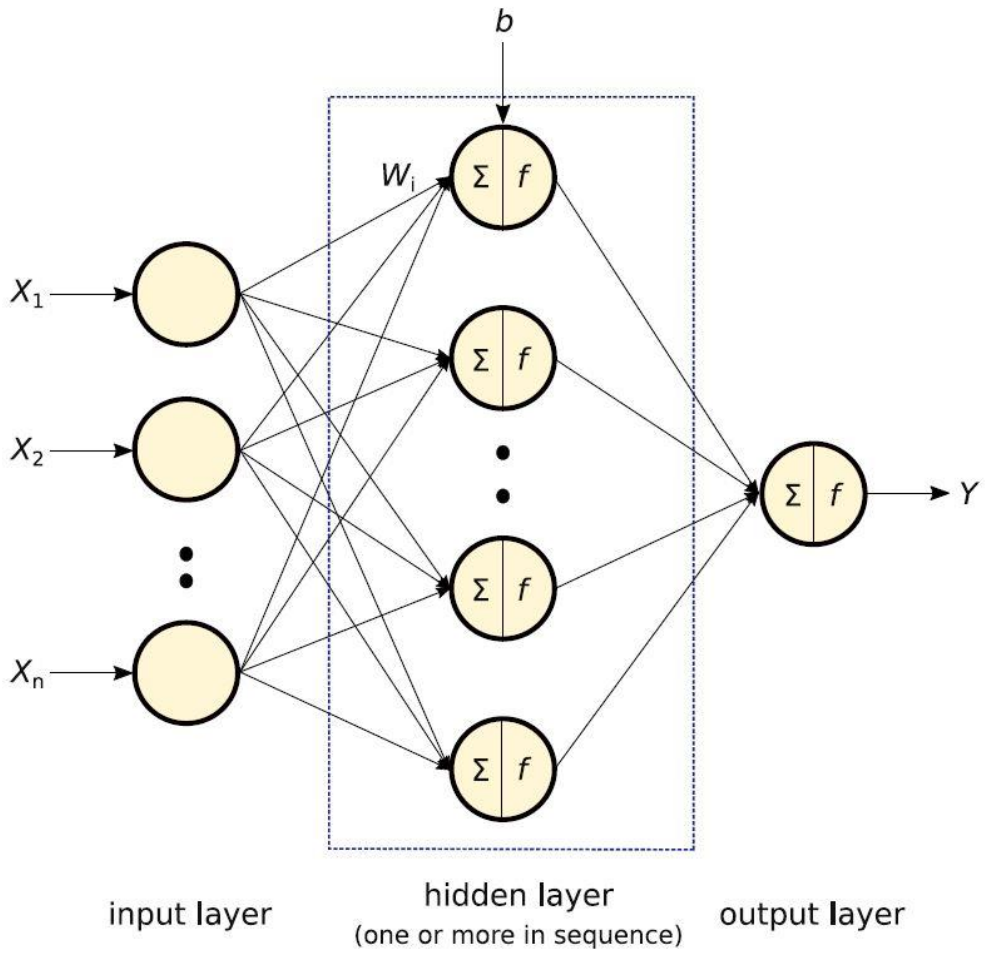

Figure 2.4 The architecture of ANN (Dutta and Upreti, 2021)

The ANN can approximate a nonlinear process based on historic data and achieve accurate and reliable predictions even under certain condition of operations. However, this approach requires very precise and formatted data to achieve the best performance (Ali, 2005; Jarmulak, 1997; Tu,1996). As the results, ANN is effective and fault tolerant for industrial process modelling.

2.3.1 *Different Types of ANN*

There are three most widely used ANN in industrial applications and other areas:

1 Feedforward neural networks.

2 Radial basis function neural networks.

3 Recurrent neural networks.

Feedforward neural network are the traditional type of neural network model mentioned before. Radial basis function (RBF) neural networks use radial basis functions as the activation function in the hidden layer (Du, 2018). This network usually only has one hidden layer and there are no

weights in the hidden layer. The training of RBF neural network uses a method of two-step training algorithm. The first step is unsupervised and involves selections of the center vectors of radial basis functions. Then, the output layer weights will be fitted using regression approach. A recurrent neural network has feedback from the network outputs or hidden neuron outputs to the inputs. The training of recurrent neural network involves a key parameter, time. The most typical types of recurrent neural network are long-short-term-memory model (LSTM) and Elman networks.

There are some other types of ANN, such as self-organizing radial basis function-based neural network, ANN using the technique of nonlinear auto-regressive exogenous (NARX), and ANN using the approach of nonlinear auto-regressive moving average (NARMA). Self-organizing radial basis function-based neural network adjusts the weights of hidden neurons and keep the model attuned with the real-time dynamics to improve the accuracy of predictions (Han, 2012). In the ANN using NARX, the model output variables are related to its past values in a time series. The input and output variable values of the externally determined driving series supplemented by the ANN inference model (Billings, 2013). NARMA is similar as NARX. The approach of auto-regressive regresses past values and the method of moving average involves error modelling to effectively represent general discrete-time nonlinear systems (Chetouani, 2008; Erguzel & Akbay ,2014).

In summary, the advantages of ANN model are listed below,

1 ANN can achieve accurate and consistent solutions in different conditions of processes based on historic process operational variables and approximate nonlinear process.

2 ANN is effective and fault tolerant.

3 ANN can store information in continuous memory locations, which have a high potential for real-time implementations.

However, ANNs have some disadvantages. Conventional neural networks always suffer from the issue of overfitting, local optimal and slow convergence.

2.3.2 *Applications of ANN*

ANN has many successful applications in many chemical, biochemical, and biomedical engineering. ANN has been utilised in PID schemes (Chen & Huang, 2004; Du et al., 2018), internal model control (IMC) (Nahas, 1992; Lim, 2010), and predictive control schemes

(Gomm, 1997). Bhat and McAvoy (1989) presented a work on using traditional neural network to model the response of pH in a continuous stirred tank reactor (CSTR). Compared with auto-regressive moving average (ARMA) model, ANN gave better performance in that application than ARMA model. Nahas et al. (1992) introduced a nonlinear internal model control strategy based on neural network models. They demonstrate that the proposed strategy using neural network outperform the conventional PID control. Normandin et al. (1994) presented an approach of control of a continuous stirred tank fermenter using a neural network model in a predictive control strategy where the inlet substrate flow rate is selected in order to maximize an objective cost function. Neural network gave very good performance in this predictive strategy. Lightbody and Irwin (1995) presented the adaptive ANN-based control for non-isothermal CSTR process to control the product concentration. The adaptive ANN-based controller outperformed the PI controller in terms of set-point tracking. Macmurray and Himmelblau (1995) proposed an externally recurrent ANN model for model predictive control. The results of externally recurrent ANN model indicate it has better performance than first principle model for model predictive control. Shah and Meckl (1995) proposed a strategy of adaptive ANN based model by using radial basis function-based ANN for non-isothermal CSTR. Bittanti and Piroddi (1997) proposed the application for shell and tube heat exchanger by using a control strategy based on ANN. ANN performed better than PID and linear controller. Syu and Chang (1997) presented a recurrent backpropagation neural network (RBPN) for the on-line adaptive pH control of penicillin acylase fermentation with Arthrobacter viscous. A moving-window type of training data was provided to train RBPN to enhance the effective on-line learning of this network. The RPBN achieved successful performance for the pH control of penicillin fermentation. A model based on the neural network with using the technique of spread encoding is proposed by Gomm et al. (1996) and it can achieve accurate long-range predictions and better control performance compared with conventional PI controller.

Díaz et al. (2001) presented the investigation of the use of adaptive ANNs to control the temperature of air exit of a compact heat exchanger. The controller based on IMC scheme can be adapted online based on different performance criteria, energy consumption, minimisation of target error and controller stabilization criteria. The neurocontroller can adapt to major structural changes as well as to minimize the amount of energy cost. Engell and Fernholz (2003) proved the predictive controller based on the technique of radial-basis functions networks have improved the process control for at a real plant. The controller using neural network outperformed the linear controller. Ou and Rhinehart (2003) used neural networks for each sub-model, and terms the prediction model as a grouped neural network (GNN). GNN was incorporated into a general nonlinear model predictive control (NMPC) structure to control a

distillation column. They demonstrated the effectiveness of this approach for a highly nonlinear process. Chu et al. (2003) proposed an MPC using recurrent ANN models. The comparison with linear controller and PI controller indicates the strong performance of recurrent ANN models. Chen and Huang (2004) presented an improved conventional PID control scheme using linearized neural network model. This controller outperformed recursive linear model- based, and self-tuning PID controllers for a pH neutralization process. Varshney and Panigrahi (2005) proposed a controller based on multilayer neural network for a heat exchanger in a closed flow air circuit. They demonstrated the controller based on multilayer neural network is more robust than conventional PID controller. Xiong and Zhang (2005) introduced a batch-to-batch iterative optimal control strategy using recurrent neural network for a methyl methacrylate polymerization process. They demonstrated the effectiveness of the proposed approach for the issue of model plant mismatches and unknown disturbances. Åkesson and Toivonen (2006) proposed a novel controller based on neural networks for optimal MPC of constrained nonlinear systems. This controller achieved optimal performance with different structure of controller, such as centralized structure and decentralized structure. Nagy (2007) introduced the enhanced Optimal Brain Surgeon algorithm for the determination of the optimal ANN topology. The controller based on this novel neural network outperformed the linear controller and traditional PID controller for alcoholic fermentation process. A novel controller based on multi-rate pseudo linear RBF neural networks was proposed by Yu and Yu (2007). It reduced the CPU time and achieved desired long-range predictions. Ekpo and Mujtaba (2007) proposed ANN-based predictive control utilised control vector parameterization and sequential quadratic programming-based optimisers for the batch polymerization of methyl methacrylate. It achieved better performance than controller based on ANN. Gonzaga et al. (2009) developed a feed-forward ANN to predict the polyethylene terephthalate viscosity for the polymerization process. The proposed model was integrated in the process control system and achieved robust performance. The control system based on ANN can be applied in servo and regulatory problems of the process.

Yu et al. (2010) presented an ANN model to control the Fenton process for textile wastewater treatment. The accurate predictions of pH and effluent COD achieve by ANN are helpful to the process control and biological process. Alshehri et al. (2010) developed an ANN controller to predict and control the salt concentration in the treated oil for a crude oil desalting process. Damour et al. (2010) proposed a nonlinear predictive controller based on neural networks for an industrial crystallization process. It achieved better efficiency than conventional PID controller. Robustness and stability for the neural closed loop control system are discussed by Fernandez De Canete et al. (2010) by using the harmonic balance equations. Lim et al. (2010)

developed a controller based on ANN with decentralized structure for a hydrolyzation process. The proposed controller achieved better control performance than conventional PID controller. Han et al. (2011) proposed a MPC method based on a self-organizing RBF neural network to control dissolved oxygen concentration in activated sludge wastewater treatment processes. The self-organizing RBF neural network can adjust the hidden neurons and keep the model adapted with the dynamic of the process. Paengjuntuek et al. (2012) developed a controller based on multi-layer feed forward ANN for batch crystallization process and outperformed linear cooling control and theoretical model-based optimal control. Imtiaz et al. (2013) developed an inverse neural network for temperature control of a biochemical CSTR and its effect on ethanol production. And the performance of the controller is better than traditional PID controller. Rani et al. (2013) proposed adaptive linear network (ADALINE) based soft sensors and the application in inferential control of a multicomponent distillation process. ADALINE achieved better performance than Levenberg-Marquardt controller in terms of accuracy, training time and memory space required for training. Porrazzo et al. (2013) presented an optimizing control system based on neural network for a seawater-desalination solar-powered membrane distillation unit. Tayyebi and Alishiri (2014) developed an ANN-based inverse model controller that achieves results without overshoot and reduced computational burden for multi-stage flash distillation process. Li and Li (2016) developed a neural network Wiener model and applied for an intensified continuous reactor. The Wiener model is a hybrid model which contains two parts: a linear state space model identified based on nonlinear first-principal model and a neural network to predict the nonlinear controlled quality variables. Du et al. (2018) an introduced radial basis function-based neuro controller for activated sludge process. They demonstrate the effectiveness and stability of the proposed controller. Wu and Christofides (2019) proposed an economic model predictive control system based on technique of control Lyapunov-barrier function for non-isothermal CSTR. To ensure the systems stability, recurrent neural network was used in this controller. Carvalho et al. (2020) proposed a feedback control system based on neural network and compared with conventional PI controller-based feedback control loop for heat exchangers. The proposed controller achieved better performance than PI controller. Monticeli and Ornaghi Júnior (2021) proposed an ANN approach for the prediction of the TG behavior of vegetal fibers. The results of the study indicate that the application of ANN for a highly non-linear application requires large amount of training data in order to achieve accurate predictions. An artificial neural network was developed for the estimation of the thermal conductivity of multi-walled carbon nanotubes (MWCNTs)-CuO/water nanofluid by Rostami et al (2021). They used ANN as a new approach to predict the thermal conductivity leading a contribution in achieving more accurate predictions of the process states. Esfe et al. (2021)

developed a well trained artificial neural network using Bayesian regularization back propagation algorithm. The model was used to predict the dynamic viscosity of $MWCNT-Al_2O_3$/oil SAE40 nanofluid at different concentration of inputs and temperatures. ANN gave good performance on prediction with > 99.9% correlation coefficient. Wang et al. (2021) developed a novel hybrid method of ANNs with response surface methodology for the optimisation of dark fermentation. This approach was applied to a complex system with significant variation among different critical operational conditions and achieved reliable optimisation of dark hydrogen fermentation process. Fan et al. (2022) presented a study on the dynamic viscosity of water – ethylene glycol/WO_3 – MWCNTs hybrid nanofluid with the influence of nanoparticle and temperatures. The process data were modeled by ANN and ANN achieved reliable and robust performance. A study of optimisation of fertilizer rates and water table levels for different crops and fields by feedforward neural networks was introduced by Grenon et al. (2022). The effectiveness of neural networks was demonstrated with the comparison between feed-forward neural network, deep feed-forward neural network, general regression neural network, bi-directional LSTM (Bi-LSTM), LSTM, gated recurrent unit and radial basis function neural network. Hassan et al. (2022) built an ANN model for the prediction of the thermal degradation and flexural strengths of the Bi_2O_3- polybenzoxazine matrix and its nanocomposite. The value of correlation coefficients is above 0.99 which demonstrated that accuracy of ANN model. Glosh et al. (2022) developed an ANN-based fiber optic hybrid multi-grating sensor for the measurement of Pb^{2+} ions concentration. The sensor achieved a high sensitivity as 2.55±0.06365 nm/nM. It indicated that the high generalisation capability of ANN approach. Sahoo & Biswal (2022) introduced using ANN model to predict mechanical properties of acrylic acid and monomers acrylonitrile with reinforcement of fish bone powder (FBP). The result of the predictions was close to the experimental results. Dam et al. (2022) introduced an approach for determination of the superficial velocity and volume fraction in a simulated two-phase (oil–water) flow system by using ANN model. The predictions of the ANN model are close to the experimental results of the process.

2.4 Bootstrap Aggregated Neural Networks

Model robustness is one of the most important criteria that need to be considered when applying advanced modelling techniques to actual industrial processes. Model robustness is crucial in real industrial applications. To improve the robustness and stability of neural networks, combining several developed neural networks into one model were investigated by many researchers (e.g. Cho & Kim, 1995; Hashem, 1997; Jacobs et al., 1991; Ji & Ma, 1997; Jordan & Jacobs, 1994; Perrone & Cooper, 1993; Raviv & Intrator, 1996; Rosen, 1996; Sharkey, 1996; Sridhar, Seagrave & Bartlett, 1996; Taniguchi & Tresp, 1997; Wolpert, 1992; Xu et al., 1992). In early 90s, Breiman (1996) proposed a novel approach to aggregate multiple forecasting models based on the technique of bootstrap resampling to enhance the robustness of model. The model accuracy was significantly improved (Breiman, 1996). The bootstrap resampled data can be considered as a replication of the original modelling data. There will be various performances of neural network when they are trained by different sets of bootstrap resampled data, even these networks are correlated and model the same process. Beyond this work, bootstrap aggregated neural network was proposed by Zhang (1999b).

2.4.1 *Structure of Bootstrap Aggregated Neural Networks*

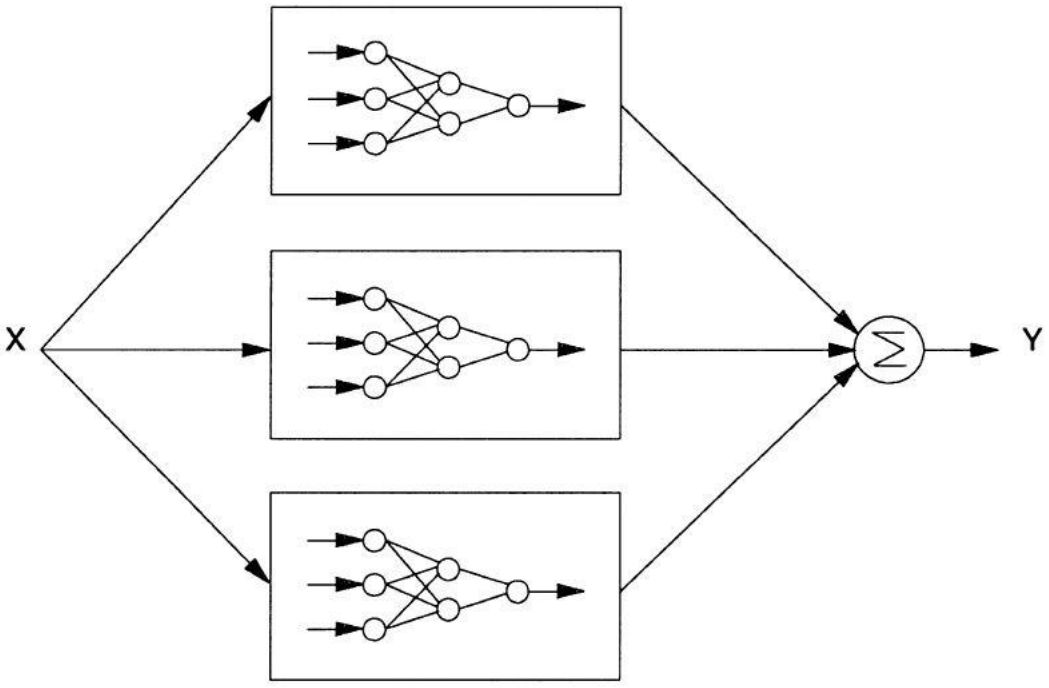

Figure 2.5 The structure of bootstrap aggregated neural networks

The structure of bootstrap aggregated neural networks is given in Figure 2.5. Bootstrap aggregated neural networks contain multiple individual neural networks. These neural networks are independently developed to each other, even they model the same chemical process, because they are developed from different sets of bootstrap resampled data, there are differences among these networks. Predictions achieved by those individual neural networks will be weighted and combined to achieve a more reliable and robust prediction of the model output variables. The combination of the bootstrap aggregated neural network outputs can be formulated as,

$$f(\boldsymbol{X}) = \sum_{i=1}^{n} w_i f_i(\boldsymbol{X}) \tag{2.34}$$

where $f_i(\boldsymbol{X})$ represents the *i*th neural network's predictive function, w_i is the aggregating weight for combing the *i*th neural network, $\boldsymbol{X}$ is a vector of model inputs.

Selecting an appropriate value of aggregating weight, w_i, can achieve accurate and reliable results for the application. The method of multiple linear regression could be applied to adjust the weights of bootstrap aggregated neural networks. However, the approach of multiple linear regression will lead to undesirable and unreliable estimation because of the high correlation between these individual neural networks (Zhang, 1999b). Consider $\boldsymbol{y}$ as a vector of desired target values and $\widetilde{\boldsymbol{y}_i}$ as the prediction of *i*th neural networks, and *n* is the number of neural networks in the bootstrap aggregated neural network, the predictions of all individual neural networks can be given as,

$$\widetilde{\boldsymbol{Y}} = [\,\widetilde{\boldsymbol{y}_1}\,\widetilde{\boldsymbol{y}_2}\,\ldots\widetilde{\boldsymbol{y}_n}] \tag{2.35}$$

where $\widetilde{\boldsymbol{Y}}$ is a matrix of predictions from individual neural networks. The prediction of bootstrap aggregated neural network, $\widetilde{\boldsymbol{y}}_b$, can be calculated by,

$$\widetilde{\boldsymbol{y}}_b = \widetilde{\boldsymbol{Y}}\boldsymbol{w} = w_1\widetilde{\boldsymbol{y}_1} + w_2\widetilde{\boldsymbol{y}_2} + \cdots + w_n\widetilde{\boldsymbol{y}_n} \tag{2.36}$$

Using the linear regression to adjust the aggregating weights, $\boldsymbol{w}$, the least squares estimation is given by,

$$\boldsymbol{w} = (\widetilde{\boldsymbol{Y}}^T\widetilde{\boldsymbol{Y}})^{-1}\widetilde{\boldsymbol{Y}}^T\boldsymbol{y} \tag{2.37}$$

Because the individual neural networks are highly correlated to each other, $\widetilde{\boldsymbol{Y}}^T\widetilde{\boldsymbol{Y}}$ will be singular or very close to singular. Therefore, the aggregating weights, $\boldsymbol{w}$, will be very sensitive to the errors between predictions and actual values and noises of data. The multiple linear regression approach will lead to poor performance. This situation also occurs in the work by Breiman (1994) and constrains need to be added to the adjustment of aggregating weights.

The aggregating weights of bootstrap aggregated neural networks can be obtained by the approach of PCR, due to the high correlation among individual neural networks. The

predictions of individual neural networks, $\widetilde{\boldsymbol{Y}}$, can be decomposed using PCA as,

$$\widetilde{\boldsymbol{Y}} = \boldsymbol{t}_1\boldsymbol{p}_1^T + \boldsymbol{t}_2\boldsymbol{p}_2^T + \cdots + \boldsymbol{t}_n\boldsymbol{p}_n^T \tag{2.38}$$

where $\boldsymbol{t}_i$ and $\boldsymbol{p}_i$ are *i*th score vector and loading vector respectively.

The bootstrap aggregated neural networks output can be obtained as a linear combination of the first a few principal components of $\widetilde{\boldsymbol{Y}}$ using PCR. The prediction, $\widetilde{\boldsymbol{Y}}_b$ can be represented as,

$$\widetilde{\boldsymbol{Y}}_b = \boldsymbol{T}_k\boldsymbol{\theta} = \widetilde{\boldsymbol{Y}}\boldsymbol{P}_k\boldsymbol{\theta} \tag{2.39}$$

where $\boldsymbol{T}_k$ is the matrix of the first *k* score vectors, $\boldsymbol{P}_k$ is a matrix of the first *k* loading vectors.

The least squares estimation of $\boldsymbol{\theta}$ is shown below,

$$\boldsymbol{\theta} = (\boldsymbol{T}_k^T\boldsymbol{T}_k)^{-1}\boldsymbol{T}_k^T\boldsymbol{y} = (P_k^T\widetilde{\boldsymbol{Y}}^T\widetilde{\boldsymbol{Y}}\boldsymbol{P}_k)^{-1}\boldsymbol{P}_k^T\widetilde{\boldsymbol{Y}}^T\boldsymbol{y} \tag{2.40}$$

The aggregating weight, $\boldsymbol{w}$, can be calculated by using PCR as,

$$\boldsymbol{w} = \boldsymbol{P}_k\boldsymbol{\theta} = \boldsymbol{P}_k(\boldsymbol{P}_k^T\widetilde{\boldsymbol{Y}}^T\widetilde{\boldsymbol{Y}}\boldsymbol{P}_k)^{-1}\boldsymbol{P}_k^T\widetilde{\boldsymbol{Y}}^T\boldsymbol{y} \tag{2.41}$$

The performance of adjusted aggregating weights through PCR performed well (Zhang, 1999b). The appropriate number of principal components can be found through the method of cross-validation. Several numbers of principal components are selected and the modelling performance is tested on the testing data set. The bootstrap aggregated neural network has the smallest testing error is considered as having the most appropriate number of principal components.

This approach of developing a bootstrap aggregated neural network can be summarised as the follow steps,

1. Using the approach of bootstrap re-sampling with replacement (Efron & Tibshirani,1993) to generate new replications of training data set from the original training data set. The distribution of replications of training data is similar to that is the original training data set.
2. A neural network with appropriate structure is developed from each set of bootstrap replications.
3. After training these individual neural networks, the predictions of individual networks are combined together through PCR.

The confidence bounds can also be calculated by calculating the standard errors of individual models in bootstrap aggregated neural networks (Zhang, 1999b). The confidence bounds provide more information to the process operator about the reliability of model predictions.

2.4.2 *Bootstrap Aggregated Neural Network Applications*

Zhang (1999a) proposed the bootstrap aggregated neural networks with the approach of PCR for nonlinear process modelling including inferential estimation of polymer quality in a simulated batch polymerization process. Because of the strong generalisation capability of neural networks for nonlinear process, the bootstrap aggregated neural network achieved more accurate and robust estimations than a single neural network. The confidence bounds for the model predictions can be computed. It can be considered as an evaluation criterion to decide the predictions are acceptable or not. A novel bootstrap aggregated neural network using a sequential training algorithm was introduced by Zhang (2002). The sequential training algorithm minimize the training errors and the correlations among individual neural networks to address the question of how many neural networks should be combined. The computational effort of bootstrap aggregated neural network had been reduced due to this sequential training. The effectiveness of the new sequential training algorithm was illustrated by modelling a water tank.

The bootstrap aggregating neural network using the Levenberg-Marquardt optimisation algorithm and the sequential training algorithm was applied to an industrial polypropylene polymerization process for the inferential estimation of polymer MI (Zhang et al., 2006). The correlation analysis for the process operational data and quality data had been carried out. The model achieved significantly improved performance on estimate the polymer MI with reduced computation costs.

Al-Mahrouqi and Zhang (2008) proposed a control strategy using bootstrap aggregated neural networks and ant colony optimisation for a fed-batch fermentation process. The confidence bounds estimated from individual neural networks are incorporated in the optimisation objective function and wide confidence bounds are penalised. The improvement of optimal control policy is demonstrated. This approach also was applied to a reactive polymer composite moulding process (Mohammed & Zhang, 2003). Kaunga et al. (2013) developed a bootstrap aggregated neural network for the modelling of the chemical durability of HLW glass in a nuclear waste processing process. They indicated that developed bootstrap aggregated neural network can give accurate and reliable prediction of chemical durability.

Khaouane et al. (2017) reported a bootstrap aggregated neural network (BANN) model for the rejection process of charged and uncharged organic compounds by nanofiltration and reverse osmosis membranes. To develop the BANN model, 436 rejections of 42 charged and uncharged organic compounds were selected as process data set. Different training data were resampled

from original data by using the bootstrap resampling technique. Each individual neural network was built, validated and tested with the resampled training data. The predictions were achieved by combining the predictions of different neural networks through simple averaging. Good performance of BANN was demonstrated in this study.

Osuolale and Zhang (2018) developed bootstrap aggregated neural networks for estimations of exergy efficiency and product qualities and the developed models were used to maximize exergy efficiency and product quality. The confidence bounds from the individual neural network predictions were incorporated in the optimisation. The results of the application for a crude distillation system indicated the improvement in the exergy efficiency with no additional costs of equipment.

Szafranek (2019) estimated the quality of the out-of-sample short-term headline inflation forecasts achieved by a combination of bagged ANN models. This study utilised a thick modelling approach to improve the accuracy of the models and prevent the issue of overfitting with bootstrap aggregation. The results were remarkable.

Lian et al. (2020) introduced a novel bootstrap aggregated classification tree neural network (BACT-ANN) for the prediction of rainfall occurrences and amounts over the Langat River Basin, Malaysia. The daily rainfall series for the years 1975–2012 at four rainfall stations were selected as the training data. The simulation of the rainfall occurrence had been achieved appropriately by BACT-ANN with critical requirements. This approach outperformed the stochastic nonhomogeneous hidden Markov model in simulating variance, distribution and correlation of rainfall amount.

Lu et al. (2021) developed a novel bagging fuzzy neural networks for 72 hours forecast of low temperature chilling injury. To develop the capability of single fuzzy neural network, the playback extracted training data subsets were utilised for the development of the bagging model. The novel bagging fuzzy model can reduce the uncertainty of the predictions of the single forecasting model. It gives lower mean absolute error than result of single fuzzy network.

2.5 Deep Learning

As a more recent research area of AI, deep learning becomes a very promising area for research and industry. It has already been applied to many areas related to our life (Goswami, 2020), such as

1. Image and voice recognition using the technique of deep leaning by Google.

2. Deciding which video and movie is being popular in the future and provide the valuable content to customers.
3. Predicting future actions for advertisers by Facebook.
4. Identification of cancer cells in tissue samples.
5. Face recognition.
6. Driver assistance systems.

Deep learning is a subcategory of machine learning in which a model is developed from process data set (Goswami, 2020). Model based on the technique of deep learning has a deep structure with many layers in the network. Deep network always contains multiple hidden layers as shown in Figure 2.6.

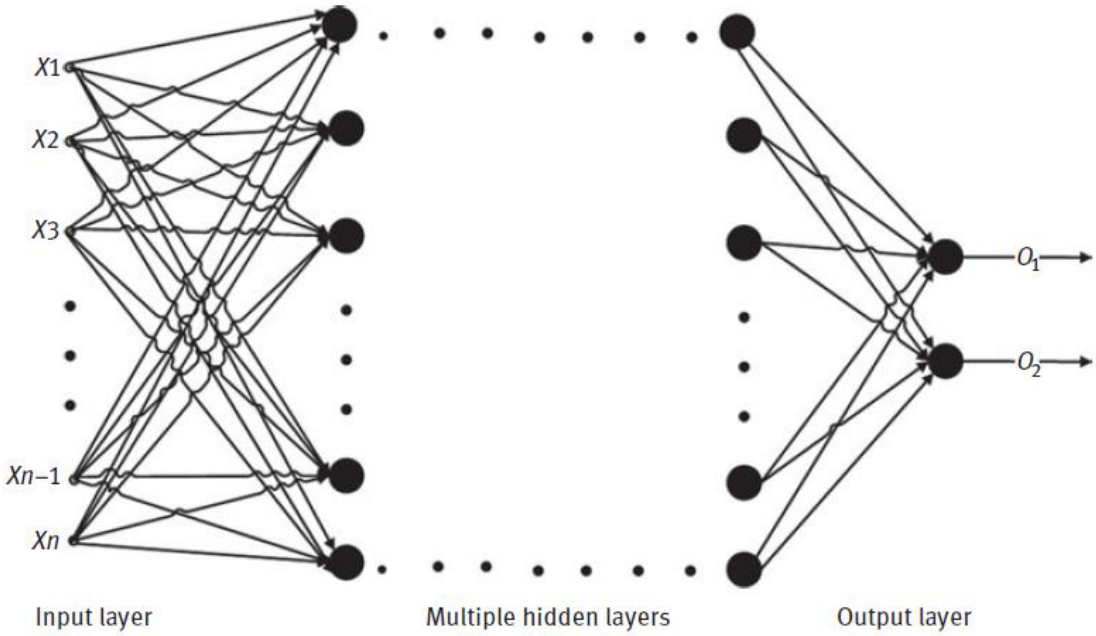

Figure 2.6 Diagram of a deep learning network (Goswami, 2020)

There is no evaluation criterion to define how many layers in the network that it can be consider as deep learning (Deng & Yu, 2014). Compared with the conventional data-driven models, deep learning can achieve more accurate and reliable results. Deep learning-based models can deal with massive amount of data and the performance increase significantly when training by large amount of data, and advanced GPUs can reduce the training time of deep learning network models rapidly (Goswami, 2020).

2.5.1 *Commonly Used Deep Learning Methods*

The restricted Boltzmann machine (RBM) is widely used in deep learning models. RBM is a building block of deep belief networks (DBN). The parametrization can be shared by RBM with DBN layers. And there are very effective training algorithms for training RBM. Figure 2.7 shows the structure of RBM and DBN is illustrated in Figure 2.8.

