공학박사 학위논문

Elaborating Digital Twin of Offshore Process Systems Using Machine Learning: Fault Detection, Prediction, and Diagnosis

2023 년 2 월

서울대학교 대학원

조선해양공학과

이 나 영

Elaborating Digital Twin of Offshore Process Systems Using Machine Learning: Fault Detection, Prediction, and Diagnosis

지도 교수 서 유 택

이 논문을 공학박사 학위논문으로 제출함
2022 년 12 월

서울대학교 대학원
조선해양공학과
이 나 영

이나영의 공학박사 학위논문을 인준함
2023 년 2 월

Abstract

Nayoung Lee

Department of Naval Architecture and Ocean Engineering

The Graduate School

Seoul National University

Recently, interest in digitalization is gradually increasing in the field of shipbuilding and marine. In particular, the digital twin enables monitoring by synchronizing the data of the real system with a virtual model in real-time and can be seen as a major platform that integrates various technologies related to digitalization. The digital twin consists of four major components: data, communication, model, and service. Among them, in the data and communication sector, much progress has been made due to the simultaneous development of new technologies such as the Internet of Things, big data, cloud, and 5G, and international standards such as ISO 23247, ISO/IEC30172, and 30173 have been established. It became. On the other hand, development in models and services, especially in the service sector, is relatively slow. The reason for this is that existing simulation models and analysis techniques lack the ability to handle and process sensor data collected and updated in real-time, and machine learning and data-based analysis techniques, which have recently emerged, are not immediately applicable to marine systems. It can be pointed out that additional engineering is required because it is not possible.

In this study, an anomaly detection model based on a machine learning model, a hazard detection model through sensor data prediction, and a process predictive maintenance model was modified and verified for use in marine systems.

In first part, an anomaly detection model based on correlation between sensors that can be applied to process systems is proposed. This model is a modification of the MSCRED model. A two-dimensional correlation matrix calculated over time using Multivariate Time Series Data is used to generate a two-dimensional correlation matrix over time, and Conv-LSTM ED (Convolutional Long A reconstruction matrix is obtained using the -short Term Memory Encoder Decoder model, and a residual matrix is calculated through the difference with the input value. The anomaly score was calculated through the residual matrix calculated in this way in a different way from the existing method. This reflects the characteristics of the offshore process and enables monitoring of the entire process system even when there is no abnormal situation. Finally, the process state can be monitored by enabling classification of fault cases using the clustering technique for time-series abnormal scores. To verify the proposed anomaly detection model, a pilot-scale Mono Ethylene Glycol (MEG) regeneration process was used, and the model was trained using four normal operation data, one starting operation situation and four abnormal data. The model was verified using the abnormal situation data of pilot plant. As a result of learning using several normal driving data, the performance of the model was improved by performing time series synthesis based on specific normal data with high accuracy. As a result of the verification, anomaly detection was performed with an accuracy close to 88%, and as a result of clustering the anomaly score that came out as a result, it was confirmed that clustering was performed for each anomaly situation.

In the second part, Deep learning-based time series prediction framework for the lab-scale hydrate formation experiment was developed. This framework suggests methodologies to use the experimental data as the input of real-time time series prediction model, which can be scaled up for the field use using transfer learning. Preventing gas hydrate formation is critical in offshore gas and oil production systems. Several models can predict hydrate formation, however, these empirical approaches have limitations due to dependency on geometries and fluid characteristics of the systems. The trends of hydrate formation or risk are considered statistical, which means there is no definite model to describe its behavior. Herein, we present a novel framework based on a combination of feature reduction methods and several deep learning models to predict the hydrate formation trend through the multivariate sensor data. Transition and segregation trends during hydrate formation were predicted in real-time using sequential time series data from the last 60 seconds. We employed various deep learning models (Dense, LSTM, GRU, ARLSTM), layers, and dropout to investigate and enhance the prediction ability of each model. Two groups of experimental data (200rpm, 600rpm) were used for training and testing the prediction to examine the universal applicability of the model. Transfer learning in training the model was employed to apply the discrete experimental set to time-series data and enhance the accuracy. The results with higher layer numbers and a dropout rate of 0.2 ~ 0.6 showed the best performance. ARLSTM showed the smallest error among deep learning models and predicted the good trend of kinetic characteristics (transition and segregation part) during the hydrate formation. This approach based on deep learning can be adopted for risk and issue detection of pipelines in the gas production system. The research questions in this chapter are as follows :

In the third part, a novel framework using data-driven prognosis with fully deep learning models is suggested. Prognosis should be accompanied by fault detection, propagation prediction, and root cause diagnosis. This study proposes a framework for performing these tasks using deep learning-based methodologies. First, the feature extraction to latent space using a Convolutional Auto Encoder (CAE) to perform fault detection. Then, near-real-time prediction on the latent vectors using the Recurrent Neural Network (RNN). Online machine learning using transfer learning was applied to increase the accuracy of prediction in unlearned situations, since the fault case trajectory propagation of the chemical process is difficult to predict. Also autoregressive prediction using the trained RNN has been done to predict Remaining Useful Life (RUL). After that, the T^2 index for the prediction result was calculated, and the contribution on T^2 index was calculated by SHAP, a model agnostic eXplainable Artificial Intelligence (XAI) technique. This framework was tested and verified through CSTR and TEP datasets and showed the better prediction performance than other previous prognosis schemes.

Keyword: Digitalization, Digital Transformation(DX), Digital Twin, Machine Learning, Process, Cyber Physical Twin, Anomaly Detection, Time Series Prediction, Prognosis

Student Number: 2018-28756

Table of Contents

List of Table

List of Figures

Chapter 1. Introduction

1.1. Research background

Recently, interest in digitalization is gradually increasing in the field of shipbuilding and marine. In particular, the digital twin enables monitoring by synchronizing the data of the real system with a virtual model in real-time and can be seen as a major platform that integrates various technologies related to digitalization. The digital twin consists of four major components: data, communication, model, and service. Among them, in the data and communication sector, much progress has been made due to the simultaneous development of new technologies such as the Internet of Things, big data, cloud, and 5G, and international standards such as ISO 23247, ISO/IEC30172/3 have been established.

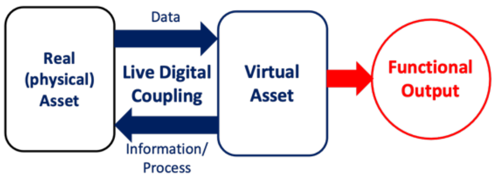


Fig. 1-1. simplified configuration of the digital twin

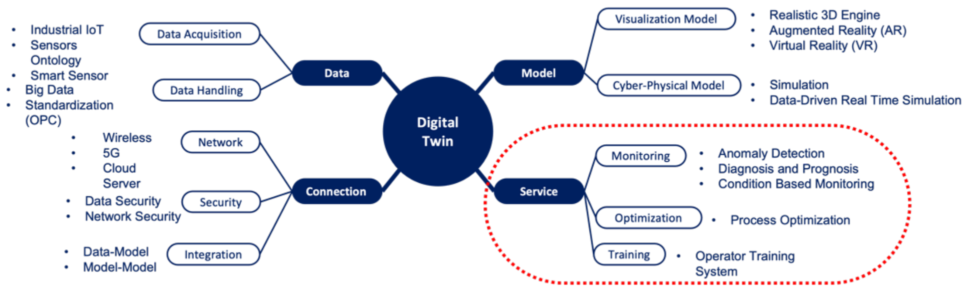


Fig. 1-2. Components of the digital twin

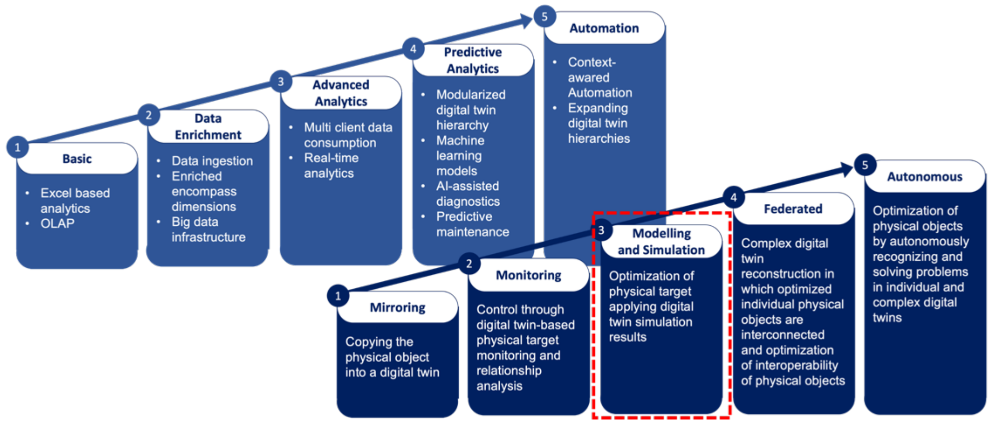

1 Basic
Excel based analytics
OLAP
2 Data Enrichment
Data ingestion
Enriched encompass dimensions
Big data infrastructure
3 Advanced Analytics
Multi client data consumption
Real-time analytics
4 Predictive Analytics
Modularized digital twin hierarchy
Machine learning models
AI-assisted diagnostics
Predictive maintenance
5 Automation
Context-awared Automation
Expanding digital twin hierarchies
1 Mirroring
Copying the physical object into a digital twin
2 Monitoring
Control through digital twin-based physical target monitoring and relationship analysis
3 Modelling and Simulation
Optimization of physical target applying digital twin simulation results
4 Federated
Complex digital twin reconstruction in which optimized individual physical objects are interconnected and optimization of interoperability of physical objects
5 Autonomous
Optimization of physical objects by autonomously recognizing and solving problems in individual and complex digital twins

Fig. 1-3. stages of digital twin and generic data analytics maturity model

On the other hand, development in models and services, especially in the service sector, is relatively slow. The reason for this is that existing simulation models and analysis techniques lack the ability to handle and process sensor data collected and updated in real-time, and machine learning and data-based analysis techniques, which have recently emerged, are not immediately applicable to marine systems. It can be pointed out that additional engineering is required because it is not possible.

In this study, an anomaly detection model based on a machine learning model, a hazard detection model through sensor data prediction, and a process predictive maintenance model was modified and verified for use in marine systems.

In the Chapter 2, a plant-wide anomaly detection algorithm using Multi-Scale Convolutional Recurrent Encoder-Decoder (MSCRED) has been proposed. Automating the process plant anomaly detection using artificial intelligence is currently widely studied topic and is attracting attention of many researchers. But still, it is a challenging task to detect changes for the entire process, not for each single sensor data, and to perform anomaly detection and classify them by fault type. This is because most of the anomaly detection algorithms perform without information on the degree of association between sensors. Accordingly, this study addresses the existing problems by expanding the Multi Scale Convolutional Recurrent Encoder Decoder (MSCRED) model, which allows the correlation between sensors to be matched in the form of 2D matrix to perform analysis detection based on changes in sensor correlation on a specific time window. In addition, the vulnerability of existing MSCRED models is compensated by replacing the anomaly score calculation method and threshold-based anomaly detection method with clustering technique using time series distance with normal data. The framework presented in this study was verified using the operational data of the actually operated pilot scale Mono Ethylene Glycol (MEG) regeneration plant, and additional studies were conducted on which data to use as training data to improve the accuracy of the model.

The research questions in this chapter are as follows:

- How to detect system wide anomaly using process sensor data and machine learning model?
- Using MSCRED model on real world sensor data works well?
- How does the training data affect on the performance of the detection model?
- How to set the threshold for the given system?
- How to classify the anomaly score result into a fault modes?

In the Chapter 3, Deep learning-based time series prediction framework for the lab-scale hydrate formation experiment was developed. This framework suggests methodologies to use the experimental data as the input of real-time time series prediction model, which can be scaled up for the field use using transfer learning. Preventing gas hydrate formation is critical in offshore gas and oil production systems. Several models can predict hydrate formation, however, these empirical approaches have limitations due to dependency on geometries and fluid characteristics of the systems. The trends of hydrate formation or risk are considered statistical, which means there is no definite model to describe its behavior. Herein, we present a novel framework based on a combination of feature reduction methods and several deep learning models to predict the hydrate formation trend through the multivariate sensor data. Transition and segregation trends during hydrate formation were predicted in real-time using sequential time series data from the last 60 seconds. We employed various deep learning models (Dense, LSTM, GRU, ARLSTM), layers, and dropout to investigate and enhance the prediction ability of each model. Two groups of experimental data (200rpm, 600rpm) were used for training and testing the prediction to examine the universal applicability of the model. Transfer learning in training the model was employed to apply the discrete experimental set into time-series data and enhance the accuracy. The results with higher layer numbers and a dropout rate of 0.2 ~ 0.6 showed the best performance. ARLSTM showed the smallest error among deep learning models and predicted the good trend of kinetic characteristics (transition and segregation part) during the hydrate formation. This approach based on deep learning can be adopted for risk and issue detection of pipelines in the gas production system.

The research questions in this chapter are as follows:

- How to select the feature from the many sensor data?

- How to predict the event which are not like traditional time series prediction; non-stationary, non-periodical with statistical and single time occurrence event?
- Training machine learning model with the experimental data which has set of multiple iterative time series data?
- How to make the near-real time prediction model to use for field use?

In the Chapter 4, a novel framework using data-driven prognosis with fully deep learning models is suggested. Prognosis should be accompanied by fault detection, propagation prediction, and root cause diagnosis. This study proposes a framework for performing these tasks using deep learning based methodologies. First, the feature extraction to latent space using an Convolutional AutoEncoder (CAE) to perform fault detection. Then, near-real-time prediction on the latent vectors using the Recurrent Neural Network (RNN). Online machine learning using transfer learning was applied to increase the accuracy of prediction in unlearned situations, since the fault case trajectory propagation of the chemical process is difficult to predict. Also autoregressive prediction using the trained RNN has been done to predict Remaining Useful Life (RUL). After that, the T^2 index for the prediction result was calculated, and the contribution on T^2 index was calculated by SHAP, a model agnostic eXplainable Artificial Intelligence (XAI) technique. This framework was tested and verified through CSTR and TEP datasets and showed the better prediction performance than other previous prognosis schemes.

The research questions in this chapter are as follows:

- What is the best way to build the prognosis framework using different deep learning models?
- How to boost the accuracy of the prediction on fault cases, when the fault case data is hard to get before it occurs?
- How to predict the Remaining Useful Lifetime (RUL) using the deep learning-based prediction model?
- Does diagnosis with an explainable artificial neural network on the prediction model available?

Chapter 2. System-wide Anomaly Detection*

2.1. Introduction

Automatic anomaly detection and diagnosis **[1, 2]** are now largely getting attention, with the rising demand for the digitalization of process systems. Chemical process plants usually consist of many components like vessels, equipment, and pipelines. Sensors and controllers are attached to those components and gather data for the constant time interval. Indicators and controllers are the main categories of values we can get from the chemical plant. Indicators monitor the status like pressure, temperature, and liquid level. Controllers transmit values including Set Point (SP), Present Value (PV), and Valve Opening (OP). These measured values indicate the status of each component and the system and are used for monitoring. Sensor data in chemical processes has been measured for a long time since they are values closely related to the process yield, efficiency, and safety **[3]**. However, interpreting these values need domain knowledge, which makes it hard to automate. But with the recent rise of digital storage technology and big data-related technologies, the volume of collected data has been increasing and the algorithms for data utilization are becoming more sophisticated. Methodologies have been developed that enable machines to automatically perform necessary algorithms using data collected using machine learning and deep learning technologies. In addition, advances in sensor hardware and sensor data processing

* This chapter is partially adapted from Plant-wide Anomaly Detection and Single-Value Monitoring using Conv-LSTM in Pilot-Scale MEG Regeneration Plant with authors N. Lee, H. Kim, J. Jang, and Y. Seo. (Preparing Submission)

technology have opened the way for various uses of the collected sensor data. With this, technologies that can automatically perform anomaly detection and failure cause analysis using sensor data using data-driven methods are starting to attract attention **[4, 5]**. Here we will suggest the monitoring and anomaly detection method using a deep learning model trained with historical sensor data. Though many AI methods do not use field data to ensure their accuracy **[6]**, this study examines the proposed anomaly detection algorithm with real-world data which is obtained from the pilot plant.

2.2. Related Work

2.2.1 Fault monitoring and identification of the chemical plant

Anomaly detection of a chemical process is a widely researched topic. Classical division for a process fault detection is like the following **[7]** : (1) data methods and signal methods (limit checking and trend checking, dimension reduction (PCA) [2], spectrum analysis and parametric models, pattern recognition (neural nets)) (2) Process model-based methods: parity equations, state observations, parameter estimation, nonlinear models (neural nets) (3) knowledge-based models : Expert systems, Fuzzy Logic **[8]**.

2.2.2 Process Anomaly Detection Using AI

Most successful deep learning applications fall into the category of supervised learning. Assuming that the amount of data is sufficient, supervised learning can now show guaranteed performance. Anomaly detection, however, cannot use supervised learning because anomalies are extraordinary in the whole data and not always present in the historical data **[4, 9]**. Instead, model training is performed using unsupervised learning such as clustering to find the normal state and detect anomalies by the deviance from the defined normal state.

Basically, Anomaly detection is a same task as defining the normal condition and finding its representation within the format of the given data. And AI model is trained entirely by the normal data given from operation. However, in chemical process plant (or in other applications), it is hard to define the normal status since its condition is constantly changing by demand and its range of normal status is large.

Within the domain of unsupervised models, the encoder-decoder model (also called an autoencoder model) is recently gaining its popularity due to its efficiency with a rather easy model architecture among the data-driven methods **[10]**. Also new ML methods such as Variational Auto Encoder – Generative Adversarial Network (VAE-GAN) was explored on fault detection on chemical process **[11]**.

Table 2-1. Literature related to the major methodology in the ML area

Literature	Method	Taxonomy
Takehisa Yairi et al. (2001)	K-Means	Unsupervised, Classic ML, distance
S. Ramaswamy et al., (2000)	KNN	Unsupervised, Classic ML, distance
F. T. Liu, et al., (2008)	Isolation Forest	Unsupervised, Outlier detection, Trees
P. Malhotra, et al., (2015)	LSTM-AD	Semi-supervised, Deep Learning, forecasting

M. Sakurada and T. Yairi. (2014)	Autoencoder	Semi-supervised, Deep Learning, reconstruction
P. Malhotra, et al., (2016)	Enc-Dec AD	Semi-supervised, Deep Learning, reconstruction
D. Park et al., (2018)	LSTM-VAE	Semi-supervised, Deep Learning, reconstruction
Y. Su et al., (2019)	OmniAnomaly	Semi-supervised, Deep Learning, reconstruction
C Zhang et al.,(2019)	MSCRED	Semi-supervised, Deep Learning, reconstruction

Table 2-2. Literature related to the system correlation based anomaly detection on time series sensor data using Deep Learning

Literature	Method	Outperforms	Domain
D. Hallac et al., (2017)	TICC	GMM, DTW, K-means	Driving sensor data
D. Song et al.,(2018)	Deep r-RSJBE (LSTM based)	LSTM, CNN	physical activity monitoring dataset, Sussex-Huawei Locomotion (SHL) dataset
K. Jiang et al.(2019)	CNN+BiLSTM	AlexNet, CNN, BiLSTM	Network Traffic Intrusion Detection
I.S.Thaseen el al. (2020)	CFS+ANN	CNN+BiLSTM, Naïve BayesSVM, Devision Tree, Random Tree, CNN, AlexNet	Network Intrusion Detection

2.2.3 Plant-wide Anomaly Detection

There are several advantages to developing a deep learning model that performs anomaly detection for the entire process with a single algorithm. There are some studies on plant-wide anomaly detection without deep-learning, as principal component analysis **[12]** or local outlier factor **[3]**. The reason for using a deep learning model that utilizes data from the entire process is that, first, creating a combined model reduces the number of models to be trained much more than creating an individual anomaly detection algorithm model for each signal. Second, it is possible to detect anomalies in consideration of the connectivity between the sensors. Third, when an abnormality is detected, it can be explained in which part and for what reason in one deep learning model. However, this plantwide anomaly detection is not quite well researched in this field. MSCRED **[13]** considers every sensor data and its correlation by introducing a signature matrix. This signature matrix is a correlation matrix for a given time-series window. For a given time window, the signature matrix shows the system status by showing the average value of sensor value and the correlation between different pairs of the sensor time-series data. multiscale anomaly detection is enabled by a multiple encoder which scales system into many stages.

MSCRED has been used steadily in the field of anomaly detection in systems utilizing multivariate sensor data **[14]** and has been used as a basis for new algorithms **[15]** and demonstrated their usefulness. It is also receiving a lot of attention in the industrial field, which is thought to be due to MSCRED's efficient combination of association analysis and time series prediction to be considered in the industry. It is noteworthy that the cases used for predictive maintenance in the field of manufacturing were used.

2.2.4 Timeseries Anomaly Detection

Many of the process anomaly detection proposed so far have mainly been algorithms for performing fault diagnosis based on past data **[16]**. For this reason, although various anomaly detection models and algorithms have appeared, they have not been actively introduced in the actual process. What is needed in the actual process is the prognosis, which predicts process abnormalities that will occur in the future and prevents them in advance, because this process was impossible using existing algorithms. As a method to compensate for this, a method of combining a recently introduced time series prediction model with an anomaly detection model may be introduced. Predictive maintenance is possible if the propagation of future sensor data is predicted using the existing process sensor data and the existing anomaly detection algorithm is introduced for the predicted value. A time series is not simply a set of values according to a time index, but a case in which the value of the previous time step affects the next time step. Time series-based forecasting has been mainly done in the fields of stock and weather forecasting and control systems.

Within the fault identification model, it is able to put timeseries model inside the anomaly detection architecture. RNN cells can be used after the encoder, and pass the output value to the decoder. Also, classical autoregressive models can also be used as a bridge between the encoder and decoder, but not implemented in this paper.

It also enables dynamic fault detection in chemical process domain. In MSCRED framework, employing conv-LSTM model allows the time-series prediction.

2.3. Experiment: Pilot-plant data for MEG regeneration process

The proposed model for abnormally detection is applied in the experimental plant (pilot scale) from MEG regeneration process, which is to recover a high concentration of MEG aqueous solution (Lean MEG) from low concentration of MEG aqueous solution (Rich MEG) and remove the salts inside the aqueous solution **[17]**.

2.3.1. Experimental apparatus and procedure

Detailed description regarding the materials, equipment & sensor information employed in the present study, are included in our previous work **[25]**. The sensor data measured were recorded every 60 seconds through data acquisition system.

The pilot plant of the MEG regeneration process consists of 4 main units: storage tank, pretreatment, distillation column, reclamation as shown in **Fig. 2-1**. The rich MEG solution (feed stream), of which the concentration of MEG is around 50 wt%, was made with mixing the MEG and water at 40 ºC in two feed tanks (T101A, T101B) in advance. It is noted that the operating conditions in each equipment described are based on the normal operation. The feed entered the pre-treatment vessel (V101) with a constant mass flow rate (200±10kg/hr). The purpose of vessel was to eliminate a small amount of divalent salt from the solution under the conditions of 80 ºC and 150 kPa. However, we didn't include the observation of divalent salts in this paper. After the pre-treatment unit, the water from the solution stream was evaporated in the distillation column (C101) to increase the MEG concentration in the solution to 80 – 90wt%, called lean MEG.

The operating temperature of reboiler in distillation unit was controlled with the range of 125 ºC to 155 ºC. The bottom product of the distillation column, lean MEG, was split into two streams and one of them was entered into the reclamation unit. The purpose of the reclamation vessel (V111) was to flash the lean MEG solution to vapor phase and remove the residual monovalent salt (NaCl in this work) as the liquid phase under the vacuum condition. The vaporized lean MEG from the reclamation vessel were cooled down to liquid phase and mixed with the stream from bottom of distillation column. The combined stream returned to the feed tanks (T101 A/B). The NaCl liquid slurry was removed through filter units (F111 A/B).

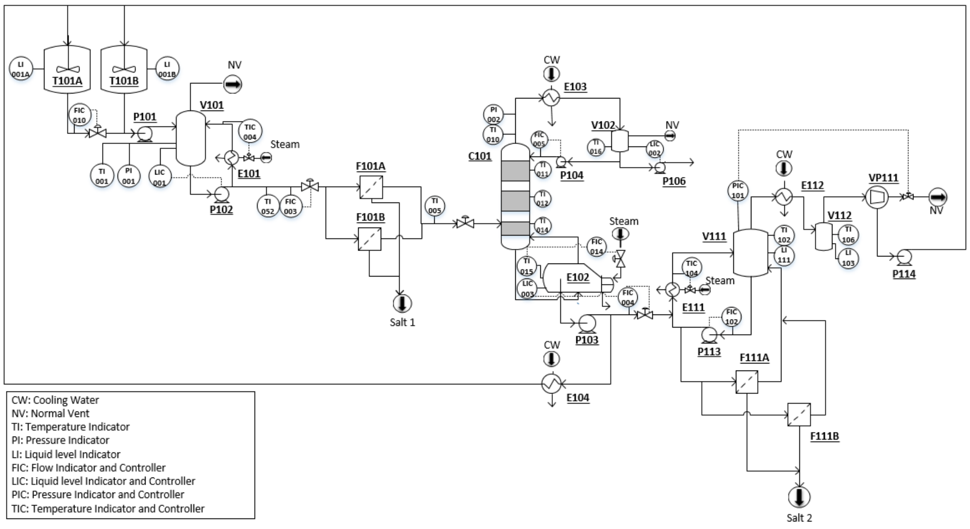


Fig. 2-1. Process flow diagram of pilot-scale MEG regeneration system

Table 2-3. Description of operating conditions of units for the pilot-scale MEG regeneration plant.

Unit	Name	Object	Size	Operatin g condition	Design condition (P/T)
Feed	T101A/	Storage	1.20 m^3×2	F : 200	-

tank	B	tank		kg/hr T : 313 K P : 100 kPa	
Pretreatment	V101	Flash tank	0.95 m(ID) ×1.40m(H)	F : 200 kg/hr T : 353K P : 150 kPa	-
	E101	Recycle heater	1.02m(ID)×0.01m(L)	-	1. Shell side : 1080kPa/483K 2. tube side : 900kPa/403K
	F101A/B	Filter	-	-	750kPa/383K
Distillation column	E103	Condenser	2.12m(ID) ×0.012m(L)	-	1. Shell side : 600 kPa/348K 2. tube side : 280kPa/413K
	V102	Reflux drum	0.043m^3	T : 372 – 374 K P : 100 - 105 kPa	280 kPa/373K
	C101	Tower	0.2545m(ID) ×8.00m(H)	-	280 kPa/473K
	E102	Reboiler	0.168 m^3	T : 398 – 418 K P : 105 kPa	1. Shell side : 280kPa/473K 2. tube side : 1080kPa/483K
	E104	Column Bottom cooler	1.61m(ID) ×0.015m(L)	-	1. Shell side : 600kPa/348K 2. tube side : 750kPa/473K
Reclamation	V111	Flash tank	0.55m(ID) ×1.05m(H)	F : 20 kg/hr T : 400 - 402K P : 11 kPa	450 kPa/413K
	E111	Reclamation inlet heater	0.20m(ID) ×0.018m(L)	-	1. Shell side : 1080kPa/483K 2. tube side : 600kPa/453K
	E112	OVHD condenser	1.61m(ID)×0.015m(L)	-	1. Shell side : 600kPa/348K 2. tube side : 450kPa/413K
	V112	OVHD receiver	2.12m(ID)×5.50m(H)	-	450 kPa/413K

	VP111	Vacuum pump package	-	-	-
	F111A/B	Filter	-	-	600kPa/433K

Table 2-4. Description of sensors for the pilot-scale MEG regeneration plant

Variables	Sensor name	Description
Flowrate	FIC 4010_PV	Flowrate from feed tank (T101A/B) to pretreatment (V101)
	FIC_4003_PV	Flowrate from pretreatment(V101) to filter (F101A/F101B)
	FIC_4005_PV	Reflux flowrate of condenser (V102) to distillation column (C101)
	FIC_4004_PV	Flowrate from bottom of column (E102) to reclamation vessel(V111)
	FIC_4014_PV	Steam flowrate to reboiler
	FIC_4102_PV	Reflux flowrate of reclamation vessel(V111)
Pressure	PI_4001_PV	Pressure of pretreatment vessel (V101)
	PI_4002_PV	Pressure of distillation column (C101)
	PIC_4101_PV	Pressure of reclamation vessel (V111)
Temperature	TI_4001_PV	Temperature of pretreatment vessel (V101)
	TIC_4004_PV	Flow temperature after pretreatment heat exchanger(E101)
	TI_4005_PV	Flow temperature before distillation column (C101)
	TI_4010_PV	Column internal temperature sensor 1 - Overhead temperature
	TIC_4011_PV	Column internal temperature sensor 2 - 1st packing column
	TI_4012_PV	Column internal temperature sensor 3 - 2nd packing column
	TIC_4014_PV	Column internal temperature sensor 3 - 3rd packing column
	TI_4015_PV	Column internal temperature sensor 4 - Reboiler temperature
	TI_4016_PV	Column internal temperature - condenser receiver tank(V102)
	TI_4102_PV	Temperature of reclamation vessel (V111) - top
	TI_4106_PV	Temperature of overhead receiver to reclamation vessel (V112)

	TIC_4052_PV	Flow temperature after pretreatment vessel to filter (F401A/B)
Level (liquid level percent)	LI_4001A_PV	Liquid level percent of feed tank (T101A)
	LI_4001B_PV	Liquid level percent of feed tank (T101B)
	LIC_4002_PV	Liquid level percent of condenser receiver tank(V102)
	LIC_4003_PV	Liquid level percent of reboiler (E102)
	LIC_4111_PV	Liquid level percent of pretreatment (V101)
	LIC_4103_PV	Liquid level percent of reclamation vessel (V111)
	LIC_4104_PV	Liquid level percent of overhead receiver to reclamation vessel (V112)

2.3.2. Operation case

In this work, the simulation contains normal and abnormal operations. The normal operations consist of 4 cases, which contain the operating conditions of equipment and streams within the normal operating tolerance range. Each normal operation case is not a continuous process. A case name with 'normal-total' is a process in which each normal operation case is sequentially combined over time, and there is a discontinuity in the connection range of each case. The normal-total data and each of four cases were used to train the models.