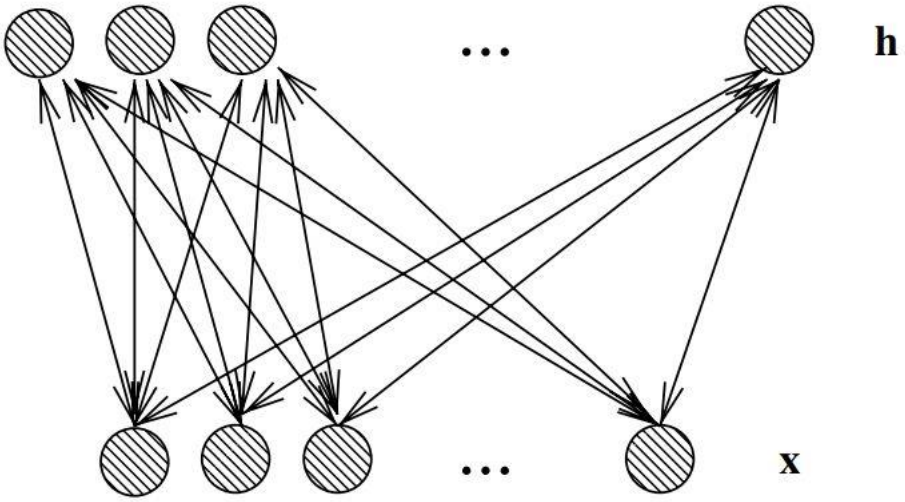

Figure 2.7 Undirected diagram of a RBM model (Bengio, 2009)

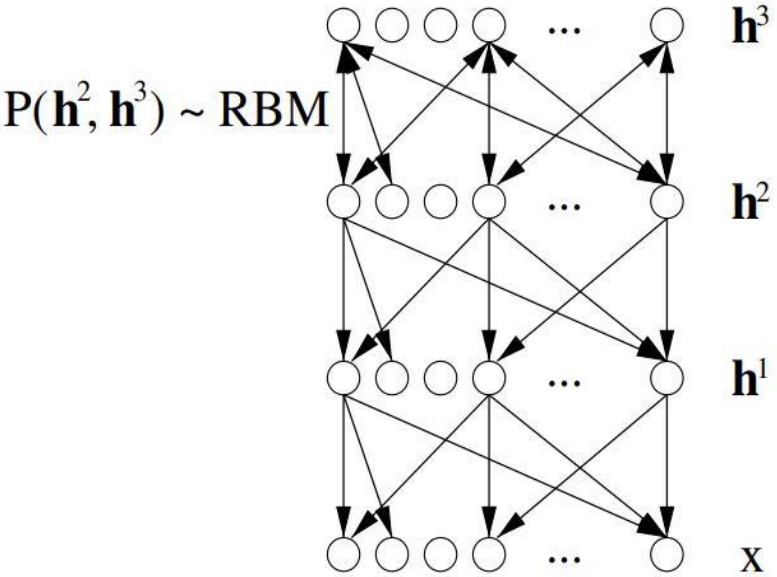

Figure 2.8 Deep belief network (Bengio, 2009)

In Figure 2.7, **x** is the input units, **h** is the hidden units. There is no connection between the neurons in the same layer which is the difference with Boltzmann machine. In Figure 2.8, the

top two layers can be considered as a RBM model. $P(\boldsymbol{h}^2, \boldsymbol{h}^3)$ means the joint conditional probability function of the top two layers. It can be considered that a DBN model is constructed with stacking RBMs (Bengio, 2009). The DBN models was invented by Hinton (2006) for image recognition, cluster and generation and motion capture data.

Another popular model based on the technique of deep learning is DBN based autoencoder. A DBN based autoencoder is comprised of two DBN models. It is shown in Figure 2.9.

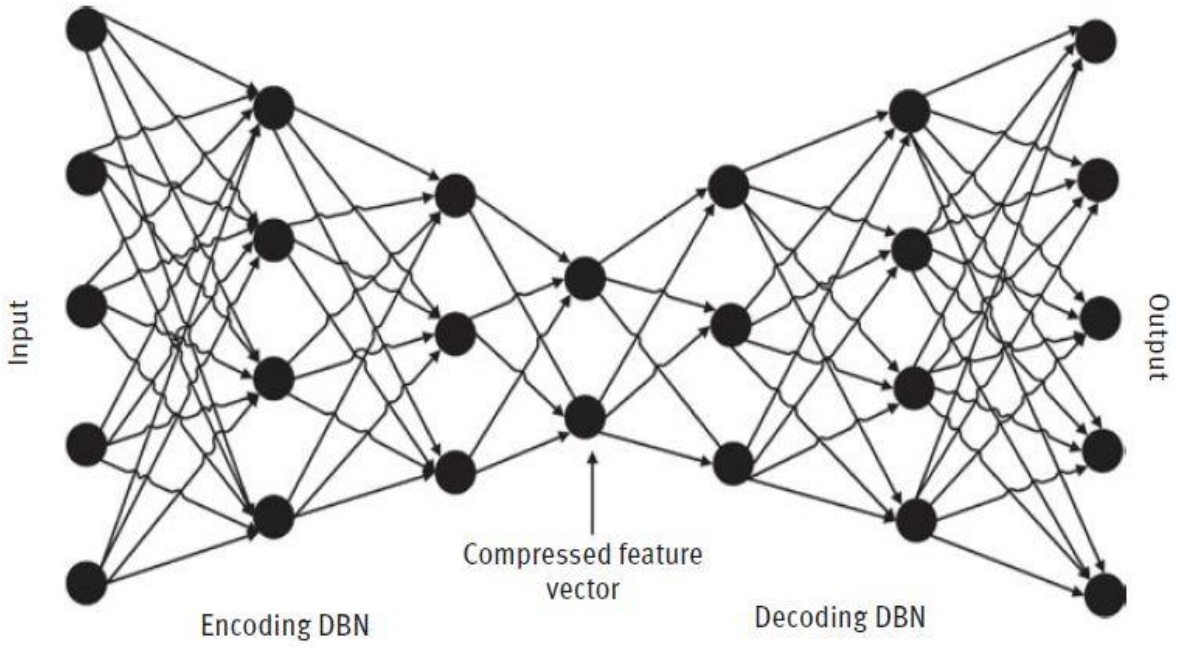

Figure 2.9 Structure of a DBN based autoencoder

The first DBN is called encoding DBN which does the encoding operation to compress the input variables into low dimension feature vectors. The second DBN model decodes the compressed feature vectors to the original data set. It means that the target outputs of an autoencoder are its inputs. Autoencoder has been successfully used in data compression, picture search and information retrieval (Bengio, 2009; Goswami, 2020).

2.5.2 *Deep Learning for Chemical Processes*

The technique of deep learning has been widely used in many high-tech areas. In this section, some applications for industrial processes are introduced.

Data-driven model based on deep learning have some applications in chemical industry for fault classification and monitoring. A novel chemical fault diagnosis model using stacked denoising autoencoder was proposed by Jiang et al. (2016). A DBN model was proposed by Zhang et al.

(2017) for the fault classification in Tennessee Eastman (TE) process. The fault classification model achieved outstanding performance, the diagnosis rate of fault 3 which is one of the most difficult to diagnose achieved to 95%. Li et al (2018) developed a novel method based on deep reinforcement learning to control molecular weight distribution of polymer. The learned control policies achieved by deep reinforcement learning are reliable and robust for the optimisation. Wu et al. (2018) presented a study on using the technique of convolutional neural network (CNN) for fault classification application in TE process. The proposed model reached average fault diagnosis rate 88.2% of all the 20 fault types. In 2019, a novel approach based on deep reinforcement learning with deep deterministic policy gradient was presented for a semi-batch polymerization process by Yan et al (2019). This approach was illustrated to perform multivariate control policies over non-linear polymerization process in simulation environment. It indicated that deep reinforcement learning has better capability of handling complicated chemical process than traditional controller. You and Arumugasamy (2020) presented an adaptive neural fuzzy inference system (ANFIS) model to predict the polycaprolactone molecular weight of enzymatic polymerization process. ANFIS achieved better performance on predicting polycaprolactone molecular weight than conventional models. Karg and Lucia (2021) demonstrated the advantage of capability of deep neural networks for an industrial polymerization process to overcome the problems of moving horizon estimation and nonlinear model predictive control. Yuan et al (2021) introduced a predictive model, STA-ConvBiLSTM, which combines the techniques of Bi-LSTM and CNN. Spatiotemporal attention (STA) was further introduced to this method to avoid high target relevant interactions from being discarded. This novel method improved the accuracy of temperature predictions of a delayed coking unit. Zapf and Wallek (2022) presented a study to developed deep learning based models for multi-objective optimisation of maximizing production margin while minimizing CO_2emissions. The deep artificial neural network can handle large numbers of inputs and achieved reliable optimisation.

Deep learning shows great performance for industrial processes. The data-driven empirical models based on deep learning will be investigated in this work.

2.6 Process Control

The development of process control approaches based on AI does not require the detailed physical and chemistry knowledge about the applications. Process control approaches based on data-driven soft sensor can achieve accuracy and optimal control policy from the historic process data. It adds effectiveness of control strategy in real-time implementations (MacMurray

& Himmelblau, 1995). Data-driven models can provide commendable predictive performances and robustness when dealing with varying dynamics and uncertainty in most chemical processes (Boroushaki, 2003; Chen & Huang, 2004; Galluzzo & Cosenza, 2010). Figure 2.10 shows an AI-based controller.

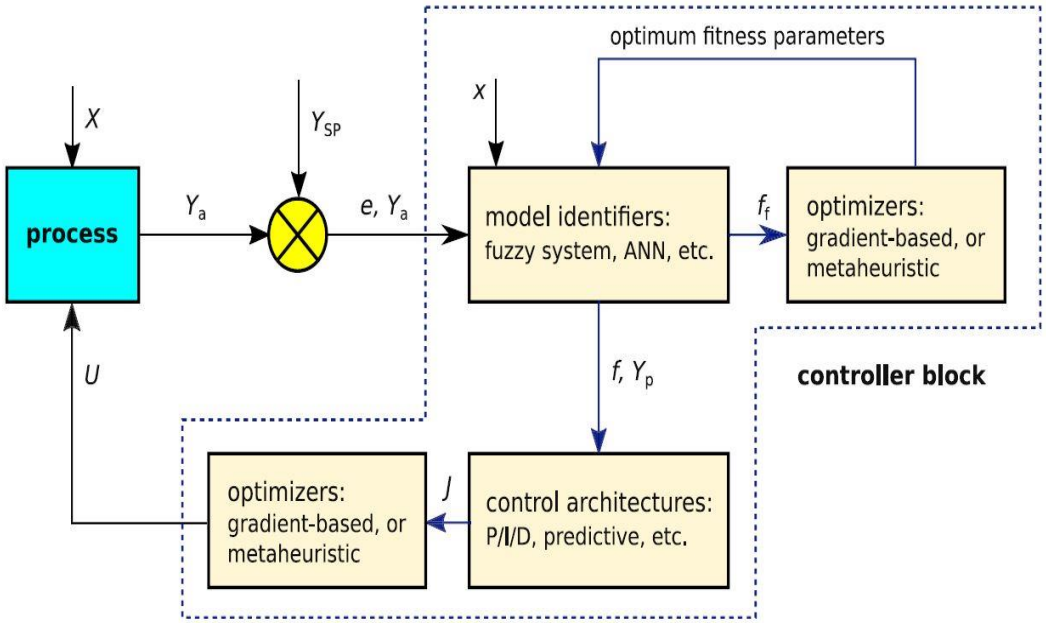

Figure 2.10 A basic implementation of an AI-based controller (Dutta and Upreti, 2021)

From Figure 2.10, the error term, *e*, together with the process variables (*X*) and output (feature), is transferred to model identifiers, such as ANN, fuzzy system, etc. The errors are calculated by output Y_a and desired output Y_{sp}. The model identifiers use gradient algorithms to fine-tune the model to achieve accurate predictions Y_p from the process variables *X*. The objective of model is to minimize the error between Y_p and Y_a. Then the controller will need to optimise a performance index *J* utilising traditional optimiser. The controller may be based on NMPC (Sarimveis & Bafas, 2003), IMC (Varshney & Panigrahi, 2005), PID control (Hojjati, 2003), or fuzzy logic-based control (Galluzzo, 1991). The calculated control action *U* is applied to the process and, as a result, the system is expected to move toward the desired state of minimum *e* or optimum *J*. The subsequent output Y_a gets directed to the controller, which in turn identifies the model as needed, and determines the next control action *U*. After several iterations, the process will reach the optimal status.

2.6.1 *Single-objective Optimisation*

The single-objective optimisation is to achieve the optimal control policy to minimize or maximize a value of key quality variable through an objective function. This type of optimisation only has one optimisation objective.

The nonlinear optimisation can be formulated as

$$\min_{u_j} f(x) \tag{2.42}$$

subject to:

$$d(x) = 0$$

$$k(x) \leq 0$$

$$u_L \leq u_j \leq u_U$$

where $f(x)$ is the optimisation objective function, the control policy u_j is constrained between the lower bound u_L and the upper bound u_U, $k(x)$ and $\mathrm{d}(x)$ are the nonlinear inequality and equality constrains respectively.

For common nonlinear process optimisation, sequential quadratic programming (SQP) is normally used (Schittkowski, 1986).

2.6.2 *Reliable Multi-objective Optimisation of Batch Processes*

Mukherjee and Zhang (2008) propose a reliable multi-objective optimisation control strategy for batch processes using bootstrap aggregated neural networks. In terms of batch process optimisation control, $\boldsymbol{Y_p}$ represents the predicted quality variables, $\boldsymbol{U} = [u_1,\ u_2, \ldots,\ u_N]$ is a vector of control actions (manipulated variables), t_f is the end time of the batch process, and function *f* is the trained prediction function of data-driven empirical model. The network model can be formulated as,

$$\boldsymbol{Y_p}(t_f) = \boldsymbol{f}(\boldsymbol{U}) \tag{2.43}$$

After the training of the network the standard error of the model can be predicted as,

$$\sigma_e(t_f) = g(\boldsymbol{U}) \tag{2.44}$$

where $g(\boldsymbol{U})$ is the formula to calculated the prediction standard errors. The smaller σ_e is the

more reliable the model prediction is.

Therefore the multi-objective optimisation function can be given as,

$$\boldsymbol{F}(\boldsymbol{U}) = \begin{bmatrix} h[Y_p(t_f)] \\ \sigma_e(t_f) \end{bmatrix} \tag{2.45}$$

$$\min_{U} \gamma$$

subject to

$$\boldsymbol{F_i}(\boldsymbol{U}) - \boldsymbol{W_i}\gamma \leq \boldsymbol{F_i^*}$$

$$\boldsymbol{Y_p}(t_f) = f(\boldsymbol{U})$$

$$\sigma_e(t_f) = g(\boldsymbol{U})$$

$$u_L \leq u_j \leq u_U \quad j = 1, 2, \ldots, N$$

where $h[Y(t_f)]$ is the process optimal objectives, γ represents a scalar variable (an auxiliary variable making the new single objective function), W_i is the weighting parameter for the ith objective, F_i^* is the desired goal value for the ith objective, and the control policy has a value constrains between the lower bound u_L and the upper bound u_U .

$\boldsymbol{F_i^*} = [F_1^*, F_2^*, \ldots, F_m^*]$ contains several objectives which are associated with the objective function, $\boldsymbol{F(x)} = \{F_1(x), F_2(x) \ldots, F_m(x)\}$, where m is the number of objectives. The problem formulation allows the objectives to be under- or over-achieved enabling the designer to be relatively imprecise about the initial design goals. The weighting coefficients, $\boldsymbol{W} = \{W_1, W_2 \ldots, W_m\}$, control the relative degree of goals. The weighting vector, W_i, enables the optimiser to express a measure of the relative trade-offs between the objectives. The optimisation can also incorporate hard constrains into the design by setting a particular weighting factor to zero (i.e., $W_i = 0$).

Accurate predictions of quality variables can be obtained by using bootstrap aggregated neural network models. The control actions calculated by the multi-objective optimisation are reliable because enhancing the model prediction confidence is incorporated as addition optimisation objectives (Mukherjee and Zhang, 2008).

2.7 Summary

A review of AI technology and statistical modelling has been carried out. With the great development of AI technology and computational ability, many advanced AI techniques are applied to process control systems for industrial chemical and biochemical processes.

Regression models were reviewed, such as single variable linear regression, multiple linear regression, PCR, PLS and ANN. Advanced techniques have been applied with conventional empirical models. There are many successful ANN applications utilised in process control methods such as PID schemes, IMC and predictive control schemes. However, there are limitations of ANNs. It suffers the issues of overfitting, under fitting due to training trapped in local optima and slow convergence.

The enhanced bootstrap aggregated neural network was introduced. It improves the robustness and accuracy of neural network. The confidence bounds can be incorporated in the optimisation of industrial processes. It enhances the reliability of the optimisation function and improves the control policy for actual industrial processes. A sequential training algorithm can be used to reduce the computation costs in training bootstrap aggregated neural network. Bootstrap aggregated neural network has many successful applications, such as on a crude distillation system, nuclear waste process, a fed-batch fermentation process, etc.

A brief introduction of deep learning was given. The advanced deep learning technique has the ability to be used in process control for complex chemical processes. However, there are few models and applications based on deep learning for regression and optimisation in chemical processes. The single-objective optimisation and multi-objective optimisation has been introduced. It provides the knowledge for optimisation control based on deep learning.

Chapter 3. Inferential Estimation of Polymer Melt Index Using Deep Belief Network

3.1 Introduction

Deep learning has drawn a lot of attention in recent years. Hinton (2006) gave the first introduction of deep learning to solve the issue of lack of generalisation capability of conventional data-driven modelling techniques. According to research, neural networks with shallow structure can lack of representation capabilities and demonstrate limitations in certain learning tasks (Bengio et al. 2005). Specifically, these networks struggle with approximating highly-variable functions that have extreme changes in specific regions. To accurately represent such regions, numerous neurons need to be added to layers. And an adequate amount of training samples is required to ensure desirable generalisation, i.e. giving good performance on unseen data. However, if the training samples are limited, shallow networks may not be able to correctly represent highly varying functions. Recently, it has been suggested that deep neural networks with more than two layers of nonlinearities can effectively represent highly varying functions (Bengio et al. 2005). Many industrial chemical processes are highly nonlinear and difficult to be modelled by first principle models. Conventional neural networks suffer the problem of lack of generalisation capability due to over-fitting or under-fitting due to training trapped in local optima. To overcome this issue, the technique of deep learning can be utilised. Deep belief network (DBN) is one kind of well-known data-driven deep learning model. Compared with conventional feed-forward neural networks, deep belief network has a deep structure with advanced learning algorithms. It has shown strong generalisation capability and many successful applications in many areas such as speech recognition, image classification and fault diagnosis (Jiang et al., 2016; Toledano et al., 2018; Abdul et al., 2020). Few applications of DBN for industrial processes have been reported. In this chapter, a DBN model is developed for the inferential estimations of polymer MI in an industrial polypropylene polymerization process. By using deep learning technique, large amount of industrial process data samples without the corresponding labels (i.e. the corresponding target values) can also be used by DBN model in the learning procedure. It improves the performance of DBN and achieves accurate estimations of MI.

This chapter is organized as follows, Section 3.2 presents DBN model and the main principles of restricted Boltzmann machines (RBMs) and back-propagation are introduced. In Section 3.3, the case study of an industrial polypropylene polymerization process is given. The selection of DBN model architectures is discussed and the polymer melt index estimation results are given

in Section 3.4. Section 3.5 summarises the conclusions of this chapter.

3.2 Deep Belief Network

3.2.1 *Structure of Deep Belief Network*

The limitation of traditional neural networks is that they usually have shallow structures. There are typically no more than three layers in a conventional neural network model. Many actual industrial processes are commonly highly nonlinear. The shallow architecture of feed-forward neural network could lead to the lack of representation capability when modelling highly complex nonlinear processes (Bengio et al 2006; 2007). To approximate various operating regions of a process, the model needs more hidden neurons added to the hidden layers. It is suggested that networks with deep structures can achieved reliable results in recent research (Bengio et al, 2006). DBN has been successfully applied to many research areas, such as classification and recognition (Tang, Salakhutdinov & Hinton 2012). In a DBN model, several restricted Boltzmann machines (RBMs) are stacked and combined as one learning network. DBN is developed with a deep structure based on deep learning technique. Figure 3.1 presents the basic architecture of DBN.

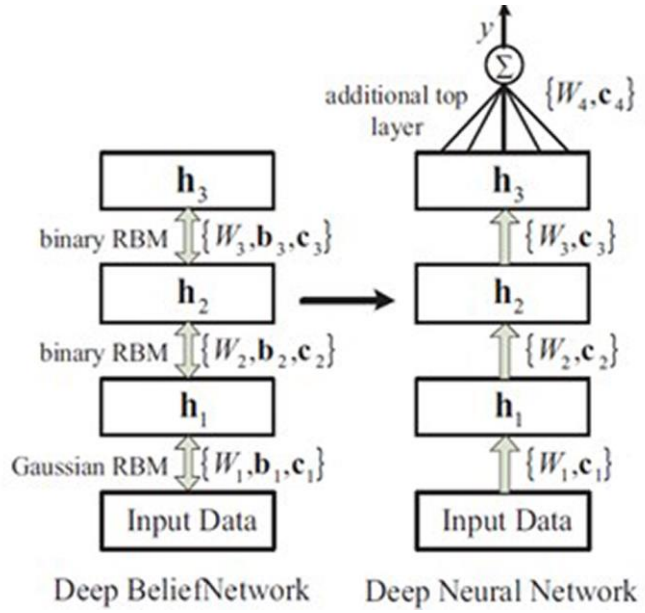

Figure 3.1 The architecture of DBN

The DBN shown in Figure 3.1 has five layers, an input layer, an output layer and three hidden layers. In this figure, $\mathbf{W}_1$ to $\mathbf{W}_4$ are the weights of the network, $\mathbf{b}_1$ to $\mathbf{b}_3$ and $\mathbf{c}_1$ to $\mathbf{c}_4$ are bias of the network. It can be considered that DBN is a combination of stacking RBMs. Each hidden layer of DBN is regarded as one single RBM. Compared with traditional Boltzmann machine,

the neurons in a hidden layer of DBN are not connected to each other. However, the layers in a network have symmetrical connections with each other. The units in hidden layers are binary units and the visible input layer units are Gaussian units. The first phase of training is unsupervised training and the process operational data are used to train the DBN model without any target variables involved. RBMs are generative models that learn to model the joint probability distribution of process variables. They are initially trained by an unsupervised learning algorithm to learn the latent feature representations of the input data. The training objective of RBMs is to maximize the likelihood of the observed data. The likelihood function is given by the product of the probabilities of each training example (Hinton et al. 2006). The unsupervised training helps the DBN to mine more correlations than the conventional feed-forward neural network. The weights are adjusted in a desired region before the supervised training phase. After unsupervised training, DBN is fine-tuned by the backpropagation algorithm in the supervised training phase.

3.2.2 *Restricted Boltzmann Machines*

In the 1980s, Paul Smolensky developed Restricted Boltzmann machine (Smolensky, 1986). Hinton et al. (2006) developed DBN by stacking RBMs as the layers of DBN. A DBN contains stacked RBMs as shown in Figure 3.1.

To understand the basics of RBM, the probability function between visible units and hidden units needs to be introduced first. Equation (3.1) shows the probability function,

$$P(\boldsymbol{v}, \boldsymbol{h}) = (\exp\{-E(\boldsymbol{v}, \boldsymbol{h})\})/Z \tag{3.1}$$

where Z represents a normalizing factor, **v** represents the vector of visible layer, **h** represents the vector of hidden layer. The probability P(**v**) increases when the energy function decreases. In the RBM, the energy function is given by,

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\boldsymbol{b}^T\boldsymbol{v} - \boldsymbol{c}^T\boldsymbol{h} - \boldsymbol{h}^T\boldsymbol{W}\boldsymbol{v} \tag{3.2}$$

where, **W, b** and **c** are parameters of the function. It should be noticed that the vector **v** and the vector **h** are both binary-valued. The state of $\boldsymbol{v_i}$ is 1 if unit i is on and 0 otherwise. Binary RBMs are used as hidden layers in a DBN model. When **v** and **h** are known, the conditional probability can be calculated by,

$$P(\boldsymbol{h_j}|\boldsymbol{v_i}, \theta) = sigm(\textstyle\sum_{i=1}^{|\boldsymbol{v}|} \boldsymbol{W_{ij}}\boldsymbol{v_i} + \boldsymbol{b_j}) \tag{3.3}$$

$$P(\boldsymbol{v_i}|\boldsymbol{h_j}, \theta) = sigm(\textstyle\sum_{j=1}^{|\boldsymbol{h}|} \boldsymbol{W_{ij}}\boldsymbol{h_j} + \boldsymbol{c_i}) \tag{3.4}$$

where $sigm(x) = 1/(1 + e^{-x})$ is the sigmoid function.

However, binary RBM cannot be used to deal with continuous variables. To overcome this issue, (3.2) can be extended to energy function of Gaussian RBM,

$$E(\boldsymbol{v},\boldsymbol{h}) = \sum_i \frac{(v_i - a_i)^2}{2\sigma_i^2} - \boldsymbol{c}^T\boldsymbol{h} - \boldsymbol{h}^T\boldsymbol{W}\boldsymbol{v} \tag{3.5}$$

where a_i is the mean of Gaussian distribution, σ_i is the standard deviation of Gaussian distribution for input neuron. The samples of input data are commonly normalized to zero mean and unit variance in practical applications. Therefore, (3.5) can be changed to,

$$E(\boldsymbol{v},\boldsymbol{h}) = \frac{1}{2}\boldsymbol{v}^T\boldsymbol{v} - \boldsymbol{b}^T\boldsymbol{v} - \boldsymbol{c}^T\boldsymbol{h} - \boldsymbol{h}^T\boldsymbol{W}\boldsymbol{v} \tag{3.6}$$

Hinton also described other forms of RBMs (Hinton 2012), but the DBN in this chapter only uses Gaussian RBM and binary RBM.

3.2.3 *Learning Algorithm for RBM*

The objective of training RBM is to maximize the probability P(**v**), which can be achieved by minimizing the energy function. From Gibbs sampling, **h** can only be sampled from **v** of visible layers. the gradient at a visible point **v** can be formulated as:

$$\frac{\partial log P(\boldsymbol{v})}{\partial\theta} = \frac{\partial log \sum_h P(\boldsymbol{v},\boldsymbol{h})}{\partial\theta} = \frac{\sum_h e^{-E(\boldsymbol{v},\boldsymbol{h})}\left(\frac{\partial[-E(\boldsymbol{v},\boldsymbol{h})]}{\partial\theta}\right)}{\sum_h e^{-E(\boldsymbol{v},\boldsymbol{h})}} - \frac{\sum_{\tilde{\boldsymbol{v}}}\sum_{\boldsymbol{h}} e^{-E(\tilde{\boldsymbol{v}},\boldsymbol{h})}\left(\frac{\partial[-E(\tilde{\boldsymbol{v}},\boldsymbol{h})]}{\partial\theta}\right)}{\sum_{\tilde{\boldsymbol{v}}}\sum_{\boldsymbol{h}} e^{-E(\tilde{\boldsymbol{v}},\boldsymbol{h})}}$$

$$= \sum_h P(\boldsymbol{h}|\boldsymbol{v})\frac{\partial[-E(\boldsymbol{v},\boldsymbol{h})]}{\partial\theta} - \sum_{\tilde{\boldsymbol{v}}}\sum_h P(\tilde{\boldsymbol{v}},\boldsymbol{h})\frac{\partial[-E(\tilde{\boldsymbol{v}},\boldsymbol{h})]}{\partial\theta} \tag{3.7}$$

where θ={**W**,**b**,**c**} is a vector of the network parameters. The positive part of the equation is conditional expectation of $\frac{\partial[-E(\boldsymbol{v},\boldsymbol{h})]}{\partial\theta}$. Computing the positive term in (3.7) is easy because the visible vector **v** has been known. The conditional probabilities for the binary RBM can be given by,

$$P(h_j = 1|v) = \frac{e^{C_j + W_j v}}{1 + e^{C_j + W_j v}} = sigm(c_j + W_{:,j}\boldsymbol{v}) \tag{3.8}$$

$$P(v_i = 1|h) = \frac{e^{b_i + W_i^T h}}{1 + e^{b_i + W_i^T h}} = sigm\left(b_i + W_{i,:}^T\boldsymbol{h}\right) \tag{3.9}$$

$W_{:,j}$ is the *j*th row of **W**, $W_{i,:}$ is the *i*th column of **W** It can be seen that the weights for binary RBM input units **v** and hidden units **h** are symmetrical. For the Gaussian RBM, these two terms can be formulated by,

$$P(h_j = 1|v) = \frac{e^{c_j + W_j v}}{1 + e^{c_j + W_j v}} = sigm\left(c_j + W_{:,j}\boldsymbol{v}\right) \tag{3.10}$$

$$P(v_i|h) = \frac{1}{\sqrt{2\pi}}\left\{-\frac{1}{2}(v_i - b_i - W_{i,:}^T\boldsymbol{h})^2\right\} \sim N(b_i + W_{i,:}^T\boldsymbol{h}, 1) \tag{3.11}$$

$P(v_i|h)$ can be considered as following a normal distribution with mean of $b_i + W_{i,:}^T \boldsymbol{h}$ and unit variance.

The other part of learning algorithm, the negative term in equation (3.7), is the $\frac{\partial[-E(\boldsymbol{v},\boldsymbol{h})]}{\partial\theta}$ of joint distribution $P(\tilde{\boldsymbol{v}}, \boldsymbol{h})$ with parameter θ. There is an intractable problem to calculate it. To overcome this issue, an approach of contrastive divergence (CD) was presented by Hinton et al. (2006).

The CD deals with the approximation of negative terms of Equation (3.7):

$$\sum_{\tilde{v}} \sum_h P(\tilde{\boldsymbol{v}}, \boldsymbol{h}) \frac{\partial[-E(\tilde{\boldsymbol{v}},\boldsymbol{h})]}{\partial\theta} \tag{3.12}$$

The method of Gibbs sampling can solve this problem effectively. Figure 3.2 shows the Markov chain and CD for RBM. The first step of sampling is to sample $\boldsymbol{h}^{(t)}$ from $P(\boldsymbol{h}|\boldsymbol{v} = \boldsymbol{v}^{(t-1)})$, then sample $\boldsymbol{v}^{(t)}$ from $P(\boldsymbol{v}|\boldsymbol{h} = \boldsymbol{h}^{(t)})$. After the infinite iterations of sampling steps, the $\boldsymbol{v}^{(\infty)}$ and $\boldsymbol{h}^{(\infty)}$ are the ideal results from the Markov chain. However, in practical situation, just one iteration of Gibbs sampling can achieve a satisfied result and the learning algorithm works well. Therefore, to obtain $\boldsymbol{v}^{(1)}$ and $\boldsymbol{h}^{(1)}$ is adopted by CD algorithm.

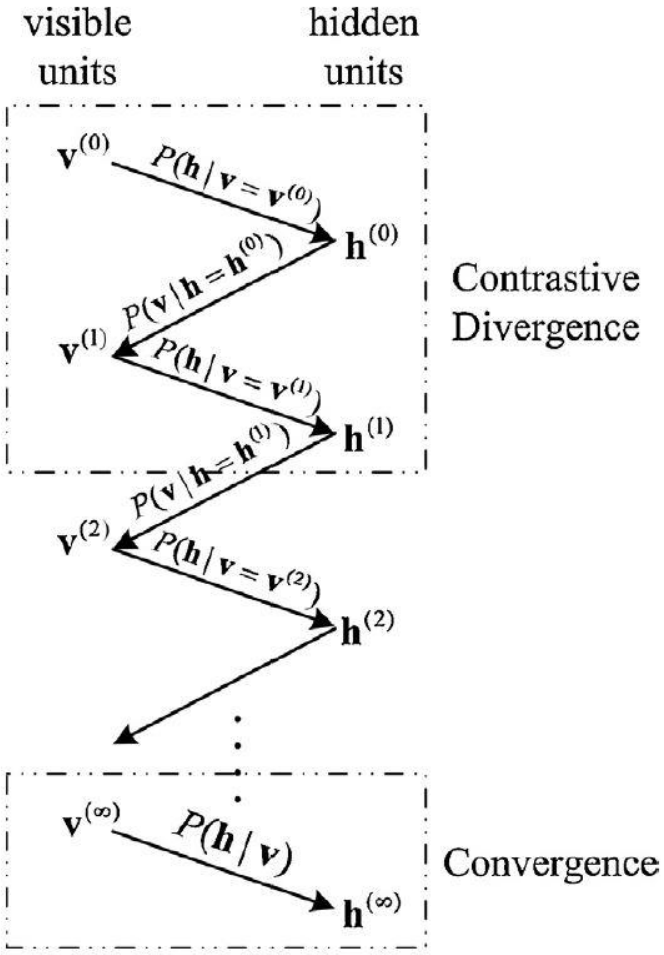

Figure 3.2 Markov chain in two-stage Gibbs sampler and CD

For both binary and Gaussian RBMs, the term $\frac{\partial[-E(\boldsymbol{v},\boldsymbol{h})]}{\partial\theta}$ is computed as:

$$\frac{\partial[-E(\boldsymbol{v},\boldsymbol{h})]}{\partial W_{ij}} = h_j v_i \tag{3.13}$$

$$\frac{\partial[-E(\boldsymbol{v},\boldsymbol{h})]}{\partial b_i} = v_i \tag{3.14}$$

$$\frac{\partial[-E(\boldsymbol{v},\boldsymbol{h})]}{\partial c_j} = h_j \tag{3.15}$$

And the gradient of log-likelihood function is given as,

$$\Delta W_{ij} = \frac{\partial \log P(\mathbf{v})}{\partial W_{ij}} = \sum_h P(\boldsymbol{h}|\boldsymbol{v}) \cdot h_j v_i - \sum_{\tilde{v},h} P(\tilde{v},\boldsymbol{h}) \cdot h_j \tilde{v}_i \approx E\left(h_j^{(0)}\middle|\boldsymbol{v}^{(0)}\right) \cdot v_i^{(0)} - h_j^{(1)} v_i^{(1)} \approx$$

$$E\left(h_j^{(0)}\middle|\boldsymbol{v}^{(0)}\right) \cdot v_i^{(0)} - E\left(h_j^{(1)}\middle|\boldsymbol{v}^{(1)}\right) \cdot E\left(v_i^{(1)}\middle|\boldsymbol{h}^{(0)}\right) = sigm\left(c_j + W_{:,j}\boldsymbol{v}^{(0)}\right) \cdot v_i^{(0)} - sigm(c_j +$$

$$W_{:,j}\boldsymbol{v}^{(1)}) \cdot sigm(b_i + W_{i,:}^T \boldsymbol{h}^{(0)}) \tag{3-16}$$

$$\Delta b_i = \frac{\partial \log P(\mathbf{v})}{\partial b_i} = \sum_h P(\boldsymbol{h}|\boldsymbol{v}) \cdot v_i - \sum_{\tilde{v},h} P(\tilde{v},\boldsymbol{h}) \cdot \tilde{v}_i \approx v_i^{(0)} - E\left(\boldsymbol{v}_i^{(1)}\middle|\boldsymbol{h}^{(0)}\right) = v_i^{(0)} -$$

$$sigm(b_i + W_{i,:}^T \boldsymbol{h}^{(0)}) \tag{3-17}$$

$$\Delta c_j = \frac{\partial \log P(\mathbf{v})}{\partial c_j} = \sum_h P(\boldsymbol{h}|\boldsymbol{v}) \cdot h_j - \sum_{\tilde{v},h} P(\tilde{v},\boldsymbol{h}) \cdot h_j \approx E\left(\boldsymbol{h}_j^{(0)}\middle|\boldsymbol{v}^{(0)}\right) - E\left(\boldsymbol{h}_j^{(1)}\middle|\boldsymbol{v}^{(1)}\right) =$$

$$sigm\left(c_j + W_{:,j}\boldsymbol{v}^{(0)}\right) - sigm\left(c_j + W_{:,j}\boldsymbol{v}^{(1)}\right) \tag{3-18}$$

where $\boldsymbol{h}^{(0)}, \boldsymbol{v}^{(1)}$ *and* $\boldsymbol{h}^{(1)}$ are sampled from Markov chain. In Eq (3-16,17,18), the conditional expectation is used instead of the binary states sampled from the one-step Markov chain.

For the Gaussian units, the equations can be formulated as,

$$\Delta W_{ij} = \frac{\partial \log P(\mathbf{v})}{\partial W_{ij}} = \sum_h P(\boldsymbol{h}|\boldsymbol{v}) \cdot h_j v_i - \sum_{\tilde{v},h} P(\tilde{v},\boldsymbol{h}) \cdot h_j \tilde{v}_i \approx E\left(h_j^{(0)}\middle|\boldsymbol{v}^{(0)}\right) \cdot v_i^{(0)} - h_j^{(1)} v_i^{(1)} \approx$$

$$E\left(h_j^{(0)}\middle|\boldsymbol{v}^{(0)}\right) \cdot v_i^{(0)} - E\left(h_j^{(1)}\middle|\boldsymbol{v}^{(1)}\right) \cdot E\left(v_i^{(1)}\middle|\boldsymbol{h}^{(0)}\right) = sigm\left(c_j + W_{:,j}\boldsymbol{v}^{(0)}\right) \cdot v_i^{(0)} - sigm(c_j +$$

$$W_{:,j}\boldsymbol{v}^{(1)}) \cdot (b_i + W_{i,:}^T \boldsymbol{h}^{(0)}) \tag{3-19}$$

$$\Delta b_i = \frac{\partial \log P(\mathbf{v})}{\partial b_i} = \sum_h P(\boldsymbol{h}|\boldsymbol{v}) \cdot v_i - \sum_{\tilde{v},h} P(\tilde{v},\boldsymbol{h}) \cdot \tilde{v}_i \approx v_i^{(0)} - E\left(\boldsymbol{v}_i^{(1)}\middle|\boldsymbol{h}^{(0)}\right) = v_i^{(0)} - (b_i +$$

$$W_{i,:}^T \boldsymbol{h}^{(0)}) \tag{3-20}$$

$$\Delta c_j = \frac{\partial \log P(\mathbf{v})}{\partial c_j} = \sum_h P(\boldsymbol{h}|\boldsymbol{v}) \cdot h_j - \sum_{\tilde{v},h} P(\tilde{v},\boldsymbol{h}) \cdot h_j \approx E\left(\boldsymbol{h}_j^{(0)}\middle|\boldsymbol{v}^{(0)}\right) - E\left(\boldsymbol{h}_j^{(1)}\middle|\boldsymbol{v}^{(1)}\right) =$$

$$sigm\left(c_j + W_{:,j}\boldsymbol{v}^{(0)}\right) - sigm\left(C_j + W_{:,j}\boldsymbol{v}^{(1)}\right) \tag{3-21}$$

3.2.4 *Supervised Training through Back-propagation*

Back-propagation is the most commonly used supervised training approach to train neural networks. After the unsupervised training phase, back-propagation algorithm will fine tune the whole network weights in the supervised training phase. The errors between the network outputs and the corresponding labels (targets) are computed and back propagated to the previous layer. Equation (3.22) shows the error terms,

$$\boldsymbol{Err}_j = \boldsymbol{O}_j\left(1 - \boldsymbol{O}_j\right)\left(\boldsymbol{T}_j - \boldsymbol{O}_j\right) \tag{3.22}$$

where $\boldsymbol{O}_j$ represents the network output for a training sample, $\boldsymbol{T}_j$ is the corresponding target value for the *j*th output neuron. The error term of hidden layers is formulated as,

$$\boldsymbol{Err}_j = \boldsymbol{O}_j\left(1 - \boldsymbol{O}_j\right)\sum_k \boldsymbol{Err}_k \boldsymbol{w}_{jk} \tag{3.23}$$

where $\boldsymbol{w}_{jk}$ is the vector of weights connecting output layer and the last hidden layer, $\boldsymbol{Err}_k$ is the error term of output layer. During training, the weight updating is transferred from the output layer to the input layer. The formulas of weight updating are given as,

$$\boldsymbol{w}_{ij} = \boldsymbol{w}_{ij} + \eta \boldsymbol{Err}_j \boldsymbol{O}_i \tag{3.24}$$

$$\boldsymbol{c}_j = \boldsymbol{c}_j + \eta \boldsymbol{Err}_j \tag{3.25}$$

where η is learning rate of the training process, $\boldsymbol{w}_{ij}$ and $\boldsymbol{c}_j$ are the vectors of weights and bias respectively. The learning rate needs to be properly selected. A large learning rate may miss the minimum whereas a small learning rate usually leads to slow training speed.

As described earlier, the training of DBN contains an unsupervised training phase and a supervised training phase. The initial weights are adjusted to an appropriate region in the unsupervised training procedure. The whole network is then fine-tuned by backpropagation in the supervised training phase to achieve accurate modelling results. The profuse latent information extracted from input variables during the unsupervised training is more interpretable. This semi-supervised method improves the robustness and generalisation capability of model with a deep architecture.

3.3 Polypropylene Polymerization Process

Advanced monitoring, control, and optimisation techniques are essential in modern industrial chemical processes to overcome the issue of high cost and improve the efficiency of production (Gao et al. 2018). In this chapter, DBN is used to develop soft sensors for a polypropylene production plant in China. In this plant, two continuous stirred tank reactors (CSTR) and two

fluidized-bed reactors (FBR) are used to produce polypropylene as shown in Figure 3.3. Propylene, hydrogen, and catalyst are fed to reactors. Reactants for the growing polymer particles are these gases and liquids fed to the reactors. They are also the provider of the heat transfer media. Melt index of polymer is a key polymer quality variable and should be closely monitored and controlled. MI of polypropylene depends on many factors like catalyst, reactor temperature and concentration of the reaction materials. For example, hydrogen can increase the polymerization rate of polypropylene. It mainly increases the initial polymerization rate of propylene (Soares & Hamielec, 1996). The hydrogen concentration regulates the molecular weight of polypropylene. Hydrogen can also delay the decay rate of catalyst. Due to the difficulty of measuring polymer MI in this process, the relationship between MI and some process variables which can be measured easily during the process need to be found. The inferential estimation of MI can be obtained by soft sensors. As this industrial process is very complicated, it is difficult to develop first principle models linking polymer MI with easy-to-measure process variables. Therefore, nonlinear data-driven models need to be utilised in developing soft sensors for this process.

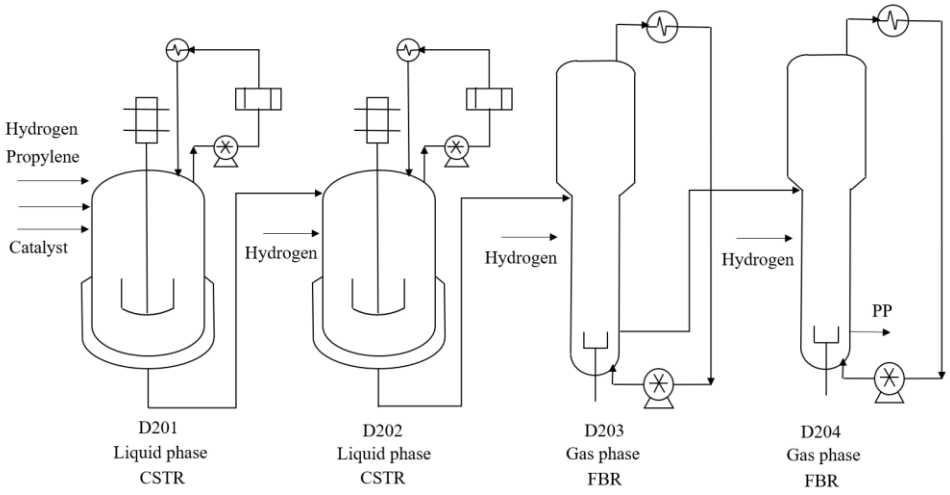

Figure 3.3 The propylene polymerization process

The polypropylene grades are related to some key variables, such as reactant composition, reactor temperature and catalyst properties. Based on the researches of Zhang et al. (2006), the feedstock of D201 are propylene, hydrogen and catalyst. The co-monomer is added to D204. Several grades of polymers were produced within one month. Industrial process operational data covering this time period are available for this study. In this process, polymer MI were measured by offline analysis and logged every two hours. On-line measurements of process variables were logged every half hour. In fact, not all the process variables of the original data have high correlation with the MI of the polymer. In order to enhance the performance of the

model, the method of statistical correlation analysis had been carried out for this study. The correlation analysis of the process data and quality data are shown in Table 3.1. High correlations exist between MI of polymer in reactor D204 and hydrogen concentration in reactor D201 and reactor D202. MI of polymer in reactor D201 is highly relevant with the hydrogen concentration and feed rate in reactor D201. Feed rate of hydrogen, concentration of hydrogen in D201 and D202 and MI of polypropylene in reactor D201 and D204 are shown in Figure 3.4 to Figure 3.6 respectively. Due to the industrial confidentiality, the units of these variables are not disclosed.

	H_2 in reactor D201	Feed rate of hydrogen in reactor D201	H_2 in reactor D202
MI in reactor D201	0.9618	0.8763	0.8615
MI in reactor D204	0.9219	0.8422	0.9358

Table 3.1 Correlation between hydrogen concentration, feed rate of hydrogen and MI.

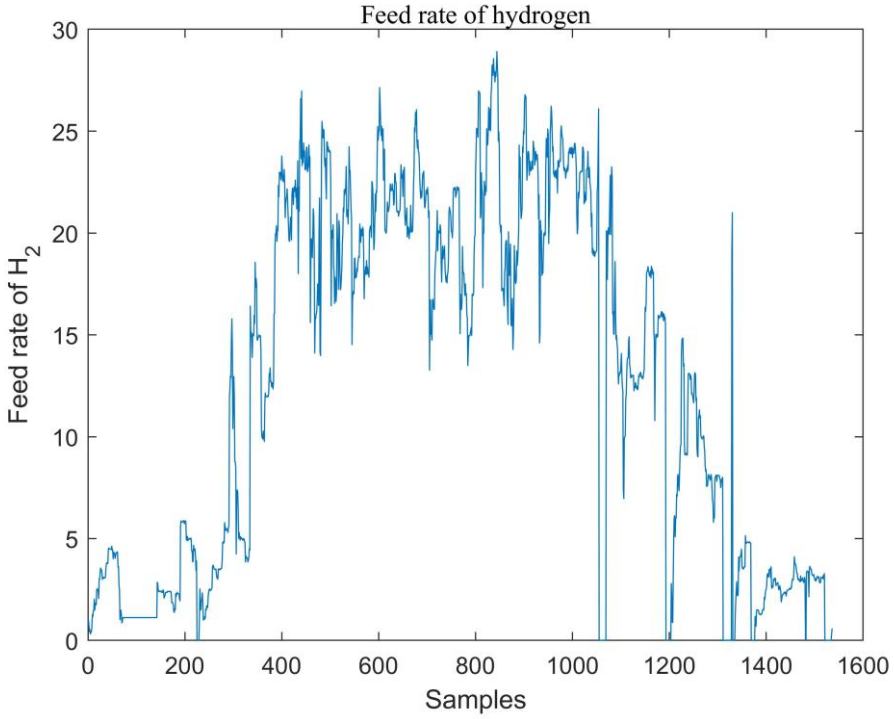

Figure 3.4 Feed rate of hydrogen

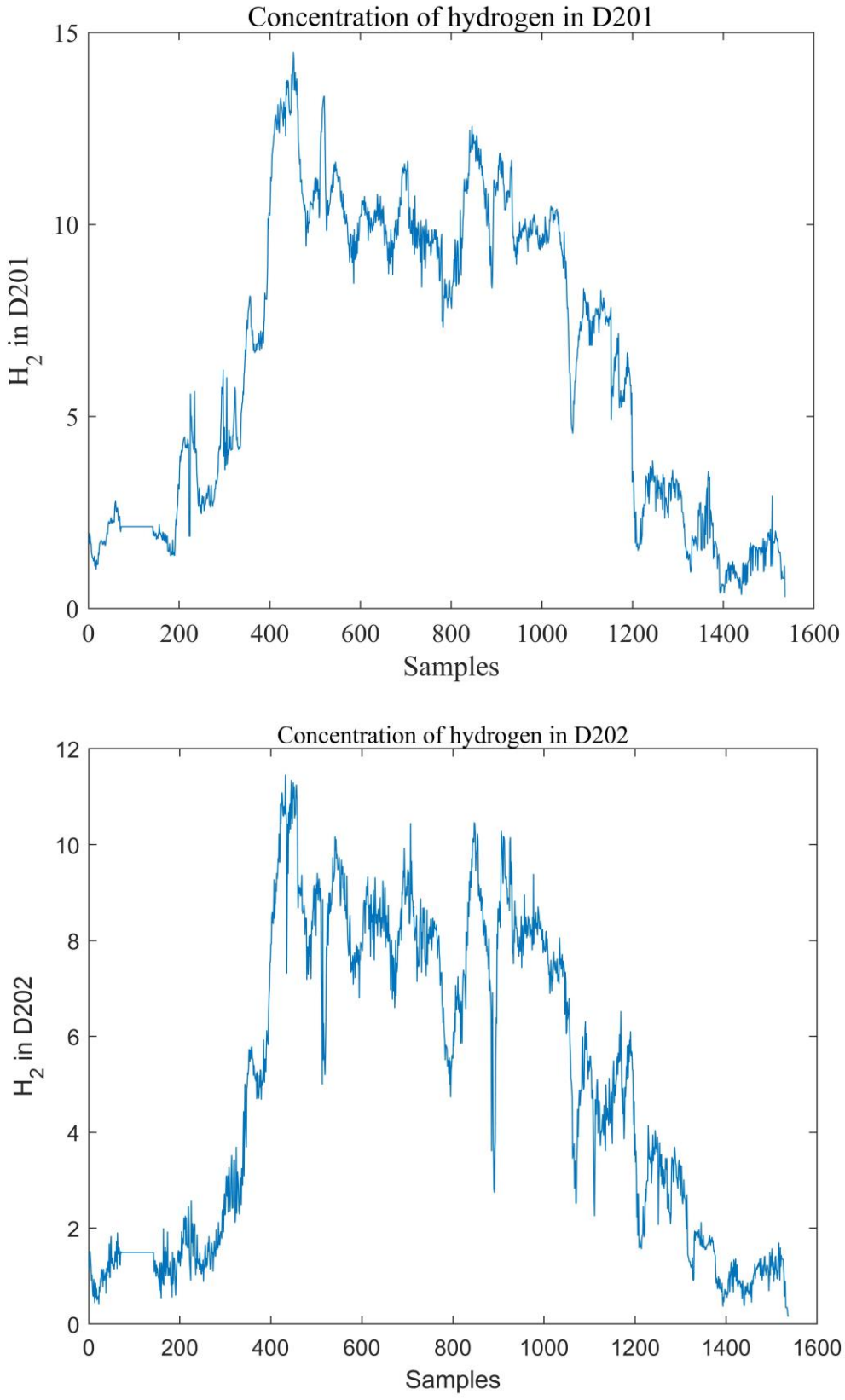

Figure 3.5 Concentration of hydrogen in D201 (top) and D202 (bottom)

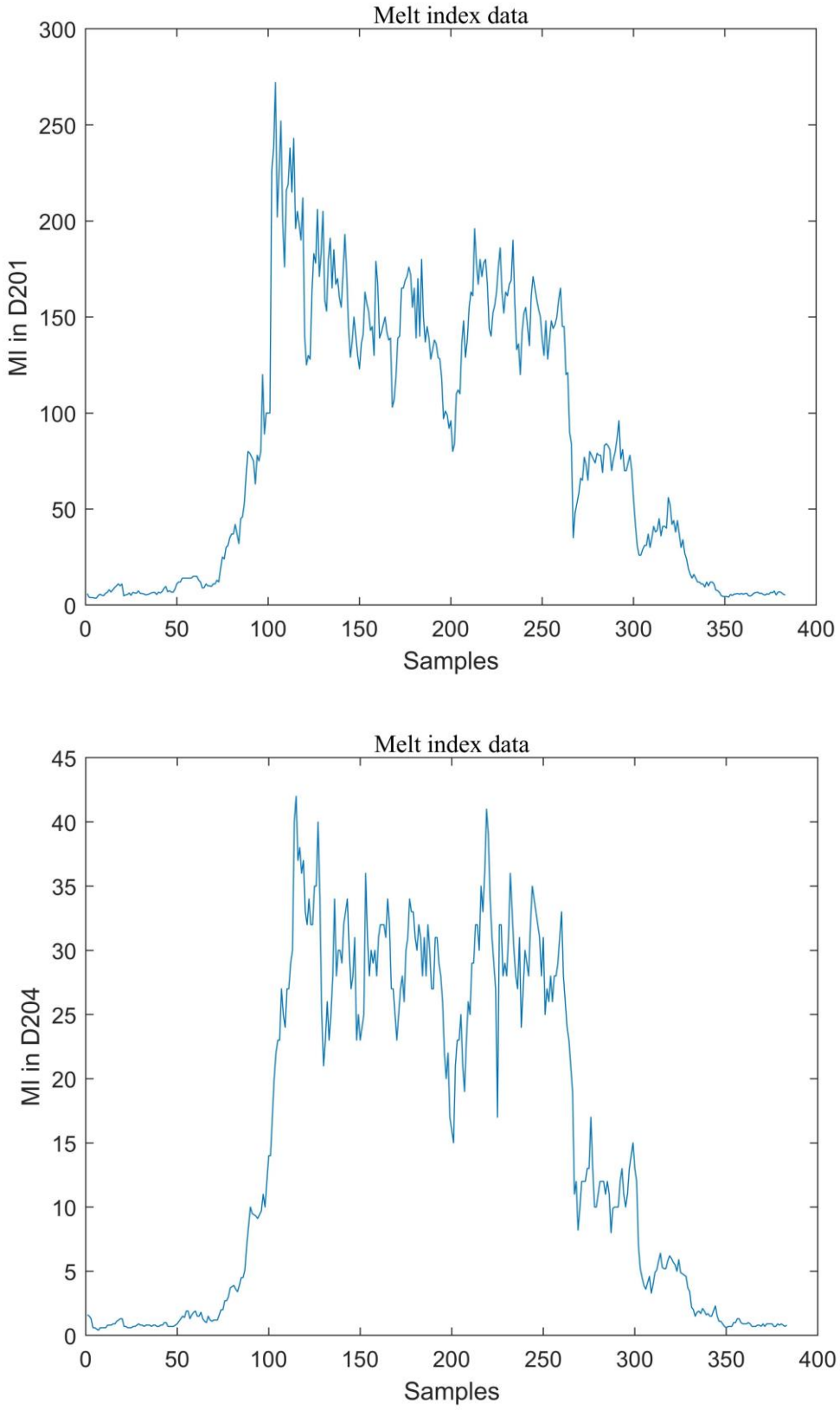

Figure 3.6 Melt index in D201 (top) and D204 (bottom)

From Figure 3.6, it can be observed that the MI data cover quite wide range. Thus, the data are suitable for developing data-driven models. Soft sensor should extract the information from limited process data and quality data to obtain accurate estimation of MI. From the Table 3.1 displayed, it can be seen that MI is highly correlated with hydrogen feed rate and concentration. The time delay of the industrial process can be found based on the cross-correlations analysis. Figures 3.7 to 3.10 show the cross-correlation analysis between hydrogen concentration, feed rate of hydrogen and MI of the polymer. From Figure 3.7, there is no time delay between the hydrogen concentration and MI in reactor D201. The maximum value of the cross-correlation is at 0-time lag. Figure 3.8 shows that there are 2 hours delay between the feed rate of hydrogen in reactor D201 and MI in reactor D201. Because the maximum value of the cross-correlation is at -2h. Figure 3.9 indicated that there was almost 1.5h time delay between the hydrogen concentration in reactor D201 and MI in reactor D204. From Figure 3.10, there was one and a half hour time delay exist between the hydrogen concentration in reactor D202 and MI in reactor D204. To improve the accuracy of estimation of MI, it is recommended to examine the cross-correlations between the MI and the process variables that have been shifted by multiple units of the smaller sampling interval (0.5 h). Figure 3.11 shows the maximum cross-correlations between the hydrogen concentration, feed rate of the hydrogen and MI occur at zero time lag.

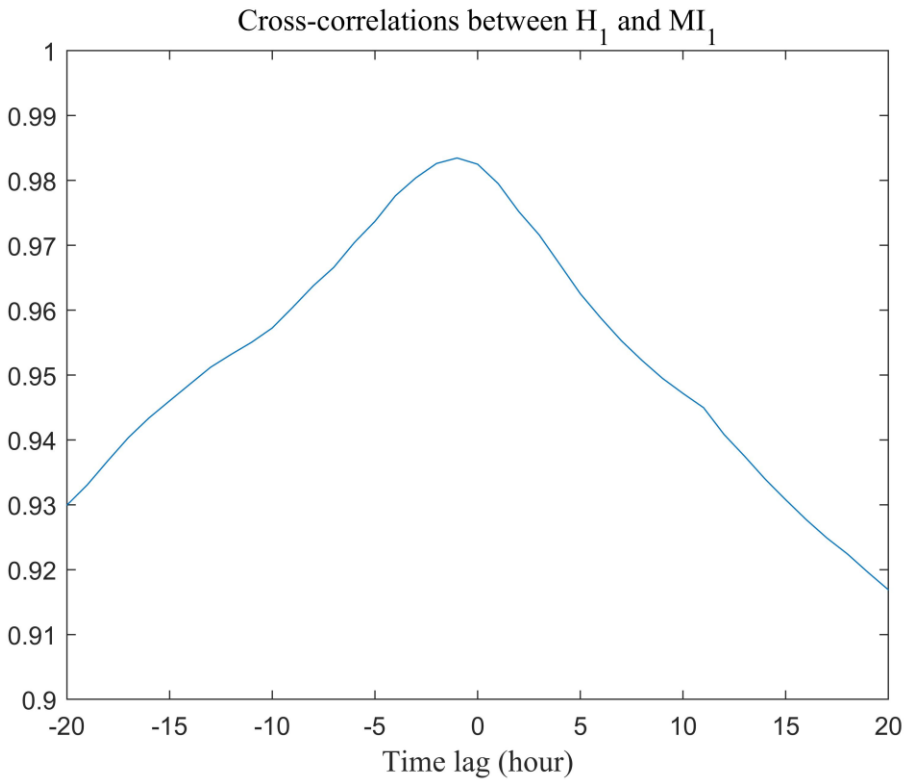

Figure 3.7 Cross-correlation between hydrogen concentration and MI in D201

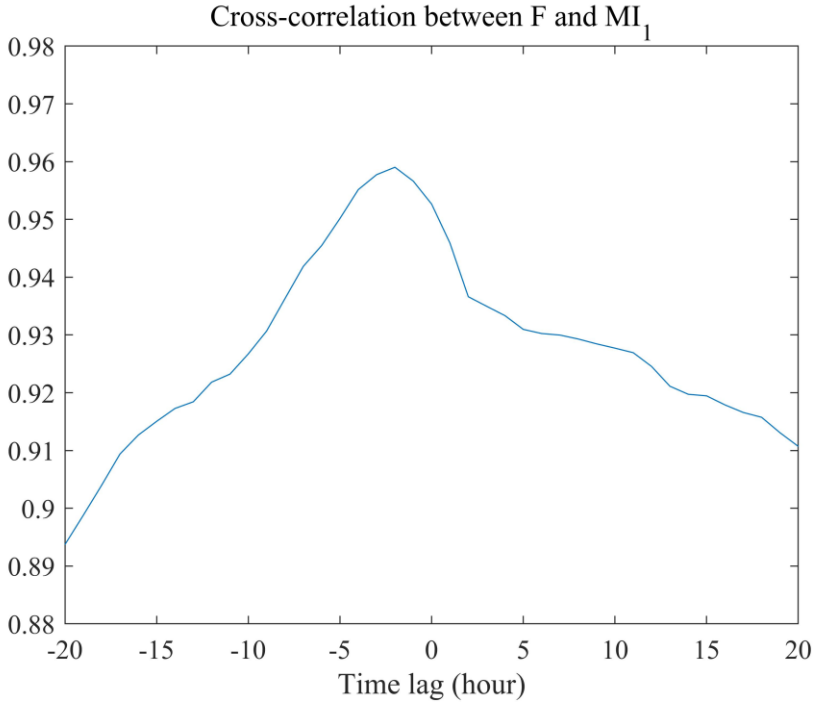

Figure 3.8 Cross-correlation between feed rate and MI in reactor D201

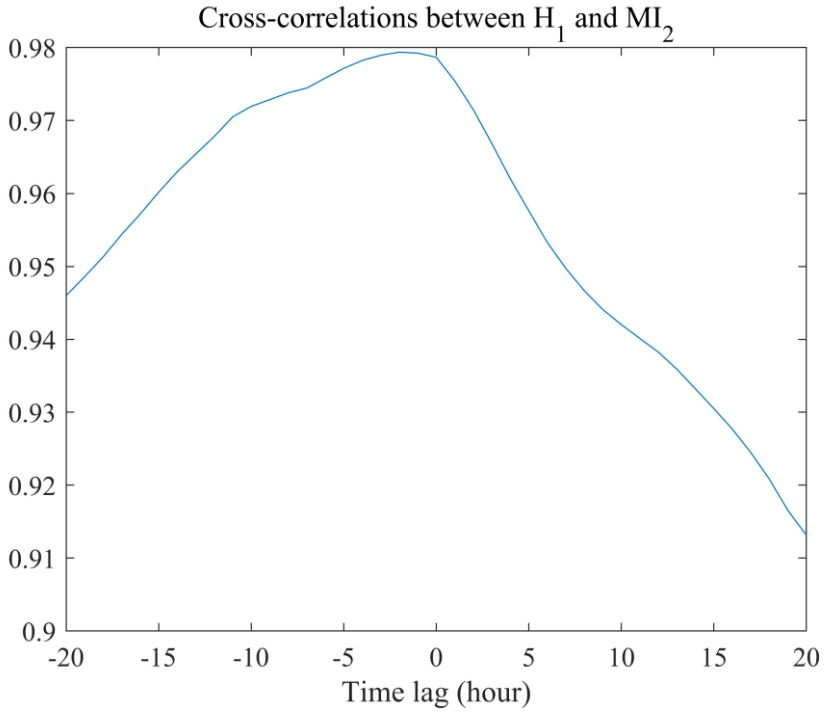

Figure 3.9 Cross-correlation between hydrogen concentration in D201 and MI in D204

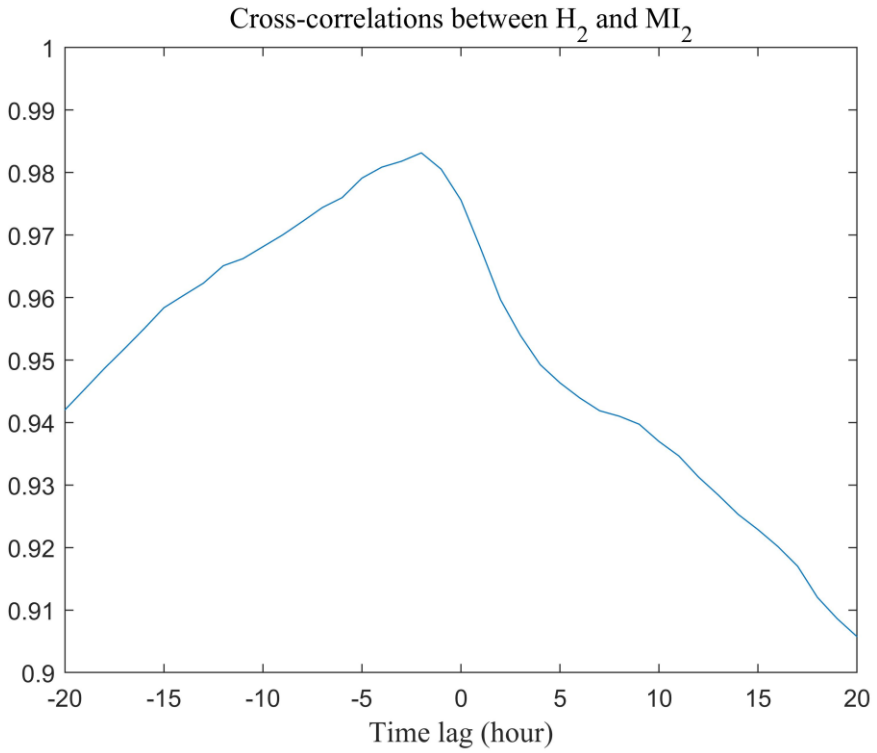

Figure 3.10 Cross-correlation between hydrogen concentration in D202 and MI in D204

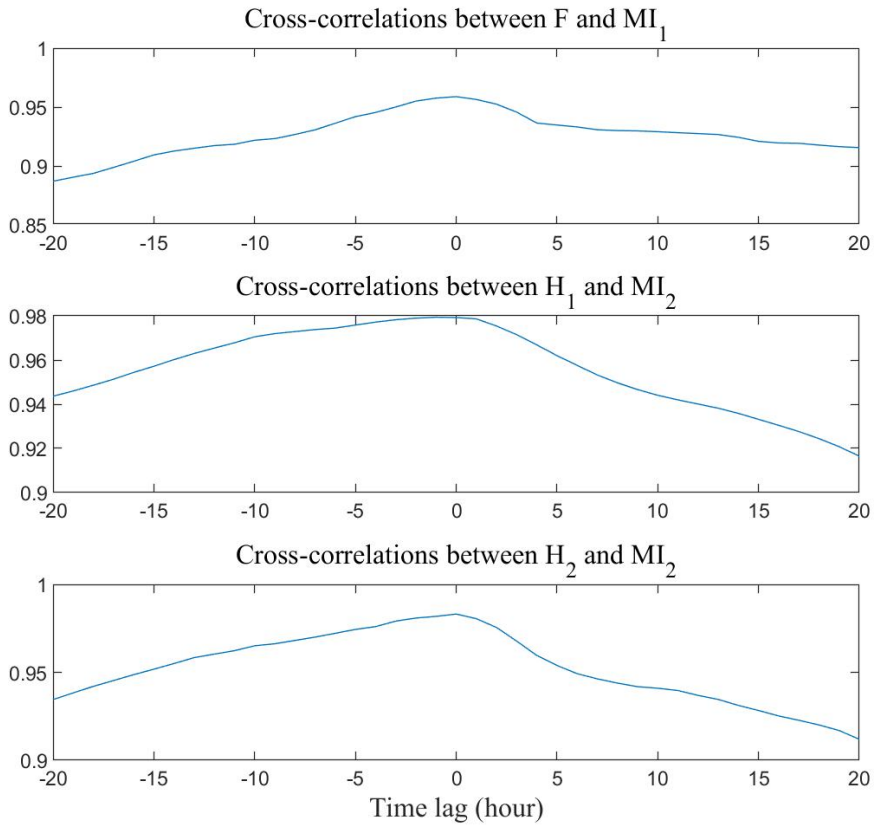

Figure 3.11 Cross-correlations between the time-shifted process variables and MI: (top) $F\,(t-9)$ and $\mathrm{MI}_1(t)$; (middle) $H_1(t-7)$ and $\mathrm{MI}_2(t)$; (bottom) $H_2(t-6)$ and $\mathrm{MI}_2(t)$.

From a few attempts of different model constructions, the models which have the best performance are chosen. The data-driven model for inferential estimation of MI can be represented as,

$$MI_1(t) = f_1[H_1(t), H_1(t-1), H_1(t-2), F(t-9), F(t-10), F(t-11)] \quad (3.26)$$

$$MI_2(t) = f_2[H_1(t-7), H_1(t-8), H_1(t-9), H_2(t-6), H_2(t-7), H_2(t-8)] \quad (3.27)$$

where MI_1 and MI_2 are the MI in D201 and D204 respectively, H_1 and H_2 are the concentrations of hydrogen in D201 and D202 respectively, and F is the hydrogen feed rate to D201 and the sampling interval of process variables are 0.5h

The original process data set contains 1534 samples of process operational data and 383 samples of quality data (MI) which are available for the establishment of data driven DBN models. It can be seen that the amount of process variable samples is larger than the amount of quality variable samples. This is due to that fact that measurements of process variables were logged every half hour while MI data were logged every 2 hours. There are only 383 samples of process variables that have corresponding quality variables. However, the rest of process variable samples can be utilised by DBN in the unsupervised training phase. By such means, DBN can capture much valuable information from process data and, as a result, the estimation of MI achieved by DBN can be improved.