For verifying the algorithm used, four abnormal operation cases have tested, which means the operating condition goes outrange of the normal conditions. **Table 2-5.** summarized what is the event and how to take action for solving the event for each issue case. **Issue-1** had the increase of liquid level percent of LIC003 in reboiler of column unit (E102). The liquid level percent of LIC003 was measured exceeded the operating range for normal operation. To decrease of the liquid level percent of LIC003, the flow rate of lean MEG at the bottom of the column should be increased. The value of FIC004 was manipulated to increase. **Issue-2** described abruptly shutdown of feed flowrate controlled by FIC010 at 1.6

hr of operation. Accordingly, the liquid level percent (LIC001) of vessel V101 was reduced. Afterward, the FIC010 turned on with 250 kg/hr, which is higher than the normal operating condition of 200 kg/hr at 2.6 hr. Hence, the liquid level precent of V101 (value of LIC001) started to increase. **Issue-3** case indicates a situation in which the problem continues to become more serious after the problem has occurred. The initial problem happened from the decrease of steam flowrate for reboiler (FIC014) at 1.5 hr**.** Liquid level percent of condenser (V102) at column (LIC002) results in decreasing due to lowering the operating temperature of distillation column (C101). The operator rapidly turned down the reflux flowrate from the condenser (FIC005) to zero for 5 minutes at 1.66 hr of operation time. Afterward, the value of FIC005 returned to original value (20 kg/hr). However, liquid level percent at LIC002 kept increasing to 77.2%, which is still abnormal operating condition. **Issue-4** showed the abnormal operation with emergency shutdown**.** The issue occurred from the failure of heat exchanger, E111. Due to the failure of E111, the operating temperature of reclamation vessel, V111, was dropped and measured in TI102. Lower operating temperature led to decrease the amount of vapor to be flashed and increased the liquid level percent of V111, measured by LIC111. However, meaningful action has not been taken at LIC111 and the measure value of LIC111 (liquid level percent of V111) increased continuously. The case was decided to emergency shutdown at around 12 hrs. The experimental profiles is described in section 5 for comparison with the model results.

Table 2-5. Summary of each abnormal operation case (issue) with event and action

Case name	Event	Action for solving the event
Issue-1	Liquid level percent of LIC003 was increased.	Increase the flowrate of FIC004 (at 5.8 hr)
Issue-2	Feed flowrate of FIC010 was abruptly shutdown.	FIC010 restarted with target flowrate.
Issue-3	Steam flowrate for reboiler (FIC014) was decreased at 1.5 hr. Liquid level percent of condenser at column (LIC002) was decreased.	FIC005 manipulated to stop the reflux flowrate to C101 from V102 for 5 minutes at 1.66 hr.
Issue-4	Heat exchanger, E111, failed. Temperature of V111(TI102) was dropped and liquid level percent of V111 (LIC111) was increased.	N/A. The operation was emergency terminated.

2.4. Model Framework

In this study, we applied modified MSCRED framework as shown in Fig. 2. This framework suggests the combination of signature matrix and conv-LSTM ED model with Anoamly score calculation. conv-LSTM ED model has convolutional encoder followed by conv-LSTM with attention, followed by convolutional decoder. the model has been optimized for Mean Squared Error (MSE) with Adam stochastic gradient descent.

Chemical process is a complex system with multivariate time series data of the networked sensors **[21]**, where its measured data are physically interconnected and influence each other throughout the system. Also the measured values are closely interconnected with the static and dynamic status of the previous data. here, sensor data from the chemical plant is dependent with physical dependency by its adjacency of system and temporal continuity of each data points. This spatio-temporal dependency affects a lot in system monitoring. However, there is not many existing machine learning models that physically implements both spatial and temporal relationships. For spatially related (e.g. image) data, Convolutional Neural Network (CNN) is used for processing. For temporary related (e.g. audio signal) data, Recurrent Neural Network (RNN) is used for processing. To reflect both spatial and temporal data, conv-LSTM model has been used **[26]**.

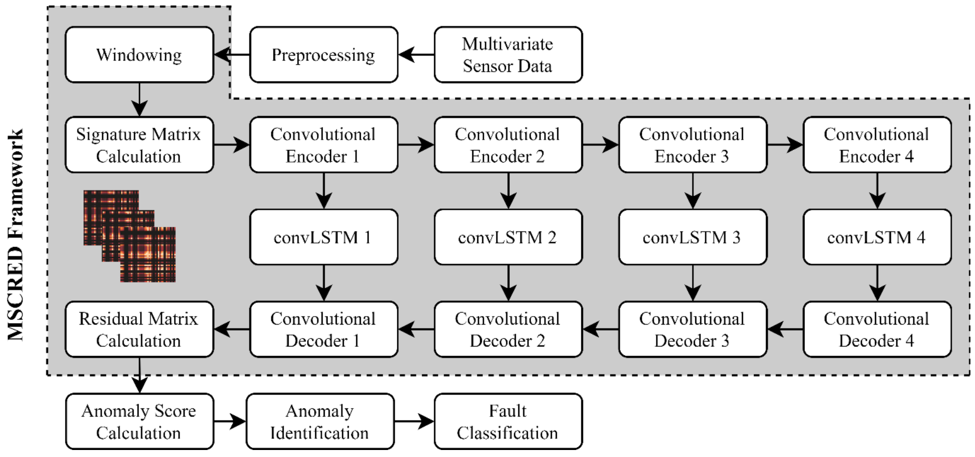


Fig. 2-2. Configuration of the modified MSCRED framework with 4 encoding-decoding layers

2.4.1. Signature Matrix

The signature matrix was introduced to represent the correlation of two given time-series signals. With two input sensor data(ith and jth) under the same window of length ω at time t, $x_i^{\omega} = (x_i^{t-\omega}, x_i^{t-\omega-1}, \dots, x_i^{t})$ and

$x_j^{\omega} = (x_j^{t-\omega}, x_j^{t-\omega-1}, \dots, x_j^{t})$, The (i,j) component of the signature matrix at time t M^t is m_{ij}^t , and this is the sum of the component-wise multiplication divided by window length ω.

$$m_{ij}^{t} = \frac{\sum_{\delta=0}^{\omega} x_i^{t-\delta} x_j^{t-\delta}}{\omega} \tag{2-1}$$

Signature matrix using above correlation value showed better performance than using correlation coefficients, such as Kendall or Spearman coefficients. Calculating this value for the whole combination of sensors makes the signature matrix M^t which is a symmetric matrix However, unlike the signature matrix, the reconstructed matrix, the output of the Conv-LSTM ED model, is not always a symmetric due to the non-linearity of conv-LSTM model and nature of the encoder-decoder architecture.

2.4.2. MSCRED Model

2.4.2.1. Conv-LSTM

Conv-LSTM **[26]** model is an extended version of an LSTM model, which allows larger dimension input for the conventional LSTM model. This model was initially used in video processing **[26]** and was used to find out how the motion of pixels changes over time in the data in the form of a moving image in which several images are overlapped over time. Although this study does not deal with image or image processing, the conv-LSTM model was chosen since the correlation calculation matrix between sensor data used as an input value has the same data format as the image. This allows conv-LSTM to handle the spatiotemporal data

In the Conv-LSTM model, each cell undergoes the following computational steps.

$$
\begin{aligned}
i_t &= \sigma(W_{xi} * \mathcal{X}_t + W_{hi} * \mathcal{H}_{t-1} + W_{ci} \circ \mathcal{C}_{t-1} + b_i) \\
f_t &= \sigma\left(W_{xf} * \mathcal{X}_t + W_{hf} * \mathcal{H}_{t-1} + W_{cf} \circ \mathcal{C}_{t-1} + b_f\right) \\
\mathcal{C}_t &= f_t \circ \mathcal{C}_{t-1} + i_t \circ \tanh(W_{xc} * \mathcal{X}_t + W_{hc} * \mathcal{H}_{t-1} + b_c) \qquad (2\text{-}2) \\
o_t &= \sigma(W_{xo} * \mathcal{X}_t + W_{ho} * \mathcal{H}_{t-1} + W_{co} \circ \mathcal{C}_t + b_o) \\
\mathcal{H}_t &= o_t \circ \tanh(\mathcal{C}_t)
\end{aligned}
$$

$\mathcal{X}$ is the feature maps, W is a convolutional kernel, $\mathcal{H}$ is a hidden state, $\mathcal{C}$ is cell output, b is the bias parameter of a given layer, f, i and o is gate tensor, σ is the sigmoid function, $\circ$ is Hadamard product, and $*$ convolutional operator. In the MSCRED framework, temporal attention was adopted in the current state estimate in the conv-LSTM model to selectively give weight to previous timesteps. To do this, the output of the feature map is given as follows:

$$\mathcal{H}_t = \Sigma_{i\in(t-h,t)}\alpha^i\mathcal{H}^i, \alpha^i = \frac{\exp\left\{\frac{Vec(\mathcal{H}^t)^T Vec(\mathcal{H}^i)}{\mathcal{X}}\right\}}{\Sigma_{i\in(t-h,t)}\exp\left\{\frac{Vec(\mathcal{H}^t)^T Vec(\mathcal{H}^i)}{\mathcal{X}}\right\}} \quad (2\text{-}3)$$

Here, $\mathcal{H}$ is the last hidden state, h is step length, α is importance weights of previous steps through a softmax function, Vec means vector, and $\mathcal{X}$ is arbitrary rescale factor (here, $\mathcal{X} = 5$). This is slightly different from the original attention mechanism **[27]** as this does not employ transformers and context parameters.

2.4.2.2. Encoder-Decoder Model

The encoder-decoder structure is also called an autoencoder, and this methodology is widely used for detecting abnormalities not only in the field of chemical engineering **[28]** but in general time-series data **[9, 16, 18]**. and has proven its usefulness. In addition, research on process anomaly detection using many advanced autoencoder models initiated in the field of machine learning is also actively underway. As a method using an autoencoder, a deep learning model is created in which the dimension is reduced first and then increased again. In addition, learning is conducted to modify the internal parameters so that the normal data can be used to produce output data identical to the input data. The produced output is called reconstructed data. When the test data is similar to the data used for the training data after completion of the training, the reconstructed data is almost the same as the input data. However, when test data show different characteristics from the training data, the reconstructed data is different from the input data. It results that the larger the difference between input and reconstructed data when the more different the abnormal value from the normal data value used as the training data.

2.4.2.3. Encoder

Encoders are responsible for dimension reduction and can take several different forms. The general model employs the form of a basic dense neural network and reduces the number of cells in the anomaly detection. In this paper, the model uses the convolution operation functions as an encoder **[29]**, which is typically used in image processing **[30]**. At the same time, the conv-LSTM model is employed for each encoding step so that the results from multiple encoding steps could be used. In other words, the four-step encoder model using fully convolutional layer **[31]** is combined.

$$\mathcal{X}^t = f(W * \mathcal{X}^t + b) \qquad (2\text{-}4)$$

Here the function f denotes activation function, $\mathcal{X}^t$ is output feature map of the previous layer at time t, W is convolutional kernels of size $k \times k \times d$, b is a bias parameter, $*$ is convolutional operation. Selection of the number of the convolutional layer is arbitrary, however, this can be affecting the size of the result since encoding allows to capture of the interconnectivity between sensors.

2.4.2.4. Decoder

The convolutional decoder has a symmetric architecture with the convolution encoder, with its layer and the size of the kernels. As shown in Eq. (5), The last decoder can decipher the values that passed through the conv-LSTM in the last encoder, and compute the other layers in reverse order. If not in the last layer, decoders must go through the process of adding the value of the previous output to the current output, to sum up the branches that entered the encoder and conv-LSTM.

$$\mathcal{X}^{t,l-1} = \begin{cases} f(W^{t,l} \circledast \mathcal{H}^{t,l} + b^{t,l}) & l = L \\ f(W^{t,l} \circledast [\mathcal{H}^{t,l} \oplus \mathcal{X}^{t,l}] + b^{t,l}) & l \neq L \end{cases} \tag{2-5}$$

Here the function f denotes activation function, $\mathcal{X}^t$ is output of decoder at time t, W is filter kernels which is same as convolutional kernel of each layer of size $k \times k \times d$, b is a bias parameter, l is decoder number, L is the largest decoder number, $\circledast$ is deconvolutional operation, and $\oplus$ is concatenation operation.

2.4.2.5. Loss Function

Since this is an encoder-decoder model as whole, the output of the model is the reconstructed matrix. Loss function is the sum of element wise difference (L2 norm) between reconstructed matrix and signature matrix used as original input for whole time series.

$$\mathcal{L} = \sum_{t} \sum_{c=1}^{s} \sum_{i=1}^{n} \sum_{j=1}^{n} \left\| \mathcal{X}_{i,j,c}^{t,0} - \hat{\mathcal{X}}_{i,j,c}^{t,0} \right\| \tag{2-6}$$

Where s is number of windows, n is number of sensors, t is the temporal length of the input data, i is the row index of the matrix, j is column index of the matrix, s is number the of windows and c is the index of windows. Adam optimizer was employed to minimize the loss.

2.4.3. Anomaly Score

Anomaly score is a sum of values in elements of Loss matrix in the original paper **[21]**. Here we introduce the state monitoring variable L(t), which processed under deep learning model to show single time-series monitoring.

$$\mathcal{L}(t) = \sum_{c=1}^{s} \sum_{i=1}^{n} \sum_{j=1}^{n} \left\| \mathcal{X}_{i,j,c}^{t,0} - \hat{\mathcal{X}}_{i,j,c}^{t,0} \right\| \tag{2-7}$$

Where s is number of windows, n is number of sensors, t is the temporal length of the input data.

2.4.3.1. Anomaly Score Calculation

The anomaly score is calculated over time. The score is greater when the value change is farther away compared to the normal state. However, the large deviation of the values at a specific point does not mean that the score may increase and be vulnerable even if the overall time series pattern changes. This is because the conv-LSTM model can detect patterns on a continuous time series and may vary depending on the model configuration. That is, the anomaly score can produce different results depending on which anomaly detection model is used.

In original MSCRED framework, anomaly score is calculated as a total sum of absolute values from reconstructed matrix, which is only larger than a predefined threshold value for each time window. However, in this work, anomaly score is a simple sum of absolute values from the whole reconstructed matrix for each time window regardless of the threshold value. The calculated score can see how the value propagates before the abnormal situation occurs. By this method, it is possible to monitor the status change during the non-anomaly state. Also, this allows the setting of a system-wide threshold rather than setting the threshold for each sensor, which is harder to optimize the threshold value. Threshold value is set by examining the output value of the normal case data. Using the threshold value before summing up the elements of the matrix makes hard to see the state of the process before the faults occur.

In addition, empirical analysis of many different abnormal conditions is required to obtain the threshold, but in the case of chemical processes, there are not many analyses on the anomaly cases which leads to the failure of the process. Unlike mechanical systems, where the development of the existing anomaly detection methodology is made, the chemical process may already be a serious problem even if minor sensor changes do not lead to major changes. Simply because the range of change is large does not mean that the problem is big. To prevent this situation, the threshold setting was excluded to prevent the prediction of a specific anomaly state if the threshold is set without analysis. and instead, time series classification was introduced to compensate for this, so that the normal state and anomaly state could be classified through clustering between the derived anomaly scores.

2.4.4. Fault Classification by Anomaly Score Clustering

Time series clustering is a general clustering technique performed according to time series **[32]**. This process was performed to replace the general threshold setting method for anonymous detection. Clustering uses distance to find similar clusters. For distance metric, correlation coefficients(including Kendall and Spearman correlation coefficient), Euclidean, MAPE(Maximum Averaged Percentage Error), and CBD(Compression Based Dissimilarity) **[33]** has been used in this paper. The distance was used to identify how the anomaly score is far from the normal case and thence classify the fault. Here, In addition, a dendrogram was written according to the degree of association so that it could be visualized.

2.5. Results

2.5.1. Environmental Settings

Environment : Google Colab with CPU Configuration of Intel(R) Xeon CPU 2.30GHz Dual CPU with Ubuntu 18.04

2.5.2. Anomaly Score Result for Issues

The anomaly score above a threshold indicates that the anomalies are more likely to appear. In this study, the threshold for anomaly score was calculated from the statistical distribution of the normal data's model output. The normal value was put into the trained conv-LSTM model as an input value, and then the distribution of model output (anomaly score) was checked. After that, the boundary value of the distribution was set to threshold. The threshold value is set as constant value regardless of the situation, which is related only to the type of normal data used in training. Currently, it is a threshold value derived as a maximum value when trained using all four normal data, and is 0.94.

Fig. 2-3 shows experimental and anomaly score trends with event moments for each issue (Issue-1, -2, -3, -4 as (a), (b), (c), (d), respectively). Anomaly scores are obtained from the model with different windows lengths of 10, 30, and 60. It should be noted that the time interval for each window is 0.015hr (53sec). Since the model output can be obtained after the largest input length, model result starts from 0.883h, not from 0h. Overall, the period where the calculated anomaly score exceeded the threshold was observed similar to the section in which the event occurred in the experiment (see **Fig. 2-3**). In addition, when comparing the above anomaly detection results with the control point, it was seen that the anomaly scores decreased at the same point with the control action had been taken.

Depending on the size of the window, the value of the anonymous score appears differently. If the window size is small, it is suitable to catch more than short cycles, and if the window size is large, it is suitable to detect Anomaly with long cycles. Issue-1 (**Fig. 2-3a**) and Issue-3 (**Fig. 2-3c**) have some similar tendencies, which can be seen as cases where the cycle of abnormal situations is not clear or noisy.

On the other hand, Issue-4 (**Fig. 2-3d**) shows the different trend depending on the size of window. It shows a similar score value on different size of window at the beginning, but the calculated score shows a different trend after 7 hrs of operation. The trend for 10 size of window shows similar trend to size of 60 but different to size of 30. This means that the period of the abnormal pattern appears only as a combination of a shorter period of 10 or less and a longer period of 60 or more compared to an intermediate length pattern of about 30. According to the experimental data, the temperature of V111 vessel (TI102) drops sharply after 10 hours. As a result, the liquid level (LIC111) increases and finally exceeds the normal operating range (around 60%). The results shows that the score calculated from window size of 10/60 tend to follow better than the scores calculated with a window size of 30.

In the case of Issue-2 (**Fig. 2-3b**), the score calculated with each size of window shows the different trend for each section. At 2.5hr to 3.5hr, the score in window size of 10 is significantly higher than in windows 30 and 60, so it can be considered that an abnormal situation with a short cycle may have occurred at this time. From 3.5 hr to 7 hr, the score at window 30 is remarkably high, so it may be thought that the cycle of abnormal situations may have been a little longer than at 2.5 hr to 3.5hr.

Table 2-6. Accuracy of anomaly detection by Issues and windows

	Issue 1	**Issue 2**	**Issue 3**	**Issue 4**
Max Index	**87.2%**	**78.9%**	49.8%	**88.0%**
Mean Index	78.6%	54.5%	49.5%	79.5%
Window 10	84.5%	64.0%	**58.6%**	67.6%
Window 30	82.7%	55.3%	49.5%	39.1%
Window 60	73.6%	58.2%	49.5%	86.7%

Table 2-6. shows the accuracy of the anomaly detection result by different window length. However, each window length detects different anomaly so detection with a single window is not enough. For Issue-4, detection accuracy difference between window 30 and 60 is about 47%, which is very high. So, to derive the single system-wide anomaly score, anomaly scores obtained by different windows should be combined. Max index identifies anomaly if one of the scores among all windows exceeds threshold. Mean

index averages all scores (in this case, averaged score of window length of 10, 30, 60). As a result of comparing them, in the case of performing anomaly detection by selecting the largest value among the three(Max index), the result was higher than that of separately checking the results for each window. In particular, For Issue 2 the accuracy of max index showed highest, which is higher than 2nd highest detection result by more than 14%.

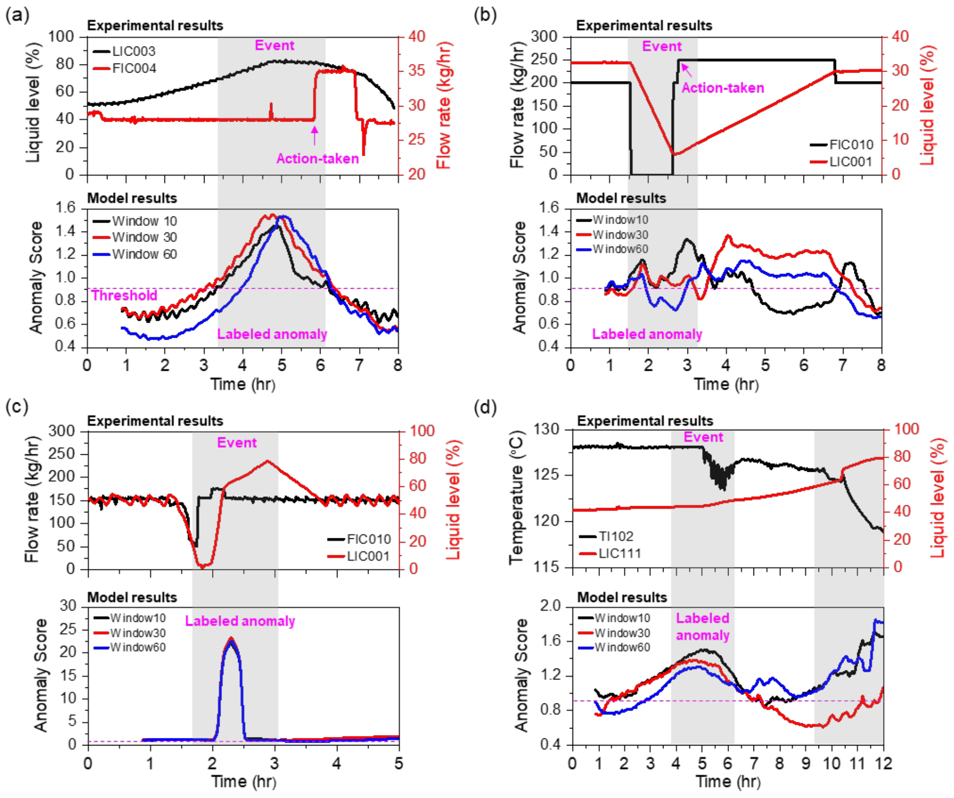


Fig. 2-3. Experiment and calculated anomaly score trend for issue 1(a), issue 2(b), issue 3(c), and issue 4(d) with different with different window size was (10, 30, 60). Pink dotted line denotes threshold obtained from distribution of normal data's anomaly score . Grey marked area shows event from experiment case and labeled anomalies. Calculated threshold value is 0.94.

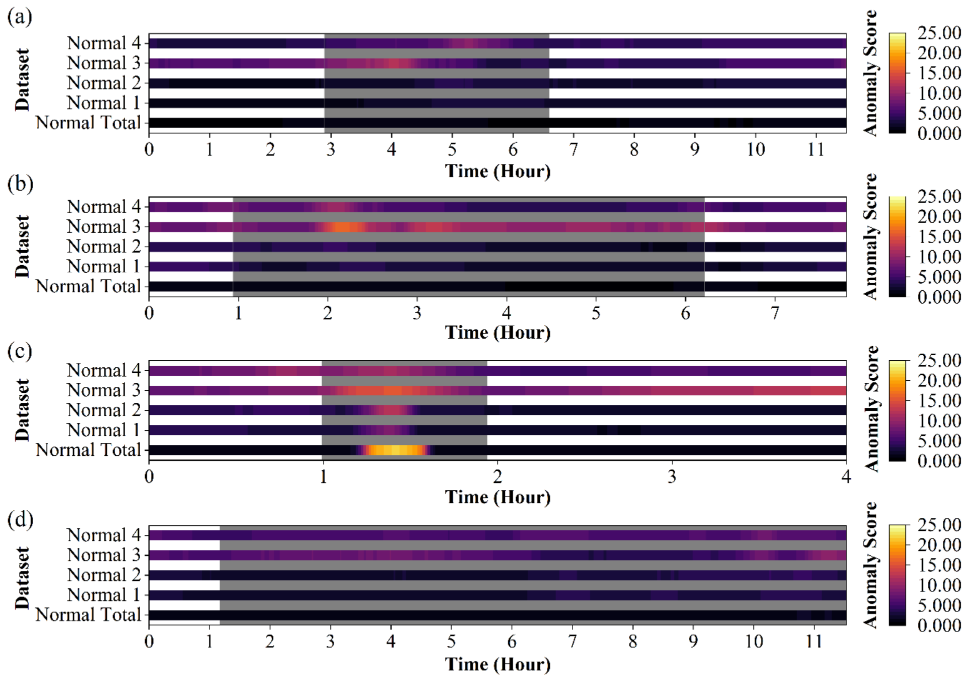


Fig. 2-4. Heatmap of anomaly score on (a) issue 1, (b) issue 2, (c) issue 3, (d) issue 4with different models which trained with different train data : Normal 1~4, Normal Total.

2.5.3. Result of anomaly scores with different train data(train1~4, Sum of train data, synthesized train data)

The challenge of deep learning-based technology is to require huge amount of data for better performance and its performance can be susceptible to the quality of train data/model structures **[34].** In this paper, we learned that the internal parameters and performance of the anomaly detection model may be greatly influenced depending on what data is used as an input, and conducted a case study. Four separately collected data were considered normal data, which have slightly different in its characteristics. In addition, results using synthetic data were added. This was then performed to further find out whether it would be possible to perform training using the synthesized time series data. Synthesis used TimeGAN. As a result, there were quite a few differences depending on what train data was used. In addition, when all train data were combined and used, it was found that the anomaly score was significantly lower than that of each use. This is believed to be because when learning using each case of trains 1 to 4, only each case of trains 1 to 4 is determined to be normal, but when learning using the whole, all cases are determined to be normal, so if any of trains 1 to 4 are determined to be normal, it is considered normal. So using the synthetic data which includes every feature of the separated train data or the parallel use of different models trained with different training dataset would be recommended.

2.5.4. Anomaly Score Timeseries Clustering

The results of the time series clustering data above show that the average score value is well separated for each issue. This is the result of clustering six different windows (length of 1, 5, 10, 30, 60, and 90) using Issue1 to 4, normal data (valid). Each is clustered using different methods, and clustering results are slightly different.

Figure shows anomaly score value as output of model trained with normal data 3. This anomaly score is separated using several time series-based clustering methods. In Figure, (a), (b), and (c) are clustering techniques based on correlation coefficient, which best isolate issues. (a)The results of clustering based on Pearson, (b) Kendall, and (c) Spearman coefficients. (d) and (e) are clustering results based on the Maximum Average Precious Error (MAPE) and Euclidean distance, respectively, indicating that they separate normal data well, but not in the separation of Issue. (f) uses the CBD technique of clustering after performing compression by determining a section, and the distance for each case is calculated around the default value of 0.5, but it is difficult to say that clustering was performed according to the purpose. Among them, (a) based on Pearson correlation coefficient and (b) based on Kendall correlation coefficient shows the most obvious clustering for each case and the largest difference between normal data and other Issue data. Through this result, it can be confirmed that when a specific clustering technique is used, the presence or absence of a steady state can be monitored only with the Anomaly score result value, and based on this, which issue can be inferred.