The data set for the supervised training phase were separated into a training data set, a testing data set and an unseen validation data set. The partition of data sets for estimating MI_1 is presented by Table 3.2. The partition of data sets for estimating MI_2 is presented by Table 3.3.

Data sets	Percentage	Number of samples
Training data	50%	192
Testing data	22%	85
Unseen validation data	28%	106

Table 3.2 Partition of data sets for estimating MI_1

Data sets	Percentage	Number of samples
Training data	52%	200
Testing data	18%	68
Unseen validation data	30%	115

Table 3.3 Partition of data sets for estimating MI_2

The selections of model structure can be determined by training data set and testing data set through cross-validation. The unseen validation data are useful to test the performance of the final developed DBN models.

It can be seen from Tables 3.2 and 3.3, 277 samples of training and testing variables were selected to fine tune DBN by backpropagation for MI_1 and DBN only use 268 samples of training and testing variables to fine tune DBN in supervised training phase for MI_2. During the unsupervised training phase of DBN models, only input data are required and target values are not required. Those input data samples without the corresponding output data are named as 'unlabeled' process data. Therefore, in the unsupervised training phase of DBN models, samples of process variables without the corresponding MI data can also be utilised. However, those 'unlabeled' process variables could not be used by other traditional neural networks for inferential estimation of product quality. For comparison, conventional neural network models were also developed.

3.4 Results and Discussions

The model structures need to be determined first. In this study, 25 DBN models with different architectures was developed and compared to each other. The one giving the best performance on the testing data set was regarded as having the appropriate structure. These DBN models have one visible layer (input layer), one additional top layer (output layer) and two hidden layers. Based on the systematic hyper-parameter optimisation, the learning rate in unsupervised training phase is selected as 0.01. The learning rate in supervised training phase is 0.0015. The structures of 25 DBN models are shown in Table 3.4. Figures 3.12 and 3.13 present the sum of squared errors (SSE) on training data set and testing data set respectively for these 25 DBN models for estimating MI_1.

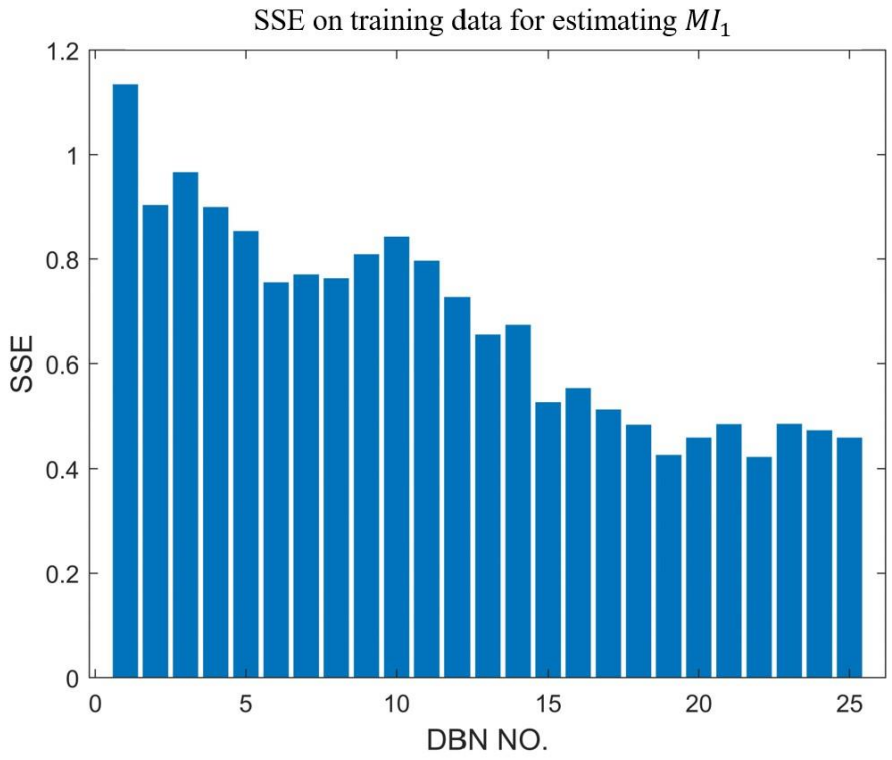

Figure 3.12 SSE on training data for estimating MI_1

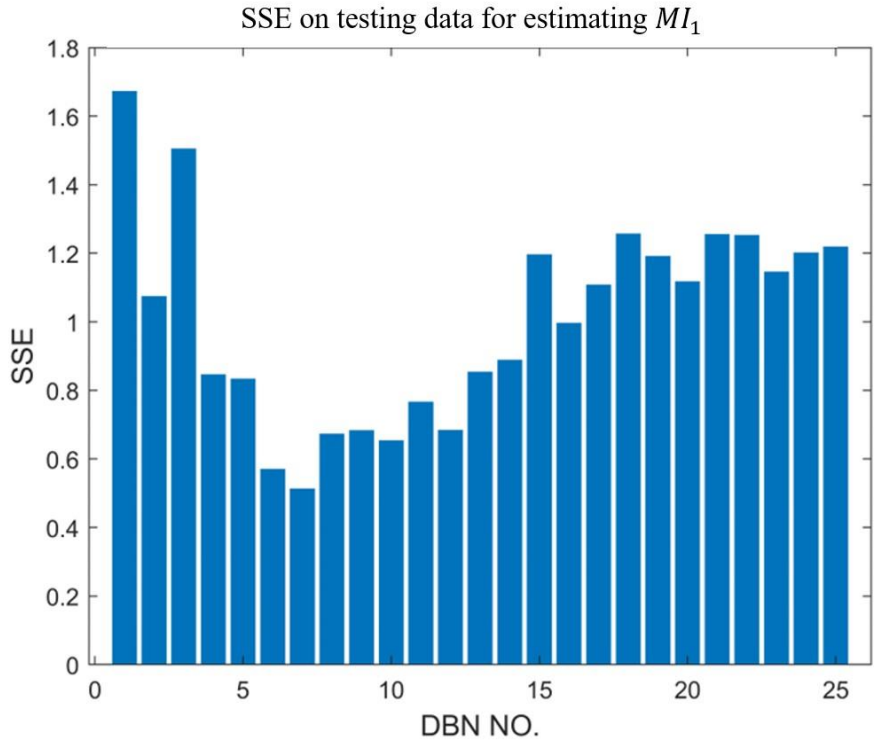

Figure 3.13 SSE on testing data for estimating MI_1

Figure 3.12 shows a general trend that the model errors on the training data reduce as the number of hidden neurons increase. However, this trend is not shown on the testing data. Figure 3.13 shows that the model errors decrease initially but later increase as more hidden neurons are used. Figure 3.13 shows that the 7th DBN model gives the best performance on the testing data set. The 6th DBN model gives the second lowest value of error on the testing data set. The 12th to the 25th DBN models have lower training errors than the 7th DBN model. However, those models give larger testing errors than the 7th DBN model. Thus, the 12th to the 25th DBN models are likely have suffered from over-fitting and their structures are not appropriate to be selected. From the results given by Figures 3.12 and 3.13, the number of neurons in the first hidden layer can be considered as 5. From Table 3.4, it can be seen that these 25 DBN models have close numbers of neurons in the first and second hidden layers. The first step of this investigation is to confirm that the 7th DBN gave the best performance among these 25 DBN models. To avoid the situation that some DBN models not included in Table 3.4might give better performance, the second step is to further investigate the number of neurons in the second hidden layer. Nine additional DBN models with neurons in the second hidden layer ranging from 2 to 10 are developed. The SSE values on the training data and testing data of these DBN models are shown in Table 3.5.

From Table 3.5, it can be seen that the training error of the 7th DBN is the smallest. However, its testing error is not the smallest. The testing errors from the 6th to the 9th DBN increased. Therefore, it can be considered that the 6th to the 9th DBNs are over-fitted. The 4th DBN (i.e. the 7th DBN model in Table 3.4) has the lowest testing error among all DBN models. This indicates that the 4th DBN model has better generalisation performance than other models and its structure should be adopted.

NO.	Neurons in 1st hidden layer	Neurons in 2nd hidden layer	NO.	Neurons in 1st hidden layer	Neurons in 2nd hidden layer
1	2	1	14	8	7
2	2	2	15	9	9
3	3	3	16	9	8
4	3	2	17	10	10
5	4	4	18	10	9
6	4	3	19	11	11
7	5	5	20	11	10
8	5	4	21	12	12
9	6	6	22	12	11
10	6	5	23	13	13
11	7	7	24	13	12
12	7	6	25	14	13
13	8	8			

Table 3.4 DBN models with different structures

NO.	Neurons in 1st hidden layer	Neurons in 2nd hidden layer	SSE (training)	SSE (testing)
1	5	2	0.7562	0.5819
2	5	3	0.8204	0.6193
3	5	4	0.7824	0.5945
4	5	5	0.7696	0.5118
5	5	6	0.8206	0.5773
6	5	7	0.7271	0.5742
7	5	8	0.6723	0.5859
8	5	9	0.7628	0.6071
9	5	10	0.7372	0.6322

Table 3.5 The errors of DBN models with different structures for estimating MI_1

NO.	Neurons in hidden layer	MI_1		MI_2	
		SSE (training)	SSE (testing)	SSE (training)	SSE (testing)
1	2	1.3256	0.7446	1.6025	0.6855
2	3	0.7949	0.8221	1.5185	0.7374
3	4	0.7924	0.6527	1.5035	0.6564
4	5	0.7675	0.6323	1.3650	0.6883
5	6	0.6347	0.6532	1.1009	0.8214
6	7	0.5124	1.1054	0.7844	0.8305
7	8	0.4201	0.8895	0.5108	1.6024

Table 3.6 The errors of neural networks with different structures

In order to demonstrate the advantage of using those input data samples without corresponding target values as additional training data in the unsupervised training phase, a DBN model trained only using the input data samples with the pre-existing labels in the unsupervised training phase was also developed. This is represented by DBN No. 1 in Table 3.7, where DBN No. 2 was built by using 'unlabeled' process data without corresponding MI samples as well. DBN No. 2 in Table 3.7 is in fact the 4th DBN model in Table 3.6. The two DBN models in Table 3.7 have the same structure. It can be seen that the first DBN model has larger SSE values on the training, testing and validation data set than the second DBN model. Therefore, DBN can extracted more features from the 'unlabeled' data. DBN NO. 2 gives better performance than DBN NO. 1.

DBN NO.	SSE (training)	SSE (testing)	SSE (validation)
1	1.6203	0.8905	0.7024
2	0.7696	0.5118	0.6851

Table 3.7 The errors of DBN models for estimating MI_1 with different input data

Seven conventional single hidden layer feedforward neural network models were also established for the purpose of comparison. The SSE values of these conventional feedforward neural networks with different structures on the training and testing data are given in Table 3.6. From Table 3.6, the 4th neural network has the lowest SSE on the testing data set for estimating

MI_1 and the 3rd neural network has the lowest SSE on the testing data for estimating MI_2.

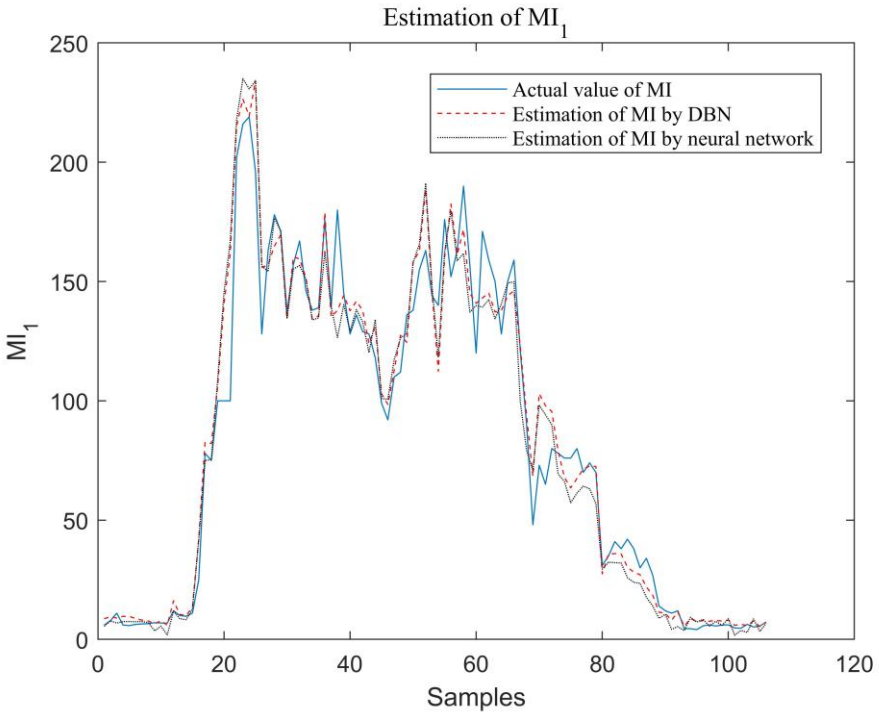

Figure 3.14 Estimation of MI_1 by DBN and neural network (validation data)

Models	SSE (training)	SSE (testing)	SSE (validation)
Neural network	0.7675	0.6323	0.8243
DBN	0.7696	0.5118	0.6851

Table 3.8 SSE of estimating MI_1

The estimations of MI_1 on the unseen validation data by DBN and conventional feedforward neural network are shown in Figure 3.14. In Figure 3.14, the solid, dashed, and dotted lines represent, respectively, the actual values of MI_1, the estimations by DBN, and the estimations by the conventional feedforward neural network. It can be seen that the estimations by the DBN model are generally closer to the corresponding actual values of MI_1 than those by the feedforward neural network. The SSE values of both DBN and neural network are presented in Table 3.8. It can be seen from Table 3.8 that the SSE of DBN on training data set is larger than that of the neural network. However, the SSE values of DBN on testing and unseen validation data set are much smaller than those of the neural network. The strong generalisation capability

of DBN was proved by the inferential estimation of MI_1. It gives better performance than the conventional feed-forward neural network. The profuse latent information from process data were extracted by DBN during the unsupervised training phase. Overall, the DBN model gives more accurate estimations of MI_1.

Figure 3.15 compares the estimations of MI_2 by DBN and conventional feedforward neural network on the unseen validation data. In Figure 3.15, the solid, dashed, and dotted lines represent, respectively, the actual values of MI_2, the estimations by DBN, and the estimations by the conventional feedforward neural network. From Figure 3.15, it can be seen that both models give similar performance when MI values are high. However, when MI values are low, the DBN model gives better estimations. Table 3.9 shows the SSE values in the estimation of MI_2. The SSE of DBN on training data is larger than that of neural network. The SSE values of DBN on testing and unseen validation data set are much smaller than those of the neural network model. The results in Figure 3.15 and Table 3.9 indicate that the estimations of MI_2 achieved by DBN are more reliable and accurate than those from the conventional feedforward neural network.

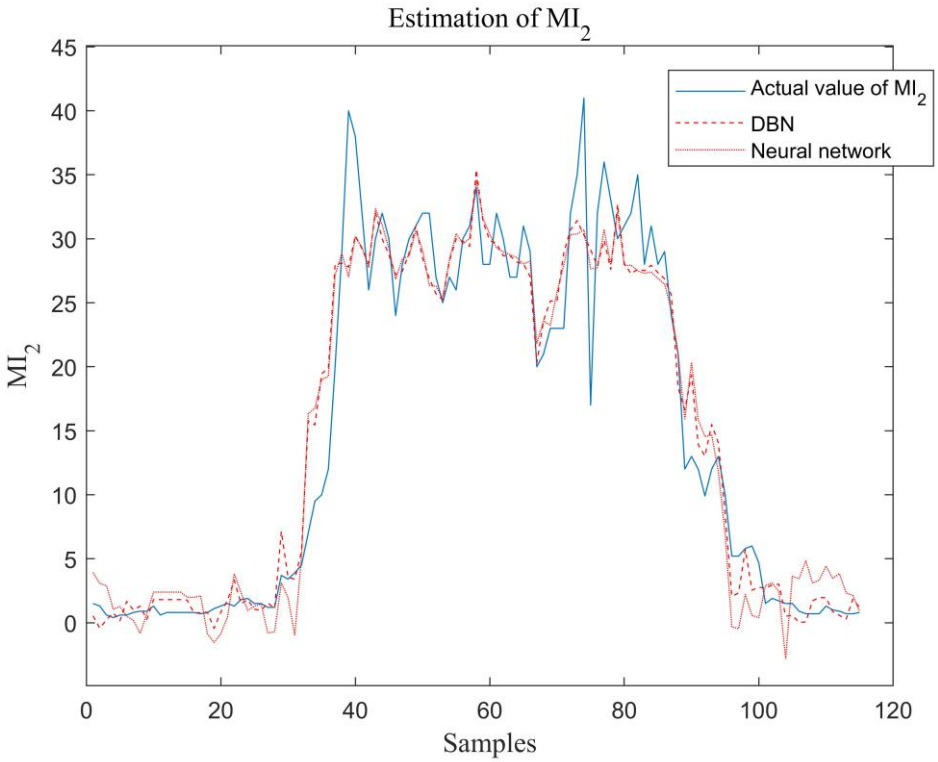

Figure 3.15 Estimation of MI_2 by DBN and neural network (validation data)

Models	SSE (training)	SSE (testing)	SSE (validation)
Neural network	1.5035	0.6564	0.9915
DBN	1.5170	0.4342	0.8560

Table 3.9 SSE of estimating MI_2

3.5 Conclusions

DBN models for on-line inferential estimation of polymer melt index in an industrial polymerization process are developed in this chapter. DBN can be developed with a deep structure. The profuse latent information from the process variables can be extracted by DBN. The 'unlabeled' process data, which cannot be utilised by conventional neural network models, can be used in the unsupervised training stage of DBN. It is shown in this chapter that the accuracy of inferential estimation of polymer MI can be improved by this means. Selection of DBN structure is investigated in the chapter. The appropriate structures of DBN for the estimation of MI_1 and MI_2 are selected. DBN has much better performance compared with the results from conventional feedforward neural networks. The study demonstrates that DBN is very suitable for developing nonlinear data-driven models for the inferential estimation of polymer melt index. The proposed DBN model could be extended for developing multi-step ahead prediction models in the future. The network structure of DBN can be further optimised to improve the robustness.

Chapter 4. Developing Robust Nonlinear Models through Bootstrap Aggregated Deep Belief Networks

4.1 Introduction

As stated in Chapter 3, DBN based on the technique of deep learning shows strong generalisation capability on the actual industrial polypropylene polymerization process. However, the DBN model needs to be developed with an appropriate structure to achieve the desired results. Developing a reliable model with an appropriate structure for the different processes is very time-consuming as various network structures need to be evaluated. Although DBN can extract more latent information from process data, the unsupervised training phase cannot guarantee the latent features from process data is appropriate for the supervised training phase. It causes the model to be non-robust and the predictions could become unreliable. And the most important issue of the DBN model is that the model may not be robust or reliable when applied to complex nonlinear processes. Therefore, the robustness and accuracy of the DBN model need to be improved.

When applying advanced modelling techniques to actual industrial processes, model robustness or reliability is one of the most important criteria that need to be considered. Model robustness can be used to judge the performance of models in real industrial applications. Among the various techniques for improving model robustness, Bates and Granger (1969) prove that the method of combining multiple forecasting models has been shown to be very effective. Wolpert (1992) proposed the stacked model through combination of several single neural networks. A linear combination of neural networks for chemical process was introduced by Sridhar et al. (1996) and the results show the accuracy of neural network has been improved through combining multiple neural networks. Bootstrap aggregated neural network was proposed by Zhang (1999b). Every neural network in that model is developed with different training data sets obtained bootstrap re-sampling of the original training data. With the comparison between bootstrap aggregated neural network and conventional neural network, it shows the performance of bootstrap aggregated neural network is much better. This type of aggregated models can achieve great performance in many applications with improved robustness such as fault diagnosis (Zhang, 2002), estimation of polymer quality variables (Zhang et al., 1997), and prediction of chemical and reaction yield (Monemian et al., 2010). It has been shown that bootstrap aggregated neural networks can also be utilised for many different chemical processes and achieve desired results (Khaouane et al, 2017; Zhang et al., 2008, 2011, 2013, 2018;

Osuolale and Zhang, 2018; Szafranek, 2019; Lian et al., 2020; Lu et al., 2021). Based on the idea of stacking several networks to enhance model performance, many approaches were proposed in recent years. Stacked extreme learning machines was given by Zhou et al. (2015) and deep analytic network was proposed by Low and Teoh (2017).

This chapter presents a bootstrap aggregated deep belief network (BAGDBN) to improve the robustness and accuracy of deep belief network (DBN) models. In the proposed approach, several replications of the original modelling data are generated by using bootstrap re-sampling with replacement. A DBN model is developed on each replication and these individual DBN models are combined to form a BAGDBN model. The effectiveness of the proposed approach is demonstrated on two application examples, modelling a conical water tank and inferential estimation of polymer melt index in an industrial polypropylene polymerization process. It is shown that BAGDBN models can give more accurate and reliable predictions than single DBN models.

The chapter is organized as follows, Section 4.2 introduces BAGDBN model and the main algorithm. Modelling a conical water tank for multi-step prediction of water level which is presented in Section 4.3. Section 4.4 gives inferential estimation of MI in an industrial polymerization process. The chapter is summarised in Section 4.5.

4.2 Bootstrap Aggregated Deep Belief Networks

The main idea of BAGDBN is to develop multiple DBN models and then combine them to improve model prediction reliability and accuracy. In order to increase the diversity of these individual DBN models, each DBN model is developed from a replication of the original modelling data set generated through bootstrap resampling. For resampling a replication of process and quality data, the new data set need to be resampled from the original data set by using bootstrap resampling technique. Figure 4.1 demonstrates bootstrap resampling with replacement. It can be seen that sample 4 of the original data was resampled twice. However, sample 7 was not resampled in the particular case. It needs to be mentioned that the dynamic data need to be arranged in such a way as shown in Figure 4.2, where the quality variable is only related to the time lagged process variables in the same row. Each row in the matrix includes process variables and the related quality variables and is considered as a sample of the original data set. After this data arrangement, bootstrap re-sampling with replacement can be carried out

for the time-series data as shown in Figure 4.3. A DBN model is developed based on each of these replications.

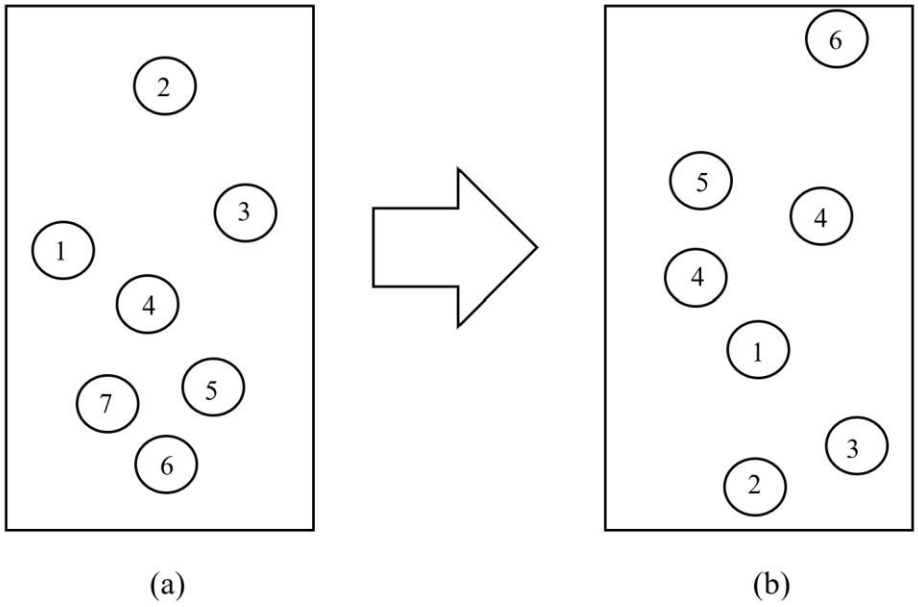

Figure 4.1 Bootstrap resampling: (a) Data samples in the original data set; (b) data samples in the resampled data set.

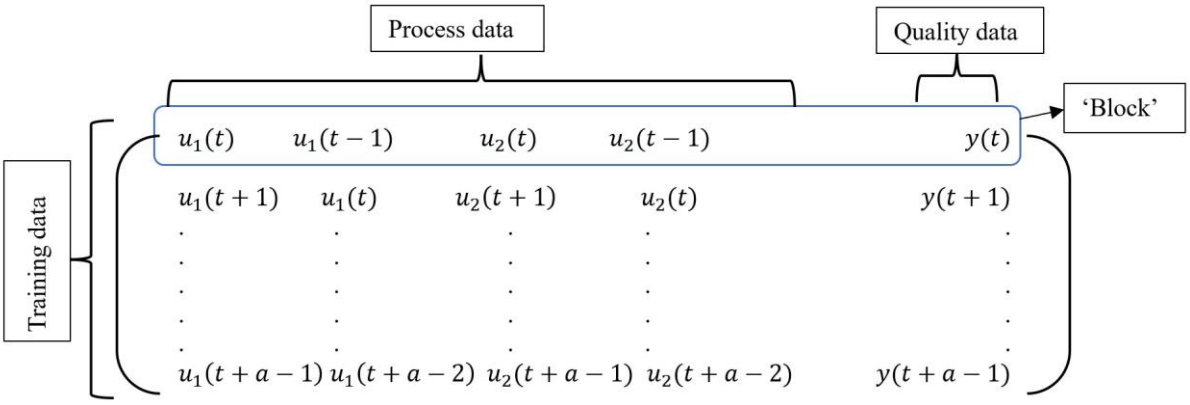

Figure 4.2 Arrangement of time lagged process and quality data

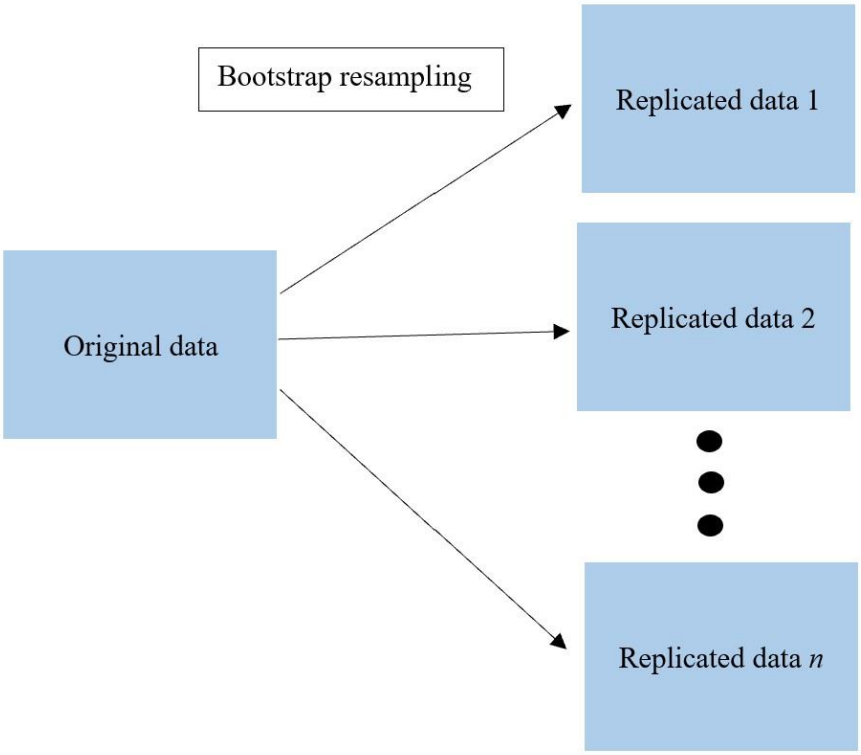

Figure 4.3 Bootstrap resampling replications of the original data

These developed DBN models are then combined. A figure of BAGDBN architecture is shown in Figure 4.4. These individual DBN models in a BAGDBN are trained to find the relationship between process data and quality data of processes. Predictions from these individual DBN models are then combined to obtain the final prediction of the BAGDBN model. The output of a BAGDBN can be formulated as,

$$f(X) = \sum_{i=1}^{n} w_i f_i(X) \quad (4\text{-}1)$$

where $f(X)$ is the output of BAGDBN, $f_i(X)$ is the *i*th DBN output, w_i is the aggregating weight for the *i*th BAGDBN, *n* is the number of DBN models in the BAGDBN model, and *X* is the vector of inputs. Aggregating weights w_i can have big effects on overall prediction and need to be determined properly for good modelling performance. In this chapter, the aggregating weights, w_i, are set as the same value of 1/*n* for simplicity. It means the output of BAGDBN is an average of DBN outputs. It is shown in this study that this approach gives quite good performance. Principal component regression can also be used in determining the aggregating weights (Zhang, 1999b).

The BAGDBN model shown in Figure 4.4 contains *n* (e.g. *n* = 30) individual DBN models. This would suggest that the training effort of a BAGDBN model is *n* times more than training a single DBN model. However, this is not the case. In developing a single DBN model, a number of

DBN models with different structures (e.g. different numbers of layers and hidden neurons) need to be developed and the one giving the best performance on the testing data is considered to have the appropriate structure. This is also known as the cross-validation based procedure for network structure determination. In developing a BAGDBN model, this cross-validation based procedure for network structure determination could just be done for the first DBN if training time is a concern. For the building of subsequent DBN models, the appropriate structure(s) identified in building the first DBN model could be used. Thus, the computation time for building a BAGDBN model containing *n* individual DBN models is much less than *n* times the computation time for building a single DBN model. MATLAB was used as the software of the development of BAGDBN models in this chapter. Table 4.1 shows the specification of the computer.