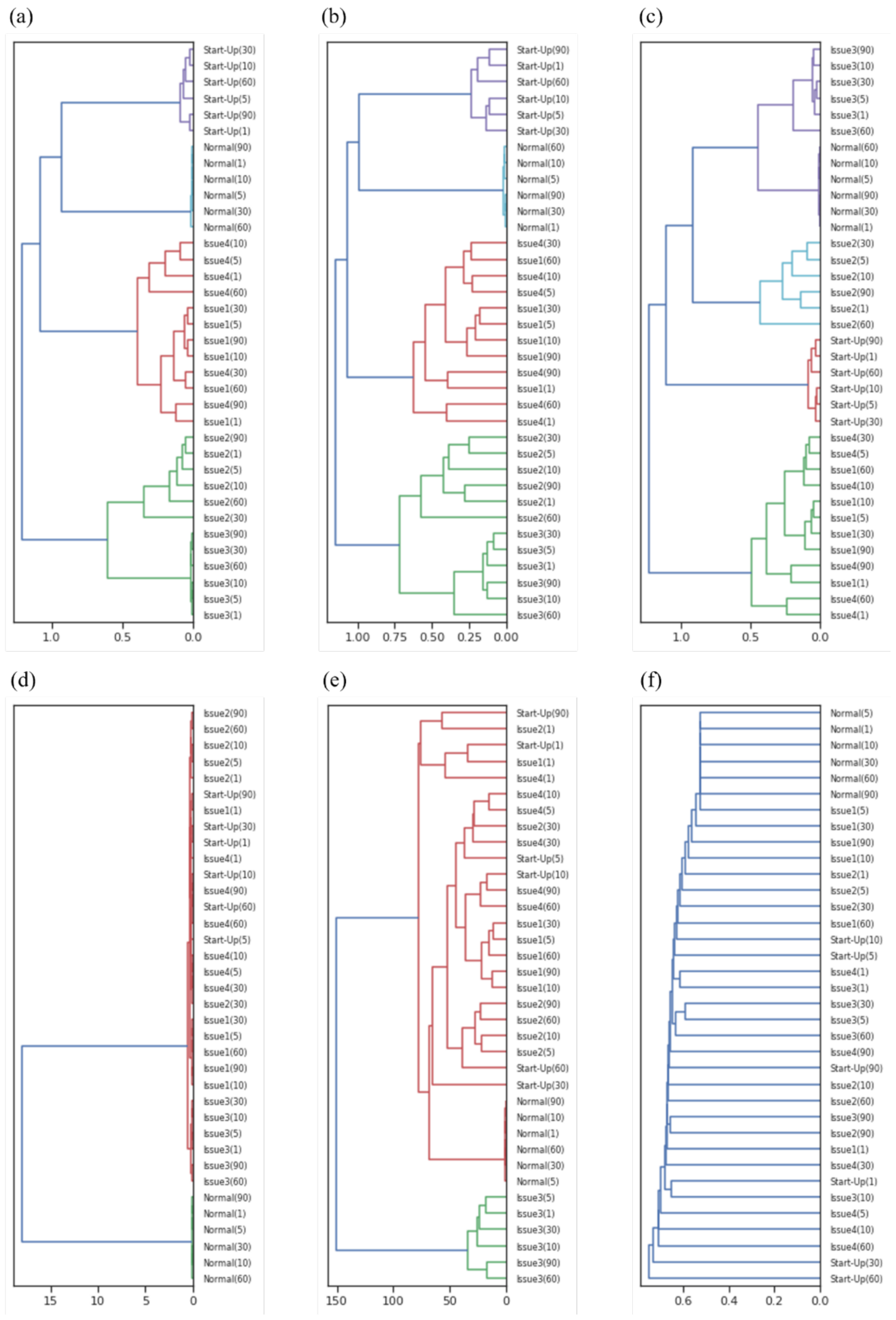


Fig. 2-5. dendrogram of clustering result using 6 clustering methods : Pearson correlation coefficient(a), Kendall correlation coefficient(b), Spearman correlation coefficient(c), MAPE based distance(d), and Euclidean distance(e),

CBD(f). Input of clustering was anomaly score for 6 case(Normal, Issue 1~4, Start-up) with 6 window size(1, 5, 10, 30, 60, 90).

2.6. Summary and Discussion

Through this study, anomaly detection was performed based on the interconnected side of process plant sensor data rather than simple sum of single sensor data. This model used a framework that modified the method of calculating the anomaly score based on the MSCRED framework for the chemical process. Also, we established methodology to process the anomaly detection using multiple windows. In addition to performing anomaly detection, the effect of the training data on the anomaly detection performance of the model was analyzed, and the situation and timing of separation of each issue were possible.

Topics need more discussion are as below :

- **Impact of adjacency in signature matrix** - When creating a signature matrix, the order of sensor data or how adjacent sensors are impacting on the output. Manually collecting related sensors and processing them in a hierarchical manner can improve performance. Additional studies are needed.
- **Finding the best training data** - The exact condition of normal data to training the data is not yet determined. more robust study is needed to determine the optimal training data for the anomaly detection. It can be seen that the results of anomaly detection vary depending on which data is used as input data. Through this, it was possible to confirm the importance of selecting learning data for the anonymous detection model. However, it has

yet to reach a general answer to which data makes the best anomaly detection model. If we can figure this out, I think it will be a great contribution to data-driven industrial animation detection.

- **Identification of fault type and cause diagnosis in real time -** Anomaly score of each issue was separated through clustering, but it cannot automatically classified. Making automatic classification can be more helpful for actual use.

2.7. Acknowledgement

This chapter is partially adapted from Plant-wide Anomaly Detection and Single-Value Monitoring using Conv-LSTM in Pilot-Scale MEG Regeneration Plant with authors N. Lee, H. Kim, J. Jang, and Y. Seo. (Preparing Submission)

Chapter 3. Multivariate Time Series Prediction**

3.1. Introduction

Gas hydrates are crystalline compounds in which hydrogen-bonded water molecules form lattice structures that encage gas molecules such as methane, ethane, and carbon dioxide **[76].** These molecules are the predominant components of natural gas, thus the gas hydrates attract attention as natural resources trapping huge amounts of methane or a gas storage medium. Clathrate gas hydrate can store approximately 170 volume of gas per volume of hydrate(STP) in theory **[82]**. From the perspective of the energy industry, gas hydrates have been an operational risk that may block the subsea flowlines transporting the produced hydrocarbon fluids. In addition, the gas hydrates have been also applied in environmental fields including carbon dioxide (CO_2) capture and storage **[84]**. Therefore, understanding the formation characteristics of gas hydrates has been central to managing the operational risks of offshore gas fields and developing novel gas storage technologies for methane and even hydrogen.

Early recognition of hydrate formation kinetics suggested the mass transfer of gas molecules into forming hydrate particles through a liquid phase, leading researchers to improve the interfacial interaction between water and gas molecules by adopting kinetic promoters including anionic surfactants such as SDS **[39-41]**. Without these promoters, gas hydrates eventually formed the film on the interface

** This chapter is partially adapted from Time series prediction of hydrate dynamics on flow assurance using PCA and Recurrent neural networks with iterative transfer learning in *Chemical Engineering Science* with authors Lee, N., Kim, H., Jung, J., Park, K. H., Linga, P., & Seo, Y. (https://doi.org/10.1016/j.ces.2022.118111).

between water and gas phases, limiting the continued transfer of gas molecules. A mechanical stirred-tank reactor was used to continuously form the gas hydrates, but increasing hydrate fraction in the aqueous phase results decreasing formation rate due to mass transfer limitation **[37]**. To avoid the blockage of the reactor connecting downstream units, water and gas flow rates need to be optimized or novel devices like impinging jets and spray nozzles must be implemented. Times New Roman pointed out that hydrate formation kinetics might be the major challenge to develop hydrate-based technologies and understanding the naturally-occurring marine gas hydrates.

There have been studies about the hydrate formation mechanism in different systems: oil-dominated **[80]**, gas-dominated **[50]**, and water-dominated systems **[59].** In a gas-dominated system, the flow regime changes from homogeneous to heterogeneous suspensions of gas hydrate particles in the liquid phase, termed as a transition point **[59]**. Upon becoming a heterogeneous flow regime, segregation of hydrate particles was observed from the continuous phase, resulting in the formation of hydrates beds on the wall. The hydrate beds eventually became blockages for the continuous liquid phase, thus considered an operational risk. The transition points may vary depending on the Gas-to-Oil Ratio (GOR), Reynolds number, the velocity of the mixed flow and liquid loading, etc. **[37, 42, 43]**. Aman et al. **[37]** investigated the hydrate fraction at the transition point and calculated the Reynolds number in the gas-water system. Chaudhari et al. **[42]** developed the correlation between the transition point, Reynolds number, capillary number, and liquid loading in oil-dominated flowlines. However, these empirical approaches have limitations in the scale-up application and may demand a scale-up factor to be verified on other scales. Davies et al. **[49]** developed the model for hydrate formation in oil-dominated flowlines to predict the pressure drop and viscosity change due to the agglomeration of hydrate particles. Lorenzo et al. **[50]** quantified the pressure drop due to the hydrate deposition and sloughing in a gas-dominated flow-loop through the pressure, and temperature profiles by using the empirical parameters. Charlton et al. **[41]** proposed the hydrate growth model and verified with the experimental results from the gas-dominated flow loop. These attempts tried to link the pressure profiles with viscosity change, resulting in a transition from homogeneous to heterogeneous flow. They manipulated the intrinsically linked parameters in the flow-loop to match the predicted and experimental results, but still, it may challenge the scale-up issue.

Recently, there has been an attempt to advance the hydrate kinetics model. Qin et al. **[71]** applied the classification/regression machine learning techniques to analyze the relationship between the hydrate fraction and the probability of hydrate plugging in the pipeline according to the independent input features such as water cut, GOR. However, the statistical methods used the classification state of the data and cannot analyze the trend of plugging risk. It is still required to develop a method for predicting the hydrate risk in pipelines through real-time sensing data.

Table 3-1. Literature related to the hydrate formation prediction

Literature	Objective	Variable	Methodology
P. J. Metaxas, et al. (2019)	Formation probability fitting	P	Fitting formulation by experiment
V. W. Lim, et al. (2020)	Formation probability fitting	P	Fitting formulation by experiment
B. X. Ferreira et al. (2022)	Near future prediction	ΔP	Multi layer perceptron

In pipeline, lots of variables (i.e. pressure, temperature) are acquired from sensors. Those sensor logging data can be classified as time series data, which is the data logged continuously with fixed time interval (time step). In this work, we tried to adopt data analysis techniques to analyze the features of the hydrate formation process in terms of pressure, temperature, and torque data. The Principal Component Analysis (PCA) technique can reduce a set of different time-series data as one single time-series data (principal component) without losing its important features. Koegh et al. **[61]** showed multivariate time series clustering with PCA as a feature reduction technique for anomaly detection. Gupta et al. **[54]** predicted the point when fatigue damage crack occurs and its length propagation by time using the PCA technique. Once we reduce the multivariate time series data into a single principal component, we can develop a data-driven model to predict the time-series trend for the desired period, i.e. temporal prediction. The sliding window is widely used in treating time series, also known as time series segmentation **[47]** . Common methods for time series prediction can be statistical methods like Auto Regressive Integrated Moving Average (ARIMA) and particle filtering, and deep learning methods like Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). Siami-Namini et al. (2018) compared the statistical model (ARIMA) and deep learning model (LSTM), where the prediction accuracy of the deep learning model was 85% better than that of the statistical model.

Deep learning is a subset of machine learning, a computational model used for regression and classification of time-series data **[65]**. The type of deep learning varies by its architecture but is classified into three large categories, Multi-Layer Perceptron (MLP), Convolution Neural Network (CNN), and Recurrent Neural Network (RNN). MLP is the most basic type of neural net architecture **[65]** with many cells in one layer and interconnected with the previous one. CNN is specialized in image data, involving convolution with filter in preprocessing step. RNN is specialized in sequential data like continuous signals, connecting cells side by side to mimic the continuous temporal behavior **[65]**. In literature, most prediction and forecasting tasks show that RNN families (RNN, LSTM, GRU) produce the best results **[79]**.

The objective of this work is to develop the time series prediction model for hydrate formation based on a novel data-driven framework. We first investigate the hydrate formation characteristics in a gas-water system with experimental results from a high-pressure autoclave under different initial conditions. Then we applied a sliding window and a PCA technique for the experimental data. The time-series-based deep learning models predicted the principal components for hydrate formation and transition from homogeneous to heterogeneous flows. After making the prediction model, the model was trained through the transfer learning by each experiment batch to enhance the performance. Using the combination of PCA and deep learning models with the transfer learning method, the developed framework was useful to predict the hydrate formation trend from the obtained data during specific periods.

Table 3-2. Literature related to the time series prediction

Literature	**Methodology**	**Comparative advantage**	**Application**	**Stationary**	**Seasonality**	**Training method**
X. Ma et al., (2015)	LSTM	MLP, NARX, SVM	Traffic speed	X	O	Single sequence
A. Sagheer and M. Kotb (2019)	LSTM	ARIMA, ERNN, GRU	Petroleum production	X	X	Single sequence

L. Kuan et al., (2017)	GRU	LSTM, GRU	Electricity load	X	O	Single sequence
Y. Wang, et al., (2018)	GRU	LSTM, SVM, ARIMA	Photovoltaic power	X	X	Single sequence
U. Ugurlu, et al., (2018)	GRU	MLP, CNN, LSTM	Electricity price	O	O	Single sequence
C. Tian et al., (2018)	Hybrid CNN-LSTM	RNN, LSTM	Electricity	X	O	Single sequence
Z. Shen et al., (2020)	Hybrid CNN-LSTM	LSTM, CNN	Financial	X	O	Single sequence
This Study	ARLSTM	Dense, LSTM, GRU	Hydrate Formation	X	X	Multiple sequence

3.2. Experiment

3.2.1 Materials and Methods

A methane gas (99.9%) was supplied by Alpha Gas (South Korea). Deionized water (99.0% purity) was used without further purification. A high-pressure autoclave was made to investigate information related to the hydrate onset time and the volumetric amount of hydrate formation while monitoring pressure, the temperature of the autoclave, and torque changes of the mechanical stirrer. The experimental apparatus is as shown in **Fig. 3.1**. The autoclave was made of 316 SUS and had an anchor-type impeller to mix the system. The impeller was located on the base of the shaft and the torque of the rotating shaft was measured by a torque sensor (TRD-10KC) having a platinum-coated connector with an uncertainty of 0.3%. Transducers measured the pressure with an uncertainty of 0.1 bar. The cell was immersed in a refrigerator to control and maintain the temperature of the cell. The platinum resistance thermometers measured the temperature of the liquid and gas phase with an uncertainty of 0.15°C. The torque, pressure, and temperature data were recorded through a data acquisition system in real-time every 10 sec.

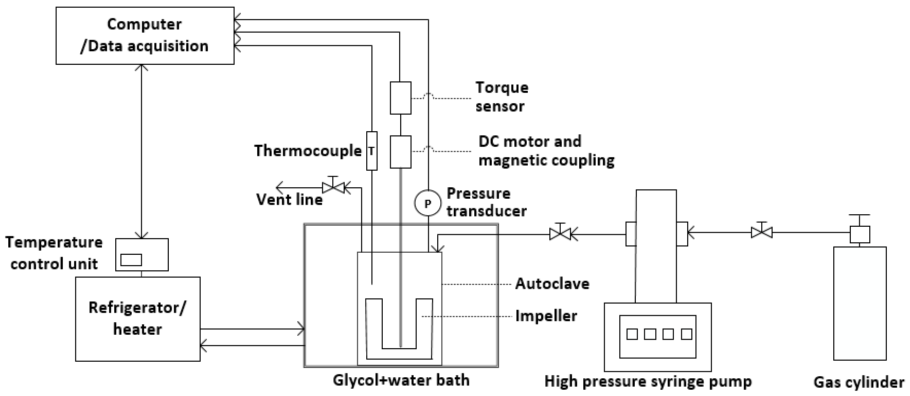


Fig. 3-1. Schematic diagram of the experimental apparatus [63]

The constant cooling method was used to investigate the hydrate formation kinetics. The cell was filled with 200 ml of aqueous solutions with inhibitors and 310 ml of methane gas. The cell was pressurized to 130 barg or 100 barg at 24 °C and mixed with the target mixing rate to saturate the liquid phase with gas till reaching a steady-state condition. The cell was cooled to 1 °C at 0.25 °C per minute by a bath circulator and maintained at 1 °C for 10 hours to consider outside temperature and resident time in the subsea pipeline. The performance of hydrate risk for agglomeration of hydrate particles was evaluated using torque changes because torque changes could be an indicator of resistance-to-flow **[58]**. In this work, a total of 16 experiments were carried out as shown in **Table 3-3**. Hydrate nucleation and growth were recognized from a rapid pressure decrease and temperature spikes from exothermic heat of hydrate formation.

Table 3-3. A summary of the tested condition for hydrate formation characteristics

Batch number	Initial pressure (barg)	Mixing rate (rpm)
1 – 4	100	200
5 – 8	130	200
9 – 12	100	600
13 – 16	130	600

To assess the characteristics of hydrate kinetics from sensing data, the relative torque is calculated as the ratio of the torque recorded during the experiment at a certain time (τ_t) to the torque measured before nucleation (τ_s) using **Eq. (3-1).**

$$Relative\ torque = \frac{\tau_t}{\tau_s} \tag{3-1}$$

The moles of consumed gas during hydrate formation was calculated from followed equation, which is based on the difference between experimentally measured pressure and estimated equilibrium pressure at a certain temperature. The methods to get the moles of consumed gas and the hydrate fraction values have been suggested in the literature as a method for studying hydrate formation in small-scale apparatus **[63]**.

$$\Delta n_{H,t} = (\frac{P_{cal}V_{cell}}{zRT})_t - \left(\frac{P_{exp}V_{cell}}{zRT}\right)_t = \left(\frac{\Delta P\ V_{cell}}{zRT}\right)_t \tag{3-2}$$

The hydrate fraction in the liquid phase was calculated using the amount of consumed gas by **Eq. (3-3) [62]**,

$$\Phi_{hyd} = \frac{V_{hyd}}{V_{hyd}+(V_w-V_{w,conv})} \tag{3-3}$$

, where V_w is the volume of water, $V_{w,conv}$ is the volume of the water converted to hydrate, and V_{hyd} is the volume of formed hydrates.

3.2.2 Experimental Results on Hydrate Formation

Fig. 3-2. shows the measured results (pressure, temperature, relative torque) and calculated hydrate fraction as a function of the time for the experiment (batch No.12 in **Table 3-3.**). It is noted that relative torque could indicate the flow-to-resistance due to the hydrate particle agglomeration **[37, 58]**. The experiment set appears the five distinct regions (labeled with A-E).

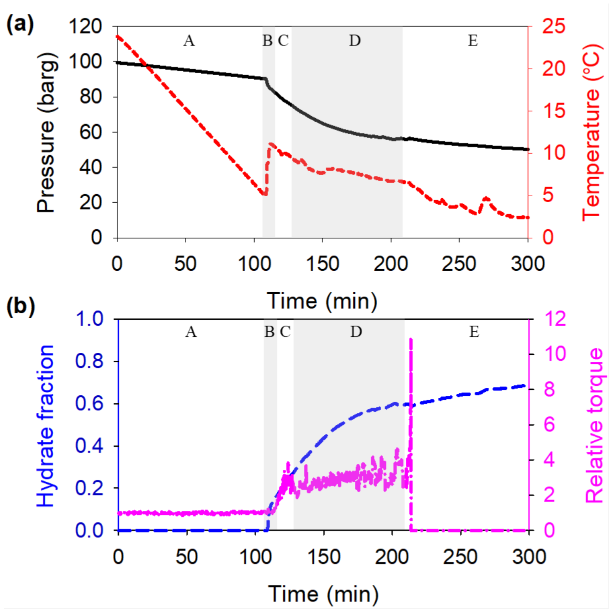


Fig. 3-2. Measured pressure, temperature (a), and calculated hydrate fraction measured relative torque (b) during the cooling and hydrate formation (experimental batch No. 12)

Each region is characterized by the rate of hydrate formation and changes in relative torque. Region A is a stage before hydrate nucleation and the pressure decreases just from the cooling effect. Then hydrate nucleation occurs in region B designated as an early stage of hydrate formation. The temperature rises instantaneously due to the exothermic heat of hydrate formation while the pressure decreases sharply due to gas consumption. The relative torque was stable with the small population of hydrate particles homogeneously dispersed in the liquid phase. Region B is relatively short.

In followed region C, the pressure was sharply dropped from the fast formation and growth of hydrate particles. In this region, the relative torque increases as increasing hydrate fraction results in the increasing viscosity of the liquid phase. The beginning of region C indicates the transition point from homogeneous to heterogeneous flow. Continuous increase of the relative torque was then followed by fluctuation in region D, which started from the segregation point. This behavior would be attributed to the break of hydrate particles with continuous mixing from increased fluid shear stress **[50]**. Segregated hydrate particles tend to repeat the deposition and breaking from the wall. After several fluctuations of the relative torque, a sudden surge of the torque and motor stoppage were observed possibly due to the thick deposition of the hydrate beds that the motor couldn't break. This is region E. It should be noted that a safety lock has been implemented for the impeller stopping when the torque is higher than 50 N·cm, this is to protect the motor and torque sensor **[77]**.

Fig. 3-3. showed the relative value changes as a function of hydrate volume fraction during hydrate formation under low (200 rpm) and high (600 rpm) mixing rates after the hydrate nucleation. Both experimental data and prediction data from the viscosity model were presented. At both mixing rates, the relative torque increased sharply as increasing the hydrate fraction from region B to region C. Similar to previous works **[59]**, the flow characteristics changed from region B to region C when the hydrate fraction reached about 0.15.

The point below the transition fraction is generally said to be able to transport hydrates safely and the “No Plug” zone **[71]**. As we mentioned, the transition point is dependent on lots of features such as liquid loading, fluid velocity, and driving force for hydrate formation (pressure, temperature, etc.). In regions D and E, the relative torque fluctuated and eventually stopped as increasing hydrate fraction. The operation of the motor stopped around 0.25 to 0.3 of hydrate fraction due to the surge in the relative torque value at a low mixing rate (200 rpm) as shown in **Fig. 3-3a**. In contrast, the relative torque at a high mixing rate (600 rpm) suddenly decreased and fluctuated with a larger amplitude (**Fig. 3-3b.**). A faster mixing rate could lead to the continuous breakage of hydrate particles avoiding the hydrates bedding. For the four of eight experiments, the motor stopped at around 0.5 - 0.6 of hydrate fraction due to over range of torque levels that may be caused by jamming and plugging of segregated hydrate beds.

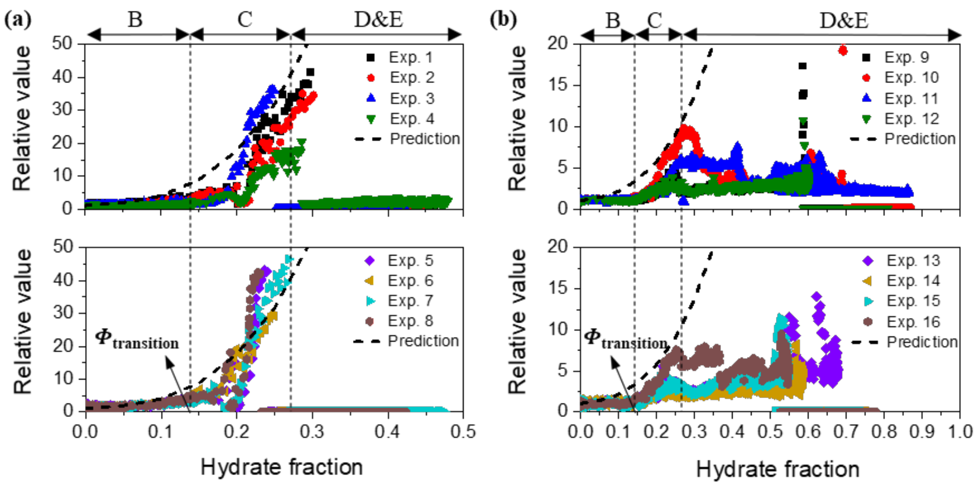


Fig. 3-3. Relative torque changes were observed during hydrate formation at 200 rpm (a) and 600 rpm (b)

Joshi et al. **[59]** observed that the flow behavior with hydrate particles is changed from homogenous to heterogeneous distribution, which then induces particle accumulation and jamming. The hydrate volume fraction at which this transition occurs is referred to as the transition hydrate fraction ($\Phi_{transition}$) **[59]**. The relative viscosity of hydrate slurry increases rapidly with increasing hydrate fraction after the transition hydrate fraction. In autoclave experiments, changes in relative torque indicate the changes in the rheological properties of hydrate slurry. To predict the transportability of hydrate slurry, we compare the relative torques obtained from the autoclave experiments to the relative viscosities estimated from the viscosity model of hydrate slurry.

The relative torque was calculated from the experimental results and **Eq. (3-1)**. The hydrate slurry behaves as a non-Newtonian fluid **[74]** and the viscosity of hydrate slurry can be developed from the viscosity of the concentrated suspension. Camargo and Palermo **[40]** presented a model for hydrate slurry viscosity as a function of particle volume fraction using the Mills model **[68]**, which considers the rheological properties of immobilized fluid trapped between the particles. As the particles form, the amount of suspended fluid decreases, and the viscosity of the fluid system increases **[66]**. Thus, the viscosity of the hydrate slurry changes depending on the hydrate fraction in the fluid. Herein, we investigate the relative viscosity (μ_r) of hydrate slurry, which is the ratio of the viscosity between hydrate slurry and continuous fluid (water). The relative viscosity (μ_r) can be derived from **Eq. (3-4) and Eq. (3-5)** due to the fractal structure of aggregates as shown in **Fig. 3-4**.

$$\mu_r = \frac{1-\Phi_{eff}}{\left(1-\frac{\Phi_{eff}}{\Phi_{max}}\right)^2} \tag{3-4}$$

$$\Phi_{eff} = \Phi\left(\frac{d_A}{d_p}\right)^{3-f} \tag{3-5}$$

, where Φ_{eff} is an effective volume fraction of immobilized fluid and Φ_{max} is the maximum packing fraction, which is typically in the range of 0.64 to 0.74 **[67]**. This work assumed the maximum packing fraction of 0.74 **[66]**.

The size of the agglomeration (*d_A*) was calculated under steady-state force balance using **Eq. (3-6).** In the equation, d_A and d_p denote the size of agglomeration and monomer size, respectively. the particle size is determined by shear stress and cohesion force between hydrate particles **[40]**. F_a is the cohesion force between hydrate particles and f is the fractal dimension assumed to be 2.5 from previous literature **[49]**. μ_c is the viscosity of the continuous phase. Φ is the volume fraction of hydrate in the slurry, and γ is the shear rate of the system. The shear rate in the autoclave was derived from the mixing rate and autoclave geometry **[69]**.

$$\left(\frac{d_A}{d_p}\right)^{4-f} - \frac{F_a\left(1-\frac{\Phi}{\Phi_{max}}\left(\frac{d_A}{d_p}\right)^{3-f}\right)^2}{d_p^2\mu_c\gamma(1-\Phi\left(\frac{d_a}{d_p}\right)^{3-f}} = 0 \qquad (3\text{-}6)$$

In this work, the monomer size was assumed from the Sauter mean diameter of the entrained droplets in a gas-dominated flow loop **[38]**. The viscosity of continuous solution (μ_c) was derived from multiflash v6.2 with the CPA infochem model set. The cohesion force (F_a) between CH_4 hydrates was assumed to be 37.5 mN/m, which Wang et al. **[83]** measured under 3.2 MPa.

The estimated relative viscosity of hydrate slurry in **Fig. 3-3.** increased exponentially with increasing hydrate volume fraction regardless of phase transition. Likely, the viscosity model could not demonstrate the fluctuation of the relative torques induced by the segregation and deposition of hydrate particles. Gas hydrate particles tend to agglomerate rapidly from the transition point. Understanding the hydrates flow behavior is central to the safe operation of subsea flowlines. Hydrate formation characteristics have been investigated for more than decades using bench-scale autoclaves as well as pilot-scale flow loops, but the probabilistic and non-linear relationship of hydrate formation with the parameters of the fluid only make it difficult to predict or analyze the hydrate formation characteristics using the analytical models.

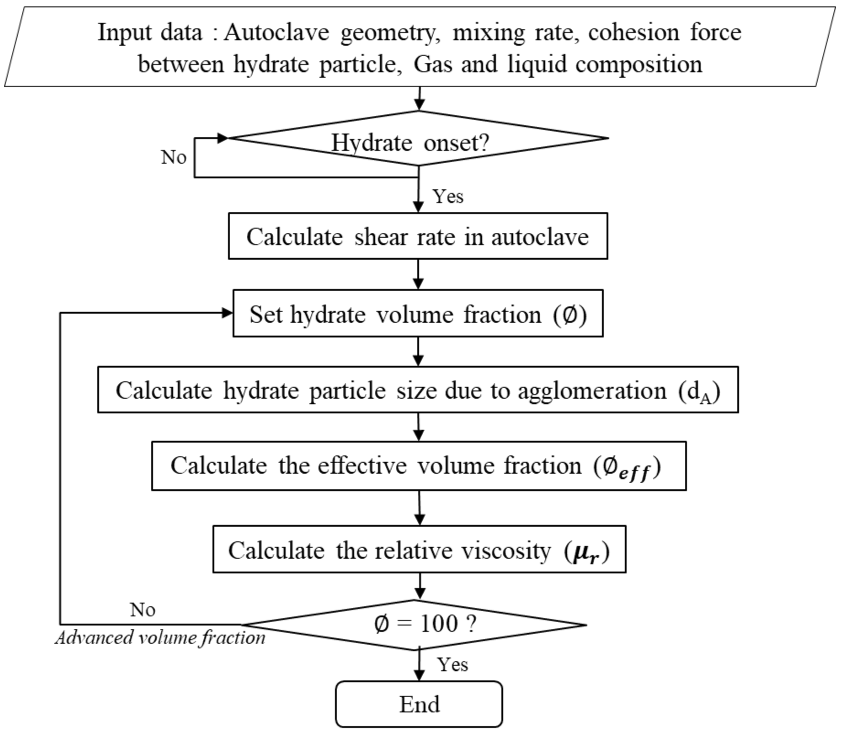

Input data : Autoclave geometry, mixing rate, cohesion force between hydrate particle, Gas and liquid composition
Hydrate onset?
No
Yes
Calculate shear rate in autoclave
Set hydrate volume fraction (∅)
Calculate hydrate particle size due to agglomeration (d_A)
Calculate the effective volume fraction (∅_eff)
Calculate the relative viscosity (μ_r)
No
Advanced volume fraction
∅ = 100 ?
Yes
End

Fig. 3-4. Flow diagram of relative viscosity model in the autoclave

3.3. Computational Methodology

To overcome the limitation of the model-based prediction of hydrate formation, the time series prediction using experimental data was adopted to analyze and forecast the hydrate formation characteristics. We predicted the hydrate formation behavior according to time and mostly focused on the two kinetic behavior: the transition point and the segregation point.

The obtained experimental data are difficult to directly use in the conventional prediction model due to following reasons:

(1) Trend of one specific sensor data (pressure, temperature, relative torque) is not a definite indicator in the prediction of transition and segregation points. Prediction cannot be dependent on specific formula or relation.

(2) Data used for time series prediction usually assume stationary or periodical repetition. However, transition and segregation behavior during hydrate formation are one-time occurrence, that is not repeated but occurs once in each experiment. Application of time series prediction for one-time occurrence is not common.

(3) Most time series prediction is typically applied on continuous non-split historical data like weather or electricity consumption **[55]**. However, hydrate formation experiments are carried out in a batch-by-batch manner and are segmented rather than continuous. Since there is no continuity between two experimental batches, simply concatenating the time series data of many batches and putting it into the existing framework should be avoided.

(4) Prediction cycle is much shorter than other continuous historical data and should carry out on a real-time basis. The kinetic behavior such as phase transition and segregation of hydrate particles occurs in a short duration. Therefore, the prediction model should receive the small and restricted length of time series data and should return following prediction ahead.

To consider the above limitation for acquired data, this work proposed a novel framework in cooperating with the prediction scheme for hydrate experimental data. Schematics of the proposed framework to train the prediction model are shown in **Fig. 3-5**. First, in preprocessing step, PCA is applied to input data to use every time series data rather than one specific sensor data. Single time series data may show good results in making predictions, but it cannot be a definite indicator for the transition or segregation points during hydrate formation. Therefore, we employed the PCA technique to include the effect of whole sensor data. Next, windowing is applied to treat data into a suitable foam for Neural Network (NN) inputs and to make a real-time prediction using a small set of adjacent historical data points. Then, in the model training phase, NN is used to predict an indeterministic system with a vague factor. Transfer learning has been carried out to preserve the characteristics of experimental data, which is done in a batch-by-batch manner and initialized after one batch finishes. After that, model validation is carried out to check if the model prediction result is reasonable and accurate. One model has been validated; the model can be deployed to use in real-time. Finally, in the operation phase, a trained neural network model can predict the future trend of PCA processed data in real-time with windowing. The proposed framework can be implemented universally with various types of input data (i.e., sensors, parameters etc.) to forecast the hydrate formation behavior. However, the input data can affect the accuracy of each model. It is noted that since PCA is a linear transformation and its parameters had been calculated/determined in the training phase, the speed of preprocessing can be done under real-time speed.

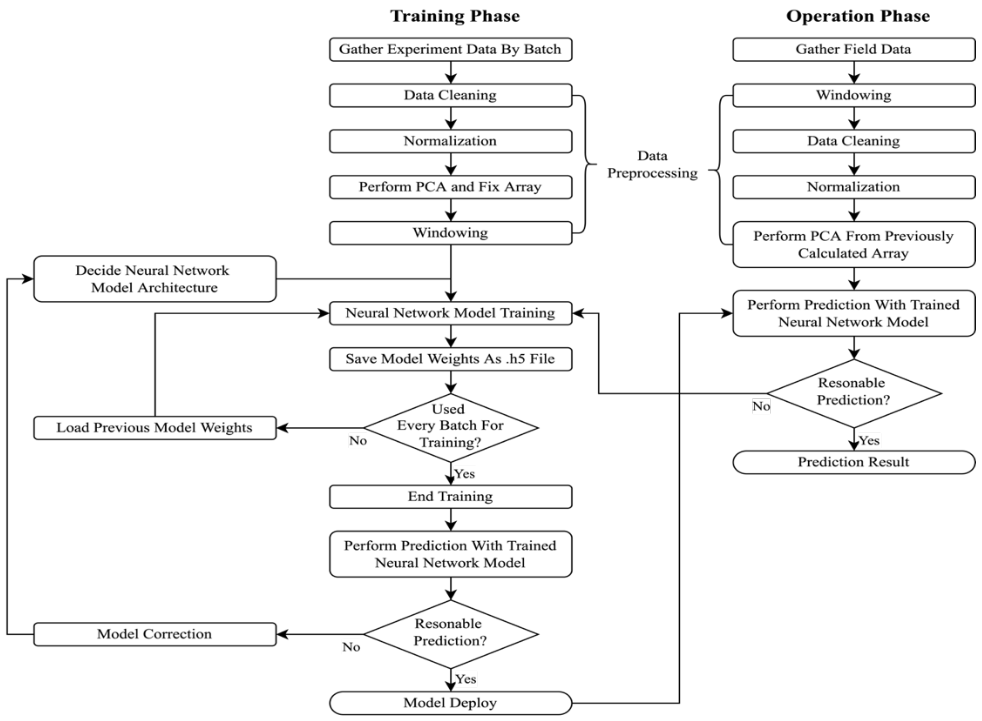

Training Phase
Operation Phase
Gather Experiment Data By Batch
Data Cleaning
Normalization
Perform PCA and Fix Array
Windowing
Data Preprocessing
Gather Field Data
Windowing
Data Cleaning
Normalization
Perform PCA From Previously Calculated Array
Decide Neural Network Model Architecture
Neural Network Model Training
Save Model Weights As .h5 File
Used Every Batch For Training?
Load Previous Model Weights
No
Yes
End Training
Perform Prediction With Trained Neural Network Model
Resonable Prediction?
Model Correction
No
Yes
Model Deploy
Perform Prediction With Trained Neural Network Model
Resonable Prediction?
No
Yes
Prediction Result

Fig. 3-5. Schematics of hydrate formation characteristics prediction model

3.3.1 Data Processing

3.3.1.1. Principal Component Analysis

In the proposed model, rather than predicting the sensor time series as it is, the prediction was performed after reducing the dimension to a latent space. A latent space can be said to be a reduced dimension in which similar data are intertwined, and PCA is a representative algorithm that performs dimensionality reduction to a latent space. Although PCA reduces the number of dimensions, it does not reduce the dimension by extracting only the important ones from the existing features but creates new variables composed of a combination of features. For example, if PCA is performed with seven sensor data, it is not reduced to a part of seven sensor data, but the importance of seven sensor data is multiplied and used as a weighted sum to obtain a combined characteristic value. If this is used, the training time of the machine learning model can be reduced, and the prediction performance can be improved because the number of features is reduced while all values of the sensor can be reflected. Machine learning, especially deep learning models, has the concept of a 'curse of dimension', in which the complexity increases exponentially as the number of features increases, resulting in lower accuracy, which can be solved by reducing the number of dimensions of the input value.

PCA is a commonly used numerical method for feature reduction and clustering. By using PCA, feature numbers are reduced and original features are reconstructed in a latent space. Latent space is a lower-dimensional manifold of high-dimensional data, where dimension means the number of features in this context. PCA is an optimization problem found below **[46]** :

$$max_{v \neq 0} \frac{v^T X^T X v}{v^T v} \tag{7}$$

, where $v \in \mathcal{R}^m$ is set of orthogonal vectors and X is matrix consists of train data with n observations and m process variables. The PCA process itself can be used as an anomaly detection tool since it performs both feature reduction and clustering **[51, 57, 73]**. This reduces the training time of the deep learning model far less than using every feature in the dataset while conserving their characteristics of them. When applied in time series data, PCA plays an additional role in denoising either.

3.3.1.2. Windowing

Windowing is a common preprocessing method in time series analysis **[39]**.Windowing is similar to convolution in the sense that it gives stride values, but it is displayed with the 1D array type. Windowing is making the original time series into many shorter pieces called windows, by sliding through the sequence shown in **Fig. 3-6**. If the window length is n and its stride (gap between two windows) is k, then the first window should be $[x_1, x_2, \ldots, x_n]$ and the second window will be $[x_{k+1}, x_{k+2}, \ldots, x_{k+n}]$, and the third one would be $[x_{2k+1}, x_{2k+2}, \ldots, x_{2k+n}]\ldots$ so on. windowing enables the real-time prediction model by training on the window, not on historical sequences. In this paper, the total window length was set as 120, with the input length of 60 and output length of 60.

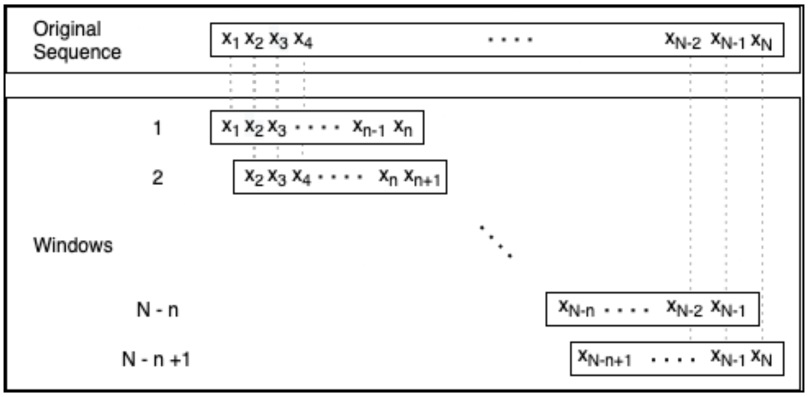

Original Sequence
$x_1\, x_2\, x_3\, x_4 \cdots x_{N-2}\, x_{N-1}\, x_N$
1
$x_1\, x_2\, x_3 \cdots x_{n-1}\, x_n$
2
$x_2\, x_3\, x_4 \cdots x_n\, x_{n+1}$
Windows
N - n
$x_{N-n} \cdots x_{N-2}\, x_{N-1}$
N - n +1
$x_{N-n+1} \cdots x_{N-1}\, x_N$

Fig. 3-6. Concept of windowing

3.3.2 Deep Learning Schematics

3.3.2.1. Model Architecture

The most basic model of deep learning is called MLP. MLP model consists of several layers with many cells. Layers are divided into three types: input layer, hidden layer, and output layer. For the deep learning model, there are many hidden layers and inside the hidden layers, many cells are only interconnected with the cells in adjacent layers and each cell in the same layer is not connected. Each cell consists of receiving part which gathers the previous layer's output values as a weighted sum and an activation function part that adds non-linear behavior. The output value of the activation function is sent to the next layer's input.

Deep learning starts with defining model architecture. The model architecture consists of the type of model layers (dense (=MLP), CNN, RNN, and other variants), the number of layers, and the cell number in each layer. Also, selection of activation functions inside the cell (linear/nonlinear activation functions (e.g. tanh, Sigmoid, Rectified Linear Unit (ReLU), Scaled Exponential Linear Unit (SeLU)…)), Selection of loss metric (e.g. Mean Squared Error (MSE), Mean Squared Percentage Error (MAPE)), selection of optimization function (e.g. Stochastic Gradient Descent (SGD), Root Mean Squared Propagation (RMSprop), Adam, Adagrad, Adamax) and additional techniques like attention **[81]** and dropout **[78]** are followed to boost the accuracy of the prediction. In this study, four model types were selected; dense, LSTM, GRU, and Auto Regressive Long Short-Term Memory (ARLSTM) as shown in **Fig. 3-8**. LSTM, GRU, and ARLSTM models are variants under the RNN family. RNN has been made for processing sequential data and applied in the areas like speech recognition.

These basic deep learning layers are made of interconnected cells between layers, and each of them receives the sum of the product of weight and cell outputs from the previous layer. The name "Dense" comes from the fact that each cell inside the layer has a full connection with every cell inside the previous layer. In time series prediction, a dense layer has the input of one window and trains to have the output of a targeted prediction sequence as shown in **Fig. 3-8a**. LSTM is the advanced version of RNN **[53, 56]** , that allows the past information to persist. LSTM cells are connected sequentially in temporal order as shown in **Fig. 3-8b**. A detailed description of the LSTM model is provided in **Fig. 3-7a**. LSTM cells are connected sequentially in temporal order as shown in **Fig. 3-8b**. GRU (Gated Recurrent Unit) cells are simplified forms of LSTM cells for faster computing and easier implementation **[48, 60]**. A typical GRU cell consists of two gates: update and output gate whereas the LSTM cell has three gates (forget, input, output) to give the memory cells ability. Like LSTM cells, GRU cells are also connected sequentially as in **Fig. 3-8c**. ARLSTM model uses the previous timestep outputs as another dimension of current step input, which is autoregressive in terms of making predictions using its output shown in **Fig. 3-8d.** Autoregressive behavior can be described below.

$$p(x) = \coprod_i p(x_i|x_1,\cdots,x_{i-1}) = p(x_1)\cdot p(x_2|x_1)\cdot\ \cdots\ \cdot (x_i|x_1,\cdots,x_{i-1}) \tag{8}$$

With these equations, the network learns temporal patterns from the input sequence with its past predictions.

A. LSTM (Long Short-Term Memory) model

As shown in **Fig. 3-7a**, LSTM model added operations inside the cell to keep cell states and controlled by three gates: input, output, forget. By this allows the model to selectively take past information into account.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{4}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{5}$$

$$\underline{C_t} = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{6}$$

$$C_t = f_t * C_{t-1} + i_t * \underline{C_t} \tag{7}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{8}$$

$$h_t = o_t * tanh(C_t) \tag{9}$$

where f_t is forget gate, i_t is input gate, C_t is cell state, o_t is output gate. h_t is hidden state at time t, h_{t-1} is hidden state at time (t-1) or the initial hidden state at time o, c_t is cell state at time t, x_t is input at time t, i_t is input gate at time t, f_t is forget gate at time t, $\underline{C_t}$ is cell gate at time t, o_t output state at time t, and $*$ is Hadamard product, σ is sigmoid function.

B. GRU (Gated Recurrent Unit) model

While LSTM cells have three gates (forget, input, output), GRU cells only have two gates (update, output) as shown in **Fig. 3-7b**. GRU integrated both cell state and hidden state while LSTM has both. This integration allows GRU layers to learn and predict faster than LSTM layer without the loss of accuracy.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{10}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \quad (11)$$

$$\underline{h}_t = tanh(W \cdot [r_t * h_{t-1}, x_t]) \quad (12)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \underline{h}_t \quad (13)$$

, where h_t is hidden state at time t, h_{t-1} is hidden state at time (t-1) or the initial hidden state at time o, x_t is input at time t, r_t is reset gate, z_t is update gate, n_t is new gate, $*$ is Hadamard product, and σ is sigmoid function.

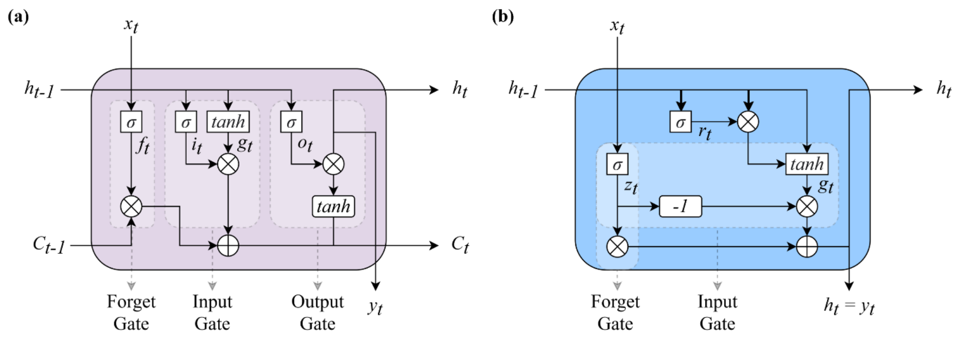


Fig. 3-7. Computational graph inside cell of (a) LSTM, (b) GRU

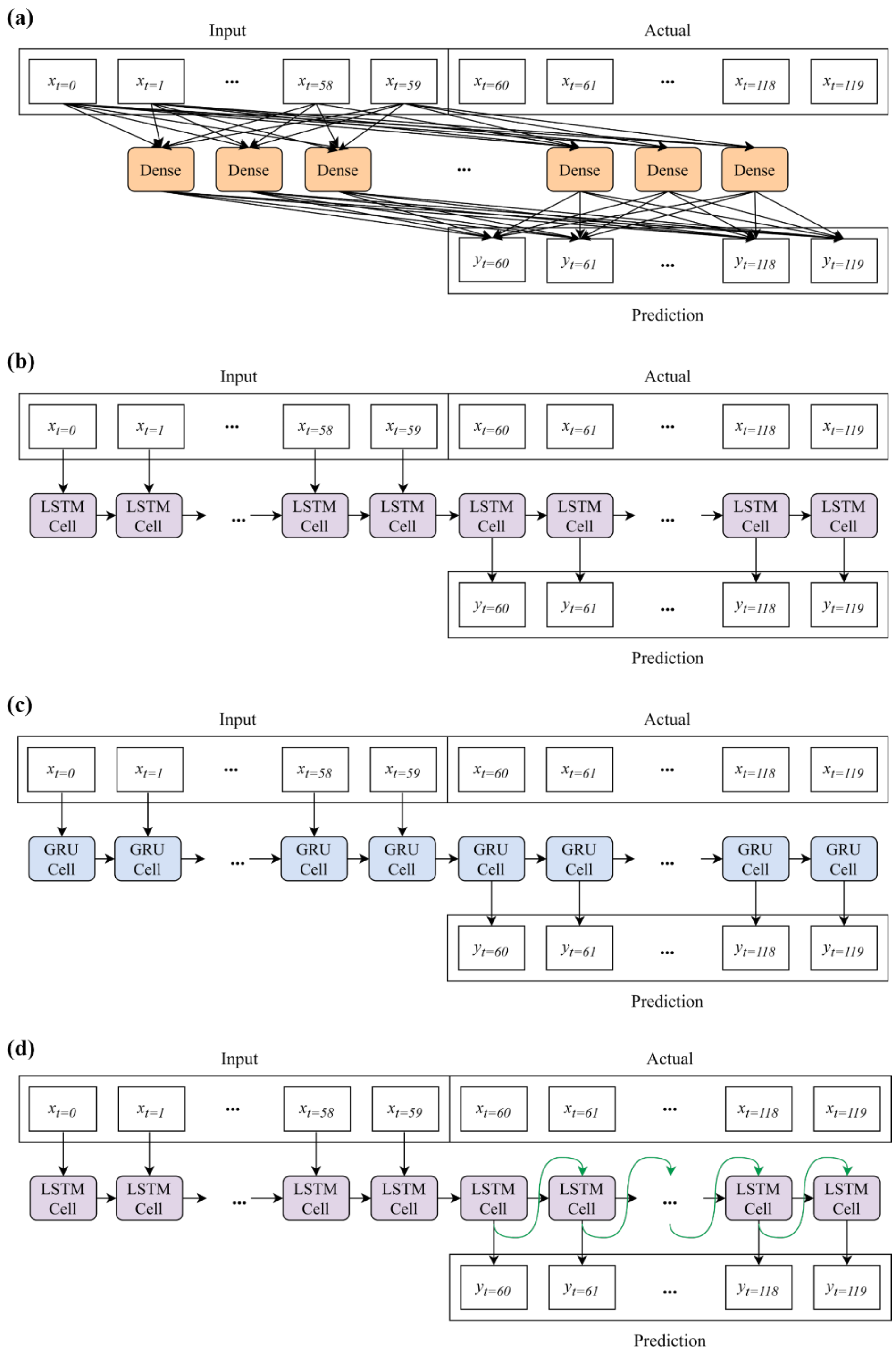


Fig. 3-8. Configuration between deep learning cells, input window, prediction. (a) Dense (b) LSTM (c) GRU (d) ARLSTM. Prediction denotes the predicted value from the deep learning model

3.3.2.2. Model Architecture Configuration

Using model types described above, model architecture has been configured as in **Table 3-4**. In whole models, ReLU was used as an activation function for every cell. In this study, the effects of the number of model layers, dropout, and the number of cells were investigated with four selected model types. Dropout was included in the model since it boosts the accuracy of the RNN models **[45, 52]**. This study compared the results through four types based on the dense models (Dense-1 to Dense-4 in **Table 3-4.**), seven types based on the LSTM models (LSTM-1 to LSTM-7 in **Table 3-4.**), three types based on the GRU models (GRU-1 to GRU-3 in **Table 3-4.**) and three types based on the ARLSTM models (ARLSTM-1 to ARLSTM-3 in **Table 3-4.**). For the dense model, architecture was set to examine the effect of stacking model layers and the dropout technique. Dense-1 and Dense-2 models are basic models with a single dense layer for a hidden layer without and with a dropout rate of 0.2, respectively. The Dense-3 model has two dense layers stacked and the Dense-4 model is two stacked dense layers, each layer with a dropout rate of 0.2. In the LSTM model, seven models have been implemented to investigate the effect of the dropout rate. LSTM-1 to LSTM-5 has a single LSTM layer for hidden layer with dropout rate of 0 (no dropout), 0.2, 0.4, 0.6, 0.8, respectively. LSTM-6 and LSTM-7 models have two stacked LSTM layers for the hidden layer with a dropout rate of 0 and 0.2, respectively. GRU models consist of two stacked GRU layers with various dropout rates (0, 0.2, 0.4). ARLSTM models are studied according to various cell numbers (32, 64, 128), which are only parameters. Noted that cell numbers of each model were set differently, as the structures of each cell are different. Base cell numbers of each model were set as the most used value, for example, 32 for LSTM and 128 for GRU.

Table 3-4. Summary of model architecture configuration

Model Name	No.	Model Architecture
Dense	1	lambda + dense (512, relu) + dense (out_steps) + reshape
	2	lambda + dense (512, relu) + dropout (0.2) + dense (out_steps) + reshape
	3	lambda + dense (512, relu) + dense (512, relu) + dense (out_steps) + reshape
	4	lambda + dense (512) + dropout (0.2) +dense (512) + dropout (0.2) + dense (out_steps) + reshape
LSTM	1	LSTM (32) + dense (out_steps) + reshape
	2	LSTM (32) + dropout (0.2) + dense (out_steps) + reshape
	3	LSTM (32) + dropout (0.4) + dense (out_steps) + reshape
	4	LSTM (32) + dropout (0.6) + dense (out_steps) + reshape
	5	LSTM (32) + dropout (0.8) + dense (out_steps) + reshape
	6	LSTM (128) + LSTM (32) + dense (out_steps) + reshape
	7	LSTM (128) + dropout (0.2) + LSTM (cell 32) + dropout (0.2) + dense (out_steps) + reshape
GRU	1	GRU (128)+ GRU (32) + dense (out_steps) + reshape
	2	GRU (128) + dropout (0.2) + GRU (32) + dropout (0.2) + dense (out_steps) + reshape
	3	GRU (128) + dropout (0.4) + GRU (32) + dropout (0.4) + dense (out_steps) + reshape
ARLSTM	1	ARLSTM (32) + dense (out_steps) + reshape
	2	ARLSTM (64) + dense (out_steps) + reshape
	3	ARLSTM (128) + dense (out_steps) + reshape

3.3.2.2. Transfer Learning by Batch

Transfer learning is the method to use knowledge from the other machine learning models **[70]** widely used in CNN Architectures. In general, the knowledge to be ransferred refers to trained cell weights of hidden layers from another dataset. Transfer learning is a method to train a model with already-trained other datasets. As we mentioned above, our experiment data have a format of discrete yet iterative time series for each experiment set. Since we are interested in the prediction of an irregular and abrupt event, simply concatenating time series data should be avoided. Due to the discontinuous points. To train the model with several experiments set by concatenating the individual sets of experiments, a transfer learning method has been employed. One experimental batch is trained at a time and then the training results are saved with a form of weight parameter. These weight parameters are transferred and loaded when training the next batch.

3.4. Results and Discussion

The working environment of the proposed approach was done on Google Colab, and GPU was used in some batches to boost the running speed. CPU/GPU Selection is mainly affecting on the speed of computation but does not plays an important role in the result of prediction. CPU Configuration is Intel (R) Xeon CPU 2.30GHz Dual CPU with Ubuntu 18.04, GPU Configuration is Tesla K80 with Cuda 11.2. Each model has been trained and saved its cell weights as .h5 files during each training process on transfer learning.

3.4.1. Preprocessing

PCA was introduced to define hydrate formation characteristics considering many factors. PCA was carried out from seven features. Four features are time-series signal from the measured sensor (pressure, relative torque, temperature, hydrate fraction). By PCA, input features are reduced to smaller dimensions that are different from previous variables but still contain the fractions of the input features. PCA calculates the parameters for linear transformation, and the set of values for multiplication is called an eigenvector. With the given training sets (batch No.1~7, 10~16 in **Table 3-3.**), PCA returns the multiplication factor for the original seven features to be transformed as 1st principal component.

Fig. 3-9a. shows the major principal component through PCA analysis during the whole experiment including before and after hydrate transition at the mixing rate of 200 rpm (batch No. 1~8 in **Table 3-3.**). It shows the distinction between regions before and after the hydrate transition fraction in its reduced feature space. Red dots and black dots can be clustered and divided. This distinction allows feature space to divide the regions by transition point, which indicates the initial point of the sudden and rapid hydrate growth. In addition, the PCA result of points before the transition point shows two different line-shaped clusters with a gap due to the different initial pressure conditions of 100 barg and 130 barg. The two different initial condition leads to the different slopes of PCA results. More experiment data with the different initial conditions would be beneficial to determine the hydrate region and the boundary shape more precisely, which supposed as linear in this paper.

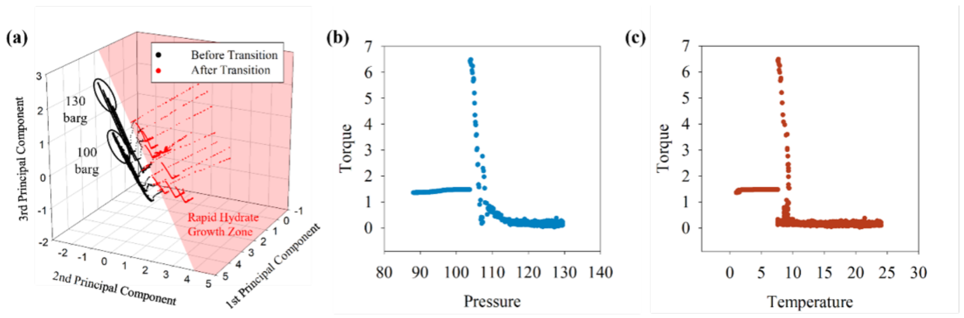


Fig. 3-9. (a) Reduced feature space of experiment data at 200rpm (1st to 8th batch) before transition (block dot) and after transition point (Red dot) of hydrate using 3 dimensional PCA results (7 features to 3 features). (b) relationship between pressure and to

As we observed in **section 2**, the hydrate transition and segregation trend could be identified from the relationship between hydrate fraction and relative torque. Hydrate fraction can be calculated from pressure and temperature by using **Eq. (3-2)-(3-3).** PCA results in **Fig. 3-9a.** show a similar trend with the relationship charts between pressure and relative torque or temperature and relative torque in **Fig. 3-9b.** and **Fig. 3-9c**. The results suggested that the PCA results can be used as an indicator of hydrate formation behavior since they can preserve the original relationships between the experimental values.

However, feature reduction itself is not time dependent. To keep the temporal information of PCA results, we use PCA data in time series format by adding a new column on pre-PCA data as shown in **Fig. 3-10**. The PCA results formatted with time series are used as an input for a deep learning model after the windowing process.

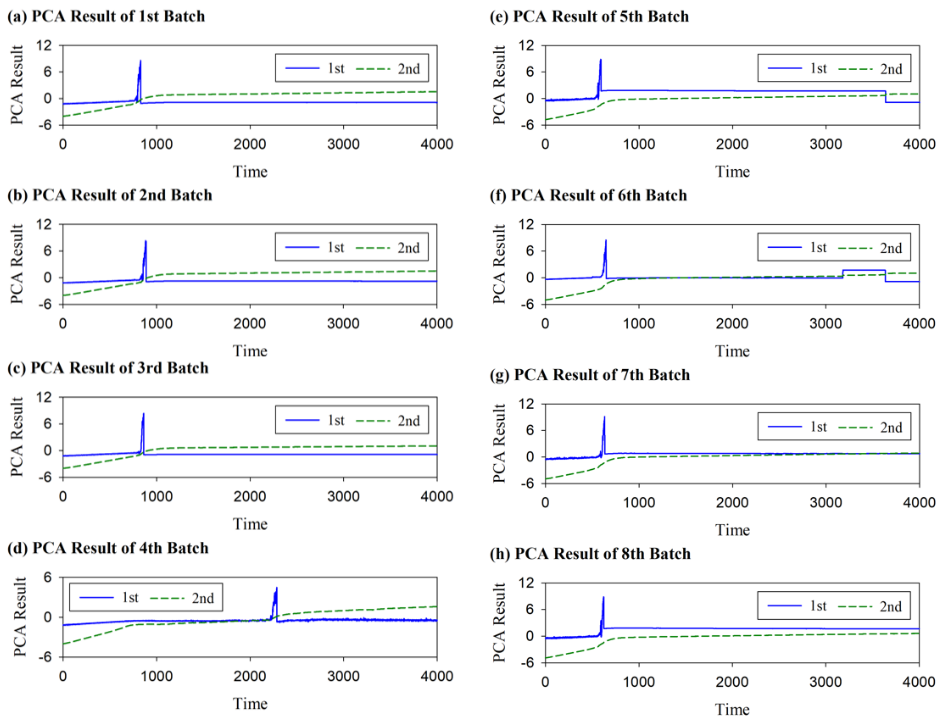


Fig. 3-10. 1st and 2nd Components of Principal Component Analysis

3.2. Prediction Results

To analyze the trend of hydrate formation behaviors with time-dependent, the model was trained with 7 experimental batches (batch no. 1 – 7 in **Table 3-3.**) and tested on the 8th batch (batch no. 8 in **Table 3-3.**). The 1st principal component from PCA was trained. Each model training was done for 20 epochs without callbacks and optimizer with Adamax. To evaluate the performance of prediction results, MSE and MAPE are adopted as loss metric and error metric, respectively, from the following equations:

$$MAE = \frac{1}{n} \cdot \sum_{i=1}^{n} \left| y_{observed} - y_{predicted} \right| \tag{3-9}$$

$$MAPE = 100 \cdot \frac{1}{n} \cdot \sum_{i=1}^{n} \frac{|y_{observed} - y_{predicted}|}{y_{observed}} \quad (3\text{-}10)$$

, where n is the length of time series, $y_{observed}$ is the observed or real value, and $y_{predicted}$ is the predicted value by the model. Typically, lesser MAPE shows better performance, since MAPE is negatively oriented values with a range of 0 to positive infinite. MAE value itself cannot be used as an absolute accuracy measure. Rather than MAE, MAPE is recommended in data with different ranges.