Component	**Specification**
Processor	Intel core i7-8650u cpu @ 1.90GHz
Ram	8GB DDR3
Storage	256 GB SSD
Operating system	Windows 10 pro

Table 4.1 Specification of the computer

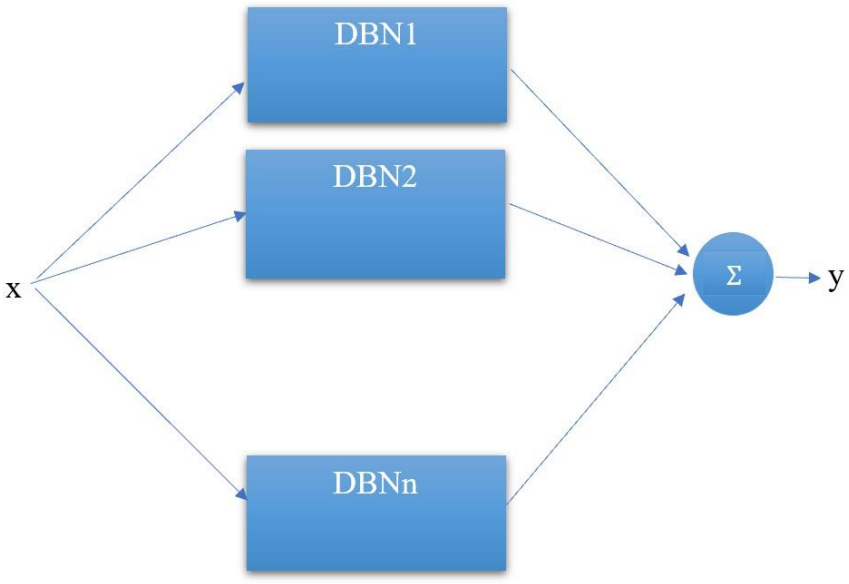

Figure 4.4 The structure of BAGDBN

4.3 Modelling of A Conical Water Tank

4.3.1 *A Conical Water Tank*

This conical water tank, shown in Figure 4.5, contains an inlet stream and an outlet stream. Inlet water flow rate regulates the water level of the tank. The rate of change in the volume of water, *V*, in the tank can be formulated using mass balance as:

$$\frac{dV}{dt} = Q_{in} - Q_{out} \tag{4.2}$$

where Q_{in} is the inlet water flow rate and Q_{out} is the outlet water flow rate. The outlet flow rate, Q_{out}, is determined by the tank level as shown in Equation (4.3):

$$Q_{out} = k\sqrt{h} \tag{4.3}$$

where *h* represents the tank level and *k* is a parameter for a fixed value opening. The volume of water in the tank is given in (4.4):

$$V = \pi h(r^2 + \frac{hr}{tan\theta} + \frac{h^2}{3(tan\theta)^2}) \tag{4.4}$$

where *r* is the radius of the bottom, θ is the angle between the tank boundary and horizontal plane. Therefore, the first principle dynamic model can be formulated as:

$$\frac{dh}{dt} = \frac{Q_i - k\sqrt{h}}{\pi(r^2 + \frac{hr}{tan\theta} + \frac{h^2}{3(tan\theta)^2})} \tag{4.5}$$

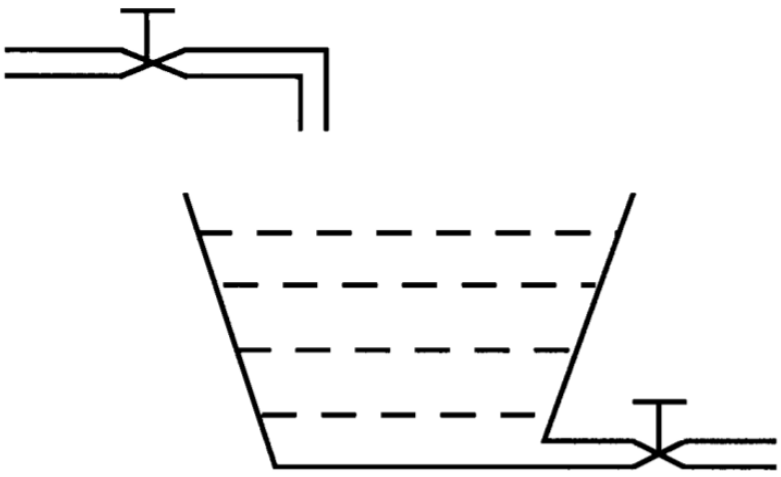

Figure 4.5 The water tank

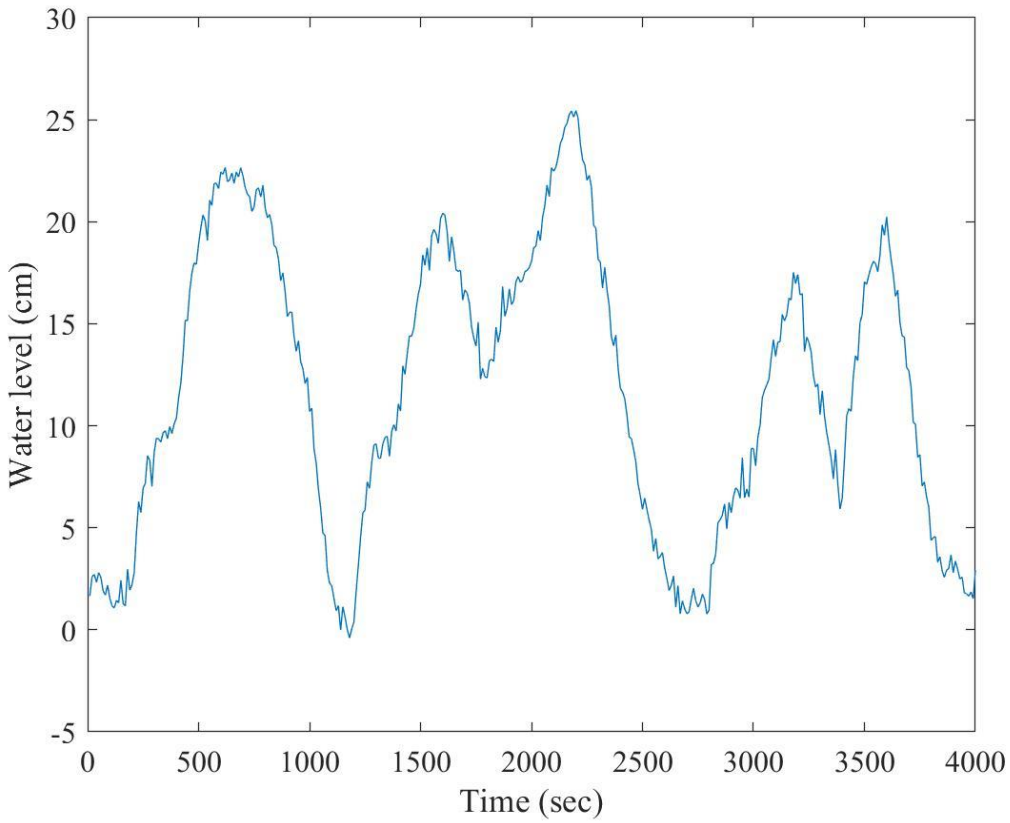

Figure 4.6 Water level

The first principle dynamic model is used to simulate the process operation using MATLAB. Figure 4.6 presents the trend of water level in the water tank. From Figure 4.6 and equation (4.5), the process is clearly nonlinear (Zhang & Morris, 2000). The parameters of the process model are set as fixed values, $k = 34.77cm^{2.5}/s$, $r = 10cm$, and $\theta = 60^{\circ}$. The sampling time used in dynamic simulation is 10 seconds. Simulated measurement noises with the distribution N (0, 0.5) are added to the tank level. It is then assumed that the first principle model is unavailable and a data-driven model has to be developed. After experimenting with first-order, second-order, and third-order dynamic models, it is found that the second-order dynamic model gives good performance. The developed multi-step ahead prediction model of water level can be represented as,

$$\hat{y}(t) = [\hat{y}(t-1), \hat{y}(t-2), u(t-1), u(t-2)] \tag{4.6}$$

where $\hat{y}$ is the prediction of water level of the tank, u is the inlet water flow rate, and t represents the discrete time.

All the process operating data were auto scaled and divided into 3 parts. The first part of data is training data set which is used to train BAGDBN model. The second part is the testing data set

which is used to determine model structures and guard against overfitting. The last part is the unseen validation data set which is used to test the final BAGDBN model to confirm the model generalisation capability. Table 4.2 shows the partition of process operating data.

Data set	Percentage	Number of samples
Training data set	47%	188
Testing data set	19%	77
Unseen validation data set	34%	133

Table 4.2 Partition of data sets for modelling water level

4.3.2 *Multi-step Ahead Prediction of Water Level*

In this case study, 30 different training and testing data sets were resampled from the original data sets by using bootstrap resampling techniques. A DBN model was developed on each replication of the training and testing data. These 30 DBN models are combined into a BAGDBN model. During the procedure of unsupervised training, the input variables of training and testing data were used to pre-train BAGDBN. After unsupervised training, BAGDBN was trained by the resampled training and testing data sets in the supervised training phase. Figures 4.7 and 4.8 show the mean squared errors (MSE) of individual DBNs on the training and testing data set and on the unseen validation data set respectively. Note that the MSE values are for scaled data.

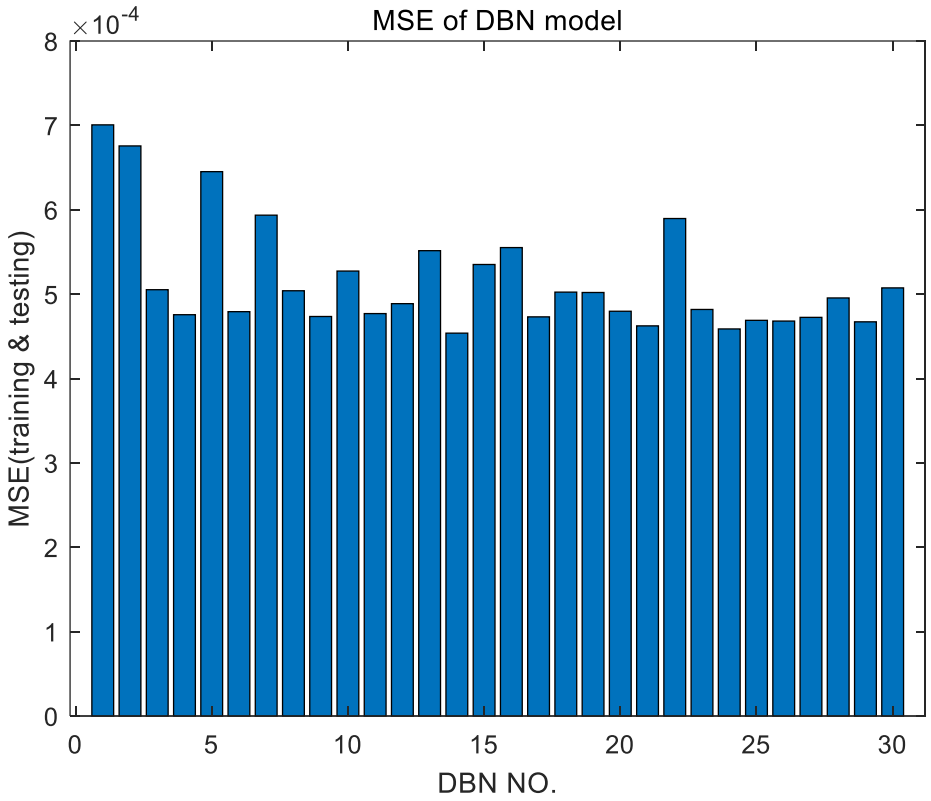

Figure. 4.7 MSE on the training and testing data of individual DBN models

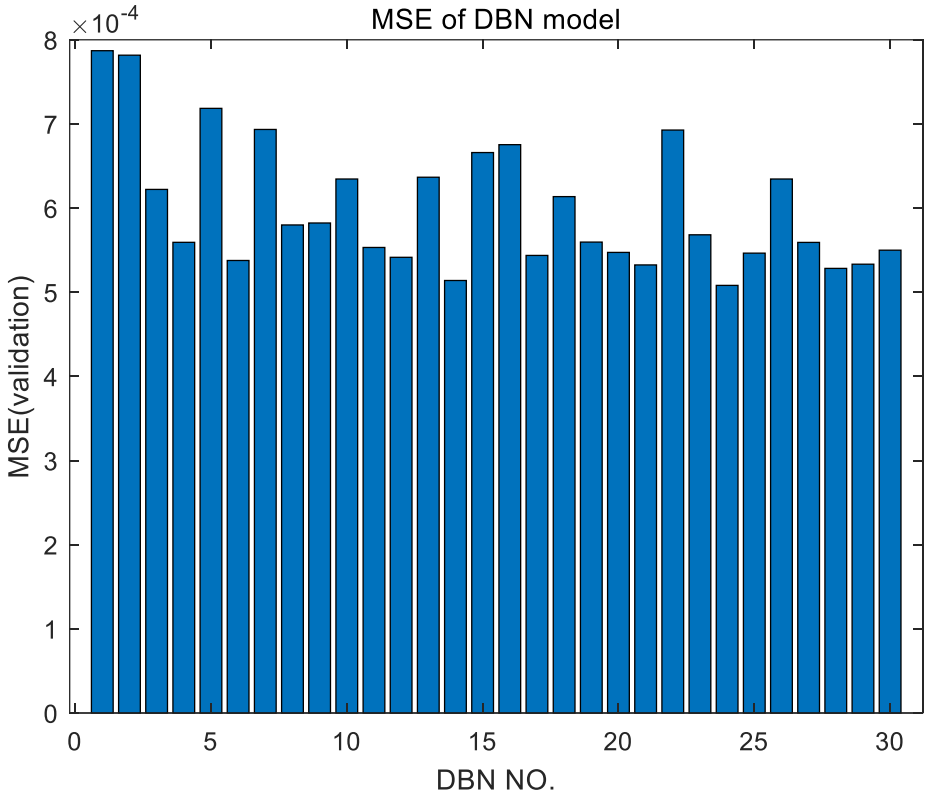

Figure 4.8 MSE on the validation data of individual DBN models

Figures 4.7 and 4.8 indicate that the individual DBN models give various performances and their performances on the training and testing data and on the unseen validation data are not consistent. The 10th and 13th DBN models give similar performances on the training and testing data sets. However, the MSE of the 10th DBN model on the unseen validation data is smaller than that of the 13th DBN model. The 28th DBN gives smaller MSE on the training and testing data than the 26th DBN, but it gives larger MSE on the validation data set than the 26th DBN. These results indicate the non-robust or unreliable nature of single DBN models.

To improve the robustness of DBN model, BAGDBN models are developed by combining the individual DBN models. Figures 4.9 and 4.10 give the MSE of BAGDBN models on the training and testing data and on the unseen validation data respectively. Note that the MSE values are for scaled data.

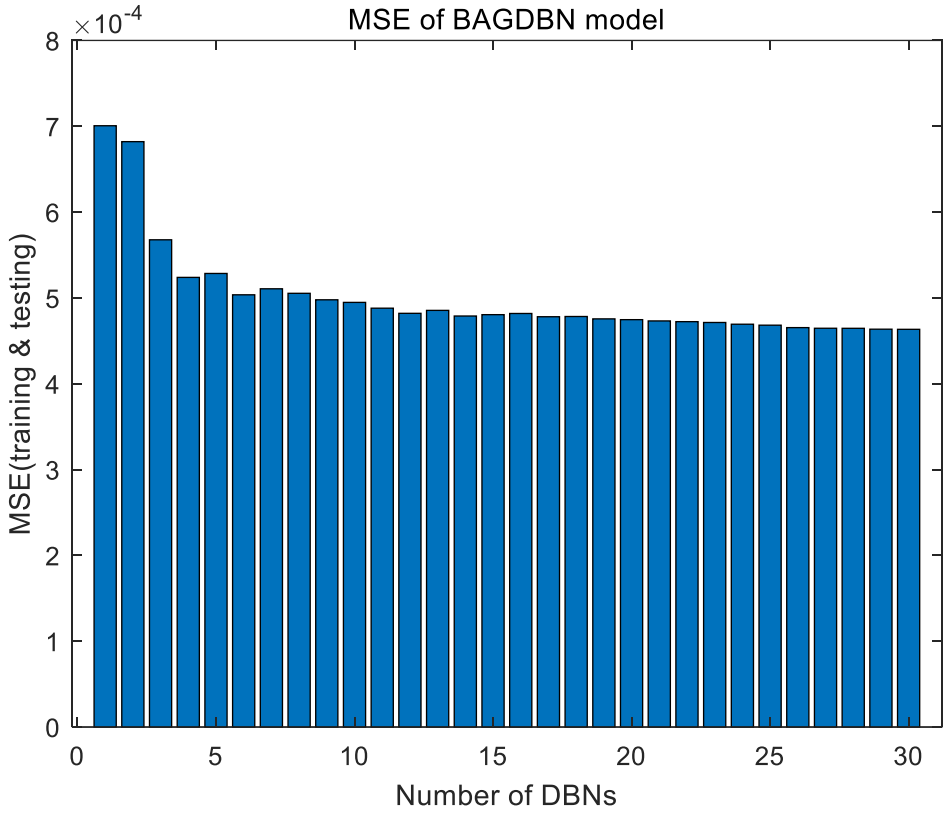

Figure 4.9 MSE on the training and testing data of BAGDBN models

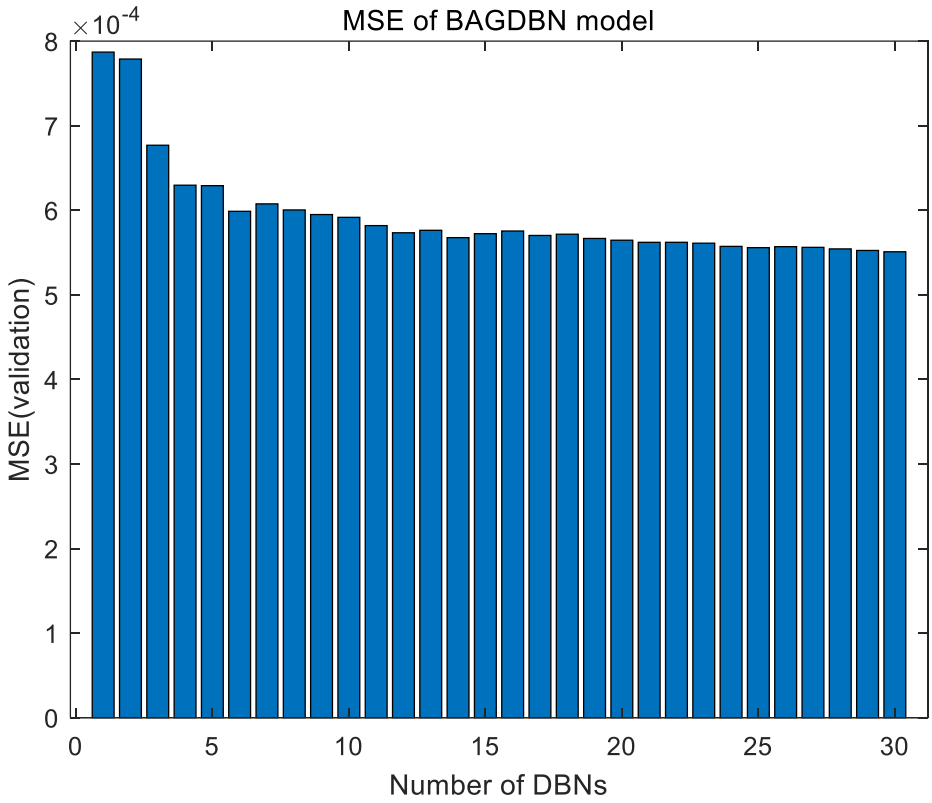

Figure 4.10 MSE on the validation data of BAGDBN models

In Figures 4.9 and 4.10, the x-axis represents the number of DBN models included in a BAGDBN model. The first bar represents the first DBN model, the second bar represents aggregating the first two DBN models, and the last bar represents aggregating all the 30 DBN models. It can be seen from Figures 4.9 and 4.10 that the MSE values decrease as more DBN models are combined. BAGDBN models give very consistent performances on the training and testing data sets and on the unseen validation data set. The results in Figures 4.9 and 4.10 demonstrate that BAGDBN models are more robust or more reliable than single DBN models. It shows that, as long as sufficient number of DBN models are included (about 10), the performance of BAGDBN models is insensitive to the numbers of individual DBN models.

Figure 4.11 shows the multi-step ahead predictions of water level and Table 4.3 compares the errors between BAGDBN model and DBN model. In the BAGDBN model, 30 DBN models are combined. It can be seen that the multi-step ahead predictions of water level achieved by BAGDBN are closer to the actual values than DBN model. From Table 4.3, it can be seen that the MSE of BAGDBN is smaller than that of DBN. It means that BAGDBN has ability to give more reliable and accuracy prediction than DBN.

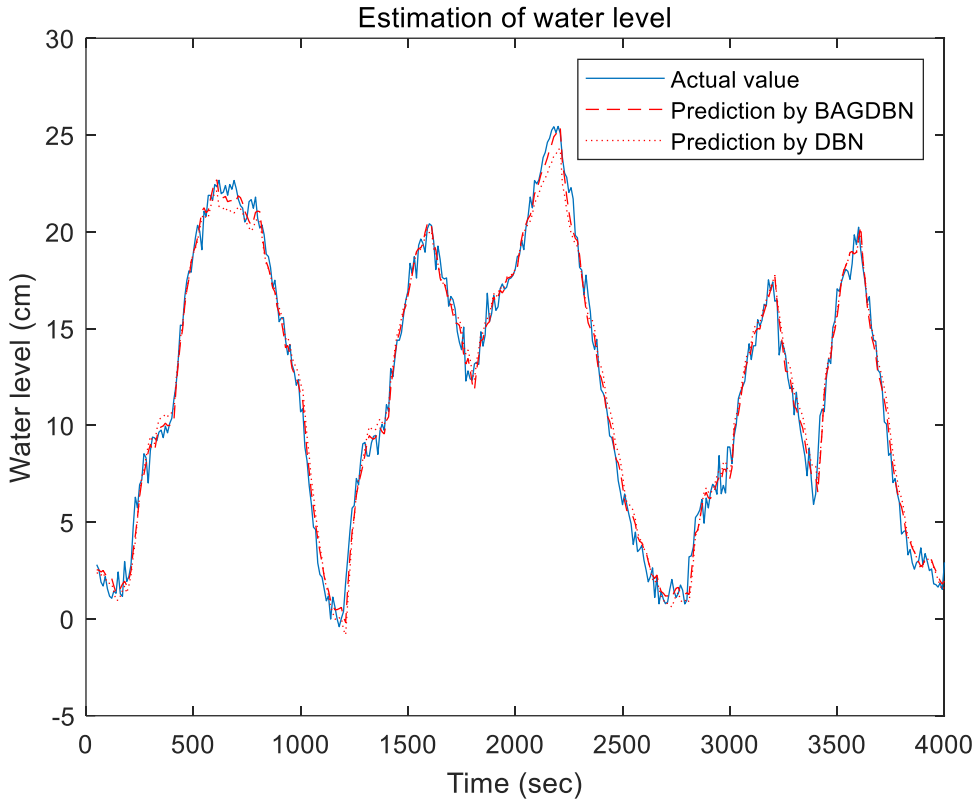

Figure 4.11 Multi-step ahead predictions of water level

Model	MSE
DBN	0.0008
BAGDBN	0.0006

Table 4.3 MSE of DBN and BAGDBN

4.4 Inferential Estimation of MI in An Industrial Polymerization Process

4.4.1 *An Industrial Polypropylene Polymerization Process*

This industrial polypropylene polymerization process is the same as in Chapter 3. It contains two CSTR and two FBR. In this industrial process, MI indicates the polymer quality and need to be monitored. To reduce the costs of production and improve the efficiency of polypropylene polymerization process, advanced monitoring technique needs to be applied to industry. Inferential estimation of MI is thus required for the advanced monitoring and control of this process.

Because process data were logged every half hour while the MI data were logged every two hours, the amount of process data is larger than quality data. It means there are a lot of process data without corresponding quality data. 1151 process samples are without corresponding quality data. In this study, only 383 pairs of input and output samples can be used in the procedure of supervised training. The advantage of BAGDBN is a combination of DBN models. It can use the 'unlabeled' process data in unsupervised pre-training. The latent information behind the process data can be extracted by BAGDBN.

The 383 operating data were separated into 3 data sets, training data set, testing data set and unseen validation set. Table 4.4 shows the partition of process operating data.

Data set	Percentage	Number of samples
Training data set	57%	217
Testing data set	15%	59
Unseen validation data set	28%	107

Table 4.4 Partition of data sets to estimate polymer melt index

The single DBN model for inferential estimation of MI can be represented as,

$$MI(t) = f[H(t), H(t-1), H(t-2), F(t-9), F(t-10), F(t-11)] \quad (4.7)$$

where MI_1 is the MI of polymer in D201, H is the concentration of hydrogen in D201, and F represents the hydrogen feed rate to D201. As a part of BAGDBN, every DBN model is developed from each replication of the training and testing data.

The number of hidden neurons for the individual DBN model is determined by the method of cross-validation as the same method given in Chapter 3. The DBN model which has lowest sum squared error on the validation data can be considered as the model with an appropriate structure.

4.4.2 *Inferential Estimation of Polymer Melt Index*

As with the previous case study, 30 replications of the original training and testing data sets were resampled from the original training and testing data set through bootstrap resampling with replacement. These data sets were used to train BAGDBN models. During the procedure of unsupervised training, the input variables without corresponding target samples were used to pre-train BAGDBN. After that, BAGDBN was fine-tuned using resampled training and testing

data sets through supervised training. Figures 4.12 and 4.13 shows the MSE values on the training and testing data set and on the unseen validation data set respectively.

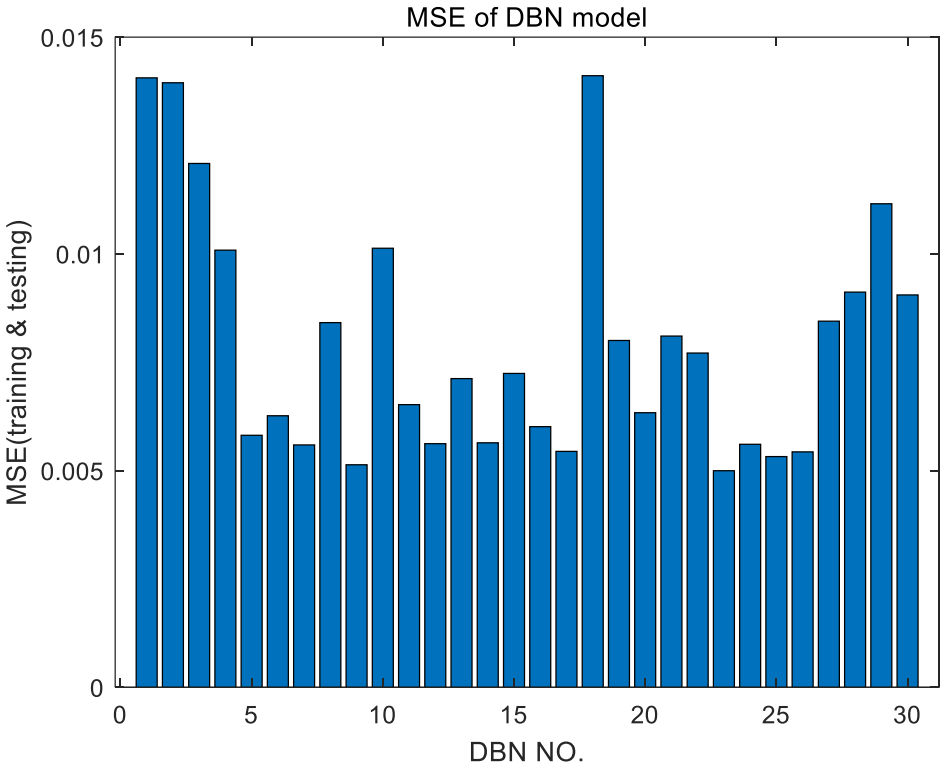

Figure 4.12 MSE of training and testing by DBN models

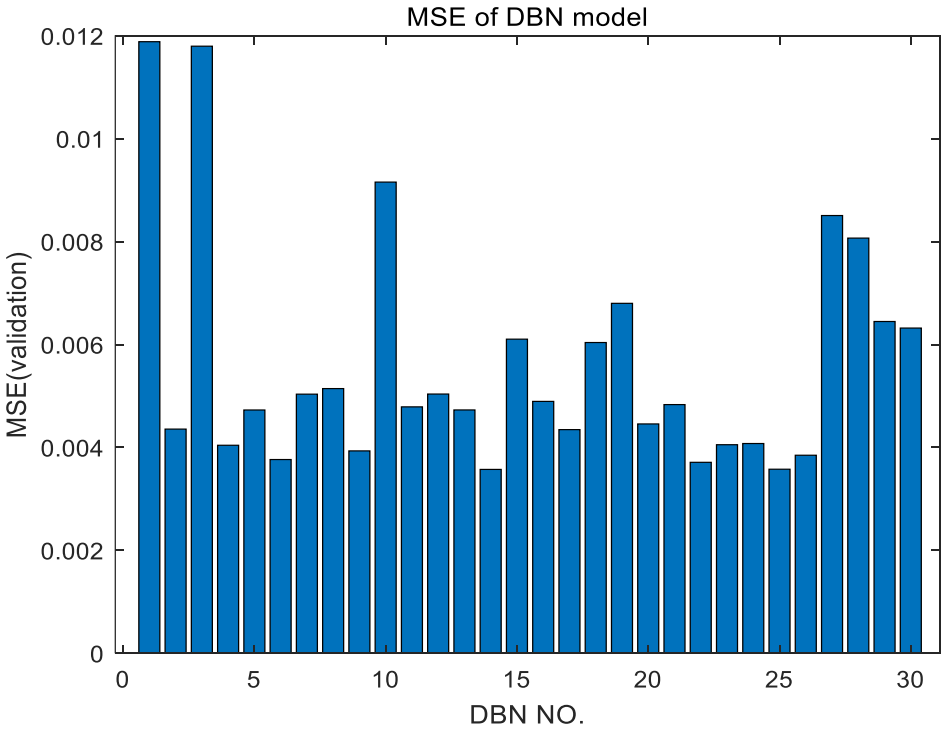

Figure 4.13 MSE of validation by single DBN models

It shows the MSE values of individual DBN models on the training and testing data and unseen validation data. Note that the MSE values are for scaled data. It can be seen from Figures 4.12 that the 1st, 2nd and 18th DBN models give similar performance on the training and testing data set. However, the MSE values of the 2nd and 18th DBN models on the unseen validation data are much smaller than that of the 1st DBN model. The 28th DBN gives smaller MSE on the training and testing data than the 29th DBN, but it gives larger MSE on the unseen validation data set than the 29th DBN. These indicate that single DBN models lack robustness or reliability.

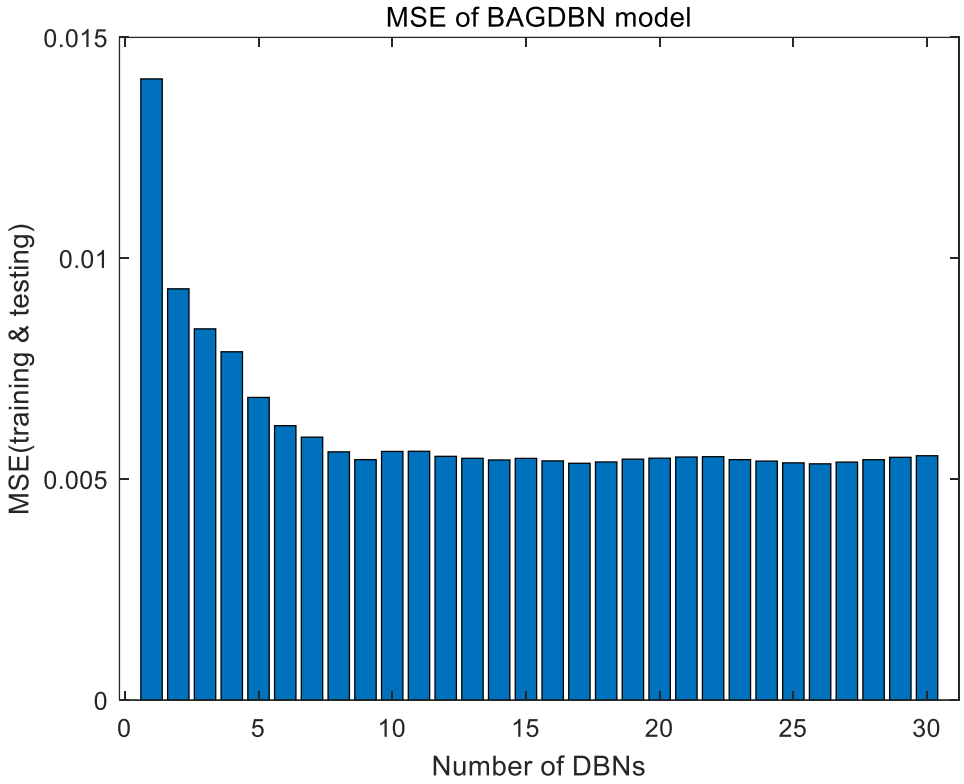

Figure 4.14 MSE of training and testing data by BAGDBN model

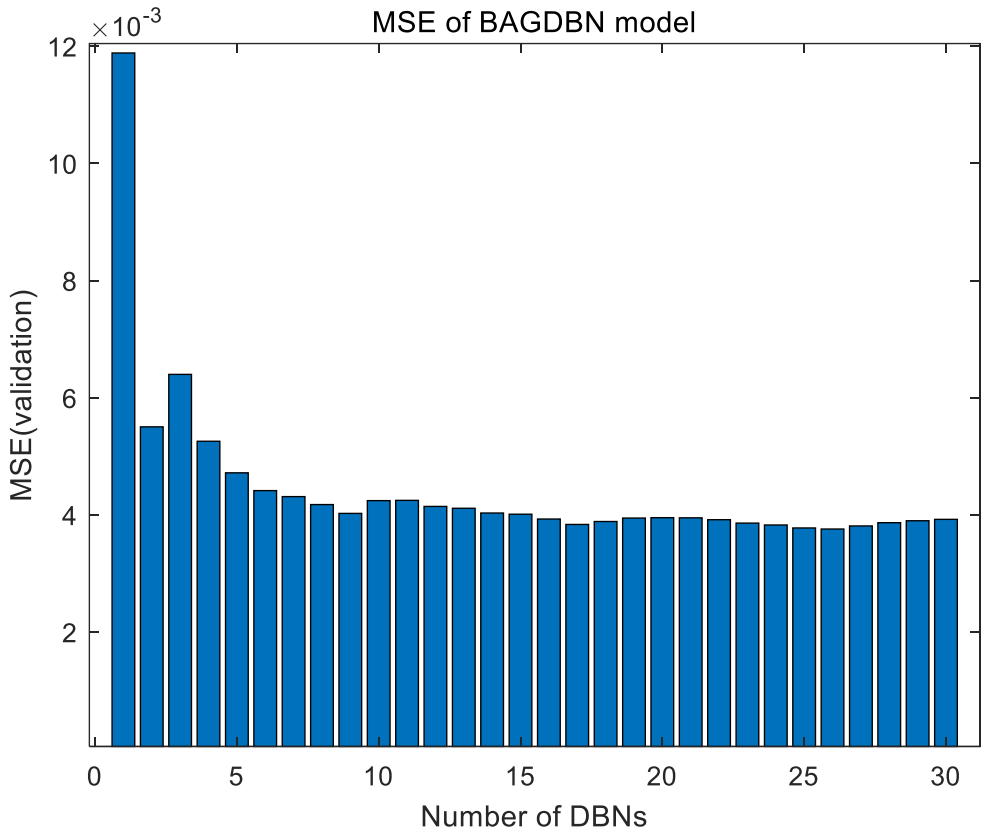

Figure 4.15 MSE of validation data by BAGDBN model

Figures 4.14 and 4.15 show, respectively, the MSE values on the training and testing data and on the unseen validation data from BAGDBN models with different number of DBN models combined. It can be seen from Figures 4.14 and 4.15 that the trend of decreasing MSE with the number of DBN models increasing is observed on both training and testing data and the unseen validation data. The results in Figures 4.12 to 4.15 indicate that BAGDBN models are more reliable than DBN models.

Figures 4.14 and 4.15 show the MSE values of BAGDBN models with different numbers of DBN models combined. In Figures 4.14 and 4.15, the first bar represents the first DBN model, the second bar represents aggregating the first two DBN models, and the last bar represents aggregating all the 30 DBN models. Again, the MSE values are for scaled data. It can be seen from Figures 4.14and 4.15 that the MSE values on the training and testing data and on the unseen validation data have similar trends. These MSE values decrease with the number of DBN models and then stabilize. Figures 4.14 and 4.15 also shows that, as long as a sufficient number of DBN models are included (about 10), the performance of BAGDBN models is insensitive to the numbers of individual DBN models. The results in Figures 4.14 and 4.15 indicate that BAGDBN models are more reliable and robust than single DBN models.

Figure 4.16 shows the estimation of MI (on the original scale) achieved by DBN and BAGDBN. Table 4.5 gives the MSE (on scaled data) from a conventional feedforward neural network, BAGDBN and DBN on the unseen validation data. It can be seen from Figure 4.16 that the BAGDBN model gives more accurate estimations than DBN. Table 4.5 shows that DBN gives smaller MSE values than conventional neural networks on the unseen validation data. The MSE values from BAGDBN are smaller than those from the conventional neural network and DBN. Hence, the advantage of BAGDBN over DBN is clear.