3.2.1. Effect of Transfer learning

To take the characteristics of individual experiment data into the continuous-time series prediction model, the transfer learning method has been used. The accuracy of the model under various experimental conditions was analyzed by training with experimental data for 1st to 7th batches at a mixing rate of 200 rpm, sequentially, then for 10th to 16th batches at a mixing rate of 600 rpm. Finally, the trained model was used for testing the performance for 8th batches at a mixing rate of 200 rpm.

MAPE was evaluated for every training batch by each model used to predict 1st principal component of PCA as shown in **Fig. 3-11**. MAPE observation shown in **Fig. 3-11.** And **Table 3-5.** shows that transfer learning is an effective method in training, as it reduces error as training proceeds. MAPE was decreased in the range of 30 to 83% depending on the model after completing to train the 7th batch compared to the 1st batch. There is a slight increase in MAPE after training the 6th batch due to some discrete points in the 6th batch data. Except for that, MAPE gradually decreased as the batch number increased, which suggested that transfer learning worked effectively for all deep learning models used in the study. When the models trained with the 600rpm data after the 200rpm data consecutively (training with 1st ~ 7th and 10th ~ 16th batch sequentially), the MAPE increased compared to training with only 200rpm data (training with 1st ~ 7th batch) as shown in **Fig. 3-11**. All models except for GRU-2 and ARLSTM-1 models show the negative values for the difference in MAPE between the 7th batch and 16th batch ($\frac{(B)-(D)}{(B)} \cdot 100$ in **Table 3-5.**). The results suggested that the prediction accuracy for the model could be higher when the model is trained with the data under constant experimental conditions, even if the amount of training data is small. Also, the slope of MAPE reduction after training 600rpm batches ($\frac{(C)-(D)}{(C)} \cdot 100$ in **Table 3-5.**) was less steep compared to using only 200rpm batches ($\frac{(A)-(B)}{(A)} \cdot 100$ in **Table 3-5.**). This is partly due to the general relation between the amount of training data and the amount of MAPE showing a logarithmic decrease. However, MAPE has decreased even in 600 rpm batches as the batch continued to transfer learning. The results suggest that putting more data contributes to lowering error, which implies more data is needed to train the model as universally applicable with higher accuracy.

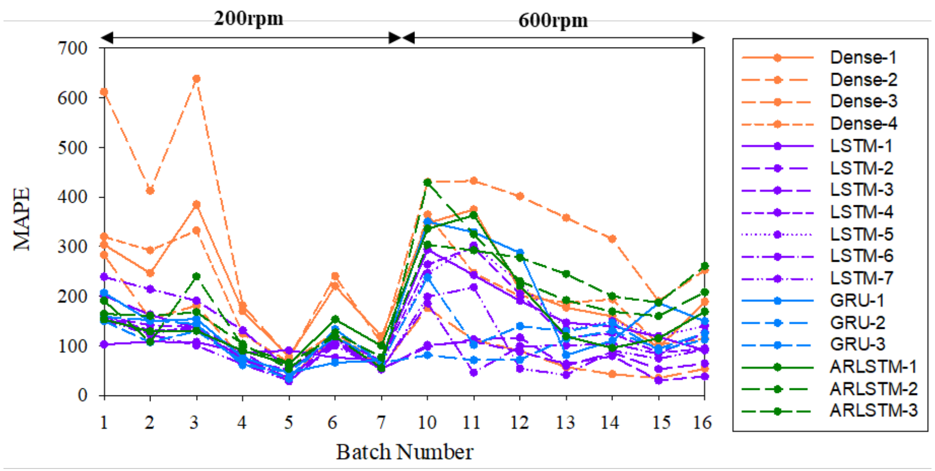


Fig. 3-11. Training loss for 1st principal component using the experiment batches at a mixing rate of 200 rpm and 600 rpm for various model architectures (Dense, LSTM, GRU, ARLSTM)

Table 3-5. MAPE by model calculated after training with 1st batch (A), after training batch 1 to batch 7 (B), after training 1st~7th and 10th batch (C), after training 1st~7th and 10th ~16th batch (D)

Model Name	Model No.	200rpm			200rpm+600rpm			MAPE Reduction after 600rpm
		1st Batch (A)	7th Batch (B)	MAPE Reduction $\frac{(A)-(B)}{(A)}\cdot 100$	10th Batch (C)	16th Batch (D)	MAPE Reduction $\frac{(C)-(D)}{(C)}\cdot 100$	$\frac{(B)-(D)}{(B)}\cdot 100$
Dense	1	304.2	119.4	60.7%	347.6	189.0	45.6%	-33.1%
	2	283.2	56.7	80.0%	176.6	53.7	69.6%	-15.0%
	3	612.2	105.1	82.8%	430.3	252.7	41.3%	-100.7%
	4	320.3	60.1	81.2%	364.7	114.9	68.5%	-18.6%
LSTM	1	103.0	71.9	30.2%	294.1	91.0	69.1%	56.3%
	2	201.5	52.4	74.0%	98.7	64.1	35.0%	-111.2%
	3	159.6	53.1	66.7%	101.5	38.1	62.4%	-6.9%
	4	159.4	70.0	56.1%	264.2	94.9	64.1%	12.5%

	5	155.4	63.6	59.1%	246.7	140.7	43.0%	-37.5%
	6	239.3	53.7	77.6%	185.8	126.9	31.7%	-144.6%
	7	151.9	52.7	65.3%	199.0	92.7	53.4%	-22.2%
GRU	1	207.0	70.7	65.8%	350.3	149.7	57.3%	-15.0%
	2	158.7	65.6	58.7%	81.4	125.4	-54.0%	208.5%
	3	149.9	67.5	54.9%	237.8	112.6	52.6%	-4.4%
ARLSTM	1	153.5	100.2	34.7%	336.2	168.9	49.8%	30.3%
	2	191.2	55.1	71.2%	429.1	208.4	51.4%	-38.4%
	3	164.5	77.0	53.2%	304.2	261.0	14.2%	-274.9%
Average				63.1%			44.4%	-30.3%

The trained prediction model was employed to locate the two major events during the hydrate formation, the transition point followed the segregation point. The experimental results of the test batch (no. 8) showed that transition point at 5830 second and a segregation point at 6250 second. The prediction results from the LSTM-1 model were presented with a graph shown in **Fig. 3-12.** for each case demonstrated in **Table 3-6.** for (a) window without transition nor segregation point, (b) window with transition point, and (c) window with segregation point. Like MAPE performance shown in **Fig. 3-11.**, the model trained with transfer learning (Case 2 and 3 in **Table 3-6.**) shows far better performance than that with only the 1st batch (Case 1 in **Table 3-6.**). In addition, transfer learning had an advantage for accuracy on repeated cycles compared to concatenated cases. The predicted results with transfer learning followed the actual value better than those with concatenated batch.

Table 3-6. Case description for time-series modeling

Case no.	**Description**
Case 1	Training only with 1st batch
Case 2	Training with 1st to 7th batch by transfer learning
Case 3	Training with 1st to 7th and 10th to 16th by transfer learning
Case 4	Training with 1st to 7th batch by concatenating
Case 5	Training with 1st to 7th and 10th to 16th by concatenating

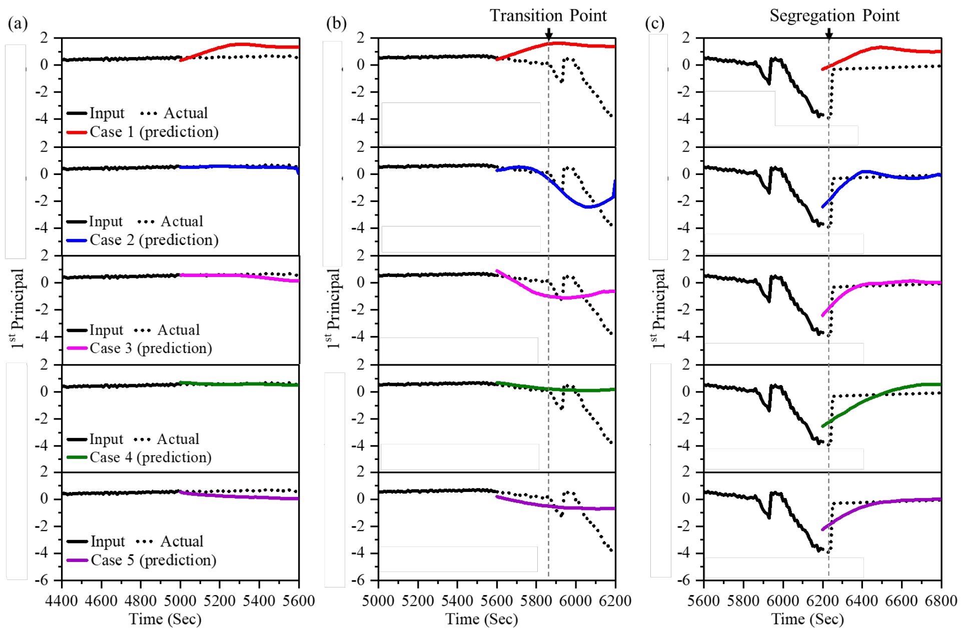


Fig. 3-12. Comparison of prediction results and true values for the test batch (No. 8) by LSTM-1 model with different cases of transfer learning. The prediction was shown at three points: (a) window without transition nor segregation point, (b) window including t ransition point, and (c) window including segregation point

These results suggested that the transfer learning method could improve the accuracy of prediction, especially on the irregular trend like transition and segregation, when the time-series data exists in an individual for each batch cycle.

The more the model is trained with many data batches, the more accurate model there will be. Further studies with a larger number of data set under various experimental condition would be beneficial to verify the scalability of the framework.

3.2.2. Effect of Deep Learning Model Configurations

In the previous section, the transfer learning method with the 1st to 7th batch (case 2 in **Table 3-6.**) showed the best performance for predicting the hydrate formation characteristics for testing the 8th batch by the LSTM-1 model. In this section, we apply the various deep learning model and configurations as shown in **Table 3-4.** To determine which model shows the best performance. **Table 3-7.** shows the forecasting results estimated with MAPE by each model for four window sections: (1) whole time series, (2) window without transition nor segregation point, (3) window with transition point, and (4) window with segregation point. Concerning MAPE results, LSTM and GRU models showed better performance compared with other models for the whole time series. In particular, the LSTM-2 model had the best performance with the lowest MAPE in the whole window.

However, the forecasting performance of each model showed a difference for each window section of hydrate formation. For windows except for the transition and segregation points (Section 1 in **Table 3-7.**), ARLSTM and GRU showed better results than other models. ARLSTM-3 showed the smallest MAPE. The model accuracy at the window of transition point (Section 3 in **Table 3-7.**) and segregation point (Section 4 in **Table 3-7.**) was dropped with higher MAPE compared to the data where the transition and segregation did not occur (Section 2 in **Table 3-7.**). For transition point, the Dense-2 model has the lowest MAPE, but GRU-based models show the best overall average MAPE results. ARLSTM and Dense models had a good performance on the prediction of segregation point with the lowest MAPE of the ARLSTM-2 model.

To investigate which models fit the best to predict the two characteristics: transition and segregation point, we calculated the averaged MAPE of window sections including transition and segregation point. With this, the GRU-3 model and ARLSTM-2 model show the best results and ARLSTM models show better results than other model types. Overall, each model has its best prediction points. This indicates that MAPE results alone cannot decide which model makes the best prediction. Also, the MAPE result only shows the difference between predictions and true values on the given window, thus it is not likely to determine if the models predict the right trend. This needs another investigation by trend-prediction rather than MAPE only. In the rightmost column in **Table 3-7.**, Dense-2, LSTM-3, GRU-3, and ARLSTM-2 show the best prediction result in each model type, respectively.

Table 3-7. MAPE is calculated by model on the whole window (1), a window without transition nor segregation point (2), a window including transition point (3), and the window including segregation point (4).

Model Name	Model No.	Whole Window (1)	Window w/o Transition & Segregation Point (2)	Window Including Transition Point (3)	Window Including Segregation Point (4)	Average of (3) and (4)
Dense	1	119.4	26.1	95.6	73.4	84.5
	2	56.7	102.2	**80.4**	85.2	**82.8**
	3	105.1	15.7	136.7	44.4	90.5
	4	60.1	19.1	177.0	81.1	129.0
LSTM	1	71.9	12.0	359.9	117.2	238.5
	2	**52.4**	37.1	110.4	89.8	100.1
	3	53.1	51.4	96.5	68.5	**82.5**
	4	70.0	43.7	101.4	77.4	89.4
	5	63.6	83.3	98.5	149.1	123.8
	6	53.7	16.8	173.3	134.9	154.1

	7	52.7	35.0	178.0	82.2	130.1
GRU	1	70.7	6.2	122.8	65.9	94.3
	2	65.6	35.4	139.9	131.0	135.5
	3	67.5	55.7	99.5	**36.1**	**67.8**
ARLSTM	1	100.2	18.9	161.2	70.8	116.0
	2	55.1	7.3	119.9	36.0	**77.9**
	3	77.0	**5.2**	112.6	53.1	82.8

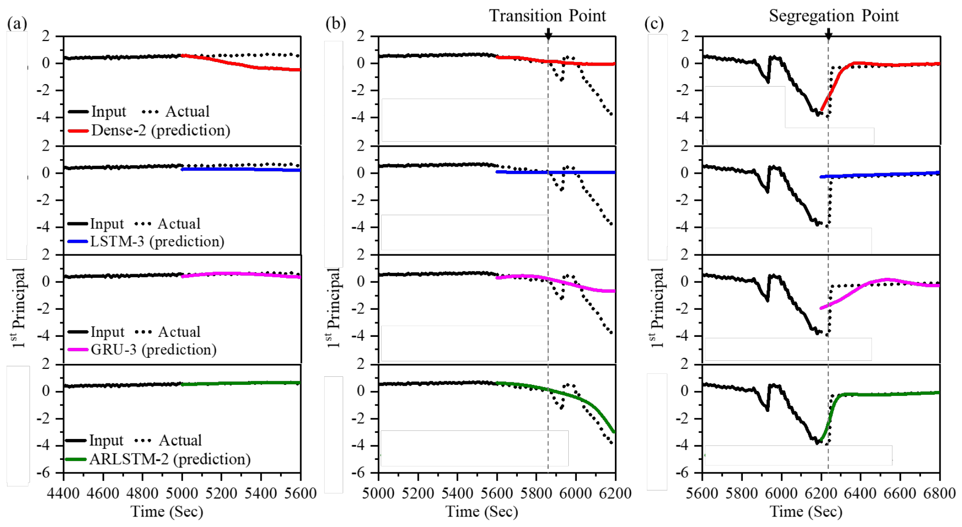


Fig. 3-13. Comparison of prediction results and true values for the test batch (No. 8) by each model; (a) Dense-2, (b) LSTM-3, (c) GRU-3, and (d) ARLSTM-2 model. Each model has shown in 3 points were (1) window without transition nor segregation point (2) window including transition point (3) window including segregation point.

Fig. 3-13. graphically expresses the results of the true and prediction values by each model type during the hydrate formation for trend evaluation. As above, the results were presented for three windows: (a) before the transition of hydrate particles, (b) when the transition occurs, and (c) when segregation occurs after the transition. The model used in **Fig. 3-13.** was chosen from **Table 3-7.**, which showed the best MAPE result on each model type (Dense, LSTM, GRU, ARLSTM). Noted that the results from other configurations for each model type are provided in **Fig. 3-14.** to **3-17.**

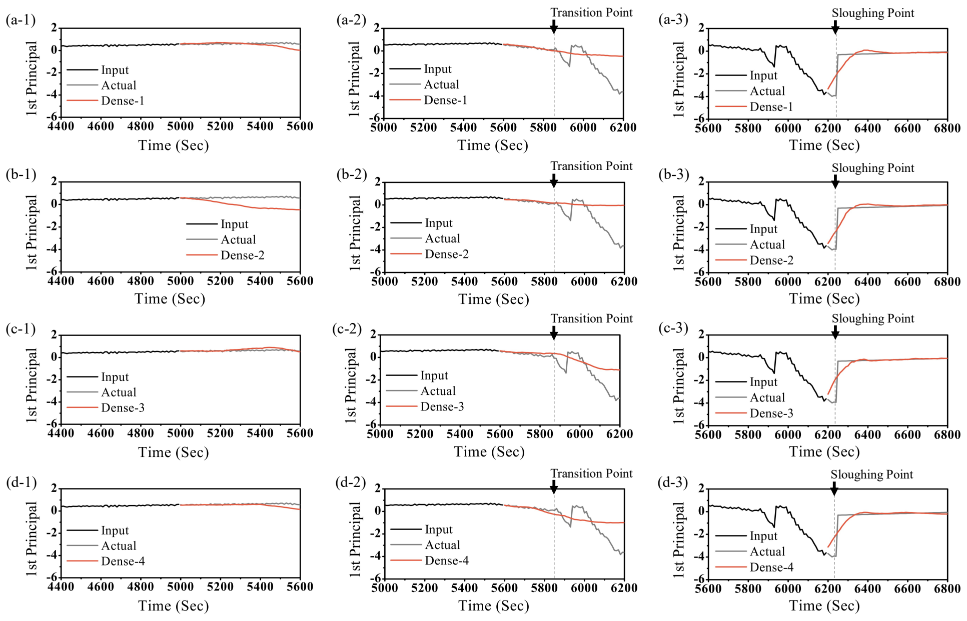


Fig. 3-14. Comparison of prediction results and true values for the test batch (No. 8) on Dense-based model

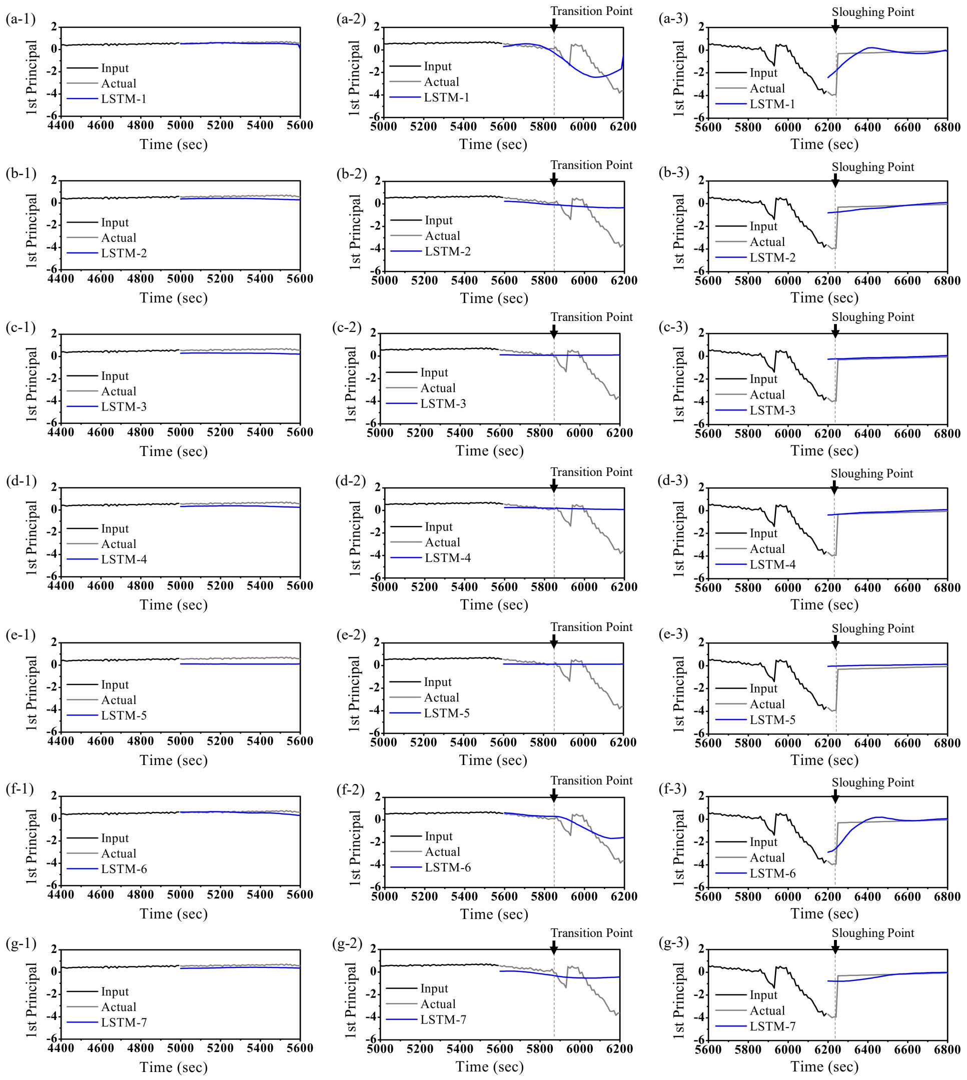


Fig. 3-15. Comparison of prediction results and true values for the test batch (No. 8) on LSTM-based model.

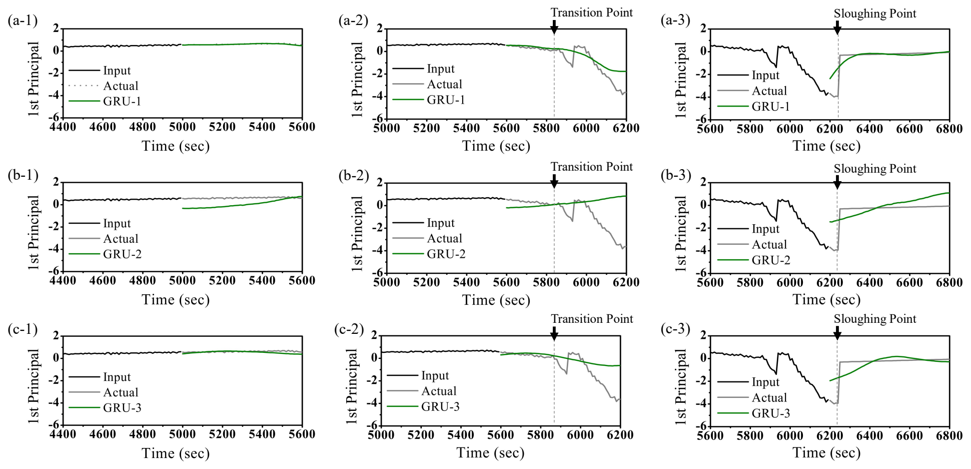


Fig. 3-16. Comparison of prediction results and true values for the test batch (No. 8) on GRU-based model

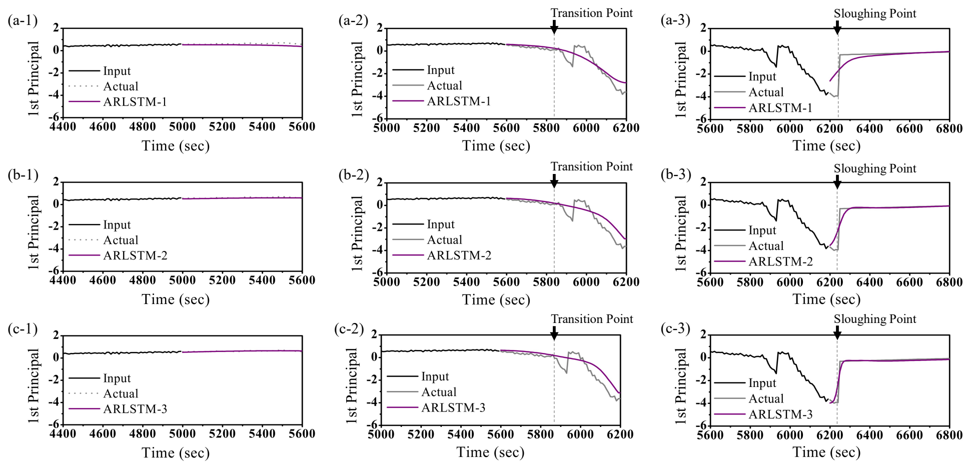


Fig. 3-17. Comparison of prediction results and true values for the test batch (No. 8) on ARLSTM-based model

The prediction by ARLSTM-2 followed the trend well with experimental results for three windows rather than other models as shown in **Fig. 3-12**. However, the results predicted by the Dense-2 model did not show good agreement with actual values even before the transition point. Even though the results followed the trend at the transition and segregation point, their magnitude and timing were not precise. Prediction results from other dense models show a similar tendency in **Fig. 3-13** The prediction with the LSTM-3 model shows poor performance compared to other model types. LSTM-3 model does not predict the trend of the transition point. Even though other LSTM models are not that good for predicting transition trends, LSTM-1 and LSTM-6 followed the trend well (see **Fig. 3-14.)**. Compared to the LSTM-based model, the GRU-based model predicts the transition trend and segregation trends but is not good at predicting the precise magnitude. GRU-2 had a particularly inaccurate performance of prediction as shown in **Fig. 3-15**.

Table 3-8. Prediction result and amount of error on transition point and segregation point for test batch (No. 8) of each model

		Transition Point		**Segregation Point**	
		Predicted Time (sec)	**Error (sec)**	**Predicted Time (sec)**	**Error (sec)**
Dense	1	5820	-10	6220	30
	2	5330	-500	6220	30
	3	5340	-490	6220	30
	4	5780	-50	6220	30
LSTM	1	5840	10	6200	50
	2	5790	-40	6200	50
	3	5660	-170	6200	50
	4	6100	270	6200	50
	5	N/A*	N/A*	6200	50
	6	5910	80	6200	50
	7	5600	-230	6200	50
GRU	1	5910	80	6200	50
	2	5600	-230	6200	50
	3	5920	90	6200	50
ARLSTM	1	5880	50	6200	50
	2	5870	40	6210	40
	3	5900	70	6220	30

* It did not show the particular point for prediction.

Table 3-8. shows the prediction result of the transition point and segregation point for the test batch (no. 8) of each model. The transition point (time: 5830sec) was defined as the first point to reach below the original 1st principal value of the actual transition point. Segregation point (time: 6190sec) was defined as the abrupt slope change from negative to positive. This shows how well the models predict each point. Dense and LSTM models show high variance between models, but ARLSTM models show consistency between models and their errors are low, though not best, compared to other model types.

Overall, ARLSTM-based models showed the best performance in the prediction of timing and trend when transition and segregation occurred (see **Fig. 3-18.** and **Table 3-8.)**. Even though ARLSTM-2 did not exactly detect the 1st trough (the initial part of phase transition), the model predicted the overall trend of the phase transition section better than the other types. In particular, the prediction results at segregation points followed the true value with a similar trend and magnitude in all ARLSTM-based models. In ARLSTM-based models, large cell numbers (64 and 128) with ARLSTM-2 and the ARLSTM-3 model showed better results than small cell numbers (32) with ARLSTM-1. Through the evaluation of MAPE and trend prediction, ARLSTM-based models show the best performance for the prediction of hydrate formation behavior, followed by GRU-based models. Dense-based models and LSTM-based models showed nonuniform performance depending on their configuration. Also, from the comparison of the several dropout rates, the effective dropout rate was showing the range of 0.2~0.6.

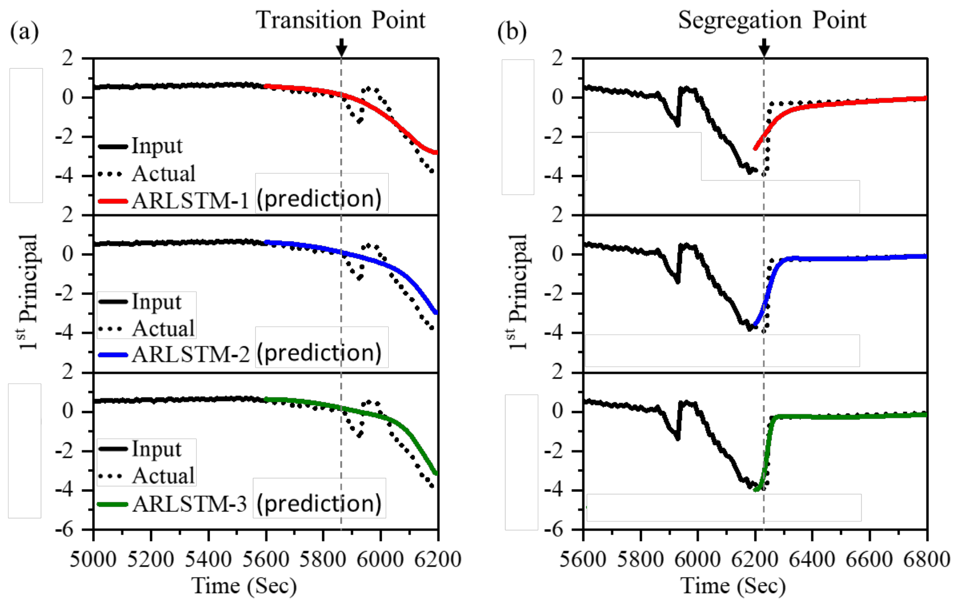


Fig. 3-18. Comparison of prediction results and actual values for the test batch (No. 8) on ARLSTM-based model at window including transition point (a) and segregation point (b)

Using the combination of PCA and deep learning models with the transfer learning method, this framework can be used more universally in a system that repeatedly gathers many sensor data for specific periods. This approach automatically clusters the hydrate risk of the given system using PCA and trains the prediction neural network to be fitted with the given system. This work suggests, for the first time, a novel framework based on the combination of PCA technique and deep learning model to demonstrate the hydrate formation characteristics. The developed model predicted the time-series hydrate formation behavior for phase transition and hydrate segregation points by using the pressure, temperature, and relative torque data. This would provide the possibility to detect the hydrate formation and plugging risks in the early stage with reliable ways for the safe operation of subsea flowlines. Further studies would be considered to predict the probability and the amount of remaining time to the point of hydrate blockage by using the frameworks. Furthermore, the data-driven deep learning method can be integrated into the governing physics to improve the limitation on generalization, which is called as a Physics Based Neural Network (PBNN) (Ren et al., 2020). Further studies using PBNN with time-series prediction would be beneficial to improve the performance of deep learning into flow assurance applications.

3.5. Conclusions

Prediction of hydrate formation characteristics with real-time data is important to safely operate the subsea flowlines. However, due to uncertain relations between measurable data and hydrate formation behavior, it was difficult to develop an accurate prediction method. With significant advancements in time series analysis and prediction models using deep learning, predicting the uncertain future trend was possible. In this work, we propose a novel data-driven framework to assess and predict the hydrate formation behavior using various deep learning models, especially the RNN family. The obtained results showed the models can predict the transition and segregation points well. Here is the comparison between the models.

1) PCA on many data can cluster the points by before/after transition point, which indicates the sudden and rapid hydrate growth. Also, PCA reduces the number of features, which makes training deep learning models more efficient.
2) Windowing was done on given time-series data to make a real-time prediction model which gets input from the near past.
3) Prediction using deep learning models (Dense, LSTM, GRU, ARLSTM) shows reasonable results on prediction transition and segregation points. Among the models, ARLSTM shows the best results. For layer number, the single layer shows larger MAPE than stacked layers. The dropout rate between 0.2~0.6 showed significant improvement in accuracy.
4) Using dataset for deep learning model, training under similar experimental conditions (training until 7th batch; only with 200rpm data) shows the best result. The data under different experimental conditions (training with

1~7 and 9~16th batch; adding 600rpm data after 200rpm data) increases MAPE but the MAPE is decreasing as training the model with multiple batches. Enough amount data is used in the training, the better the prediction results become.

3.6. Acknowledgement

This chapter is partially adapted from Time series prediction of hydrate dynamics on flow assurance using PCA and Recurrent neural networks with iterative transfer learning in *Chemical Engineering Science* with authors Lee, N., Kim, H., Jung, J., Park, K. H., Linga, P., & Seo, Y. (https://doi.org/10.1016/j.ces.2022.118111).

Chapter 4. Prognosis on System

4.1. Introduction

Prognosis is a important goal in the domain of process monitoring, yet no definitive definition or the framework has been given. It is to detect a fault before it occurs and find out its root cause to prevent in advance. Fault prognosis in chemical processes has been proposed by the several literature from **[85, 86]** to recent model using combined approach of hidden Markov and Bayesian network model **[87]**, but the framework is not yet has been standardized as the research is still in its early stage, and has only been centered around in each separated subcategory; fault detection, prediction in fault case, and root cause diagnosis. Until now, numerous studies have attempted to find and explore the new algorithms. However, in order to use this effectively, prediction is needed to find when and what faults will occur and a diagnostic that determines what causes them to occur are needed together. From the perspective of maintenance, if the control or maintenance point of a system can be predicted and informed in advance, it is called prognosis, which is the most efficient among various maintenance techniques. And ultimate purpose of fault detection is to build a system that enables prognosis within a process system, which requires fault prediction and diagnostics.

4.1.1. Deep learning based fault detection

In domain of fault detection using deep learning, there are 3 major subsets on its techniques **[88]**. (1) deep learning of feature extraction, (2) learning feature representations of normality, (3) end-to-end anomaly score learning. On the second category, learning feature representations of normality, includes the generic normality feature learning and anomaly measure-dependent feature learning. Generic normality feature learning is the methods which gaining its popularity nowadays, which includes the autoencoders **[89]-[91]**, generative adversarial networks **[92]**, predictability modeling **[93]**, and self-supervised classification **[94]**.

4.1.2. Deep learning based propagation prediction

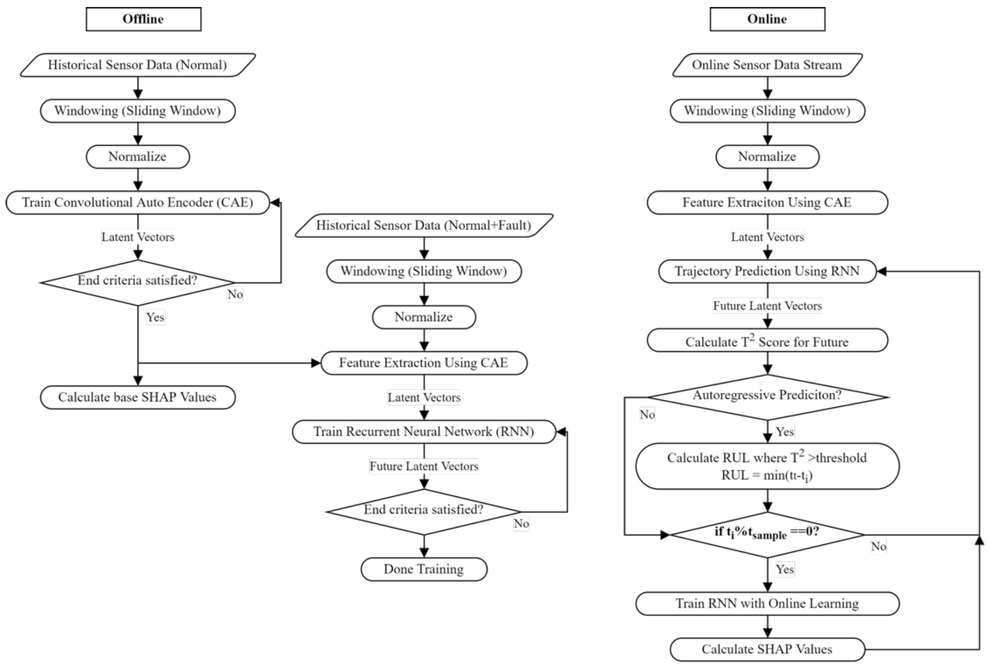
Machine learning based time series prediction methods has been successfully applied in many domains related to time series, including sensor networks. Time series prediction requires the ability to effectively handle the complex and innate features of temporal relationships and since the machine learning models have superiority in their ability to process big data with high dimensionality and representability**[95]**. Machine learning based time series prediction method is divided into 2 major subsets. One is discriminative and the other is generative. Discriminative prediction methods learn to act from the statistics from the observed data, where the methods like Support Vector Machine (SVM), Shallow neural network (In the category of classical machine learning), Convolutional Neural Network (CNN), Long-short Term Memory (LSTM), Auto-Encoder (AE), and Deep Stacking Net (DSN) (In the category of deep learning) are included. The generative prediction method considers the joint probability distribution of both observed data and target data, where the Gaussian Process (GP), Bayesian Network (BN) & Hidden Markov Model (HMM) (In the category of classical machine learning), Restricted Boltzmann Machine (RBM), Deep Belief Network (DBN), and Generative Adversarial Networks (GANs) (In the category of deep learning) are included. Other than these two major categories, clustering-based models and hybrid deep learning models are also present. The prediction model used for the prognosis for chemical process are widely studied. Some used AE, HMM **[96]**. Here, we are using the LSTM model to effectively use the autoregressive trajectory propagation prediction.

4.1.3. Deep learning based diagnosis

For fault diagnosis, we employed eXplainable Artificial Intelligence (XAI) method. Unlike fault detection and propagation prediction, XAI methods are not a standalone model but the algorithms for the given model. Here, we apply the XAI method to the fault detection and propagation prediction model to perform the fault diagnosis to identify the cause of the problem. XAI methods can be categorized by the scope, methodology and usage. By its scope, there are 3 major subsets : (1) the local, (2) global or (3) hybrid methodologies. The local explanation method calculates the contribution of input features to output in the training stage. Activation maximization, saliency map, Layer-wise Relevance BackPropagation (LRP), Local Interpretable Model-agnostic Explanations (LIME), and Shapley Additive exPlanations (SHAP) **[97]** is included. The global explanation method is about the set of decision-making rules applied by the model to analyze the global behavior of an AI model on all input variables, not each of them. Global surrogate models, Class model visualization, LIME for Global Explanations, Concept Activation Vectors (CAVs), Spectral Relevance Analysis (SpRAy), Global attribution mapping, and Neural Additive Models (NAMs) are included in this category. The hybrid XAI explanation combines the above-mentioned approaches.

Here, we are using SHAP, which is the local and model agnostic XAI method, on the deep learning model. Model is made up of two different deep learning models : one is performing the fault detection and the other performing the prediction. There are a few literatures with SHAP employed on a time series prediction model, however, VegaGarcia et al. **[98]** used DeepSHAP-based method to explain the predictions of time-series signals involving Long Short-Term Memory (LSTM) networks. Explanations were generated for each time step of each input instances.

This study introduces the novel deep learning based prognosis scheme including 3 major parts : fault detection, propagation prediction and root cause diagnosis. This study is distinguished from previous studies as the framework is 3 main parts are fully deep-learning based, and the prediction technique employs novel methodology to enhance the prediction accuracy under the unprecedented situation and to predict the RUL until the threshold.



Offline
Historical Sensor Data (Normal)
Windowing (Sliding Window)
Normalize
Train Convolutional Auto Encoder (CAE)
Latent Vectors
End criteria satisfied?
No
Yes
Calculate base SHAP Values
Historical Sensor Data (Normal+Fault)
Windowing (Sliding Window)
Normalize
Feature Extraction Using CAE
Latent Vectors
Train Recurrent Neural Network (RNN)
Future Latent Vectors
End criteria satisfied?
No
Done Training
Online
Online Sensor Data Stream
Windowing (Sliding Window)
Normalize
Feature Extraciton Using CAE
Latent Vectors
Trajectory Prediction Using RNN
Future Latent Vectors
Calculate T² Score for Future
Autoregressive Prediciton?
No
Yes
Calculate RUL where T² >threshold
RUL = min(tt-ti)
if ti%tsample ==0?
No
Yes
Train RNN with Online Learning
Calculate SHAP Values

Fig. 4-1. Schematic of fault prognosis suggested in this paper

4.2. Process prognosis framework

As it is a combination of many algorithms, it is important to connect them with the flow. Also, in the case of this study, the algorithms used were closely connected, so the framework configuration was an important part of the study. In this study, we first train the fault detection model and fix the encoder and decoder models. After that, the latent space between the encoder-decoder used for fault detection was used, and this was considered as a reduced feature dimension and used as a preprocessing step for fault prediction. T^2index value was calculated using this predicted value, and fault diagnosis was performed through contribution analysis on variables that affect the change of this index. In addition, windowing, a method of dividing the time series into pieces, was introduced to perform prediction in real time, and the code of the existing algorithm was changed and applied to apply this windowing to fault detection and diagnosis. Also, attention was paid to the interpretation of the result value as the format of input and output was changed by windowing.

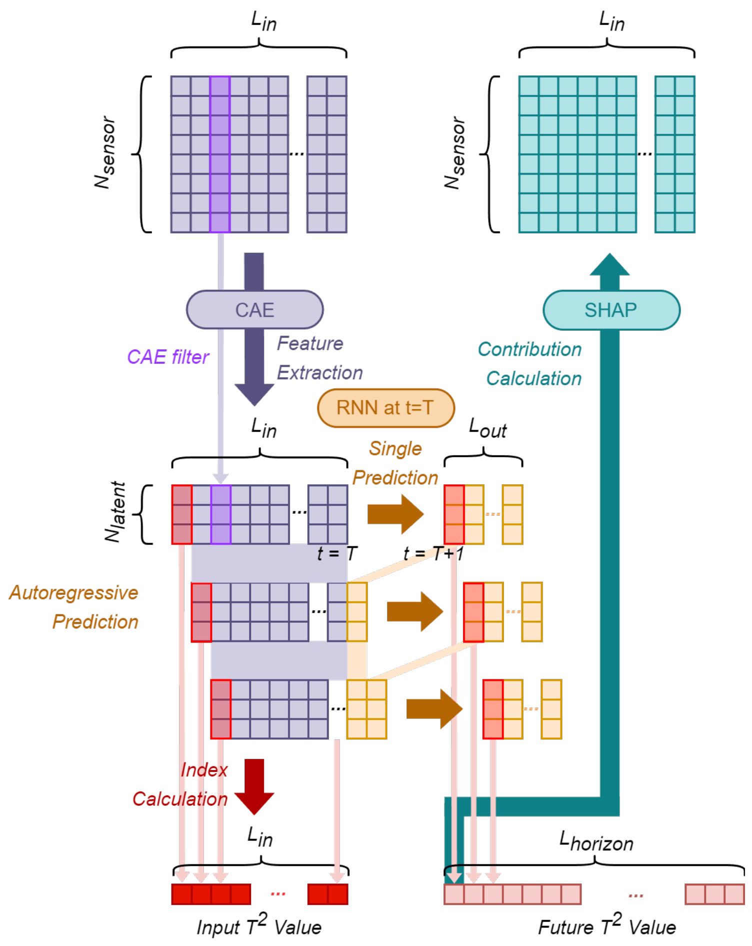


Fig. 4-2. Auto regressive prediction on current fault for calculating RUL

4.2.1. Offline modeling

4.2.1.1. Feature extraction using convolutional autoencoder

The raw data obtained from the chemical process is first processed with a sliding window with a fixed length L, followed by a normalization between 0 and 1. The data is transformed into matrices of the size $L \times 1 \times n$, where n represents the process variable number.

The model used for feature extraction is CAE. We employ CAE to deal with multivariate time-series data. It should be noted that the point-wise convolution which uses 1×1 kernel is conducted to only extract information across channels without destroying the features in time series. Although it varies from the autoencoder in that it uses a convolutional layer instead of a dense layer, the dimension reduction operation itself is the same because the kernel size is 1. The encoder part of CAE map the input to a hidden representation by the nonlinear transformation. Then, the hidden representation is reconstructed to the output through the decoder part. The model is trained by optimizing the mean squared error between input and reconstructed output.

4.2.1.2. Prediction using RNN

Here, RNN model combined with the online machine learning technique was used for prediction. Long-Short Term Memory (LSTM), is one of the most popular model amongst the RNN family. LSTM model has operations inside the cell to keep cell states and controlled by three gates : input, output, forget. By this the model can selectively take past information into account. This is expressed as :

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \underline{\mathcal{C}}_t &= \tanh W_c \cdot [h_{t-1}, x_t] + b_c \\ \mathcal{C}_t &= f_t * \mathcal{C}_{t-1} + i_t * \underline{\mathcal{C}}_t \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * \tanh \mathcal{C}_t \end{aligned} \qquad (4\text{-}1)$$

, where h_t is hidden state at time t, h_{t-1} is hidden state at time $t-1$ or the initial hidden state at time o, c_t is cell state at time t, x_t is input at time t, i_t is input gate at time t, f_t is forget gate at time t, $\underline{\mathcal{C}_t}$is cell gate at time t, o_t output state at time t, and $*$ is the Hadamard product, σ is the sigmoid function.

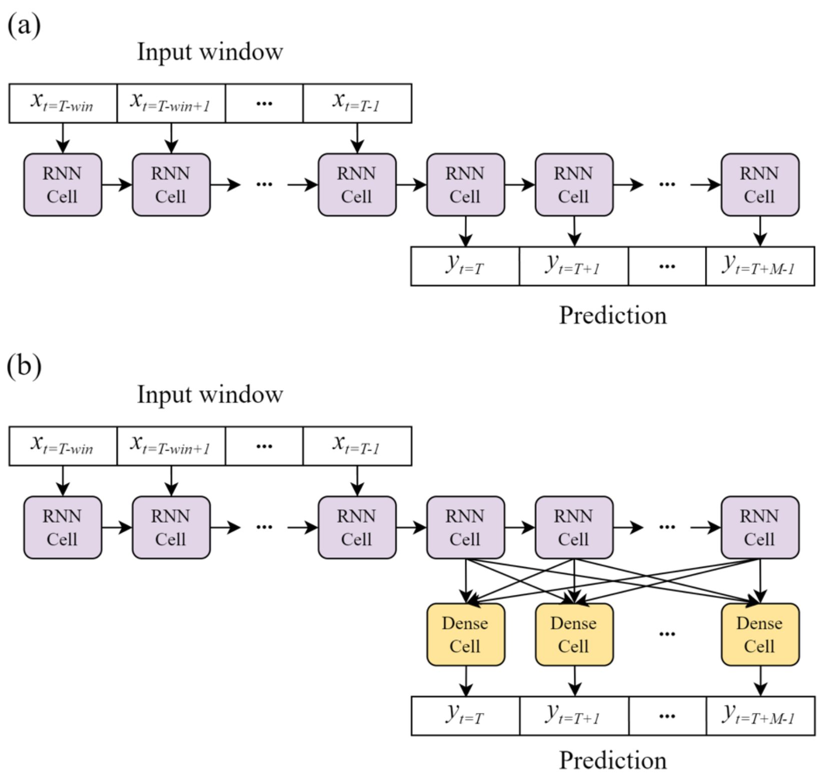


Fig. 4-3. Configuration of RNN cells

LSTM is widely studied model and proved its usefulness **[99]**. But there are not many research has been done on LSTMs with online learning. However, the combination of LSTM and online learning is very essential for fault prediction since It is difficult to detect a fault condition promptly using unsupervised data, and Predicting the fault condition using offline risk calculation based

on a normal dataset is not accurate **[100]**. Unlike general prediction, process fault prediction must predict propagation that has never been made before. This is a 'abnormal' situation where abnormal situations that occur in the process are literally not encountered before, and fault prediction must make predictions in these abnormal situations. For this reason, the accuracy is very poor if the existing recurrent neural network is used as it is.

Fault prediction is made for the latent vector obtained after passing through the encoder used in the fault detection step. These values, which are feature redux, can increase accuracy and reduce time. The latent vector is put as an input to the RNN model so that learning is performed. This paper attempted to verify using various models through mixing and matching with various models and various types of current natural network models. The training data used in the prediction model contains some of the fault data as well as pure normal data, unlike the training data in the fault detection model. Combination of CAE and LSTM is a bit similar to the of CNN-LSTM, which combines the CNN and LSTM shown in **[100]**, however, the training process is different since this study treats CAE and LSTM as the different model and train with different data.

4.2.1.3. Monitoring statistics

The indicator for process monitoring is calculated using the future latent vector predicted by LSTM predictor. The Hotelling's T^2 statistic **[101]** is a way of measuring the variation captured in the latent vector, and it is expressed as:

$$T^2 = z^T \Delta^{-1} z \tag{4-2}$$

where Δ is the covariance matrix of the latent vector z. A fault is detected when the T^2 exceeds a specific threshold, where the deviation from the normal state is observed. The threshold is determined through a kernel density estimation of T^2 with a given confidence level α. It should be noted that, in order to consider the accumulated error while passing through the LSTM predictor, the future normal latent vector which is not obtained from CAE but from LSTM is used for calculating the threshold.

4.2.2. Online monitoring

4.2.2.1. Online learning

For real-time prediction, N inputs were received and M prediction results were presented as outputs. This is to make it available in real time in the actual process. Based on this, the methods of fault detection and fault diagnostics were also changed.

After completing the fault prediction, the T^2 value was calculated using each latent variable. That is, since it is possible to calculate the future T^2value, it is possible to predict whether or not a fault will occur in the future.

Online learning means training a system by iteration of small amounts of data sequentially. Examples of applying online learning to deep learning include the case of changing the cell structure itself **[102]** and the case of using batch learning **[103]**.

It is suitable for systems that receive data continuously and have to adapt itself to rapid changes, so it can compensate for the lack of data at the time of process failure by allowing the prediction model in the process system to run online. In general, if only online learning is enabled, system performance can degrade when bad data is injected into the system. To compensate for this, the framework was configured so that it could learn using existing historical data and online learning using transfer learning while monitoring was performed. As a result of comparing several cases, the shorter the update cycle and the longer the sample returned at once, the shorter the learning time was. The time required for learning is expected to be shortened if the simulation is performed in a better environment.

4.2.2.2. RUL estimation using autoregressive trajectory prediction

RUL is an amount of time left for which a unit under test is usable **[104]**. This is an important index for calculating the lifespan of equipment and for maintenance of process plants and prevention of faults. RUL has mainly been studied in relation to mechanical equipment as shown in **[105]-[107]** rather than chemical processes. So There is a difficulty in RUL prediction in chemical processes can be viewed as the time it takes to reach a specific fault condition rather than mechanical failure. Since chemical process fault is a broad concept that includes mechanical and process condition fault, it is hard to adopt the physical model formulation as shown in literature. In this study, the RUL prediction has been made based on the auto regressive trajectory prediction of health indicator, which has been predicted by the trained RNN model of the given time. At each time window, RNN model can predict far future using autoregressive iteration, which done by putting output of prediction result as a input of next prediction. Iterative point prediction has made for the trajectory prediction. Autoregressive behavior means that the each output depends on previous observations as below :

$$p(\mathbf{x}) = \prod_{i=1}^{d} p(x_i | \mathbf{x}_{<i}) \tag{4-3}$$

where $p(\mathbf{x})$ is a density function of training samples $\mathbf{x}$. Since the LSTM model can be described as the autoregressive model perspective **[108]**, autoregressive prediction using LSTM based prediction model satisfies the innate accordance.

With using the prediction model at $t = i$ for prediction of point $t = i + 1$, the trajectory $\phi(i + 1|i)$ denotes the trajectory of the T^2 value. RUL $r(i)$ at $t = i$ is

$$\begin{gathered} \boldsymbol{r}(i) = t_j - t_i \\ j = \max_{z}(h(z) \leq 0, z > i) \end{gathered} \tag{4-4}$$

, where z is point of failure (=End Of Life (EOL), a time instance when the prediction crosses the failure threshold), $h(z)$is health indicator, which in here is same as $\phi(z)$.

Algorithm 1 Autoregressive propagation and RUL estimation

```
0: for Every time step $\mathbf{i} > 60$ do
0:     Predict future latent vectors for $t = i + 1$ with the input time length of $t = i - 59, ..., i$ with model
0:     for Every time step from i to end j do
0:         Make next input latent vector with given vectors of $t = j - 58, ..., j$ and $t = j + 1$
0:         Predict future latent vector for $t = j + 2$ with the input time length of $t = j - 58, ..., j + 1$ with model
0:         Calculate $T^2$
0:         if $T^2$ Value of predicted future latent vector for $t = j + 2 >$ Threshold then
0:             Failure Time $\leftarrow t = j + 2$
0:             RUL $\leftarrow t = j + 2 - i$
0:         end if
0:     end for
0:     Calculate $m = \mathbf{i}/n_{sample}$
0:     if $\mathbf{i}/n_{sample} == 0$ then
0:         Append input of $t = \mathbf{i} - \mathbf{n_{sample}}, ..., \mathbf{i}$ as model training input
0:         Using list of above, train model using transfer learning and get new model $\mathbf{M}(\mathbf{i})$
0:         Save weights of trained model
0:     end if
0: end for=0
```

4.2.2.3. Fault diagnosis using future T^2

Based on the future T^2 value, the contribution of input sequence of sensors to the future T^2 was calculated using the SHAP technique **[97]**. The SHAP technique used in diagnosis is a model agnostic method among XAI techniques. There are several sub-techniques in SHAP, such as treeSHAP and deepSHAP kernelSHAP, and techniques are increasingly being added. Among them, the method used in this study is kernelSHAP, which can be used for all models with defined input/output, unlike treeSHAP or deepSHAP, which are greatly affected by the type of model. In this study, the feature extractor and predictor built above are the parts that need explanation through SHAP for fault diagnosis.

Algorithm 2 KernelSHAP Algorithm

Input: classifier $\mathbf{f}$, input sample $\mathbf{x}$
Output: explainable coefficients from the linear model

```
0: $\mathbf{z_k} \leftarrow$ SampleByRemovingFeature($\mathbf{x}$)
0: $\mathbf{z_k} \leftarrow \mathbf{h_x}(\mathbf{z_k})$
0: $\mathbf{y_k} \leftarrow \mathbf{f}(\mathbf{z_k})$
0: $\mathbf{W_x} \leftarrow \mathbf{SHAP}(\mathbf{f}, \mathbf{z_k}, \mathbf{y_k})$
0: LinearModel($\mathbf{W_x}$).fit()
0: Return LinearModel.coefficients() =0
```

Algorithm of KernelSHAP is shown in **Algorithm2**. Using SHAP kernels, KernelSHAP removes features from the input data and linearizing the model influence to randomly sample coalitions.

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z_j' \tag{4-5}$$

where g as the explanation model of an ML model f, z_j' as the coalition vector, M the maximum coalition size, and ϕ_j the feature attribution for feature j, $g(z_0)$ is the sum of bias and individual feature contributions.

4.3. Case study

To prove the validity and effectiveness of the proposed prognosis methodology, two widely used simulation cases, the Continuous Stirred-Tank Reactor (CSTR) process **[109]** and the Tennessee Eastmann (TE) process **[110]**, were used.

4.3.1. Target system

4.3.1.1. CSTR Process

CSTR is a vessel in which reactants and solvents flow into the reactor while the reaction product flows which shown in **Fig. 4-4**. Simulation of the CSTR has been introduced in **[109]**. Simulations of normal and faulty data were generated every 60 min for 20 h of operation under varying conditions. There are 8 process variables : inlet flow rate, tank volume, jacket volume, heat of reaction, heat transfer coefficient, pre-exponential factor to k, activation energy, fluid density, and fluid heat capacity. The sampling interval for all variables was 1 min. There were 10 incipient faults, as listed in **Table 4-2**. These faults were initiated from the 200th sample.

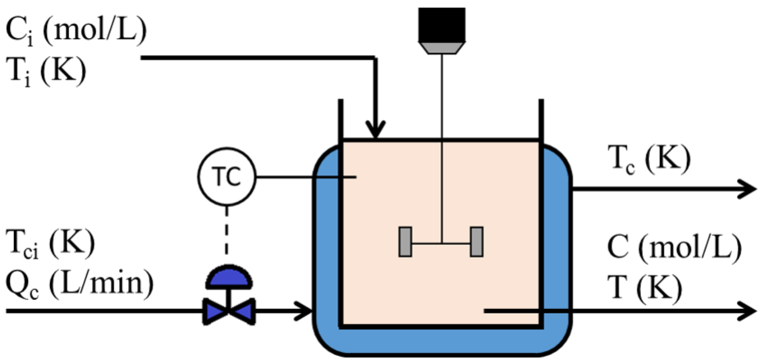


Fig. 4-4. Configuration of the CSTR simulation

Table 4-1. Constant values in the CSTR model

Parameter	Description	Value	Units
Q	Inlet flow rate	100.0	L/min
V	Tank volume	150.0	L
V_c	Jacket volume	10.0	L
ΔH_r	Heat of reaction	-2.0×10^5	cal/mol

$\boldsymbol{UA}$	Heat transfer coefficient	7.0×10^5	cal/min/K
$\boldsymbol{k_0}$	Pre-exponential factor to k	7.2×10^{10}	min-1
$\boldsymbol{E/R}$	Activation energy	1.0×10^4	K
$\boldsymbol{\rho, \rho_c}$	Fluid density	1000	g/L
$\boldsymbol{C_p, C_{pc}}$	Fluid heat capacity	1.0	cal/g/K

Table 4-2. Incipient fault scenarios in the CSTR

Fault ID	Description	Rate of fault progression	Type
1	$a = a_0 \, exp\,(-\delta t)$	0.0005	Multiplicative
2	$b = b_0 \, exp\,(-\delta t)$	0.001	Multiplicative
3	Simultaneous Faults 1 and 2	-	Multiplicative
4	$C_i = C_{i,0} + \delta t$	0.001	Additive
5	$T_i = T_{i,0} + \delta t$	0.05	Additive
6	$T_{ci} = T_{ci,0} + \delta t$	0.05	Additive
7	$C = C_0 + \delta t$	0.001	Additive
8	$T = T_0 + \delta t$	0.05	Additive
9	$T_c = T_{c,0} + \delta t$	0.05	Additive
10	$Q_c = Q_{c,0} + \delta t$	-0.1	Additive

4.3.1.2. TE process

The TE process contains five major unit operations: reactor, condenser, separator, compressor, and stripper. There are 52 variables, including 22 process variables and 19 composition variables, i.e., X1-X41, and 11 manipulated variables, i.e., X42-X52, as described in **Table 4-3**.. And the time length of the cases is 1080 samples. The TE process contains 1 normal case and 21 fault cases, as described in **Table 4-4**. In test set A, faults have 480 samples, and in the test set, B faults had 960 samples. And for all the fault cases, faults were initiated from the 160th sample. This means that even in the fault set, samples before the 160th sample are under normal conditions.