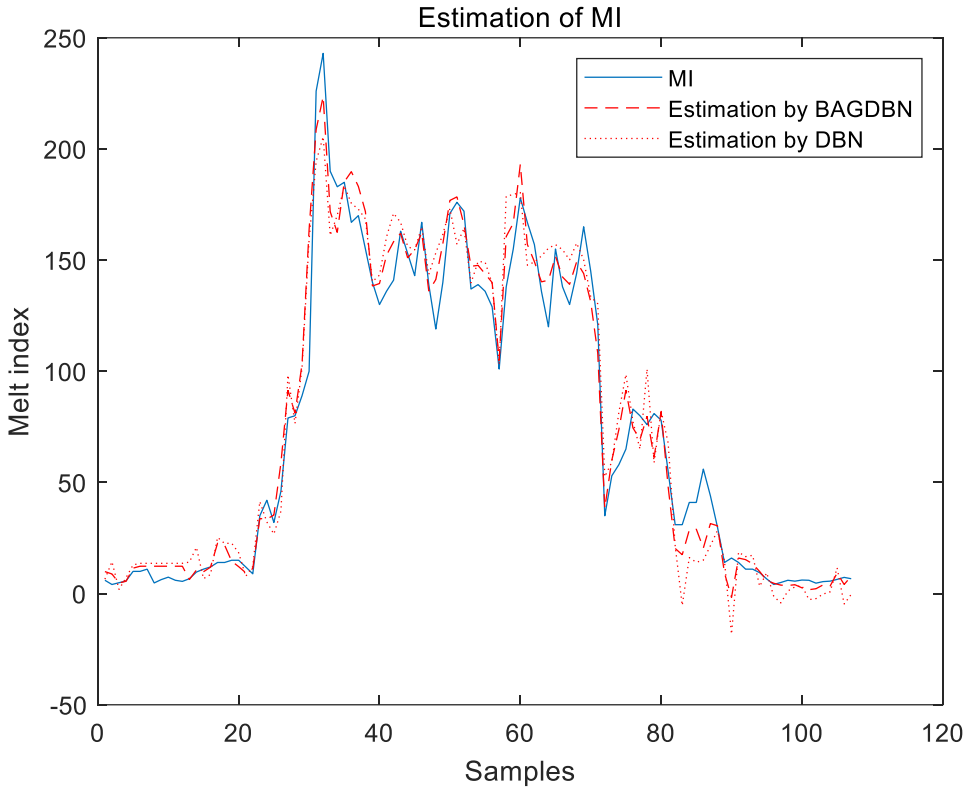

Figure 4.16 Estimation of polymer melt index

Model	MSE (training & testing)	MSE (validation)
DBN	0.0063	0.0068
BAGDBN	0.0053	0.0038

Table 4.5 MSE of DBN and BAGDBN

4.5 Conclusions

A novel data-driven modelling approach through integrating multiple DBN is proposed in this chapter. BAGDBN improves the accuracy and robustness of data-driven nonlinear models. In this study, multiple DBNs are established based on different bootstrap resampling replications from the original process modelling data set and are combined as one BAGDBN model. By aggregating multiple DBN models, the failure of a particular DBN model can be compensated by other DBN models. The effectiveness of BAGDBN is demonstrated on two application examples, dynamic modelling of a conical water tank and inferential estimation of MI in an industrial polypropylene polymerization process. A BAGDBN model gives better multi-step ahead prediction performance than a single DBN model. It is also more robust than a single DBN model in that it can give consistent good performance on different sets of data. In the estimation of polymer MI, the BAGDBN model gives more accurate and reliable estimations than DBN models. In the polypropylene polymerization process, there are a large number of process data samples without the corresponding quality data samples and they cannot be used by conventional supervised training models. However, these unlabeled data samples can be used in the unsupervised training phase of DBN and BAGDBN, which can extract more latent information to improve the estimation of polymer MI. One limitation of BAGDBN is the long training time required as more DBN models need to be trained. This could be improved in the future by developing new BAGDBN algorithms through sequential training and selective combination of individual DBN models.

Chapter 5. Data Driven Modelling and Optimisation of A Batch Reactor Using Bootstrap Aggregated Deep Belief Networks

5.1 Introduction

Batch processes are suitable for the agile manufacturing of high value-added products such as pharmaceuticals and specialty chemicals as the same reactors can be used to produce different products or different grades of products (Bonvin, 1998). Batch chemical reaction processes are typically highly nonlinear due to the fact that batch process operation covers a wide range of operation conditions from the start to the end of a batch and batch to batch variations commonly exist in practice. Optimisation of batch process operation is essential for the enhanced production efficiency and product quality. Batch process optimisation usually requires an accurate process model that can accurately predict the end of batch product quality variables. Developing accurate mechanistic models for batch processes is typically very time consuming and effort demanding. This is because a chemical reaction network usually involves a large number of reactions and some reaction pathways and/or kinetic parameters are not readily available. To overcome this difficulty, data-driven models developed from process operation and plant testing data should be capitalised. As batch chemical reaction processes are typically very nonlinear, nonlinear data-driven modelling techniques should be utilised. Machine learning techniques including neural networks and more recently deep learning, e.g., deep belief network (DBN), are effective techniques for data-driven modelling of batch processes (Zhang, 2004; Zhu & Zhang, 2020).

This chapter presents a reliable data-driven modelling and optimisation strategy for a batch chemical reactor using BAGDBN model. BAGDBN has enhanced modelling accuracy and reliability due to the combination of multiple models. Through incorporating model prediction confidence bounds from BAGDBN into the optimisation objective function, optimisation reliability can be enhanced.

This chapter is organized as follows. Section 5.2 presents the BAGDBN model. The case study is given in Section 5.3. Section 5.4 gives the results of simulation and data partition. Reliable optimal control through the incorporation of model prediction confidence bounds is introduced in Section 5.5. The results and discussions are given in Section 5.6. Conclusions of this study are drawn in Section 5.7.

5.2 Bootstrap Aggregated Deep Belief Network

Because of limitations in the amount of available data, creating an accurate and reliable DBN model is considerably challenging. The primary concept behind BAGDBN is to develop various DBN models and combine them for enhanced accuracy and robustness for the modelling of highly nonlinear batch processes. To increase the diversity of these individual DBN models, each DBN model is developed from a replication of the original batch process data set generated through bootstrap resampling with replacement (Efron & Tibshirani, 1993). The effectiveness of BAGDBN has been illustrated in Chapter 3. These individual DBN models in a BAGDBN are trained to find the relationship between process input and output data. Predictions from these individual DBN models are then combined to obtain the final prediction of the BAGDBN model. The output of a BAGDBN can be formulated as,

$$f(X) = \sum_{i=1}^{n} w_i \, f_i(X) \tag{5.1}$$

where $f(X)$ is the output of BAGDBN, $f_i(X)$ is the output of the *i*th DBN, w_i is the aggregating weight of the *i*th BAGDBN, *n* is the number of DBN models in the BAGDBN model, and *X* is a vector of model inputs. Aggregating weights w_i can have big effects on overall prediction and need to be determined properly for good modelling performance. In this Chapter, the aggregating weights, w_i, are set as the same value of 1/*n* for simplicity. It means the output of BAGDBN is an average of individual DBN outputs. It is shown in this study that this approach gives quite good performance.

5.3 Case Study

The batch chemical reactor presented by Arpornwichanop et al. (2005) is taken as a case study. In the reactor, two parallel highly exothermic reactions occur:

$$A + B \xrightarrow{k_1} C \tag{5.2}$$

$$A + C \xrightarrow{k_2} D \tag{5.3}$$

where A and B are raw materials, C is the desirable product and D is the by-product, k_1 and k_2 are the reaction rates with temperature dependence based on the Arrhenius relation. To simulate the reactor model, a mechanistic model based on the following equations are used.

Material balance in the reactor,

$$\frac{dM_A}{dt} = -k_1 M_A M_B - k_2 M_A M_C \tag{5.4}$$

$$\frac{dM_B}{dt} = -k_1 M_A M_B \tag{5.5}$$

$$\frac{dM_C}{dt} = k_1 M_A M_B - k_2 M_A M_C \tag{5.6}$$

$$\frac{dM_D}{dt} = k_2 M_A M_C \tag{5.7}$$

Energy balances around the reactor,

$$\frac{dT_r}{dt} = \frac{Q_r + Q_j}{M_r Cp_r} \tag{5.8}$$

$$\frac{dT_j}{dt} = \frac{F_j \rho_j Cp_j (T_{jsp} - T_j) - Q_j}{V_j \rho_j Cp_j} \tag{5.9}$$

$$k_1 = \exp\left(k_1^1 - \frac{k_1^2}{T_r + 273.15}\right) \tag{5.10}$$

$$k_2 = \exp\left(k_2^1 - \frac{k_2^2}{T_r + 273.15}\right) \tag{5.11}$$

$$W = MW_A M_A + MW_B M_B + MW_C M_C + MW_D M_D \tag{5.12}$$

$$M_r = M_A + M_B + M_C + M_D \tag{5.13}$$

$$Cp_r = \frac{(Cp_A M_A + Cp_B M_B + Cp_C M_C + Cp_D M_D)}{M_r} \tag{5.14}$$

$$Q_r = -\Delta H_1 (k_1 M_A M_B) - \Delta H_2 (k_2 M_A M_C) \tag{5.15}$$

$$Q_j = UA\left(T_j - T_r\right) \tag{5.16}$$

$$A = \frac{2W}{\rho r} \tag{5.17}$$

In the above equations, M_i is the amount of mole of component "i", T_j is the jacket temperature, T_{jsp} is the set point value of the temperature control system, T_r is the temperature of reactor. The other meaning of parameters is explained in the nomenclature.

The dynamic model of the batch process is developed based on the equations shows above. The process parameter values are listed in Table 5.1. The initial values of M_A, M_B, M_C and M_D are selected as 12, 12, 0, and 0 kmol respectively. In this work the initial temperature of reaction mixture was assumed as 20℃ based on the previous work (Cott and Macchietto,1989; Aziz et al., 2000; Arpornwichanop et al., 2005; Karer et al., 2007). The sampling time of the process is 1 min. The end time of the batch process is selected as 200 min.

Process parameter values	
MW_A = 30 kg/kmol	Cp_A = 75.31 kJ/(kmol°C)
MW_B = 100 kg/kmol	Cp_B = 167.36 kJ/(kmol°C)
MW_C = 130 kg/kmol	Cp_C = 217.57 kJ/(kmol°C)
MW_D = 160 kg/kmol	Cp_D = 334.73 kJ/(kmol°C)
k_1^1 = 20.9057	$\Delta H_1 = -41840$ kJ/ kmol
k_1^2 = 10000	$\Delta H_2 = -25105$ kJ/ kmol
k_2^1 = 38.9057	ρ_j = 1000 kg/m^3
k_2^2 = 17000	ρ = 1000 kg/m^3
r = 0.5 m	Cp_j = 1.8828 kJ/(kg°C)
F_j = 0.348 m^3/min	V_j = 0.6912 m^3
U = 40.842 kJ/(min m^2 °C)	

Table 5.1 Process parameter values

5.4 Results of Simulation and Data Partition.

To develop the BAGDBN model, simulated process operation data are generated. This study assumes and employs piecewise constant control via manipulated variables, because the solution's form is appropriate for implementation on a digital computer. The piecewise constant control via temperature variables was widely applied for many chemical processes optimisation and different control policies were used for different objectives in many chemical processes (Xie et al.,2002; Arpornwichanop et al., 2005; Bouhenchir et al., 2006; Wang et al.,2015; Hemalatha et al., 2018). Base on the previous work, the optimal reaction temperature fell into the range from 90℃ to 100℃ and the process yields the best results at around 95℃ (Cott and Macchietto,1989; Aziz et al., 2000; Arpornwichanop et al., 2005; Karer et al., 2007). 120 piecewise initial temperature profiles with 10 intervals each are selected to generate simulated process operational. The temperature changes between two consecutive intervals are limited to 6℃ as temperature usually has slow dynamics and too large temperature change within a short time period is infeasible. The reason why 10 intervals of temperature profile are chosen for this optimasation is that as the number of intervals increases, the degree of freedom in optimisation increases and it is shown in (Arpornwichanop, 2005) that an optimal control profile with 10

intervals performs significantly better than those with 1 or 5 intervals but only marginally worse than those with 20 or 40 intervals. Figures 5.1 shows the temperature profiles for the 120 batches. As it is difficult to identify the 120 control profiles from Figure 5.1, Figures 5.2 and Figure 5.3 present the control profiles of 18 batches. The final values of Mc of 120 batches are presented in Figure 5.4. The prediction of Mc at the final batch time (t_f) can be formulated as,

$$\hat{y}(t_f) = f[T_{jsp1}, T_{jsp2}, \dots, T_{jsp10}] \tag{5.18}$$

where $\hat{y}(t_f)$ is the prediction of Mc at the end of batch, $[T_{jsp1}, T_{jsp2}, \dots, T_{jsp10}]$ are the 10 intervals values of temperature setpoints, and $f[T_{jsp1}, T_{jsp2}, \dots, T_{jsp10}]$ is a nonlinear function.

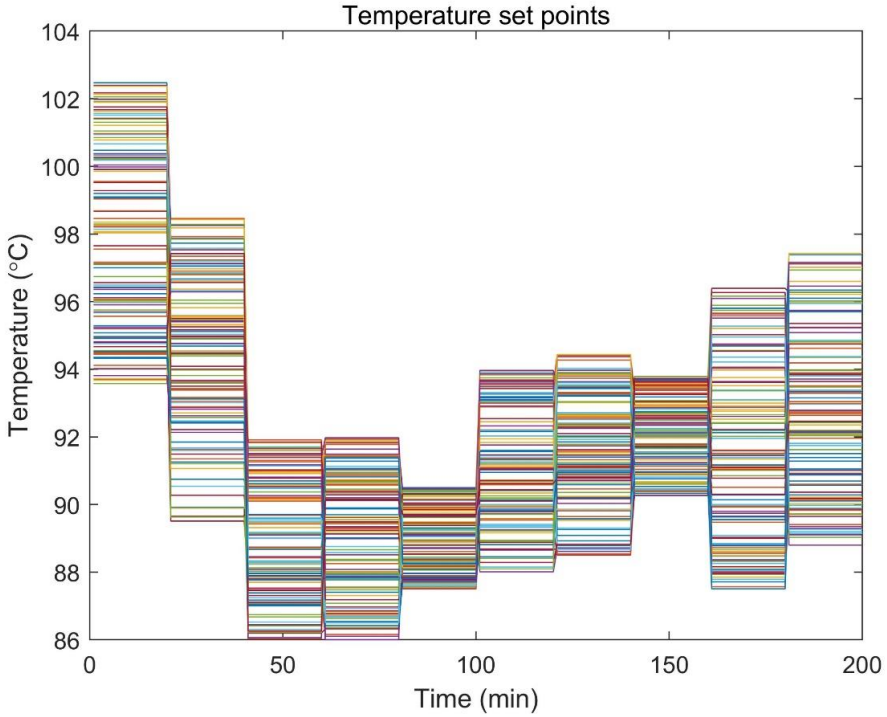

Figure 5.1 Temperature profiles for the 120 batches

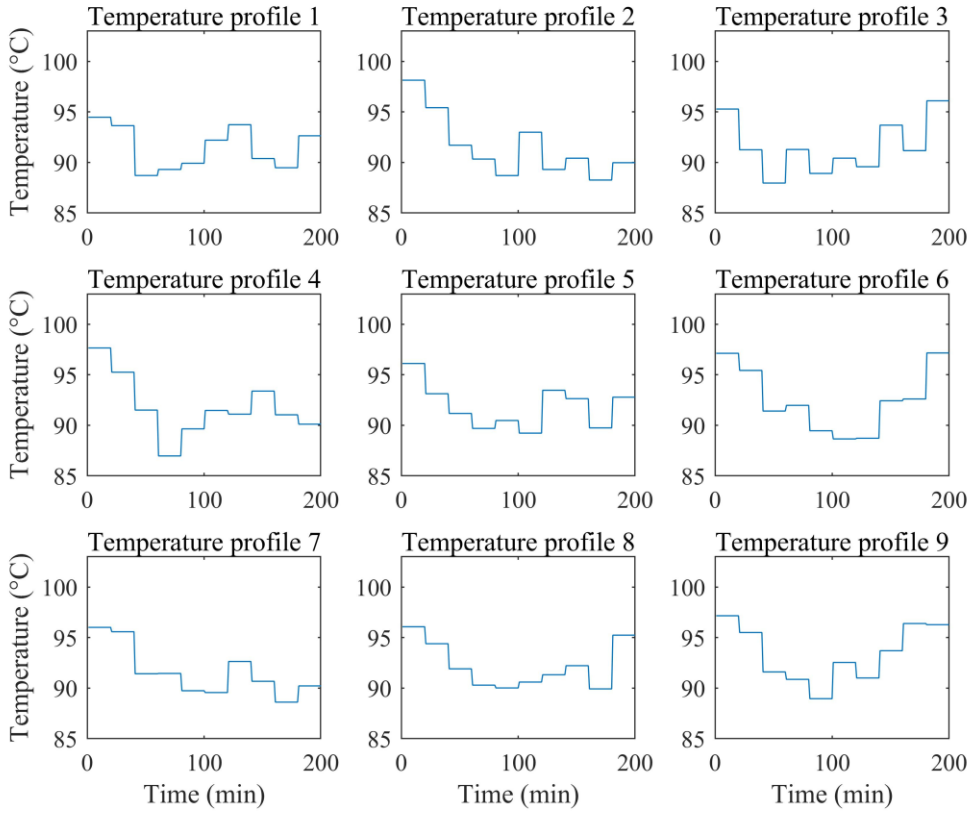

Figure 5.2 Temperature profiles (batch 1 to batch 9)

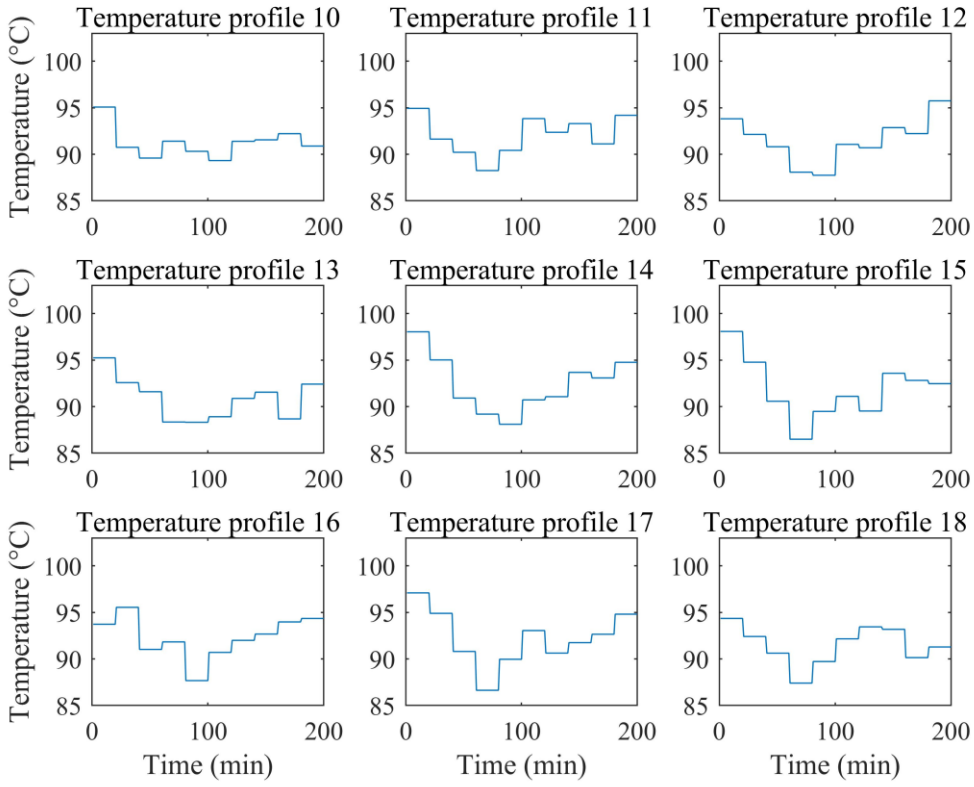

Figure 5.3 Temperature profiles (batch 10 to batch 18)

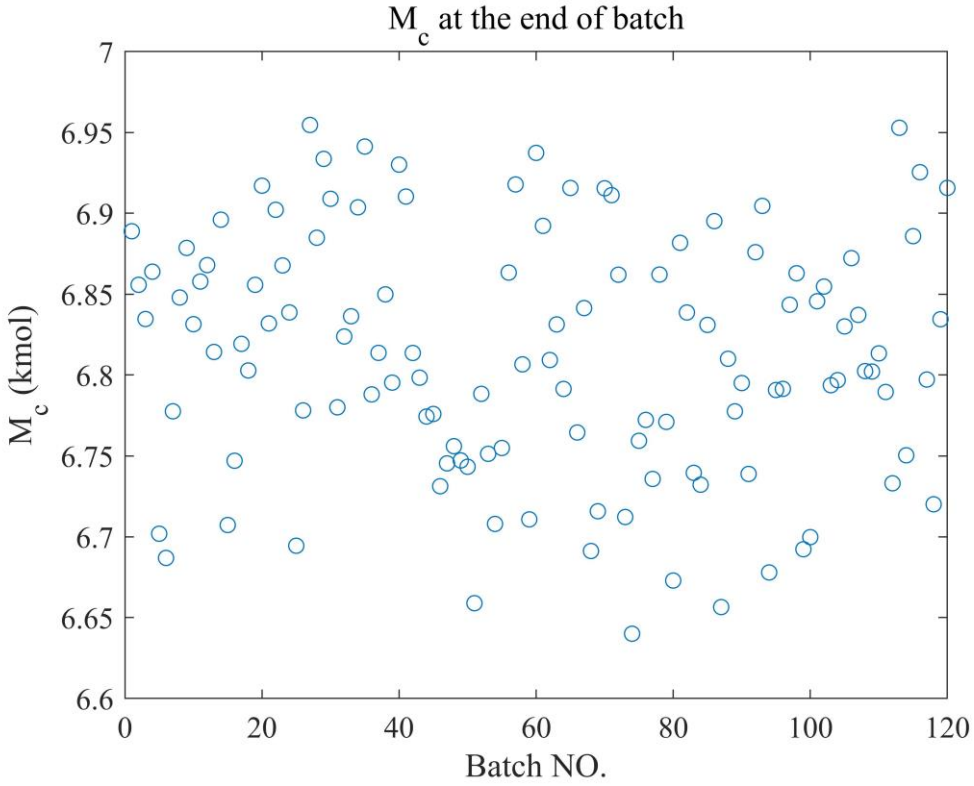

Figure 5.4 Mc at the end of batch

Because the measurement noises always exist in a real plant, the quality data of Mc at the batch end are corrupted with zero mean Gaussian noise and a standard deviation of 0.015. All the input data and output data were scaled to zero mean and unit variance before they are used in developing BAGDBN. The generated process data needs to be split into 3 subsets, training, testing and unseen validation data sets to develop the BAGDBN model. The training data set is for DBN training and the testing data is for DBN structure selection. The unseen validation data are used to test the performance of the final developed BAGDBN model. Table 5.2 shows the partition of the process data. In this study, bootstrap re-sampling with replacement is applied to the original modelling data (i.e. the training and testing data) to generate multiple replications. Each replication data set is randomly partitioned into training and testing data sets for developing a DBN model.

Data set	Number of batches
Training and testing data	95
Validation data	25

Table 5.2 Partition of data sets for BAGDBN modelling

5.5 Reliable Optimal Control through the Incorporation of Model Prediction Confidence Bounds

The objective of this batch reactor is to produce a maximum amount of product C in the fixed batch time. The manipulated variable is T_{jsp}. However there always are errors between predictions of the data-driven model and the process. To improve the performance of optimisation and reduce the impact of model errors on optimisation, the objective function can be formulated as:

$$\min_{U,t_f} J = -M_C(t_f) + \lambda \sigma_e \tag{5.19}$$

s.t.

$$Mc(t_f) = f_{BAGDBN}(U),$$

$$T_L \leq T_{jsp} \leq T_U$$

where U is a control profile, i.e. a vector of control actions containing the 10 reactor temperature setpoints, T_{jsp} is the reactor temperature set point which is bounded within its lower bound, T_L, and upper bound, T_U, t_f is the final batch time, σ_e is the model prediction standard error, and λ is the weight for model prediction standard error. The confidence bound of BAGDBN can be calculated from individual DBNs predictions. The standard error, σ_e, of the individual DBN model predictions can be calculated as:

$$\sigma_e = std(\hat{y}(t_f) - \bar{y}(t_f)) \tag{5.20}$$

where $\hat{y}(t_f)$ is the prediction of quality variables and $\bar{y}(t_f)$ is the mean of the predictions. The predictions errors of BAGDBN are assumed as normally distributed, then the 95% confidence bound can be calculated as $\hat{y}(t_f) \pm 1.96\sigma_e$. A narrow confidence bound of prediction indicates that the model prediction is reliable. In this objective function, the amount of Mc at the final batch time is maximized by the implementation of appropriate reactor temperature setpoints. The model predictions standard error σ_e is minimized in this objective function in order to penalise wide prediction confidence bound. The optimisation problem was solved by using the sequential quadratic programming (SQP) method implemented in the MATLAB Optimisation Toolbox. SQP is a numerical optimisation method for solving nonlinear programming problems with equality and inequality constraints. The basic idea of SQP is to approximate the nonlinear programming problem by a sequence of quadratic programming subproblems, which are easier to solve. The SQP approach iteratively solves a sequence of QP subproblems, where each subproblem involves finding a feasible search direction based on an

estimate of the constraints and objective function near the current iterate. This search direction is then used to update the current iterate, yielding a new candidate solution. Iterations are repeated until a satisfactory solution is reached or convergence cannot be achieved. One of the key advantages of the SQP method is that it can handle non-convex optimisation problems with general nonlinear constraints. However, it does require the computation of gradient and Hessian matrices, which can be computationally expensive, especially for large-scale problems. Additionally, it relies on the assumption that the optimisation problem is twice differentiable. Overall, SQP is a powerful optimisation method that can be very effective in certain settings, but its performance may vary depending on the specific problem at hand. After the optimisation, the model prediction will have a narrow confidence bound, indicating that the model prediction under the calculated optimal control policy is reliable. The optimal control policy is more robust than the result without considering the prediction confidence bounds (Zhang, 1999b).

5.6 Results and Discussions

5.6.1 *Prediction Results*

In this study, a BAGDBN containing 30 DBNs are developed. 30 different replications of the original training and testing data set are produced by resampling from the initial training and testing data set by using bootstrap resampling with replacement. On each replication, a DBN model is developed.

The mean-squared error (MSE) on scaled training & testing data and validation data of individual DBNs are given in Figure 5.5. Figure 5.5 indicates that the performances of DBNs on the training & testing data and validation data are not consistent. For example, the MSE of the 1st DBN is larger than that of the 2nd DBN on the training and testing data. However, the opposite results occur on the validation data achieved by the 1st and the 2nd DBNs. The situations between the 4th and the 5th DBNs, the 6th and the 7th DBNs, the 12th and the 13th DBNs, the 19th, the 20th and the 21st DBNs, the 23rd and the 24th DBNs, the 28th and the 29th DBNs are the same as the 1st and 2nd DBNs. It indicates the performance of individual DBNs is not reliable.

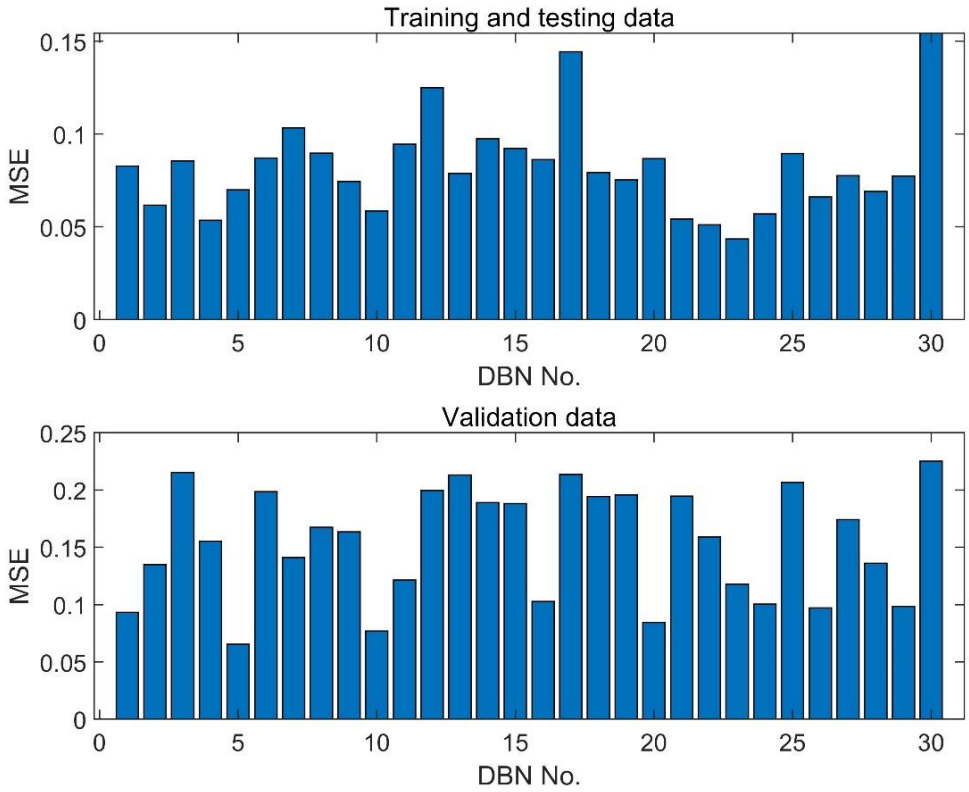

Figure 5.5 MSEs of individual DBNs

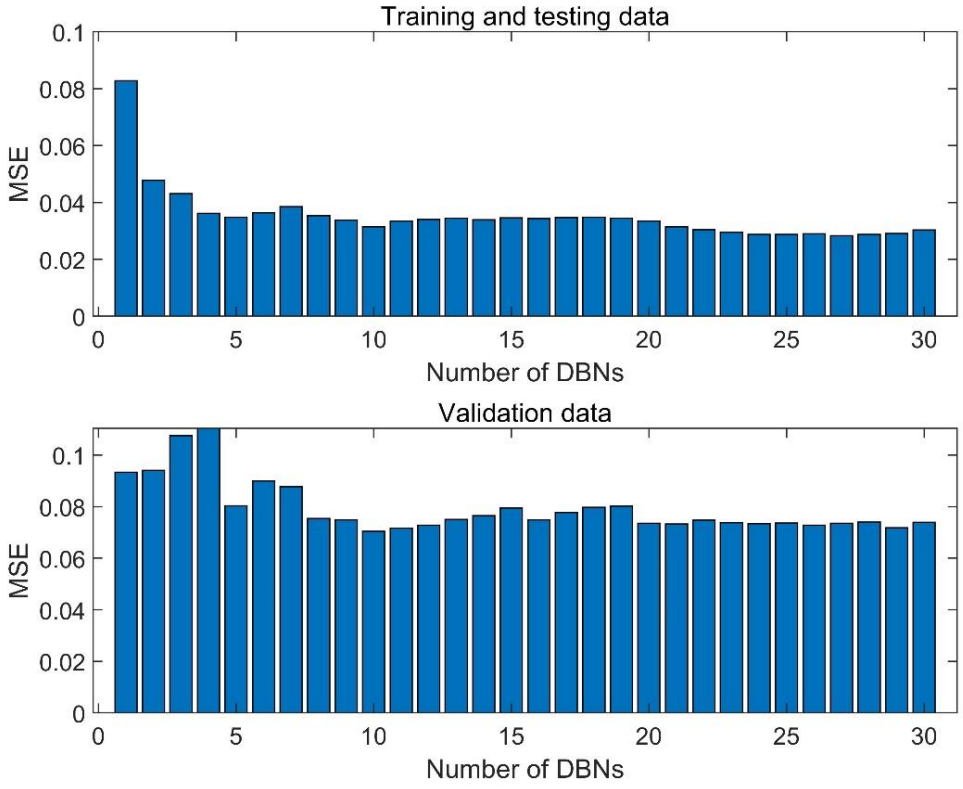

Figure 5.6 MSEs of BAGDBNs

Figure 5.6 shows the MSEs of BAGDBNs containing different number of DBNs on training & testing data and validation data. The first bar represents the first DBN model, the second bar represents aggregating the first two DBN models, and the last bar represents aggregating all the 30 DBN models. It can be observed that the MSEs on both data sets become more consistent with the number of DBNs in BAGDBN increases. And the MSEs are significantly decreased by combining more than 7 DBN models in a BAGDBN. BAGDBN gives better performance on prediction of Mc than individual DBNs. And from the comparison between Figure 5.5 and 5.6, the BAGDBN model improves the robustness of DBN model. Figure 57 shows the prediction of Mc at end time (t_f = 200 min) achieved by BAGDBN containing 30 DBNs. It can be observed that the prediction on validation data is very accurate.