TE process has total of 8 components, 4 of which are reactants (A, C, D, E) and 2 are products (G, H) and 1 is byproduct (F). Reactions are irreversible and exothermic. The reactions are shown as :

$$\begin{gathered} A\,(g) + C(g) + D(g) \rightarrow G(liq) \\ A\,(g) + C(g) + E(g) \rightarrow H(liq) \\ A\,(g) + E(g) \rightarrow F(liq) \\ 3D(g) \rightarrow 2F(liq) \end{gathered} \tag{4-6}$$

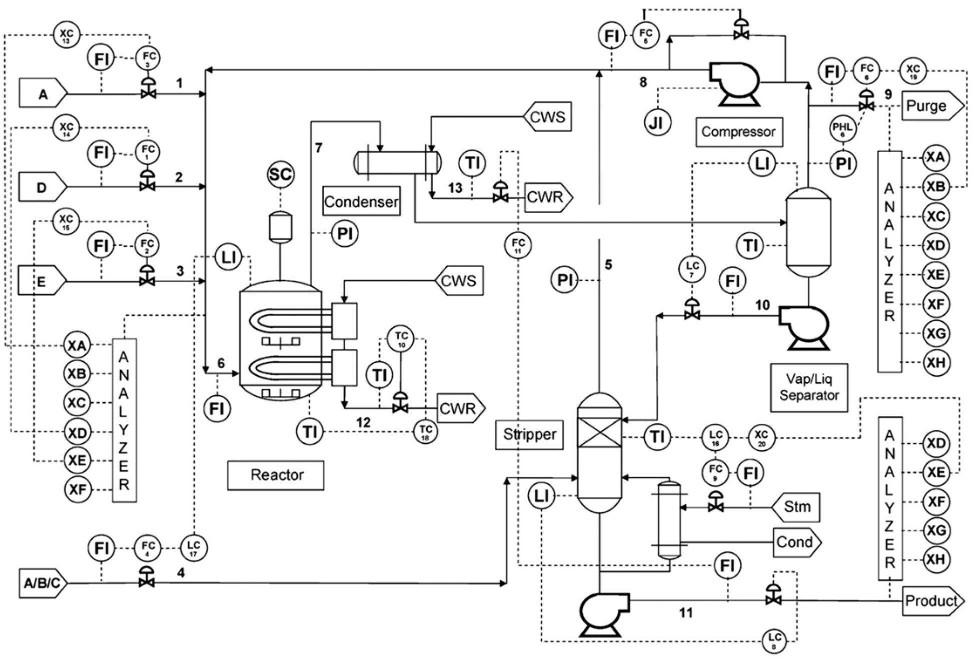


Fig. 4-5. Configuration of the TEP simulation

Table 4-3. Process variables in TEP, including manipulated variables (XMV), Continuous process measurements (XMEAS (1) ~ XMEAS (22)), sampled process measurements (XMEAS (23) ~ XMEAS (41))

Variable number	Variable name	Base case value(%)	Low limit	High Limit	Units
XMV (1)	D feed flow (stream 2)	63.053	0	5811	kg h^{-1}
XMV (2)	E feed flow (stream 3)	53.980	0	8354	kg h^{-1}
XMV (3)	A feed flow (stream 1)	24.644	0	1.017	kscmh
XMV (4)	A and C feed flow (stream 4)	61.302	0	15.25	kscmh
XMV (5)	Compressor recycle valve	22.210	0	100	%
XMV (6)	Purge valve (stream 9)	40.064	0	100	%
XMV (7)	Separator pot liquid flow (stream 10)	38.100	0	65.71	m^3h^{-1}
XMV (8)	Stripper liquid product flow (stream 11)	46.534	0	49.10	m^3h^{-1}
XMV (9)	Stripper steam valve	47.446	0	100	%
XMV (10)	Reactor cooling water flow	41.106	0	227.1	m^3h^{-1}
XMV (11)	Condenser cooling water flow	18.114	0	272.6	m^3h^{-1}

XMV (12)	Agitator speed	50.000	150	250	rpm
XMEAS (1)	A feed (stream 1)	0.25052	-	-	kscmh
XMEAS (2)	D feed (stream 2)	3664.0	-	-	kg h^{-1}
XMEAS (3)	E feed (stream 3)	4509.3	-	-	kg h^{-1}
XMEAS (4)	A and C feed (stream 4)	9.3477	-	-	kscmh
XMEAS (5)	Recycle flow (stream 8)	26.902	-	-	kscmh
XMEAS (6)	Reactor feed rate (stream 6)	42.339	-	-	kscmh
XMEAS (7)	Reactor pressure	2705.0	-	-	kPa gauge
XMEAS (8)	Reactor level	75.000	-	-	%
XMEAS (9)	Reactor temperature	120.40	-	-	℃
XMEAS (10)	Purge rate (stream 9)	0.33712	-	-	kscmh
XMEAS (11)	Product separator temperature	80.109	-	-	℃
XMEAS (12)	Product separator level	50.000	-	-	%
XMEAS (13)	Product separator pressure	2633.7	-	-	kPa gauge
XMEAS (14)	Product separator underflow (stream 10)	25.160	-	-	m^3h^{-1}
XMEAS (15)	Stripper level	50.000	-	-	%
XMEAS (16)	Stripper pressure	3102.2	-	-	kPa gauge
XMEAS (17)	Stripper underflow (stream 11)	22.949	-	-	m^3h^{-1}
XMEAS (18)	Stripper temperature	65.731	-	-	℃
XMEAS (19)	Stripper steam flow	230.31	-	-	kg h^{-1}
XMEAS (20)	Compressor work	341.43	-	-	kW
XMEAS (21)	Reactor cooling water outlet temperature	94.599	-	-	℃
XMEAS (22)	Separator cooling water outlet temperature	77.297	-	-	℃
XMEAS (23)	Stream 6 (reactor feed) component A	32.188	-	--	mol%
XMEAS (24)	Stream 6 (reactor feed) component B	8.8933	-	--	mol%
XMEAS (25)	Stream 6 (reactor feed) component C	26.383	-	-	mol%
XMEAS (26)	Stream 6 (reactor feed) component D	6.8820	-	-	mol%
XMEAS (27)	Stream 6 (reactor feed) component E	18.776	-	-	mol%
XMEAS (28)	Stream 6 (reactor feed) component F	1.6567	-	-	mol%

XMEAS (29)	Stream 9 (purge gas) component A	32.958	-	-	mol%
XMEAS (30)	Stream 9 (purge gas) component B	13.823	-	-	mol%
XMEAS (31)	Stream 9 (purge gas) component C	23.978	-	-	mol%
XMEAS (32)	Stream 9 (purge gas) component D	1.2565	-	-	mol%
XMEAS (33)	Stream 9 (purge gas) component E	18.579	-	-	mol%
XMEAS (34)	Stream 9 (purge gas) component F	2.2633	-	-	mol%
XMEAS (35)	Stream 9 (purge gas) component G	4.8436	-	-	mol%
XMEAS (36)	Stream 9 (purge gas) component H	2.2986	-	-	mol%
XMEAS (37)	Stream 11 (product) component D	0.01787	-	-	mol%
XMEAS (38)	Stream 11 (product) component E	0.83570	-	-	mol%
XMEAS (39)	Stream 11 (product) component F	0.09858	-	-	mol%
XMEAS (40)	Stream 11 (product) component G	53.724	-	-	mol%
XMEAS (41)	Stream 11 (product) component H	43.828	-	-	mol%

4.3.2. Results

4.3.2.1.Feature extraction using CAE

Using the trained CAE's encoder, sensor data of N_{sensor} with a time step length of L_{input} is reduced to N_latent with the time step length of L_{input}. Here, the number of features (N) is reduced number of time steps (L) is conserved. For CSTR process, 8 variables (N_{sensor}) were reduced to 3 latent variables (N_{latent}). and for TE process, 52 variables (N_{sensor}) were reduced to 8 latent variables (N_{latent}). The number of the latent vector is obtained empirically. CAE is trained only with the normal data, due to the characteristic of encoder-decoder. For training in the CSTR process, the normal case in the training data was used and for the TE process, the normal case in the training data was used for training. The decoder was trained accordingly with the encoder for the reconstruction but did not use in this framework. With the obtained latent vector, the T^2value of the given data was calculated as a prediction label. Also the fault detection threshold was set from the T^2value.

4.3.2.2. Trajectory prediction

Trajectory prediction of multivariate time series on latent space was performed using various RNN models. The input of prediction is the reduced feature number of N_{latent} by input time step length of L_{input} and the output of the prediction is the reduced feature number of N_{latent} by output time step length of L_{output}. Here, several variables (N) are conserved but the number of time steps (L) can be differed by the length desired for prediction. For the CSTR process, the normal case in the training data and fault case 1 to 10 in the test set A was used for model training, and fault case 1 to 10 in test set B was used for model testing. For the TE process, the normal case in the training data and fault case 1 to 21 in the test set A was used for model training, and fault case 1 to 21 in test set B.

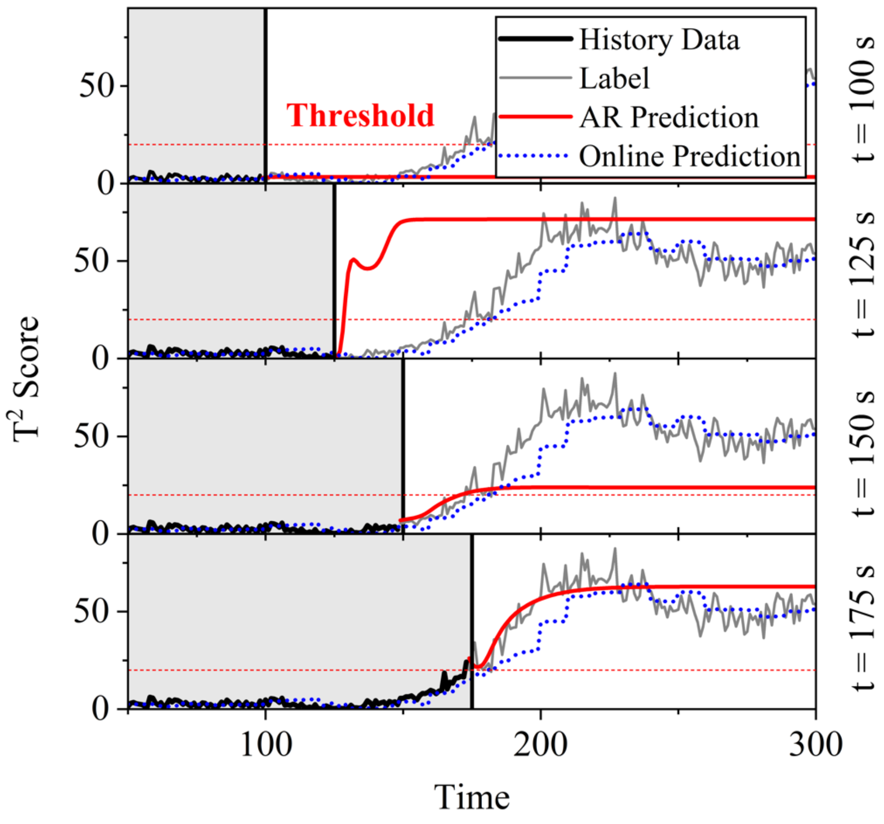


Fig. 4-6. Autoregressive prediction on current fault for calculating RUL on TEP Fault 2 (B composition, A/C ratio constant (Stream 4))

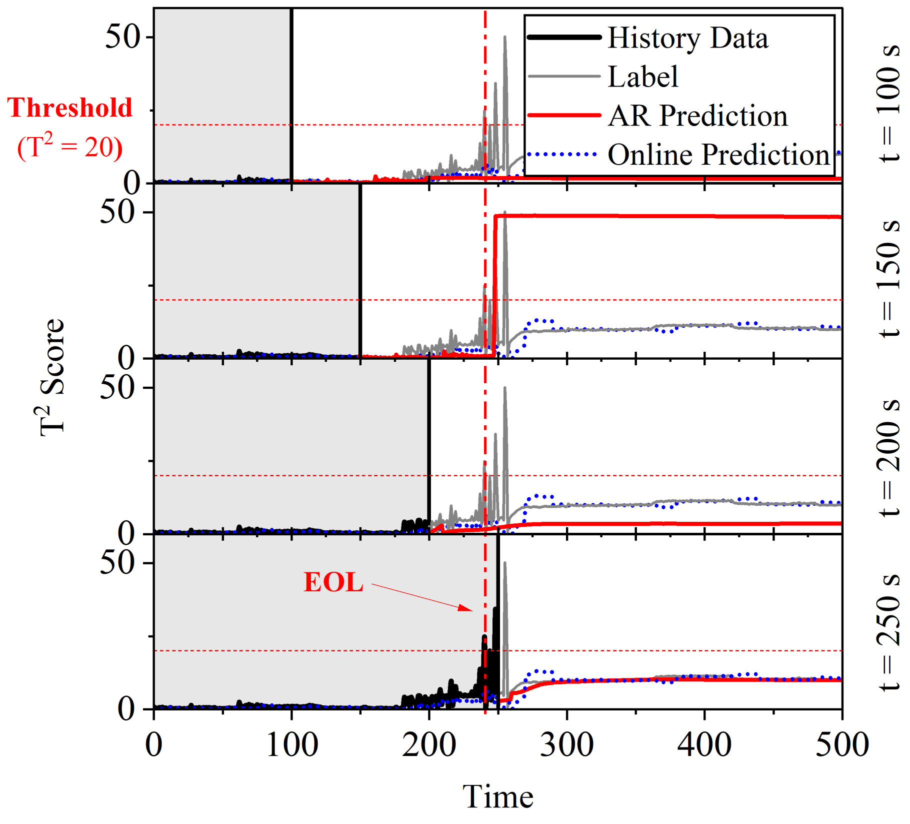


Fig. 4-7. Autoregressive prediction on current fault for calculating RUL on CSTR Fault 3

Model architectures combined with various RNN cell types were used for prediction including LSTM, LSTM + Dense, ARLSTM, GRU, GRU + Dense, ARGRU. The dense model is a simple fully-connected neural network model, ARLSTM is Autoregressive LSTM, and ARGRU is Autoregressive GRU. There were no big differences between the cell types on the prediction performance of the model shown in **Table 4-4**, but were tendencies regarding the model architectures. Basic models without stacking like LSTM and GRU showed faster performance than stacked models like RNN + Dense + Dense and RNN + RNN + Dense. As a result of the comparison, an unstacked LSTM model was used as a prediction model for further steps.

A large T^2 value means that there is a large deviation from the normal driving situation, and it means that data that is different from the data learned by the deep learning model under normal conditions is being input. In other words, the existing model refers to the problem of significantly lowering prediction performance when a new situation occurs, not the learned situation, that is, the extrapolation problem of the deep learning model, which is a data driven technique. However, in the case of using online learning using new incoming samples, prediction performance can be maintained even if a new situation appears rather than the learned situation through continuous updates. This is a very big advantage in the chemical process where it is difficult to actually generate a problem situation.

4.3.2.3. Online learning after training

Online learning was done using periodic transfer learning. when the number of incoming samples accumulates up to a certain value(l_{online}), the training process using transfer learning is initiated using newly incoming samples as the training set. This method uses makes changes to the trained internal weight parameters of the neural network model with accumulated samples. Since the sample number used as update l_{online} is smaller than the sample number used for training the original model, changes would be smaller compared to the main training phase. Here, the number of sample interval lengths for online learning(l_{online}) was set as 10, 30, and 60. Also, the epoch size for the transfer learning was set as 20, 50, and

100. As a result, the error was lower with the smaller interval length and the larger epoch size. The error was calculated as the Maximum Error Percentage Error (MAPE) on T^2 value of predicted latent vectors.

Also, a visual comparison of the trend was included in finding the tendency. By comparing the result without online learning and with online learning, there is a significant accuracy increase in prediction using online learning. Also, there is a relatively small difference between the different combinations of sample interval length and epoch size. By this, using online learning is beneficial regardless of the sample interval or epoch size. For the TE process, faults 2, 6, 13, 17, and 18 showed a dramatic increase in accuracy. Those faults have larger T^2 values than other cases, which means they showed a large difference from the normal operating behavior.

Table 4-4. Prediction errors on TEP fault case 1~21 by different RNN models

	Description	Type	Dense	Dense +Dense	LSTM	LSTM + Dense	ARLSTM	GRU	GRU +Dense	ARGRU
Fault 1	A/C feed ratio, B composition constant (Stream 4)	Step	1.900	2.136	2.570	2.364	2.086	2.846	2.146	2.737
Fault 2	B composition, A/C ratio constant (Stream 4)	Step	17.130	15.187	25.078	24.517	26.338	28.471	26.154	29.172
Fault 3	D feed temperature (Stream 2)	Step	1.425	1.511	1.582	1.996	1.567	1.632	1.553	1.667
Fault 4	Reactor cooling water inlet temperature	Step	1.354	1.495	1.463	2.025	1.511	1.673	1.587	1.646
Fault 5	Condenser cooling water inlet temperature	Step	1.509	1.685	2.153	2.132	1.917	1.584	1.778	2.173
Fault 6	A feed loss (Stream 1)	Step	29.805	47.048	53.703	38.968	50.047	45.359	55.656	40.554

Fault 7	C header pressure loss-reduced availablity (Stream 4)	Step	1.870	2.177	2.618	2.652	1.926	2.021	2.200	2.057
Fault 8	A, B, C feed composition (Stream 4)	Random	1.643	2.018	2.798	2.671	2.504	2.612	1.998	3.681
Fault 9	D feed temperature (Stream 2)	Random	1.338	1.401	1.534	1.861	1.486	1.568	1.468	1.623
Fault 10	C feed temperature (Stream 4)	Random	1.220	1.351	1.349	1.711	1.330	1.422	1.359	1.489
Fault 11	Reactor cooling water inlet temperature	Random	1.233	1.344	1.408	1.753	1.368	1.469	1.461	1.469
Fault 12	Condenser cooling water inlet temperature	Random	2.419	2.751	3.328	3.024	3.780	3.857	4.003	3.398
Fault 13	Reaction kinetics	Slow Drift	3.357	5.207	10.152	8.947	8.110	5.716	5.609	8.111
Fault 14	Reactor cooling water valve	Sticking	1.186	1.098	1.197	1.258	1.088	1.166	1.158	1.104
Fault 15	Condenser cooling water valve	Sticking	1.429	1.543	1.587	1.978	1.569	1.612	1.568	1.666
Fault 16	Unknown	Unknown	1.190	1.286	1.399	1.694	1.338	1.445	1.359	1.492
Fault 17	Unknown	Unknown	7.948	8.261	7.682	7.898	8.456	7.663	7.866	8.236
Fault 18	Unknown	Unknown	46.854	43.085	48.241	48.115	46.030	47.073	45.818	45.309
Fault 19	Unknown	Unknown	1.457	1.546	1.603	2.013	1.599	1.648	1.585	1.677
Fault 20	Unknown	Unknown	1.157	1.225	1.315	1.531	1.213	1.273	1.278	1.314
Fault 21	Unknown	Unknown	1.240	1.356	1.418	1.681	1.356	1.396	1.355	1.431

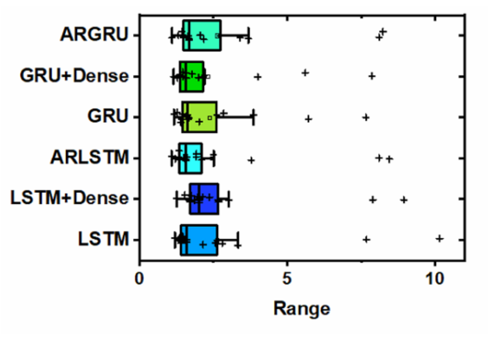


Fig. 4-8. Result of prediction by RNN model (except fault 2, 6, 18)

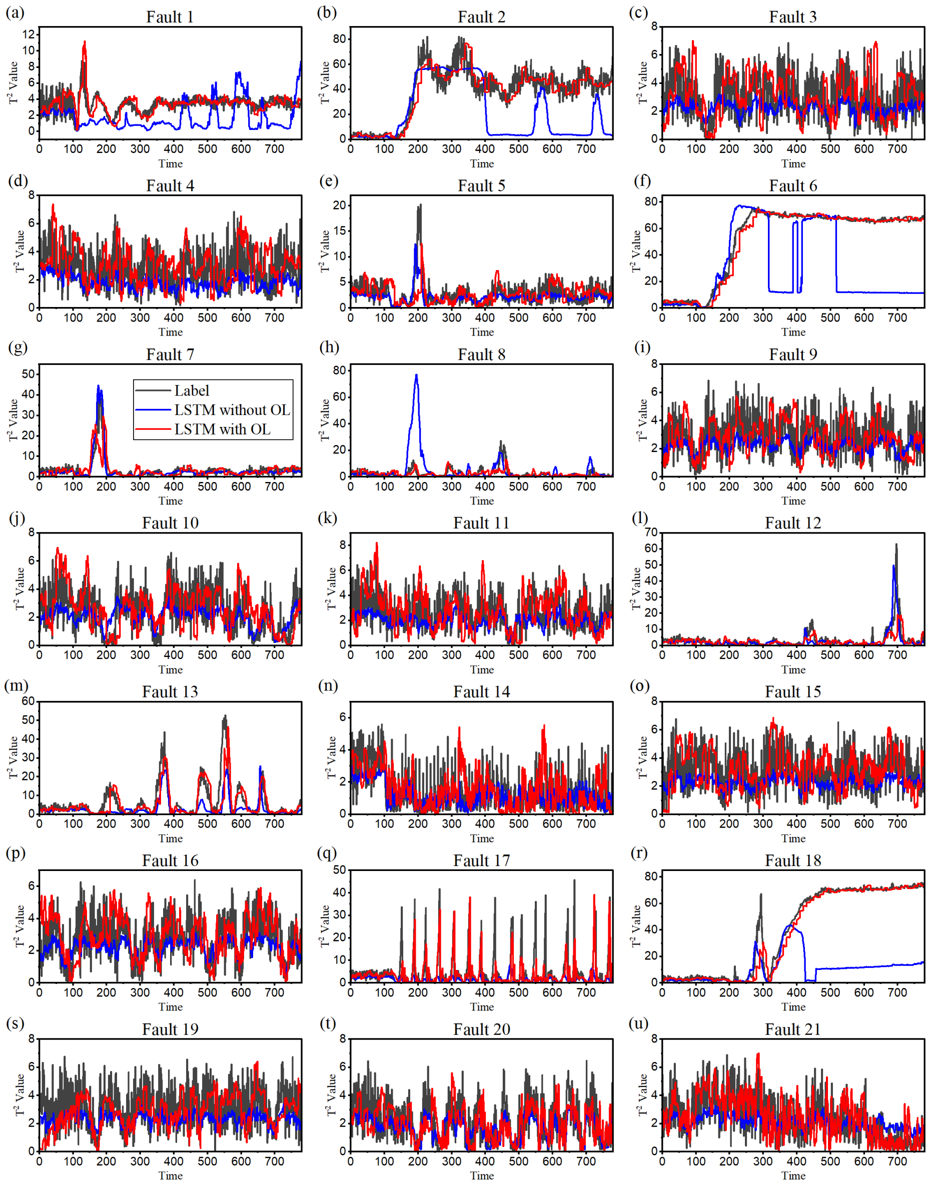


Fig. 4-9. Fault propagation prediction on TEP fault case 1~21((a)~(u)) without and with online learning (interval 10, epoch 100)

Table 4-5. Prediction errors on TEP fault case 1~21 without and with online learning with various combinations of online learning intervals (10, 30, 60) and epoch sizes (100, 50, 20)

	Description	Type	w/o OL	10/100	10/50	10/20	30/100	60/100
Fault 1	A/C feed ratio, B composition constant	Step	2.57	0.70	0.77	0.76	0.82	0.70
Fault 2	B composition, A/C ratio constant	Step	25.07	6.04	8.98	6.67	11.41	6.85
Fault 3	D feed temperature	Step	1.58	1.49	1.34	1.18	1.35	1.31
Fault 4	Reactor cooling water inlet temperature	Step	1.46	1.45	1.35	1.09	1.52	1.47
Fault 5	Condenser cooling water inlet temperature	Step	2.15	1.64	1.55	1.20	1.63	1.46
Fault 6	A feed loss	Step	53.70	2.28	3.53	2.43	6.79	7.25
Fault 7	C header pressure loss-reduced availablity	Step	2.61	2.06	1.73	1.35	1.78	1.42
Fault 8	A, B, C feed composition	Random	2.79	1.71	1.34	1.24	1.83	1.17
Fault 9	D feed temperature	Random	1.53	1.30	1.19	1.10	1.25	1.54
Fault 10	C feed temperature	Random	1.34	1.28	1.29	1.07	1.14	1.17
Fault 11	Reactor cooling water inlet temperature	Random	1.40	1.32	1.14	1.11	1.38	1.22
Fault 12	Condenser cooling water inlet temperature	Random	3.32	1.99	1.90	1.52	1.73	1.64
Fault 13	Reaction kinetics	Slow Drift	10.15	3.16	2.38	3.26	2.46	3.14
Fault 14	Reactor cooling water valve	Sticking	1.19	1.06	0.92	0.87	1.05	1.17
Fault 15	Condenser cooling water valve	Sticking	1.58	1.45	1.32	1.21	1.69	1.42
Fault 16	Unknown	Unknown	1.39	1.16	1.08	1.00	1.26	1.20
Fault 17	Unknown	Unknown	7.68	3.03	2.95	2.56	2.87	2.81
Fault 18	Unknown	Unknown	48.24	3.20	3.68	3.46	4.68	8.25
Fault 19	Unknown	Unknown	1.60	1.33	1.28	1.18	1.45	1.66
Fault 20	Unknown	Unknown	1.31	1.13	1.05	0.95	0.99	1.33
Fault 21	Unknown	Unknown	1.41	1.17	1.06	0.97	1.21	1.31

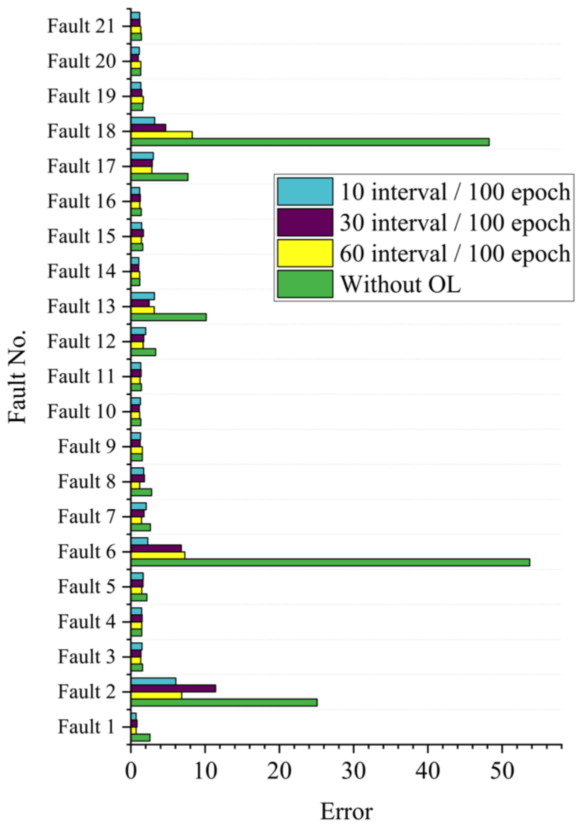


Fig. 4-10. Fault propagation prediction errors on TEP fault case 1~21 without and with online learning

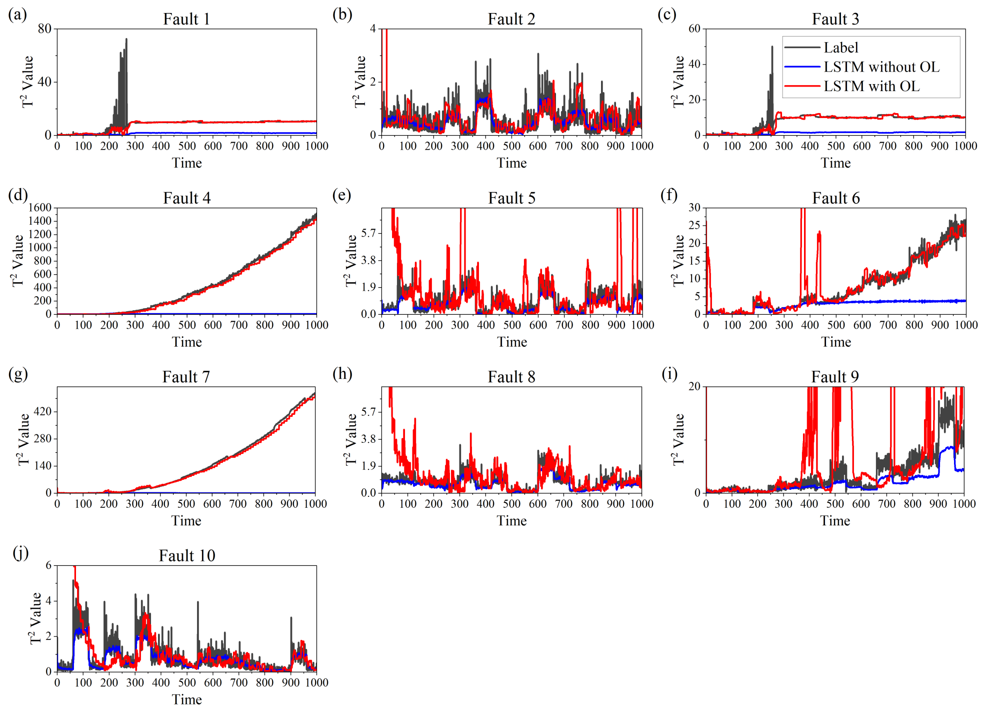


Fig. 4-11. Fault propagation prediction on CSTR case fault 1~10 ((a)~(j)) without and with online learning

Table 4-6. Fault propagation prediction on CSTR fault case 1~21 without and with online learning with various combinations of online learning intervals and epoch sizes

	Description	**Type**	**w/o OL**	**10/100**	**10/50**	**10/20**	**30/100**	**60/100**
Fault 1	Catalyst decay	Multiplicative	8.96	5.02	5.25	5.31	5.50	5.49
Fault 2	Heat transfer fouling	Multiplicative	0.37	1.01	1.30	1.10	0.72	0.73
Fault 3	Simultaneous Faults 1 and 2	Multiplicative	7.89	2.62	2.62	2.78	3.07	2.99
Fault 4	Sensor drift on Ci	Additive	653.75	36.10	43.62	212.71	193.17	616.50
Fault 5	Sensor drift on Ti	Additive	0.45	61.92	44.97	18.24	49.33	9.99
Fault 6	Sensor drift on Tc i	Additive	8.73	7.97	27.32	9.27	18.20	41.80

Fault 7	Sensor drift on C	Additive	225.29	11.31	31.11	100.95	219.65	159.19
Fault 8	Sensor drift on T	Additive	0.37	17.49	13.41	9.80	32.31	2.41
Fault 9	Sensor drift on Tc	Additive	2.86	151.81	68.15	58.50	56.22	12.07
Fault 10	Sensor drift Qc	Additive	0.52	13.48	13.49	6.90	14.24	2.22

4.3.2.4. RUL Prediction

Assuming the prediction model has a certain level of accuracy, the prediction result can be put as the model's input to get a prediction result for the further future. This is referred to as an autoregressive prediction, which iteratively makes a prediction until a certain time horizon limit using its own prediction result. This is separate sequence with online learning, however, to better predict RUL, online learning is also needed for the RUL calculation phase as included in the algorithm **Algorithm 1**. Here, the End Of Life (EOL) point is defined as the first point at which the T^2value meets the threshold. For the TE process, the threshold is 20 and For the CSTR process the threshold is 5. RUL is difference between the current time and EOL.and this can be shown in **Fig. 4-12**.