5.6.2 *Optimisation Results*

Optimisation based on single DBN models are carried out for comparison. The objective of the optimisation is to maximize the amount of the desire product C at the end batch time. The function of the optimisation can be formulated as,

$$\min_{U,t_f} J = -M_C(t_f) \tag{5.21}$$

s.t.

$$\mathrm{Mc}(t_f) = \mathrm{f_{BAGDBN}(U)},$$

$$T_L \leq T_{jsp} \leq T_U$$

where M_C is the amount of product C, t_f is the final batch time. $\mathrm{f_{BAGDBN}(U)}$ represents the function of prediction achieved by BAGDBN model via control profiles, U. The jacket temperature is given as T_{jsp}. T_L and T_U are the temperature lower bound and upper bound respectively. To solve this optimisation problem, SQP methodology was employed in this study. From Figure 5.7, it can be observed that in validation data set, the 18th batch of process validation data has the largest value of Mc. The process input variables of the 18th batch data are selected as the initial values for optimisation. The lower bound and the upper bound of input were set as 85 ℃ and 105 ℃ respectively according to the Figure 5.1. The optimisation results of using 30 individual DBNs are shown in Figure 5.8. From Figure 5.8, the prediction of DBNs under each of the optimal control policies is larger than the actual value of simulation. The results are various and the optimisation result using the 23rd DBN, which gives best performance of prediction on the training and testing data as shown in Figure 5.5, is shown in

the Table 5.3. However, this optimal control policy does not lead to the best performance when applied to the actual process (i.e. mechanistic model based simulation) as indicated by Figure 5.8.

DBN NO.	DBN model (kmol)	Process simulation (kmol)
23	7.142	6.945

Table 5.3 Optimal results of the 23rd DBN

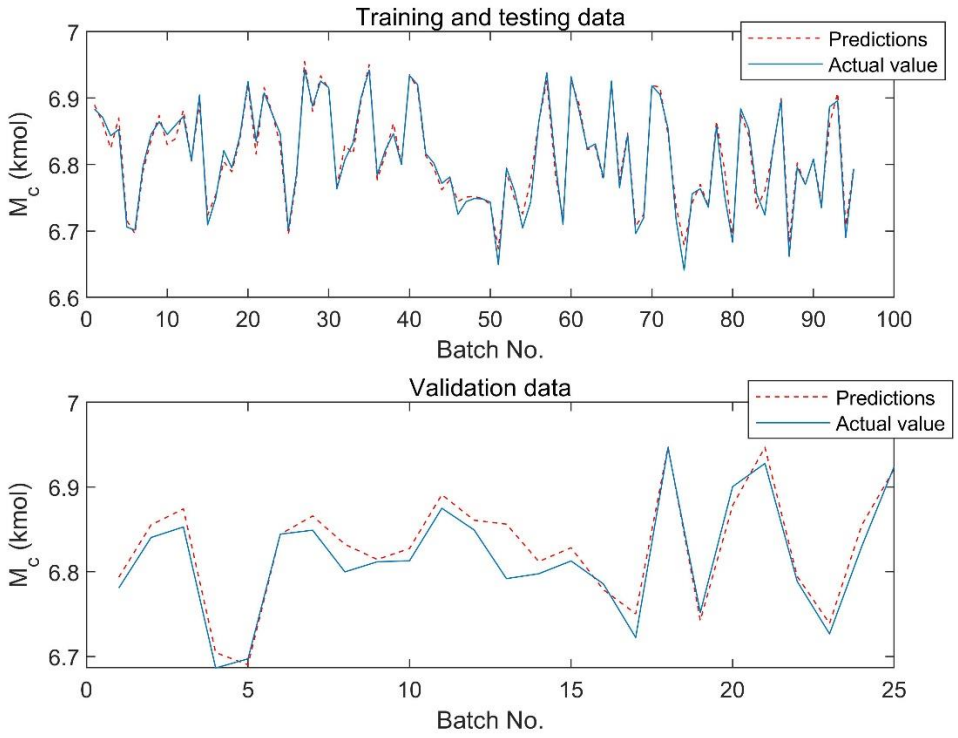

Figure 5.7 Predictions of Mc at the end time achieved by BAGDBN

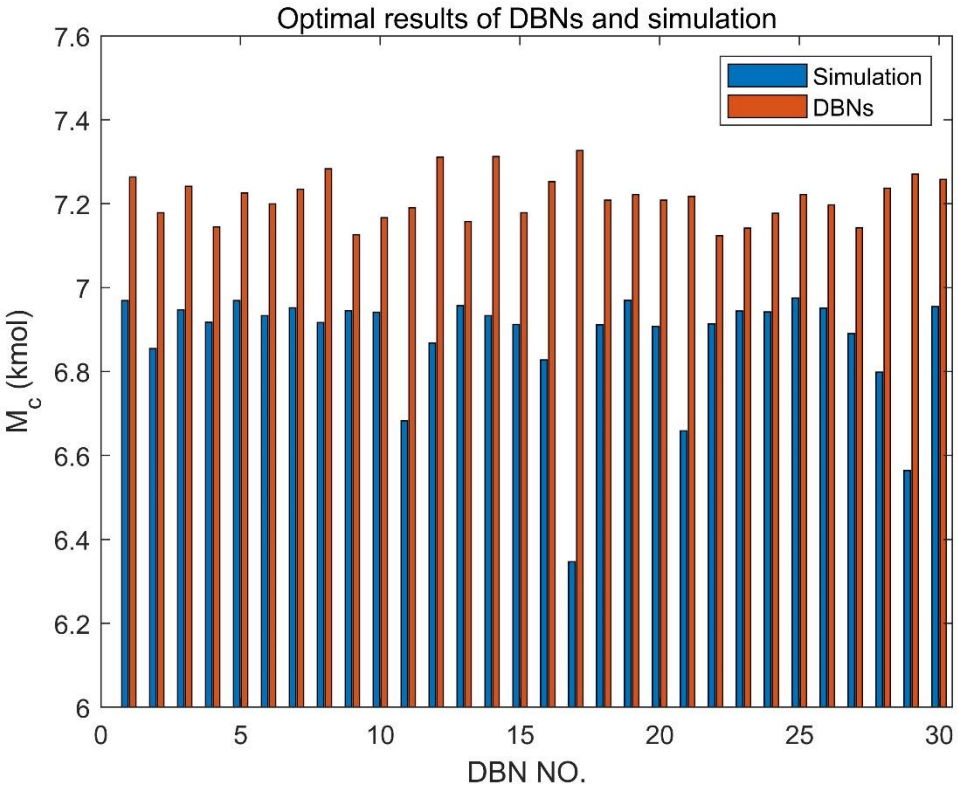

Figure 5.8 Optimisation results of 30 individual DBNs

The optimal control policies from 30 DBNs are shown in Figure 5.9. It indicates that 30 DBNs give various control policies. When applying the optimal control policies in the simulation, the results are not consistent to the predictions achieved by DBNs. For example, the control policy from the 17th DBN gives the highest prediction of Mc on the DBN model but the lowest value of Mc when applied to the actual process (i.e. the mechanistic model based simulation). Therefore, the optimisation from individual DBNs is not reliable. Figure 5.8 shows that the optimal control policy obtained using the 5th DBN give the best performance on the actual process and this optimal control policy is shown in Figure 5.10.

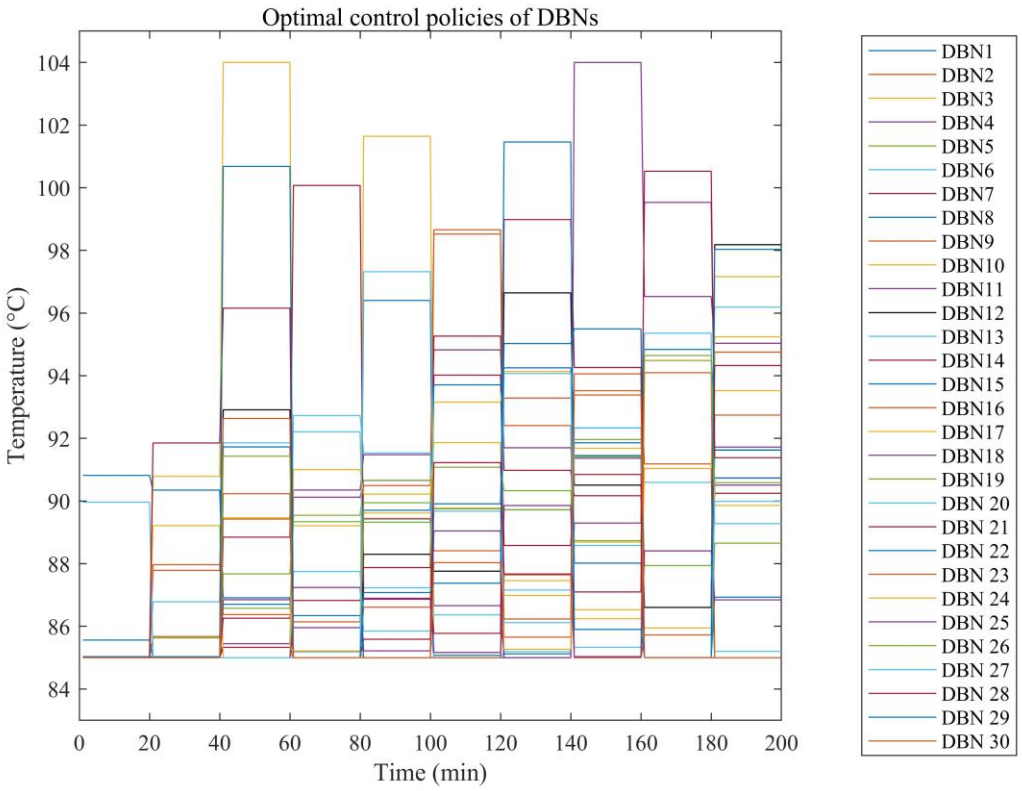

Figure 5.9 Optimal control policies from 30 individual DBNs

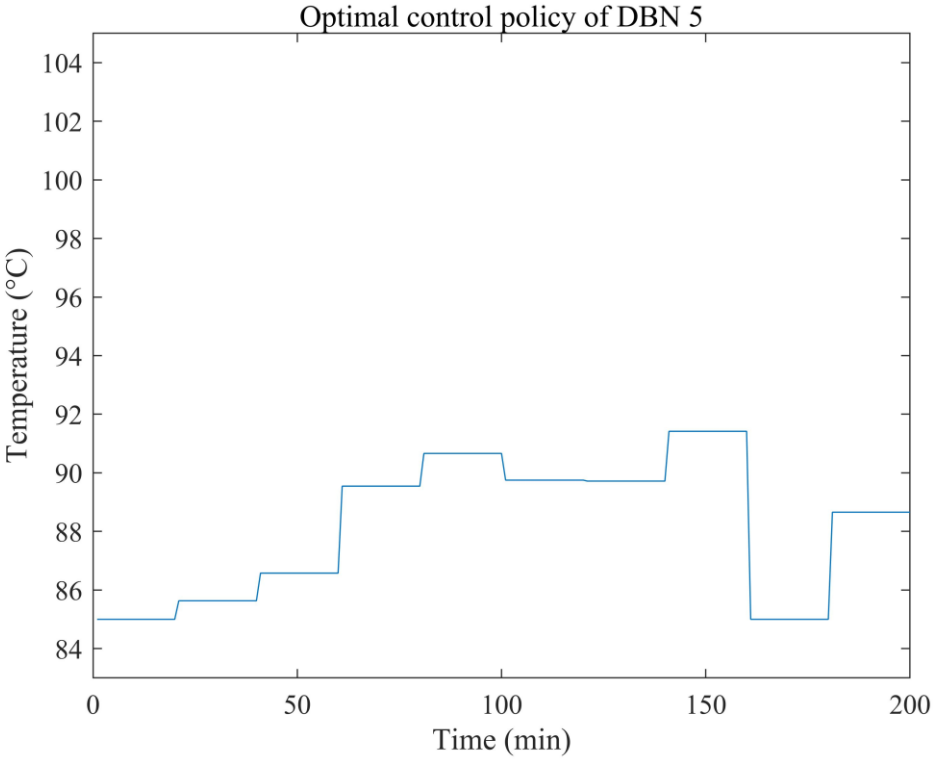

Figure 5.10 Optimal control policy from DBN 5

To optimise the batch process operation using BAGDBN through the incorporation of model prediction confidence bounds, the initial values of temperature control profile, the upper bound and lower bound are set as the same as the optimisation of single DBNs. Forty different weights for confidence bound within the range from 0 to 9.75 were tried. To show the robustness of optimisation using BAGDBN, four different BAGDBNs containing the first 20, 25, 28 and 30 DBNs were used to optimise the batch process operation.

Figures 5.11, 5.12 and 5.13 show the optimisation results of BAGDBN containing 20, 25 and 30 DBNs with different values of weights for confidence bounds. From Figure 5.11, it can be observed that the predictions of BAGDBN containing 20 DBNs decreases when the weight of confidence bound increase. The confidence bound gets narrower and the differences between the BAGDBN predictions and the true values get smaller and, hence, optimisation reliability improves. The results of simulation are at a stable value when the weight of confidence bound increases. The lower confidence bound indicates the worst-case performance. The appropriate value of the weight for the confidence bound can be determined when the worst-case performance is the best. In this case, the appropriate value for the confidence bound is 3. Figure 5.12 indicates the same situation of predictions of BAGDBN and errors between predictions and simulation as shown in Figure 5.11. However, the greatest value of Mc on the actual process (i.e. mechanistic model simulation) when the weight of confidence bound is around 4.5. In Figure 5.13, the actual value of Mc meets the largest value when the weight is around 3.75. Then error between predictions of BAGDBN and simulation and the width of confidence bounds becomes smaller with the weight increasing. The BAGDBN model containing 25 and 30 gives similar results on prediction and simulations.

Table 5.4 gives the best optimal results of BAGDBN. Comparing Table 5.3 with Table 5.4, it can be seen that the errors of BAGDBN between prediction and simulation are much smaller than those of DBNs. The amount of Mc under the optimal control policy from the BAGDBN containing 25 DBNs achieves the greatest value of 7.003 kmol on simulation. The results from BAGDBN containing 20, 25 and 30 are very closer when applying the optimal control policies in the process simulation. It indicates the optimisation from BAGDBN is very reliable.

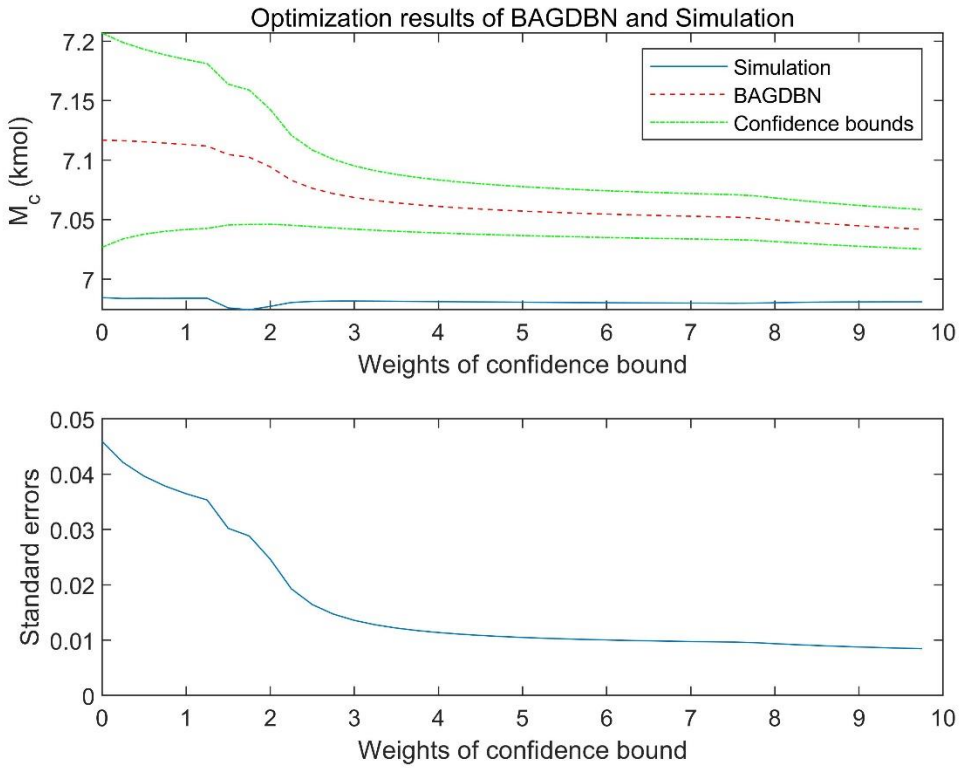

Figure 5.11 Optimisation results of BAGDBN (containing 20 DBNs)

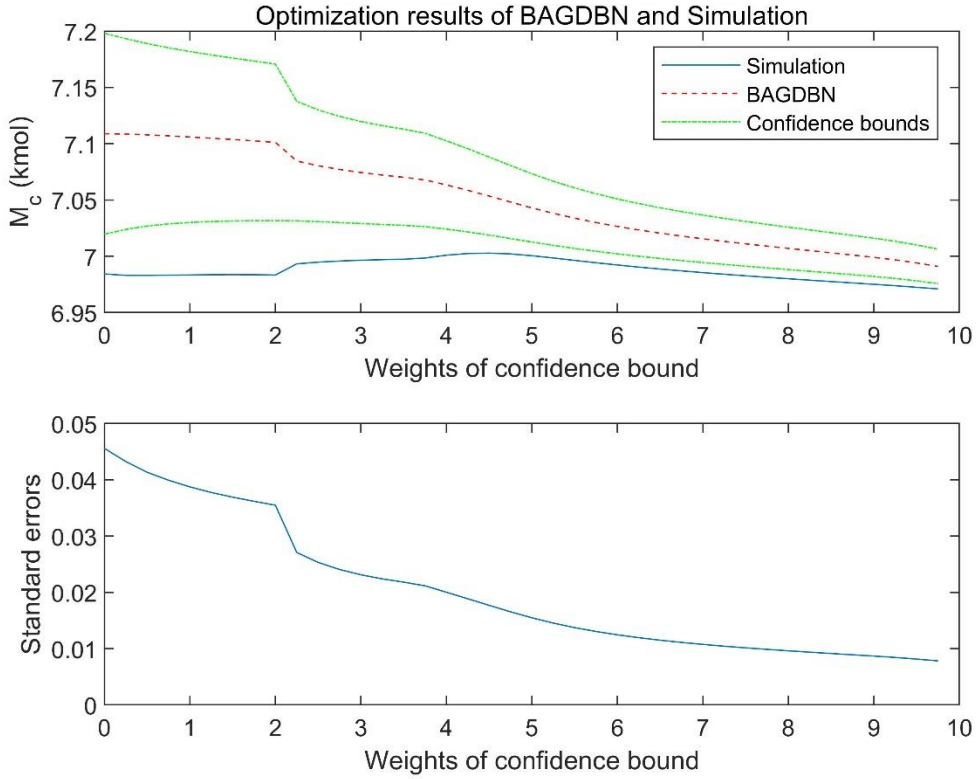

Figure 5.12 Optimisation results of BAGDBN (containing 25 DBNs)

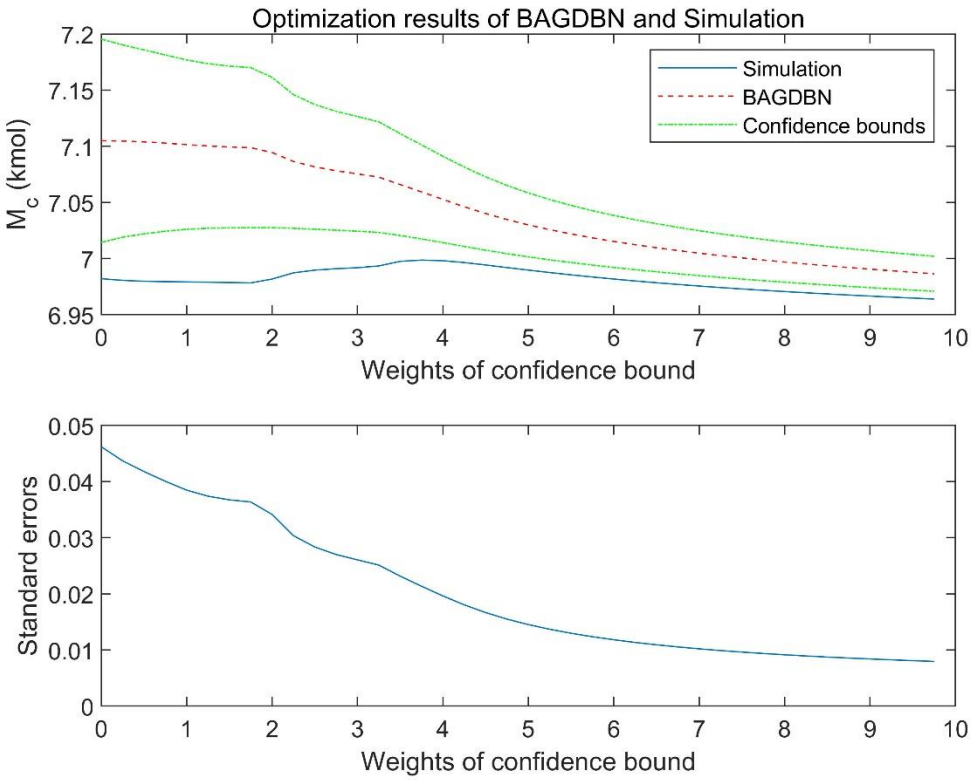

Figure 5.13 Optimisation results of BAGDBN (containing 30 DBNs)

Number of DBNs in a BAGDBN	BAGDBN (kmol)	Simulation (kmol)	Weight of confidence bound
20	7.069	6.981	3
25	7.054	7.003	4.5
30	7.059	6.999	3.75

Table 5.4 Best optimal results of BAGDBNs

To illustrate the robustness of BAGDBN on the optimisation of the batch process. A BAGDBN containing 28 DBNs was used to achieve an optimal control policy for the comparison with BAGDBNs containing 25 and 30 DBNs. Figure 5.14 shows the optimisation results of BAGDBN containing 28 DBNs and simulation at different values of weight for confidence bound. From the figure, it can be observed that the result of simulation achieves the highest value of 6.999 kmol, when the weight of confidence bound is around 3.75. The bar chart of best optimisation results of Mc from different BAGDBNs containing 25, 28 and 30 DBNs is shown in Figure 5.15. The predictions of Mc and actual values of simulations are very similar and

consistent. Figure 5.16 gives the optimal control policies from these three BAGDBNs. It indicates that three similar control policies are obtained from the optimisations when the weight for confidence bound are selected appropriate values. From Figures 5.8, 5.9, 5.15 and 5.16, BAGDBN improve the accuracy and robustness of predictions of DBN. The results from individual DBNs are not reliable and consistent. The optimal control policies calculated by individual DBNs are various and lack of stability. Optimisation using a BAGDBN model can lead to a reliable optimal control policy for batch process to achieve the maximum results of Mc.

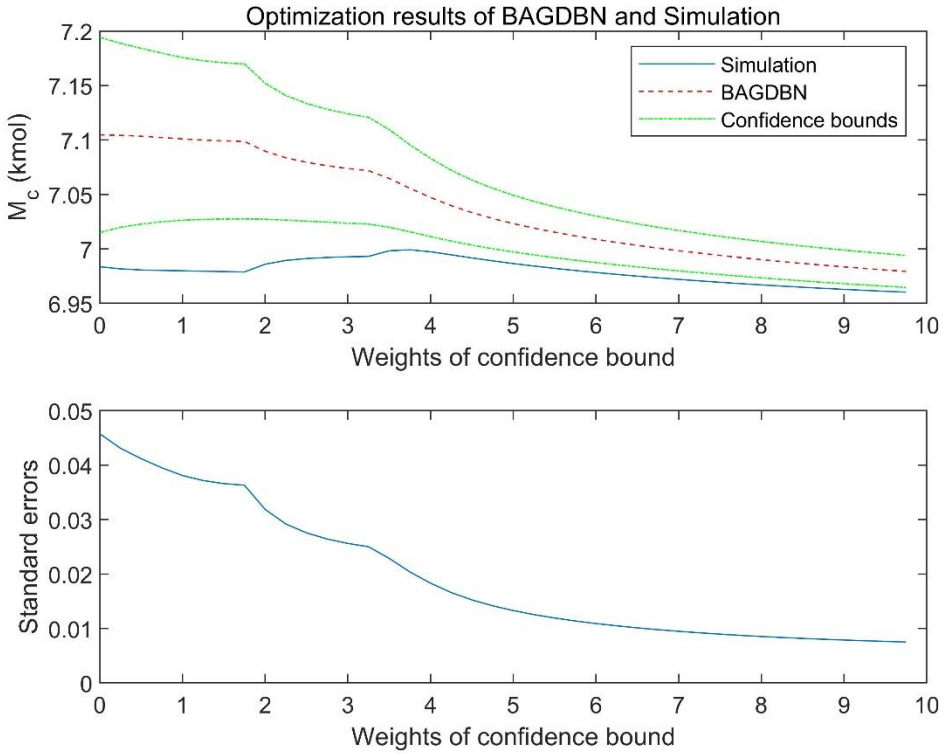

Figure 5.14 Optimisation results of BAGDBN (containing 28 DBNs)

5.7 Conclusions

In this work, a BAGDBN model based optimal control strategy is introduced for batch reactor control. A BAGDBN model is developed by using a limited amount of process operational data to predict the amount of product Mc at the end of a batch. The modelling results demonstrate that BAGDBN are more reliable and robust than a single DBN model. The model prediction confidence bound is incorporated in the optimisation objective function so that a wide confidence bound at the end of a batch is penalised. Therefore, the reliability of calculated optimal control is significantly improved. The optimal control policies calculated by individual DBNs give various and inconsistent performances. This indicates that the optimal control

policies from individual DBN models are unreliable and may not be optimal at all on the actual process. BAGDBN model improve the reliability and robustness of data driven model and, through incorporating model prediction confidence in the optimisation objective function, reliable optimal control policy can be obtained for batch processes.

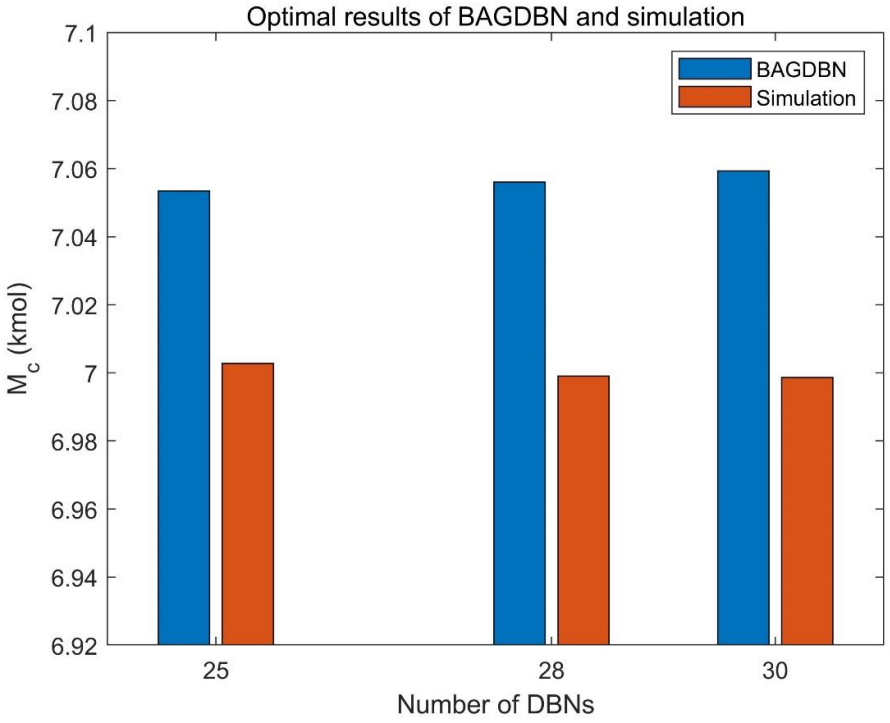

Figure 5.15 Optimisation results of BAGDBN (containing 25, 28, 30 DBNs)

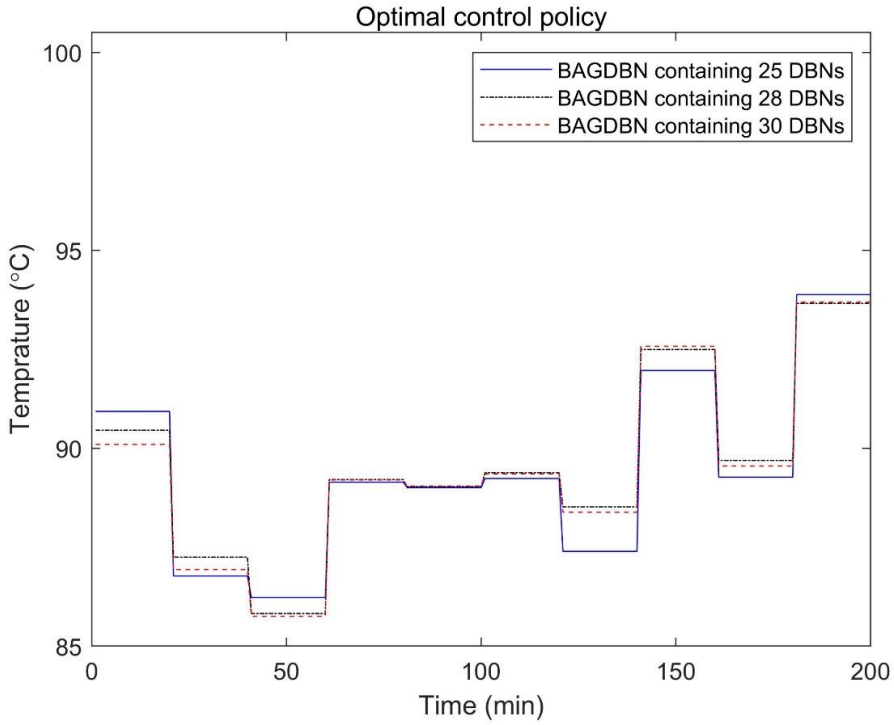

Figure 5.16 Optimal control policies from BAGDBN (containing 25, 28 ,30 DBNs)

Chapter 6. Conclusions and Recommendations for Future Works

6.1 Conclusions

This study focuses on deep learning techniques for nonlinear process modelling, optimisation, and control. The main objective of this thesis is to develop reliable and robust computational intelligence-based data-driven models for nonlinear processes. The improvement of accuracy and robustness of the DBN model was investigated. The optimisation of the DBN model structure was investigated for the modelling of chemical processes. The development of DBN and BAGDBN based soft sensors for a key quality variable in an industrial chemical process was carried out in this thesis. Reliable optimisation of batch processes using BAGDBN models was investigated. Three different processes, industrial polypropylene polymerization process, water level control of a water tank, and a batch reactor process are the cases used to judge the performance of novel data-driven modelling approaches.

Because of the lack of online measurement of polymer MI of the industrial polypropylene polymerization process, building soft sensors by finding the relationship between the easy-to-measure process variables and the difficult-to-measure quality variables is an efficient approach to overcome this issue. This polypropylene polymerization process is a plant located in China. Several grades of polymer products were produced within a campaign period of one month. During this campaign period, the process operational data were logged every half hour and the quality data, polymer MI, were logged every 2 hours. In this study, 30 different process variables were measured. However, the polymer MI is only correlated to a few process variables. Through the sensitivity analysis, hydrogen concentration and the feed rate of hydrogen were selected as process variables for the estimation of polymer MI. DBN model was developed from the recorded process and quality variables identified by the cross-correlation analysis. A conventional neural network was also developed for comparison. Because the numbers of process data samples are much larger than that of the quality data, not every process data sample has corresponding polymer MI data because the quality variables are much more difficulty to obtain than process variables. This is common in many industrial processes. Conventional data-driven models are usually developed from data sets with equal numbers of process data samples and quality data samples. Any process data samples without the corresponding quality data samples are usually discarded. Therefore, a large amount of process data that contain lots of profuse information are left and unused. The advantage of the DBN model is that it can extract the important profuse information from these unused process variables to improve the accuracy of estimation.

A reliable and accurate DBN model must be developed with appropriate architecture. The process and quality data samples are divided into three parts for the cross-validation to determine the number of neurons in each hidden layer. From the partition of the data, training data are used to train the network, testing data are used for preventing the overfitting of the model, and the unseen validation data are used to evaluate the final developed models. The SSE values were considered as the performance indicator to judge the performance of different models. DBN models with different structures were developed. The DBN which gives the lowest SSE on the testing data is considered to have the appropriate structure.

From the comparison of polymer MI estimations, the DBN model has been proved to be capable of giving more accurate online inferential estimations of polymer quality than a conventional neural network. The DBN model with an appropriate structure can extract profuse latent information from unused process data samples without corresponding quality data samples. When applying the data-driven models to highly nonlinear and complex processes, the performance of the conventional neural network is shown to be poorer than that of the DBN. It demonstrated the DBN model has stronger and more reliable generalisation capability for the actual industrial chemical process than the conventional neural network. It provided a valuable reference for data-driven empirical model applications.

The robustness of data-driven models is an important requirement for real industrial chemical process applications. Combining predictions from individual forecasting models is an effective and verified approach for improving the accuracy and robustness of conventional data-driven models. To improve the robustness of DBN models, a novel model named BAGDBN was developed in this thesis for the modelling of chemical processes. Several single DBN models with appropriate structures were developed and combined in one BAGDBN model. Because the quality data from many actual industrial chemical processes are often limited, the technique of bootstrap re-sampling with replacement was used to generate replications from the original data. These replications were used to develop the DBN models in the BAGDBN model. Each DBN model was developed on a bootstrap replication of the original modelling data set. Predictions of product quality variable from individual DBN models were combined to obtain the final prediction of BAGDBN. The failure of some DBN models can be compensated by other DBN models. Two application case studies were used for testing the performance of the BAGDBN model. The first case study was dynamic modelling of a conical water tank. The multi-step ahead predictions of water level were achieved by the BAGDBN model and single DBN model. From the results, the BAGDBN model gives more consistently accurate results on different data sets than the single DBN model. The multi-step ahead predictions of water level

achieved by BAGDBN were more accurate than the DBN model. The second application is the inferential estimation of polymer MI in an actual polypropylene polymerization process. The results also demonstrate the BAGDBN model is more accurate and robust than the DBN model. The inferential estimation of polymer MI from the BAGDBN model is more accurate and reliable than that from the DBN model.

Batch reactors play a vital role in industrial production processes as they enable the production of high value-added products including pharmaceuticals and specialty chemicals. These reactors are versatile in nature. They can be utilised for the production of various products or different grades of the same product. It is usually effort demanding to develop a detailed mechanistic model for a batch chemical process. Because the issue of time-consuming in developing and utilising mechanistic models, this type of model may not be used for the on-line optimisation of batch processes. To overcome this problem, BAGDBN can be developed for the optimisation of batch processes. A simulated batch reactor is used as a case study to demonstrate reliable optimisation control using BAGDBN models. The objective of the batch reactor is to produce a maximum amount of desired product in the fixed batch time. In this case study, 120 batches of simulated process data were obtained using different reactor temperature control profiles with 10 intervals. These data were divided into 3 parts, training data, testing data and unseen validation data. Because the measurement noises are always present in a real plant, the simulated quality data of Mc were corrupted with zero mean Gaussian noise. Comparisons were carried out between DBN and BAGDBN models. From the prediction of desired product at the final batch time achieved by both models, BAGDBN gives more accurate predictions than single DBN. This indicates that the technique of BAGDBN model enhances the accuracy and robustness of the DBN model. For obtaining better optimisation results and reliable control policy, the confidence bound of prediction calculated by the BAGDBN model is incorporated into the optimisation objective function. In addition to optimizing the process operation objectives, the width of model prediction confidence bound is minimized. By minimizing the width of model prediction confidence bound, the reliability of the optimal control policy is enhanced. From comparing the results between DBN models and BAGDBN models, it is demonstrated that BAGDBN is a more accurate and reliable data-driven model than single DBN model. From comparing the results of control policies between BAGDBN models with different numbers of DBN models, the control policies are very similar as long as BAGDBN include sufficient numbers of DBN models. It is shown that the three similar control policies from the optimisation using the BAGDBN models are reliable for the batch reactor process.

6.2 Recommendations for Future Works

Based the knowledge from this work, the suggestions and recommendations of future works to expand the scope of the present work are listed as follow,

1. To overcome the lack of online measurement for key quality variables, data-driven empirical models become very popular in past decades. Deep learning draws much attention in the research area of process control. In this work, the DBN model was developed and applied to an actual industrial chemical process. With limited process data supplied, the DBN model achieved significant performance for the estimation of key variable polymer MI. By using the method of cross-validation, the structure of the DBN model and BAGDBN model had been discussed and selected. This process can be time-consuming and resource-intensive, but it is necessary for developing robust and accurate machine learning models. Automated machine learning automates the process of hyperparameter tuning, including choosing the optimal number of layers and neurons, by performing an exhaustive search over the hyperparameter space (Cai et al., 2020). Neural architecture search employs machine learning techniques to automate the architecture search process (Ren et al., 2021). The goal is to find an optimal architecture that balances accuracy, efficiency, and other desirable properties, like interpretability or transferability. These efficient methods can be utilised for the process of hyperparameter tuning of DBN and BAGDBN models.

2. The training time of the DBN model for the modelled chemical process is impacted by the complexity of the chemical process and the amount of process data. The learning algorithm used in the supervised training phase of DBN training, the backpropagation algorithm, can be changed to other more efficient learning algorithms, such as the Levenberg-Marquardt algorithm. It may reduce the training time of the DBN model. The results of the predictions between this new model and conventional DBN should be compared and discussed in the future. DBN model with the Levenberg-Marquardt algorithm can be combined to develop a novel model.

3. BAGDBN based on the technique of bootstrap resampling was developed in this work to enhance the robustness of the DBN model. The predictions are a combination of predictions from DBN models in BAGDBN by using the simple average method. The method of principal component regression can be used in the combination phase of BAGDBN to enhance the structure of BAGDBN. The generalisation capability of BAGDBN may be improved by this method.

4. For further work on multiple-step ahead predictions of dynamic models, the technique of deep recurrent neural networks (DRNNs) can be combined with the BAGDBN model. DRNNs have been shown to perform better than traditional feedforward networks on sequential data due to their ability to capture dependencies across time. In addition, they can be stacked to form deeper architectures, allowing for even more complex feature extraction and pattern recognition. The performance of BAGDBN on multiple-step ahead predictions can be enhanced.

5. These new models mentioned can be used to predict the key quality variables and optimisation for other actual industrial chemical processes such as pharmaceuticals and food manufacturing processes. These methods and further application case studies should be investigated in future works.

References

Abdul, L., Rajasekar, S., Lin, D.S.Y., Venkatasubramania Raja, S., Sotra, A., Feng, Y., Liu, A. & Zhang, B., 2020. Deep-LUMEN assay - human lung epithelial spheroid classification from brightfield images using deep learning. *Lab on a Chip*, 2(24), pp. 4623–4631.

Al Hassan, M., Derradji, M., Ali, M.M.M., Rawashdeh, A., Wang, J., Pan, Z. & Liu, W., 2022. Artificial neural network prediction of thermal and mechanical properties for Bi2O3 - polybenzoxazine nanocomposites. *Journal of Applied Polymer Science*, 139(32), p. n/a–n/a.

Al-Mahrouqi, M.H. and Zhang, J., 2008. Reliable optimal control of a fed-batch bio-reactor using ant colony optimization and bootstrap aggregated neural networks. *IFAC Proceedings Volumes*, *41*(2), pp.8407-8412.

Ali, J.M., Hussain, M.A., Tade, M.O. and Zhang, J., 2015. Artificial Intelligence techniques applied as estimator in chemical process systems–A literature survey. *Expert Systems with Applications*, *42*(14), pp.5915-5931.

Alshehri, A.K., Ricardez-Sandoval, L.A. and Elkamel, A., 2010. Designing and Testing a Chemical Demulsifier Dosage Controller in a Crude Oil Desalting Plant: An Artificial Intelligence-Based Network Approach. *Chemical Engineering & Technology*, *33*(6), pp.973-982.

Åkesson, B.M. and Toivonen, H.T., 2006. A neural network model predictive controller. *Journal of Process Control*, *16*(9), pp.937-946.

Arpornwichanop, A., Kittisupakorn, P. and Mujtaba, I.M., 2005. On-line dynamic optimization and control strategy for improving the performance of batch reactors. *Chemical Engineering and Processing: Process Intensification*, *44*(1), pp.101-114.

Bagheri, M., Akbari, A. and Mirbagheri, S.A., 2019. Advanced control of membrane fouling in filtration systems using artificial intelligence and machine learning techniques: A critical review. *Process Safety and Environmental Protection*, *123*, pp.229-252.

Bengio, Y., 2009. *Learning Deep Architectures for AI*. Now Publishers Inc.

Bengio, Y. and LeCun, Y., 2007. Scaling learning algorithms towards AI. *Large-scale Kernel Machines*, *34*(5), pp.1-41.

Bengio, Y., Delalleau, O. and Le Roux, N., 2006. The curse of highly variable functions for local kernel machines. *Advances in Neural Information Processing Systems*, *18*, p.107.

Bhat, N. and McAvoy, T.J., 1990. Use of neural nets for dynamic modeling and control of chemical process systems. *Computers & Chemical Engineering*, *14*(4-5), pp.573-582.

Billings, S.A., 2013. Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-temporal Domains. John Wiley & Sons.

Bittanti, S. and Piroddi, L., 1997. Nonlinear identification and control of a heat exchanger: a neural network approach. *Journal of the Franklin Institute*, *334*(1), pp.135-153.

Bonvin, D., 1998. Optimal operation of batch reactors—a personal view. *Journal of Process Control*, *8*(5-6), pp.355-368.

Boroushaki, M., Ghofrani, M.B., Lucas, C. and Yazdanpanah, M.J., 2003. Identification and control of a nuclear reactor core (VVER) using recurrent neural networks and fuzzy systems. *IEEE Transactions on Nuclear Science*, *50*(1), pp.159-174.

Breiman, L., 1996. Bagging predictors. *Machine Learning*, *24*(2), pp.123-140.

de Canete, J.F., Gonzalez, S., del Saz-Orozco, P. and Garcia, I., 2010. A harmonic balance approach to robust neural control of MIMO nonlinear processes applied to a distillation column. *Journal of Process Control*, *20*(10), pp.1270-1277.

Cai, H., Lin, J., Lin, Y., Liu, Z., Wang, K., Wang, T., Zhu, L. & Han, S., 2020. AutoML for Architecting Efficient and Specialized Neural Networks. *IEEE MICRO*, 40(1), pp. 75–82.

Carvalho, C.B., Carvalho, E.P. and Ravagnani, M.A., 2020. Combined neural networks and predictive control for heat exchanger networks operation. *Chemical Industry and Chemical Engineering Quarterly*, *26*(2), pp.125-134.

Chen, J. and Huang, T.C., 2004. Applying neural networks to on-line updated PID controllers for nonlinear process control. *Journal of Process Control*, *14*(2), pp.211-230.

Chen, L.P., 2019. Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar: Foundations of machine learning.

Chetouani, Y., 2008. A non-linear auto-regressive moving average with exogenous input non-linear modelling and fault detection using the cumulative sum (Page-Hinkley) test: application to a reactor. *International Journal of Computer Applications in Technology*, *32*(3), pp.187-193.

Cho, S. B., & Kim, J. H. (1995). Multiple network fusion using fuzzy logic. *IEEE Transactions on Neural Networks*, *6*, 497–501.

Chu, J.Z., Tsai, P.F., Tsai, W.Y., Jang, S.S., Shieh, S.S., Lin, P.H. and Jiang, S.J., 2003. Multistep model predictive control based on artificial neural networks. *Industrial & Engineering*

Chemistry Research, *42*(21), pp.5215-5228.

Costarelli, D. & Spigler, R., 2013. Approximation results for neural network operators activated by sigmoidal functions. *Neural Networks*, 44, pp. 101–106.

Dam, R.S. de F., Salgado, W.L., Schirru, R. & Salgado, C.M, 2022. Application of radioactive particle tracking and an artificial neural network to calculating the flow rate in a two-phase (oil–water) stratified flow regime. *Applied Radiation and Isotopes*, 180, pp. 110061–110061.

Damour, C., Benne, M., Grondin-Perez, B. and Chabriat, J.P., 2010. Nonlinear predictive control based on artificial neural network model for industrial crystallization. *Journal of Food Engineering*, *99*(2), pp.225-231.

Desai, K., Badhe, Y., Tambe, S.S. and Kulkarni, B.D., 2006. Soft-sensor development for fed-batch bioreactors using support vector regression. *Biochemical Engineering Journal*, *27*(3), pp.225-239.

Deng, L. and Yu, D., 2014. Deep learning: methods and applications. *Foundations and Trends in Signal Processing*, *7*(3–4), pp.197-387.

Dı´ az, G., Sen, M., Yang, K.T. and McClain, R.L., 2001. Adaptive neurocontrol of heat exchangers. *J. Heat Transfer*, *123*(3), pp.556-562.

Du, X., Wang, J., Jegatheesan, V. and Shi, G., 2018. Dissolved oxygen control in activated sludge process using a neural network-based adaptive PID algorithm. *Applied Sciences*, *8*(2), p.261.

Dutta, D. and Upreti, S.R., 2021. Artificial intelligence-based process control in chemical, biochemical, and biomedical engineering. *The Canadian Journal of Chemical Engineering*, 99, pp.2467–2504.

Esfe, M.H., Eftekhari, S.A., Hekmatifar, M. & Toghraie, D., 2021. A well-trained artificial neural network for predicting the rheological behavior of MWCNT–Al2O3 (30–70%)/oil SAE40 hybrid nanofluid. *Scientific Reports*, *11*(1), pp. 17696–17696.

Ekpo, E.E. and Mujtaba, I.M., 2008. Evaluation of neural networks-based controllers in batch polymerisation of methyl methacrylate. *Neurocomputing*, *71*(7-9), pp.1401-1412.

Engell, S. and Fernholz, G., 2003. Control of a reactive separation process. *Chemical Engineering and Processing: Process Intensification*, *42*(3), pp.201-210.

Erguzel, T.T. and Akbay, E., 2014. Process control using genetic algorithm and ant colony optimization algorithm. *Journal of Intelligent & Fuzzy Systems*, *26*(1), pp.501-516.

Fan, G., A.S., E.-S., Eftekhari, S.A., Hekmatifar, M., Toghraie, D., Mohammed, A.S. & Khan, A., 2022. A well-trained artificial neural network (ANN) using the trainlm algorithm for predicting the rheological behavior of water – Ethylene glycol/WO3 – MWCNTs nanofluid. *International Communications in Heat and Mass Transfer*, 131, p. 105857.

Galluzzo, M., Cappellani, V. and Garofalo, U., 1991. Fuzzy control of pH using NAL. *International Journal of Approximate Reasoning*, *5*(6), pp.505-519.

Galluzzo, M. and Cosenza, B., 2010. Adaptive type-2 fuzzy logic control of a bioreactor. *Chemical Engineering Science*, *65*(14), pp.4208-4221.

Gao, S.Z., Wu, X.F., Luan, L.L., Wang, J.S. and Wang, G.C., 2018. PSO optimal control of model-free adaptive control for PVC polymerization process. *International Journal of Automation and Computing*, *15*(4), pp.482-491.

Ghosh, S., Dissanayake, K., Asokan, S., Sun, T., Rahman, B.M.A. & Grattan, K.T.V, 2022. Lead (Pb2+) ion sensor development using optical fiber gratings and nanocomposite materials. *Sensors and Actuators. B, Chemical*, 364, p. 131818.

Gomm, J.B., Evans, J.T. and Williams, D., 1997. Development and performance of a neural-network predictive controller. *Control Engineering Practice*, *5*(1), pp.49-59.

Gonzaga, J.C.B., Meleiro, L.A.C., Kiang, C. and Maciel Filho, R., 2009. ANN-based soft-sensor for real-time process monitoring and control of an industrial polymerization process. *Computers & Chemical Engineering*, *33*(1), pp.43-49.

Goswami, S., 2020. Deep learning–A state-of-the-art approach to artificial intelligence. In *Deep Learning* (pp. 1-20). De Gruyter.

Grenon, G., Hamrani, A., Madramootoo, C.A., Singh, B. & von Sperber, C., 2022. Neural network model predictions for phosphorus management strategies on tile-drained organic soils. *Hydrology Research*, 53(6), pp. 825–839.

Gundogdu, O., Egrioglu, E., Aladag, C.H. & Yolcu, U., 2016. Multiplicative neuron model artificial neural network based on Gaussian activation function. *Neural Computing & Applications*, 27(4), pp. 927–935.

Han, H.G., Qiao, J.F. and Chen, Q.L., 2012. Model predictive control of dissolved oxygen concentration based on a self-organizing RBF neural network. *Control Engineering Practice*, *20*(4), pp.465-476.

Haykin, S., 2004. *Kalman Filtering and Neural Networks* (Vol. 47). John Wiley & Sons.

Hinton, G.E., 2012. A practical guide to training restricted Boltzmann machines. In *Neural Networks: Tricks of the Trade* (pp. 599-619). Springer, Berlin, Heidelberg.

Hinton, G.E., Osindero, S. and Teh, Y.W., 2006. A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), pp.1527-1554.

Hinton, G.E. and Sejnowski, T.J. eds., 1999. *Unsupervised Learning: Foundations of Neural Computation*. MIT press.

Hojjati, H., Sheikhzadeh, M. and Rohani, S., 2007. Control of supersaturation in a semibatch antisolvent crystallization process using a fuzzy logic controller. *Industrial & Engineering Chemistry Research*, *46*(4), pp.1232-1240.

Honkela, T., Duch, W., Girolami, M. & Kaski, S., 2011. A one-layer dual recurrent neural network with a heaviside step activation function for linear programming with its linear assignment application. in *Artificial Neural Networks and Machine Learning - ICANN 2011*. Germany: Springer Berlin / Heidelberg.

Hotelling, H., 1933. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, *24*(6), p.417.

Hotelling, H., 1992. Relations between two sets of variates. In *Breakthroughs in Statistics* (pp. 162-190). Springer, New York, NY.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixture of local experts. *Neural Computation*, *3*, 79–87.

Imtiaz, U., Assadzadeh, A., Jamuar, S.S. and Sahu, J.N., 2013. Bioreactor temperature profile controller using inverse neural network (INN) for production of ethanol. *Journal of Process Control*, *23*(5), pp.731-742.

Jain, A.K., Mao, J. and Mohiuddin, K.M., 1996. Artificial neural networks: A tutorial. *Computer*, *29*(3), pp.31-44.

Jarmulak, J., Spronck, P. and Kerckhoffs, E.J.H., 1997. Neural networks in process control: Model-based and reinforcement trained controllers. *Computers and Electronics in Agriculture*, *18*(2-3), pp.149-166.

Ji, C., & Ma, S. (1997). Combinations of weak classifiers. *IEEE Transactions on Neural Networks*, *8*, 32–42.

Jiang, P., Hu, Z., Liu, J., Yu, S. and Wu, F., 2016. Fault diagnosis based on chemical sensor data with an active deep neural network. *Sensors*, *16*(10), 1695.

Jordan, M. I., & Jacobs, R. A. (1994). Hierarchical mixture of experts and the EM algorithm. *Neural Computation*, *6*, 181–214.

Kadlec, P., Gabrys, B. and Strandt, S., 2009. Data-driven soft sensors in the process industry. *Computers & Chemical Engineering*, *33*(4), pp.795-814.

Karg, B. & Lucia, S., 2021. Approximate moving horizon estimation and robust nonlinear model predictive control via deep learning. *Computers & Chemical Engineering*, 148, p. 107266.

Kano, M. and Ogawa, M., 2010. The state of the art in chemical process control in Japan: Good practice and questionnaire survey. *Journal of Process Control*, *20*(9), pp.969-982.

Kaunga, D.L., Zhang, J., Ferguson, K. and Steele, C., 2013, July. Reliable modeling of chemical durability of high level waste glass using bootstrap aggregated neural networks. In *2013 Ninth International Conference on Natural Computation (ICNC)* (pp. 178-183). IEEE.

Khaouane, L., Ammi, Y. & Hanini, S, 2017. Modeling the retention of organic compounds by nanofiltration and reverse osmosis membranes using bootstrap aggregated neural networks. *Arabian Journal for Science and Engineering (2011)*, 42(4), pp. 1443–1453.

Li, F., Zhang, J., Shang, C., Huang, D., Oko, E. and Wang, M., 2018. Modelling of a post-combustion CO2 capture process using deep belief network. *Applied Thermal Engineering*, *130*, pp.997-1003.

Li, H., Collins, C.R., Ribelli, T.G., Matyjaszewski, K., Gordon, G.J., Kowalewski, T. & Yaron, D.J., 2018. Tuning the molecular weight distribution from atom transfer radical polymerization using deep reinforcement learning. *Molecular Systems Design & Engineering*, 3(3), pp. 496–508.

Li, S. and Li, Y., 2016. Model predictive control of an intensified continuous reactor using a neural network Wiener model. *Neurocomputing*, *185*, pp.93-104.

Lightbody, G. and Irwin, G.W., 1995. Direct neural model reference adaptive control. *IEE Proceedings-Control Theory and Applications*, *142*(1), pp.31-43.

Lian, C.Y., Huang, Y.F., Ng, J.L., Mirzaei, M., Koo, C.H. & Tan, K.W., 2020. A proposed hybrid rainfall simulation model: bootstrap aggregated classification tree–artificial neural network (BACT-ANN) for the Langat River Basin, Malaysia. *Journal of Water and Climate Change*, 11(4), pp. 1218–1234.

Lim, J.S., Hussain, M.A. and Aroua, M.K., 2010. Control of a hydrolyzer in an oleochemical

plant using neural network based controllers. *Neurocomputing*, *73*(16-18), pp.3242-3255.

Low, C.Y. and Teoh, A.B.J., 2017, September. Stacking-based deep neural network: Deep analytic network on convolutional spectral histogram features. In *2017 IEEE International Conference on Image Processing (ICIP)* (pp. 1592-1596). IEEE.

Lu, H., Ou, Y., Qin, C. & Jin, L., 2021. A fuzzy neural network bagging ensemble forecasting model for 72-h forecast of low-temperature chilling injury. *Natural Hazards (Dordrecht)*, 105(2), pp. 2147–2160.

MacMurray, J.C. and Himmelblau, D.M., 1995. Modeling and control of a packed distillation column using artificial neural networks. *Computers & Chemical Engineering*, *19*(10), pp.1077-1088.

Ma, Y., Zhu, W., Benton, M.G. & Romagnoli, J., 2019. Continuous control of a polymerization system with deep reinforcement learning. *Journal of Process Control*, 75, pp. 40–47.

McCulloch, W.S. and Pitts, W., 1943. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), pp.115-133.

Mnih, A. and Hinton, G.E., 2008. A scalable hierarchical distributed language model. *Advances in Neural Information Processing Systems*, *21*, pp.1081-1088.

Mohammed, K.J.R. and Zhang, J., 2013. Reliable optimisation control of a reactive polymer composite moulding process using ant colony optimisation and bootstrap aggregated neural networks. *Neural Computing and Applications*, *23*(7), pp.1891-1898.

Monticeli, F.M., Neves, R.M. & Ornaghi Júnior, H.L., 2021. Using an artificial neural network (ANN) for prediction of thermal degradation from kinetics parameters of vegetable fibers. *Cellulose (London)*, 28(4), pp. 1961–1971.

Mukherjee, A. and Zhang, J., 2008. A reliable multi-objective control strategy for batch processes based on bootstrap aggregated neural network models. *Journal of Process Control*, *18*(7-8), pp.720-734.

Nagy, Z.K., 2007. Model based control of a yeast fermentation bioreactor using optimally designed artificial neural networks. *Chemical Engineering Journal*, *127*(1-3), pp.95-109.

Nahas, E.P., Henson, M.A. and Seborg, D.E., 1992. Nonlinear internal model control strategy for neural network models. *Computers & Chemical Engineering*, *16*(12), pp.1039-1057.

Normandin, A., Thibault, J. and Grandjean, B.P.A., 1994. Optimizing control of a continuous stirred tank fermenter using a neural network. *Bioprocess Engineering*, *10*(3), pp.109-113.

Osuolale, F.N. and Zhang, J., 2018. Exergetic optimisation of atmospheric and vacuum distillation system based on bootstrap aggregated neural network models. In *Exergy for A Better Environment and Improved Sustainability 1* (pp. 1033-1046). Springer, Cham.

Ou, J. and Rhinehart, R.R., 2003. Grouped neural network model-predictive control. *Control Engineering Practice*, *11*(7), pp.723-732.

Paengjuntuek, W., Thanasinthana, L. and Arpornwichanop, A., 2012. Neural network-based optimal control of a batch crystallizer. *Neurocomputing*, *83*, pp.158-164.

Pearson, K., 1901. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), pp.559-572.

Perrone, M. P., & Cooper, L. N., 1993. When networks disagree: ensemble methods for hybrid neural networks. In R. J. Mammone (Ed.), *Artificial Neural Networks for Speech and Vision*, (pp. 126). London: Chapman and Hall.

Porrazzo, R., Cipollina, A., Galluzzo, M. and Micale, G., 2013. A neural network-based optimizing control system for a seawater-desalination solar-powered membrane distillation unit. *Computers & Chemical Engineering*, *54*, pp.79-96.

Rani, A., Singh, V. & Gupta, J.R.P., 2013. Development of soft sensor for neural network based control of distillation column. *ISA transactions*, 52(3), pp. 438–449.

Raviv, Y., & Intrator, N., 1996. Bootstrapping with noise: an effective regularisation technique. *Connection Science*, *8*, 355–372.

Ren, P., Xiao, Y., Chang, X., Huang, P., Li, Z., Chen, X. & Wang, X., 2021. A Comprehensive Survey of Neural Architecture Search. *ACM Computing Surveys*, 54(4), pp. 1–34.

Rosen, B. E., 1996. Ensemble learning using decorrelated neural networks. *Connection Science*, *8*, 373–384.

Rostami, S., Toghraie, D., Shabani, B., Sina, N. & Barnoon, P., 2021. Measurement of the thermal conductivity of MWCNT-CuO/water hybrid nanofluid using artificial neural networks (ANNs). *Journal of Thermal Analysis and Calorimetry*, 143(2), pp. 1097–1105.

Sahoo, D.R. & Biswal, T., 2022. Synthesis and optimization of properties of Poly (AN-co-AA)/fish bone biocomposite by using artificial neural networks. *Polymer Bulletin (Berlin, Germany)*.

Sarimveis, H. and Bafas, G., 2003. Fuzzy model predictive control of non-linear processes

using genetic algorithms. *Fuzzy Sets and Systems*, *139*(1), pp.59-80.

Schittkowski, K., 1986 NLPQL: A fortran subroutine solving constrained nonlinear programming problems. *Annals of Operations Research*, 5(2), pp. 485-500.

Shah, M.A. and Meckl, P.H., 1995, June. On-line control of a nonlinear system using radial basis function neural networks. In *Proceedings of 1995 American Control Conference-ACC'95* (Vol. 6, pp. 4265-4269). IEEE.

Shang, C., Yang, F., Huang, D. and Lyu, W., 2014. Data-driven soft sensor development based on deep learning technique. *Journal of Process Control*, *24*(3), pp.223-233.

Sharkey, A. J. C. (1996). On combining artificial neural nets. *Connection Science*, *8*, 299–314.

Silva, R.S. and Parpinelli, R.S., 2019, January. A self-adaptive differential evolution with fragment insertion for the protein structure prediction problem. In *International Workshop on Hybrid Metaheuristics* (pp. 136-149). Springer, Cham.

Smolensky, P., 1986. *Information processing in dynamical systems: Foundations of harmony theory*. Colorado Univ at Boulder Dept of Computer Science.

Soares, J.B.P. and Hamielec, A.E., 1996. Kinetics of propylene polymerization with a non-supported heterogeneous Ziegler-Natta catalyst—effect of hydrogen on rate of polymerization, stereoregularity, and molecular weight distribution. *Polymer*, *37*(20), pp.4607-4614.

Sridhar, D. V., Seagrave, R. C., & Bartlett, E. B. (1996). Process modelling using stacked neural networks. *AIChE Journal*, *42*, 2529–2539.

Sucar, L.E. ed., 2011. *Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions: Concepts and Solutions*. IGI Global.

Syu, M.J. and Chang, J.B., 1997. Recurrent backpropagation neural network adaptive control of Penicillin Acylase Fermentation by Arthrobacter viscosus. *Industrial & Engineering Chemistry Research*, *36*(9), pp.3756-3761.

Szafranek, K., 2019. Bagged neural networks for forecasting Polish (low) inflation. *International journal of forecasting*, 35(3), pp. 1042–1059.

Tang, Y., Salakhutdinov, R. and Hinton, G., 2012, June. Robust boltzmann machines for recognition and denoising. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2264-2271). IEEE.

Taniguchi, M., & Tresp, V. (1997). Averaging regularized estimators. *Neural Computation*, *9*,

1163–1178.

Tayyebi, S. and Alishiri, M., 2014. The control of MSF desalination plants based on inverse model control by neural network. *Desalination*, *333*(1), pp.92-100.

Tham, M.T., Montague, G.A., Morris, A.J. and Lant, P.A., 1991. Soft-sensors for process estimation and inferential control. *Journal of Process Control*, *1*(1), pp.3-14.

Tibshirani, R.J. and Efron, B., 1993. An introduction to the bootstrap. *Monographs on Statistics and Applied Probability*, *57*, pp.1-436.

Toledano, D.T., Fernández-Gallego, M.P. & Lozano-Diez, A., 2018. Multi-resolution speech analysis for automatic speech recognition using deep neural networks: Experiments on TIMIT. *PloS one*, 13(10), pp. e0205355–e0205355.

Tu, J.V., 1996. Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of Clinical Epidemiology*, *49*(11), pp.1225-1231.

Varshney, K. and Panigrahi, P.K., 2005. Artificial neural network control of a heat exchanger in a closed flow air circuit. *Applied Soft Computing*, *5*(4), pp.441-465.

Venkatasubramanian, V., 2019. The promise of artificial intelligence in chemical engineering: Is it here, finally. *AIChE J*, *65*(2), pp.466-478.

Wang, Y., Yang, G., Sage, V., Xu, J., Sun, G., He, J. & Sun, Y., 2021. Optimization of dark fermentation for biohydrogen production using a hybrid artificial neural network (ANN) and response surface methodology (RSM) approach. *Environmental Progress & Sustainable energy*, 40(1), e.13485.

Werbos, P., 1974. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. *PhD thesis, Harvard University*.

Wold, S., Sjöström, M. and Eriksson, L., 2001. PLS-regression: a basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems*, *58*(2), pp.109-130.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, *5*, 241–259.

Wu, H. and Zhao, J., 2018. Deep convolutional neural network model based chemical process fault diagnosis. *Computers & Chemical Engineering*, *115*, pp.185-197.

Wu, Z. and Christofides, P.D., 2019. Optimizing process economics and operational safety via economic MPC using barrier functions and recurrent neural network models. *Chemical*

Engineering Research and Design, *152*, pp.455-465.

Xiong, Z. and Zhang, J., 2005. A batch-to-batch iterative optimal control strategy based on recurrent neural network models. *Journal of Process Control*, *15*(1), pp.11-21.

Xu, L., Krzyzak, A., & Suen, C. Y. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, *22*, 418–435.

Yao, Z.J., Bi, J. and Chen, Y.X., 2018. Applying deep learning to individual and community health monitoring data: A survey. *International Journal of Automation and Computing*, *15*(6), pp.643-655.

Ye, Z., Yang, J., Zhong, N., Tu, X., Jia, J. and Wang, J., 2020. Tackling environmental challenges in pollution controls using artificial intelligence: A review. *Science of the Total Environment*, *699*, p.134279.

You, K.W. & Arumugasamy, S.K., 2020. Deep learning techniques for polycaprolactone molecular weight prediction via enzymatic polymerization process. *Journal of the Taiwan Institute of Chemical Engineers*, 116pp. 238–255.

Yu, D.W. and Yu, D.L., 2007. Multi-rate model predictive control of a chemical reactor based on three neural models. *Biochemical Engineering Journal*, *37*(1), pp.86-97.

Yu, R.F., Chen, H.W., Liu, K.Y., Cheng, W.P. and Hsieh, P.H., 2010. Control of the Fenton process for textile wastewater treatment using artificial neural networks. *Journal of Chemical Technology & Biotechnology*, *85*(2), pp.267-278.

Yuan, Z., Yang, Z., Ling, Y., Wu, C. & Li, C., 2021. Spatiotemporal attention mechanism-based deep network for critical parameters prediction in chemical process. *Process safety and environmental protection*, 155pp. 401–414.

Zapf, F. & Wallek, T., 2022. Case-study of a flowsheet simulation using deep-learning process models for multi-objective optimization of petrochemical production plants. *Computers & Chemical Engineering*, 162, p. 107823.

Zhang, J., 1999a. Inferential estimation of polymer quality using bootstrap aggregated neural networks. *Neural Networks*, *12*(6), pp.927-938.

Zhang, J., 1999b. Developing robust non-linear models through bootstrap aggregated neural networks. *Neurocomputing*, *25*(1-3), pp.93-113.

Zhang, J., 2002, May. Sequential training of bootstrap aggregated neural networks for nonlinear

systems modelling. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)* (Vol. 1, pp. 531-536). IEEE.

Zhang, J., 2004. A reliable neural network model based optimal control strategy for a batch polymerization reactor. *Industrial & Engineering Chemistry Research*, *43*(4), pp.1030-1038.

Zhang, J., 2006. Offset-Free Inferential Feedback Control of Distillation Compositions Based on PCR and PLS Models. *Chemical Engineering & Technology*, *29*(5), pp.560-566.

Zhang, J., 2001, June. Inferential feedback control of distillation composition based on PCR and PLS models. In *Proceedings of the 2001 American Control Conference. (Cat. No. 01CH37148)* (Vol. 2, pp. 1196-1201). IEEE.

Zhang, J., Jin, Q. and Xu, Y., 2006. Inferential estimation of polymer melt index using sequentially trained bootstrap aggregated neural networks. *Chemical Engineering & Technology*, *29*(4), pp.442-448.

Zhang, J. and Morris, A.J., 2000. Long range predictive control of nonlinear processes based on recurrent neuro-fuzzy network models. *Neural Computing & Applications*, *9*(1), pp.50-59.

Zhang, J. and Pantelelis, N.G., 2011, April. Modelling and optimisation control of polymer composite moulding processes using bootstrap aggregated neural network models. In *2011 International Conference on Electric Information and Control Engineering* (pp. 2363-2366). IEEE.

Zhang, Z. and Zhao, J., 2017. A deep belief network based fault diagnosis model for complex chemical processes. *Computers & Chemical Engineering*, *107*, pp.395-407.

Zhou, H., Huang, G.B., Lin, Z., Wang, H. and Soh, Y.C., 2014. Stacked extreme learning machines. *IEEE Transactions on Cybernetics*, *45*(9), pp.2013-2025.

Zhu, C.H. and Zhang, J., 2020. Developing soft sensors for polymer melt index in an industrial polymerization process using deep belief networks. *International Journal of Automation and Computing*, *17*(1), pp.44-54.