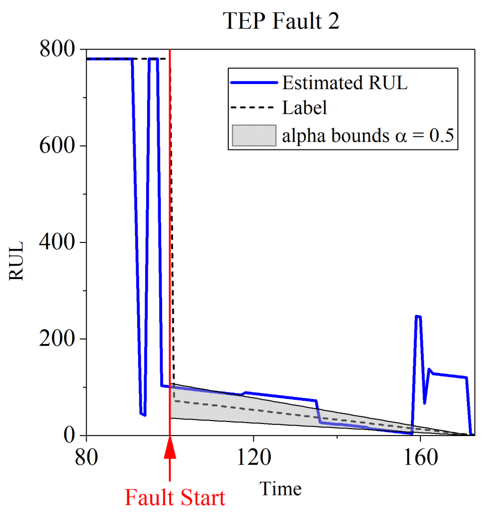


Fig. 4-12. Alpha-lambda plot of TEP Fault 2 from t = 80 to t = 173 (EOL)

4.3.2.5. Fault Diagnosis

Using the SHAP algorithm, the contribution score of the calculated T^2value of the corresponding input value of size N_{sensor} by L_{input} was obtained. Since the Shapely value obtained from the SHAP is model-dependent, constantly updating the model using online learning affects the result of the SHAP value. This means that when the model updates using online learning, the calculation of the Shapley value for the model should be renewed. The result shown in **Fig. 4-13** was re-calculated by each sample interval length l_{online}. the larger the contribution score, the more likely the sensor is the cause of the fault. In **Fig. 4-13**, sensor has the largest contribution score which correlates with the real root cause.

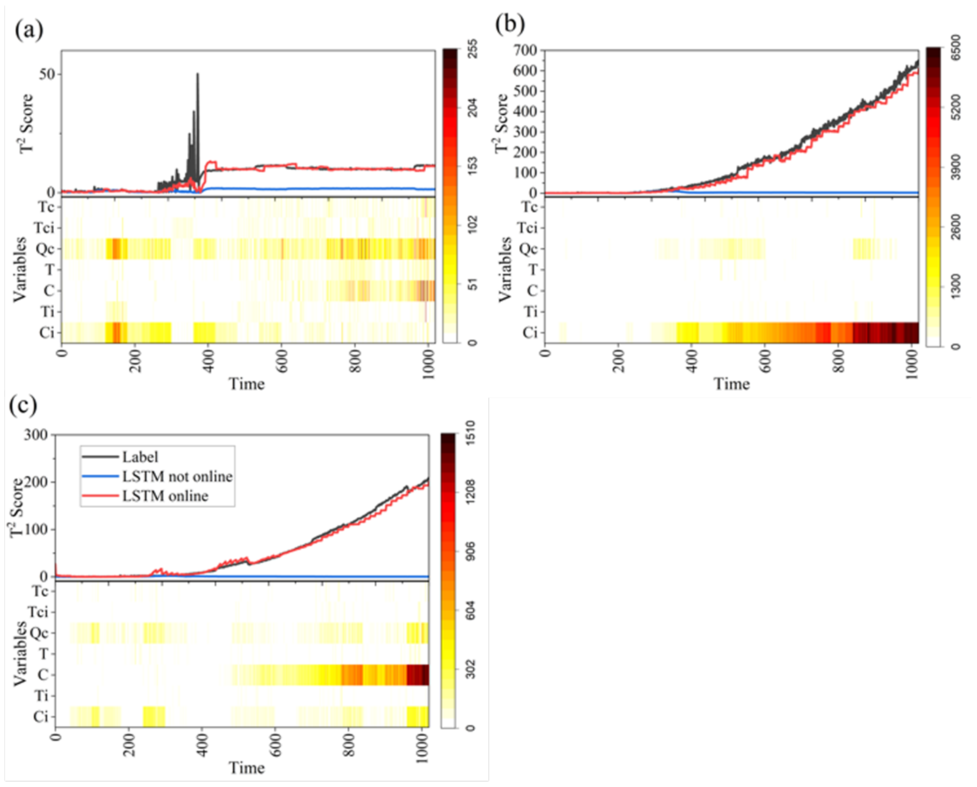


Fig. 4-13. T^2 Score of prediction result on CSTR fault cases with and without online learning, with SHAP value by time. (a) is fault case 3, catalyst decay and heat transfer learning, (b) is fault case 4, sensor drift on Ci, (c) is fault case 7, sensor drift on C

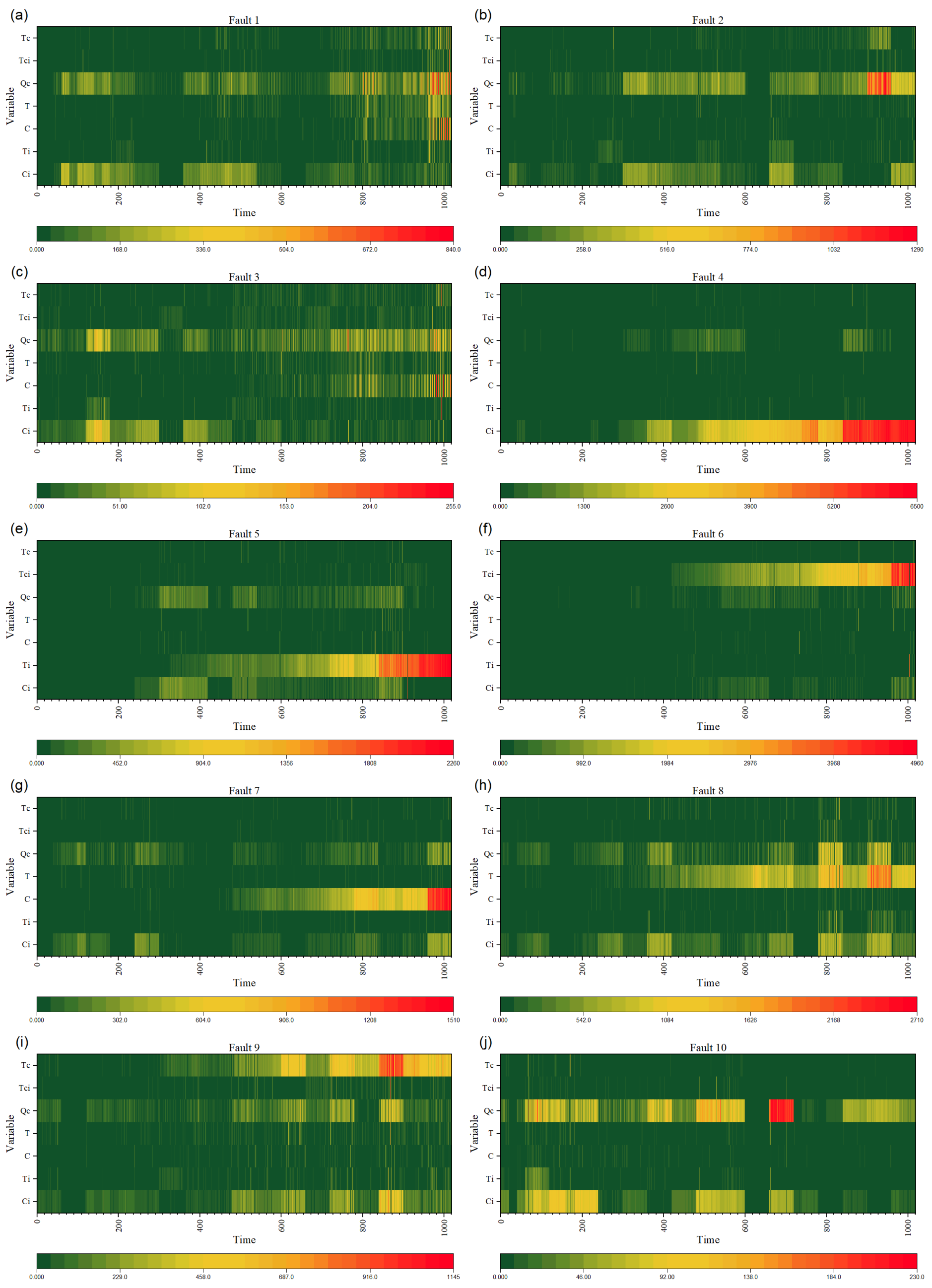


Fig. 4-14. Heatmap of SHAP values over time on CSTR fault case 1 to 10, which shown in (a) to (j) respectively

4.4. Conclusion

In this study, novel prognosis scheme for chemical process with deep-learning based anomaly detection, prediction and diagnosis has been suggested. anomaly detection has been done with CAE, prediction was done using autoregressive RNN using online learning and root cause diagnosis was done with XAI method SHAP. With progress in each parts, this framework shows the far better results than any other attempts at prognosis on chemical plant.

4.5. Acknowledgement

This chapter is partially adapted from A Fully Machine Learning Based Framework for Industrial Process Fault Prognosis Combining Autoencoder, Autoregressive RNN with Online Learning and XAI with authors N. Lee, K. Jang, K.E.S. Pilario, J. Na, and Y. Seo. (Preparing Submission)

Chapter 5. Concluding remarks

5.1. Conclusions

This study presented various models using machine learning for use in offshore systems, including an anomaly detection model based on a machine learning model, a hazard detection model through sensor data prediction, and a process predictive maintenance model.

Firstly, a plant-wide anomaly detection algorithm using Multi-Scale Convolutional Recurrent Encoder-Decoder (MSCRED) has been proposed. Herein, anomaly detection was performed based on the interconnected side of process plant sensor data rather than the simple sum of single sensor data. Conv-LSTM model used a framework that modified the method for calculating the anomaly score based on the MSCRED framework for chemical processes. Additionally, we established a methodology to process anomaly detection using multiple windows. In addition to performing anomaly detection, we analyzed the effect of the training data on the anomaly detection performance of the model, and the situation and timing of separation of each issue were obtained.

Secondly, a novel data-driven framework to assess and predict the hydrate formation behavior using various deep learning models, especially the RNN family, was proposed. The obtained results showed the models can predict the transition and segregation points well. PCA on many data can cluster the points by before/after transition point, which indicates the sudden and rapid hydrate growth. Also, PCA reduces the number of features, which makes training deep learning models more efficient. And windowing was done on given time-series data to make a real-time prediction model which gets input from the near past. Prediction using deep learning models (Dense, LSTM, GRU, ARLSTM) shows reasonable results on prediction transition and segregation points. Among the models, ARLSTM shows the best results. For layer number, the single-layer shows larger MAPE than stacked layers. The dropout rate between 0.2~0.6 showed significant improvement in accuracy. Lastly, using dataset for deep learning model, training under similar experimental conditions (training until 7th batch; only with 200rpm data) shows the best result. The data under different experimental conditions (training with 1~7 and 9~16th batch; adding 600rpm data after 200rpm data) increases MAPE but the MAPE is decreasing as training the model with multiple batches. Enough amount data is used in the training, the better the prediction results become.

Thirdly, novel prognosis scheme for chemical process with deep-learning based anomaly detection, prediction and diagnosis has been suggested. anomaly detection has been done with CAE, prediction was done using autoregressive RNN using online learning and root cause diagnosis was done with XAI method SHAP. With progress in each parts, this framework shows the far better results than any other attempts at prognosis on chemical plant.

5.2. Further study

Machine learning and deep learning-based methodologies have come a long way in about 10 years. In addition, studies that can be used in harmony with existing engineering techniques are also being studied a lot recently. However, there is no methodology established as an industry standard yet, and it will be an important task to develop an algorithm with accuracy and efficiency that can be established as a standard while identifying various machine learning techniques that are continuously developing. The biggest advantage of machine learning-based methodology is that it can be automatically optimized based on data, and that automation can be performed later using this. In other words, it is necessary to automatically process it in the era of increasing data volume using cloud and big data. However, in order to use it efficiently, it is necessary to compare and review the methodology suitable for the system and modify it to suit the system. In this study, only the diagnostic field was studied, but for complete automation, it is thought that a full framework that applies machine learning to the control field should be produced. Since the control part uses methodologies such as reinforcement learning, additional research on how to bridge is needed. Also, within this study, clear definitions and classification methods for fault cases and thresholds are needed for predictive maintenance in many process systems. However, since this is different for each system, automation can be easily achieved only when research on a methodology that can determine this regardless of the system is conducted. There is also a need to more actively borrow probabilistic methodologies. The introduction of formula-based deep learning methodologies such as Physics informed neural network (PINN) is also an important part. Since this field is still in the stage of application, it is necessary to continuously search for the optimal methodology through the introduction of various methodologies and comparison between them.

Bibliography

[1] M. Kordestani, M. Saif, M.E. Orchard, R. Razavi-Far, K. Khorasani, Failure prognosis and applications—A survey of recent literature, IEEE Trans. Rel. 70 (2019) 728–748.

[2] B.R. Bakshi, Multiscale PCA with application to multivariate statistical process monitoring, AIChE J. 44 (1998) 1596–1610.

[3] S. Heo, J.H. Lee, Fault detection and classification using artificial neural networks, IFAC-PapersOnLine 51 (2018) 470–475.

[4] M.R. Dobbelaere, P.P. Plehiers, R. Van de Vijver, C.V. Stevens, K.M. Van Geem, Machine learning in chemical engineering: strengths, weaknesses, opportunities, and threats, Engineering 7 (2021) 1201–1211.

[5] D. Li, Perspective for smart factory in petrochemical industry, Comput. Chem. Eng. 91 (2016) 136–148.

[6] M. Mowbray, M. Vallerio, C. Perez-Galvan, D. Zhang, A.D.R. Chanona, F.J. Navarro-Brull, Industrial data science–a review of machine learning applications for chemical and process industries, React. Chem. Eng. (2022).

[7] U. Ahrend, M. Aleksy, M. Berning, J. Gebhardt, F. Mendoza, D. Schulz, Sensors as the basis for digitalization: new approaches in instrumentation, IoT-concepts, and 5G, Internet of Things 15 (2021) 100406.

[8] Z.J. Baum, X. Yu, P.Y. Ayala, Y. Zhao, S.P. Watkins, Q. Zhou, Artificial intelligence in chemistry: current trends and future directions, J. Chem. Inf. Model. 61 (2021) 3197–3212.

[9] R. Chalapathy, S. Chawla, Deep learning for anomaly detection: a survey, arXiv preprint arXiv:1901.03407 (2019). https://doi.org/10.48550/arXiv.1901.03407.

[10] G. Pang, C. Shen, L. Cao, A.V.D. Hengel, Deep learning for anomaly detection: A review, ACM Comput. Surv. 54 (2021) 1–38.

[11] Y. Luo, Y. Xiao, L. Cheng, G. Peng, D. Yao, Deep learning-based anomaly detection in cyber-physical systems: Progress and opportunities, ACM Comput. Surv. 54 (2021) 1–36.

[12] R. Isermann, Fault-diagnosis systems: an introduction from fault detection to fault tolerance, Springer Science & Business Media2005.

[13] M. Ammiche, A. Kouadri, A. Bakdi, A combined monitoring scheme with fuzzy logic filter for plant-wide Tennessee Eastman Process fault detection, Chem. Eng. Sci. 187 (2018) 269–279.

[14] A. Shrestha, A. Mahmood, Review of deep learning algorithms and architectures, IEEE Access 7 (2019) 53040–53065.

[15] N. Görnitz, M. Kloft, K. Rieck, U. Brefeld, Toward supervised anomaly detection, J. Art. Intell. Res. 46 (2013) 235–262.

[16] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM Comput. Surv. 41 (2009) 1–58.

[17] M.A. Kramer, Nonlinear principal component analysis using autoassociative neural networks, AIChE J. 37(2) (1991) 233–243.

[18] B. Siegel, Industrial anomaly detection: A comparison of unsupervised neural network architectures, IEEE Sens. Lett. 4 (2020) 1–4.

[19] K. Jang, S. Hong, M. Kim, J. Na, I. Moon, Adversarial autoencoder based feature learning for fault detection in industrial processes, IEEE Trans. Ind. Inform. 18 (2021) 827–834.

[20] K.R. Kini, M. Madakyaru, Anomaly detection using multi-scale dynamic principal component analysis for Tenneesse Eastman Process, 2019 Fifth Indian Control Conference (ICC), IEEE, 2019, pp. 219–224.

[21] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, N.V. Chawla, A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data, Proceedings of the AAAI conference on artificial intelligence, 2019, pp. 1409–1416.

[22] Y. Zhang, Y. Chen, J. Wang, Z. Pan, Unsupervised deep anomaly detection for multi-sensor time-series signals, IEEE Trans. Knowl. Data Eng. (2021).

[23] Z. He, P. Chen, X. Li, Y. Wang, G. Yu, C. Chen, X. Li, Z. Zheng, A spatiotemporal deep learning approach for unsupervised anomaly detection in cloud systems, IEEE Trans. Neural Netw. Learn. Syst. (2020).

[24] L.H. Chiang, E.L. Russell, R.D. Braatz, Fault detection and diagnosis in industrial systems, Springer Science & Business Media2000.

[25] H. Kim, J. Jung, Y. Lim, E.F. May, Y. Seo, Performance degradation of the monoethylene glycol regeneration process in the presence of electrolytes: pilot-scale experiments and dynamic simulations, Ind. Eng. Chem. Res. 59 (2020) 21205–21216.

[26] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, W.-c. Woo, Convolutional LSTM network: A machine learning approach for precipitation nowcasting, Adv. Neural Inf. Process. Syst. 28 (2015).

[27] J.K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, Y. Bengio, Attention-based models for speech recognition, Adv. Neural Inf. Process. Syst. 28 (2015) 577–585.

[28] H. Zhang, C. Zhang, K. Peng, J. Dong, L. Ma, X. Zhang, Quality anomaly monitoring and comprehensive diagnosis framework for plant-wide process industries with spatio-temporal coordination, 2021 CAA Symposium on Fault Detection, Supervision, and Safety for Technical Processes (SAFEPROCESS), IEEE, 2021, pp. 1–6.

[29] J. Xu, K. Duraisamy, Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics, Comput. Methods Appl. Mech. Eng. 372 (2020) 113379.

[30] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444.

[31] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3431–3440.

[32] S. Aghabozorgi, A.S. Shirkhorshidi, T.Y. Wah, Time-series clustering–a decade review, Inf. Syst. 53 (2015) 16–38

[33] R. Cilibrasi, P.M. Vitányi, Clustering by compression, IEEE Trans. Inf. Theor. 5 (2005) 1523–1545.

[34] M.M. Najafabadi, F. Villanustre, T.M. Khoshgoftaar, N. Seliya, R. Wald, E. Muharemagic, Deep learning applications and challenges in big data analytics, J. Big Data 2 (2015) 1–21.

[35] Akhfash, M., Boxall, J.A., Aman, Z.M., Johns, M.L., May, E.F., 2013. Hydrate formation and particle distributions in gas–water systems. Chemical Engineering Science 104, 177-188.

[36] Alon, J., Sclaroff, S., Kollios, G., Pavlovic, V., 2003. Discovering clusters in motion time-series data, 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings. IEEE, pp. I-I.

[37] Aman, Z.M., Akhfash, M., Johns, M.L., May, E.F., 2015. Methane hydrate bed formation in a visual autoclave: cold restart and Reynolds number dependence. Journal of Chemical & Engineering Data 60, 409-417.

[38] Aman, Z.M., Di Lorenzo, M., Kozielski, K., Koh, C.A., Warrier, P., Johns, M.L., May, E.F., 2016. Hydrate formation and deposition in a gas-dominant flowloop: Initial studies of the effect of velocity and subcooling. Journal of Natural Gas Science and Engineering 35, 1490-1498.

[39] Brownlee, J., 2018. Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in Python. Machine Learning Mastery.

[40] Camargo, R., Palermo, T., 2002. Rheological properties of hydrate suspensions in an asphaltenic crude oil, Proceedings of the 4th International Conference on Gas Hydrates, pp. 880-885.

[41] Charlton, T.B., Di Lorenzo, M., Zerpa, L.E., Koh, C.A., Johns, M.L., May, E.F., Aman, Z.M., 2018. Simulating hydrate growth and transport behavior in gas-dominant flow. Energy & Fuels 32, 1012-1023.

[42] Chaudhari, P., Zerpa, L.E., Sum, A.K., 2018. A correlation to quantify hydrate plugging risk in oil and gas production pipelines based on hydrate transportability parameters. Journal of Natural Gas Science and Engineering 58, 152-161.

[43] Chen, C., Yuan, H., Wang, X., Lin, Y., He, Y., Wang, F., 2022. Recyclable and high-efficiency methane hydrate formation promoter based on SDS-coated superparamagnetic nano-Fe_3O_4. Chemical Engineering Journal 437, 135365.

[44] Chen, J., Wang, T., Zeng, Z., Jiang, J.-H., Deng, B., Chen, C.-Z., Li, J.-Y., Li, C.-H., Tao, L.-M., Li, X., 2019. Oleic acid potassium soap: A new potential kinetic promoter for methane hydrate formation. Chemical Engineering Journal 363, 349-355.

[45] Cheng, G., Peddinti, V., Povey, D., Manohar, V., Khudanpur, S., Yan, Y., 2017. An exploration of dropout with lstms, Interspeech, pp. 1586-1590.

[46] Chiang, L.H., Russell, E.L., Braatz, R.D., 2000. Fault detection and diagnosis in industrial systems. Springer Science & Business Media.

[47] Chu, C.-S.J., 1995. Time series segmentation: A sliding window approach. Information Sciences 85, 147-173.

[48] Chung, J., Gulcehre, C., Cho, K., Bengio, Y., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

[49] Davies, S.R., Boxall, J.A., Dieker, L.E., Sum, A.K., Koh, C.A., Sloan, E.D., Creek, J.L., Xu, Z.-G., 2010. Predicting hydrate plug formation in oil-dominated flowlines. Journal of petroleum science and engineering 72, 302-309.

[50] Di Lorenzo, M., Aman, Z.M., Kozielski, K., Norris, B.W., Johns, M.L., May, E.F., 2018. Modelling hydrate deposition and sloughing in gas-dominant pipelines. The Journal of Chemical Thermodynamics 117, 81-90.

[51] Ding, Q., Kolaczyk, E.D., 2013. A compressed PCA subspace method for anomaly detection in high-dimensional data. IEEE Transactions on Information Theory 59, 7419-7433.

[52] Gal, Y., Ghahramani, Z., 2016. A theoretically grounded application of dropout in recurrent neural networks. Advances in neural information processing systems 29, 1019-1027.

[53] Gers, F.A., Schmidhuber, J., Cummins, F., 2000. Learning to forget: Continual prediction with LSTM. Neural computation 12, 2451-2471.

[54] Gupta, S., Ray, A., Keller, E., 2007. Symbolic time series analysis of ultrasonic data for early detection of fatigue damage. Mechanical Systems and Signal Processing 21, 866-884.

[55] Hewage, P., Behera, A., Trovati, M., Pereira, E., Ghahremani, M., Palmieri, F., Liu, Y., 2020. Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station. Soft Computing 24, 16453-16482.

[56] Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation 9, 1735-1780.

[57] Huang, L., Nguyen, X., Garofalakis, M., Jordan, M.I., Joseph, A., Taft, N., 2006. In-network PCA and anomaly detection, NIPS, pp. 617-624.

[58] Huo, Z., Freer, E., Lamar, M., Sannigrahi, B., Knauss, D., Sloan Jr, E., 2001. Hydrate plug prevention by anti-agglomeration. Chemical Engineering Science 56, 4979-4991.

[59] Joshi, S.V., Grasso, G.A., Lafond, P.G., Rao, I., Webb, E., Zerpa, L.E., Sloan, E.D., Koh, C.A., Sum, A.K., 2013. Experimental flowloop investigations of gas hydrate formation in high water cut systems. Chemical Engineering Science 97, 198-209.

[60] Jozefowicz, R., Zaremba, W., Sutskever, I., 2015. An empirical exploration of recurrent network architectures, International conference on machine learning. PMLR, pp. 2342-2350.

[61] Keogh, E.J., Pazzani, M.J., 2000. A simple dimensionality reduction technique for fast similarity search in large time series databases, Pacific-Asia conference on knowledge discovery and data mining. Springer, pp. 122-133.

[62] Kim, H., Kim, J., Seo, Y., 2020. Economic benefit of methane hydrate reformation management in transport pipeline by reducing thermodynamic hydrate inhibitor injection. Journal of petroleum science and engineering 184, 106498.

[63] Kim, J., Kim, H., hoon Sohn, Y., Chang, D., Seo, Y., Kang, S.-P., 2017. Prevention of methane hydrate re-formation in transport pipeline using thermodynamic and kinetic hydrate inhibitors. Journal of Petroleum Science and Engineering.

[64] Kumar, A., Bhattacharjee, G., Kulkarni, B., Kumar, R., 2015. Role of surfactants in promoting gas hydrate formation. Industrial & Engineering Chemistry Research 54, 12217-12232.

[65] LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. nature 521, 436-444. Liu, Z., Farahani, M.V., Yang, M., Li, X., Zhao, J., Song, Y., Yang, J., 2020. Hydrate slurry flow characteristics influenced by formation, agglomeration and deposition in a fully visual flow loop. Fuel 277, 118066.

[66] Majid, A.A., Wu, D.T., Koh, C.A., 2018. A Perspective on Rheological Studies of Gas Hydrate Slurry Properties. Engineering 4, 321-329.

[67] McGeary, R., 1961. Mechanical packing of spherical particles. Journal of the American ceramic Society 44, 513-522.

[68] Mills, P., 1985. Non-Newtonian behaviour of flocculated suspensions. Journal de Physique Lettres 46, 301-309.

[69] Mitschka, P., 1982. Simple conversion of Brookfield RVT readings into viscosity functions. Rheologica Acta 21, 207-209.

[70] Pan, S.J., Yang, Q., 2009. A survey on transfer learning. IEEE Transactions on knowledge and data engineering 22, 1345-1359.

[71] Qin, H., Srivastava, V., Wang, H., Zerpa, L.E., Koh, C.A., 2019. Machine Learning Models to Predict Gas Hydrate Plugging Risks Using Flowloop and Field Data, Offshore Technology Conference. OnePetro.

[72] Ren, Y., Xu, X., Yang, S., Nie, L., Chen, Y., 2020. A physics-based neural-network way to perform seismic full waveform inversion. IEEE Access 8, 112266-112277.

[73] Ringberg, H., Soule, A., Rexford, J., Diot, C., 2007. Sensitivity of PCA for traffic anomaly detection, Proceedings of the 2007 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, pp. 109-120.

[74] Shi, B.-H., Chai, S., Wang, L.-Y., Lv, X., Liu, H.-S., Wu, H.-H., Wang, W., Yu, D., Gong, J., 2016. Viscosity investigation of natural gas hydrate slurries with anti-agglomerants additives. Fuel 185, 323-338.

[75] Siami-Namini, S., Tavakoli, N., Namin, A.S., 2018. A comparison of ARIMA and LSTM in forecasting time series, 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, pp. 1394-1401.

[76] Sloan, E.D., Koh, C.A., 2007. Clathrate hydrates of natural gases. CRC press. Sloan, E.D., Koh, C.A., Sum, A.K., 2010. Natural gas hydrates in flow assurance. Gulf Professional Publishing, Oxford, UK.

[77] Sohn, Y.H., Seo, Y., 2017. Effect of monoethylene glycol and kinetic hydrate inhibitor on hydrate blockage formation during cold restart operation. Chemical Engineering Science 168, 444-455.

[78] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research 15, 1929-1958.

[79] Torres, J.F., Hadjout, D., Sebaa, A., Martínez-Álvarez, F., Troncoso, A., 2021. Deep Learning for Time Series Forecasting: A Survey. Big Data 9, 3-21.

[80] Turner, D.J., Miller, K.T., Sloan, E.D., 2009. Methane hydrate formation and an inward growing shell model in water-in-oil dispersions. Chemical Engineering Science 64, 3996-4004.

[81] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need, Advances in neural information processing systems, pp. 5998-6008.

[82] Veluswamy, H.P., Kumar, A., Seo, Y., Lee, J.D., Linga, P., 2018. A review of solidified natural gas (SNG) technology for gas storage via clathrate hydrates. Applied Energy 216, 262-285.

[83] Wang, S., Hu, S., Brown, E.P., Nakatsuka, M.A., Zhao, J., Yang, M., Song, Y., Koh, C.A., 2017. High pressure micromechanical force measurements of the effects of surface corrosion and salinity on CH 4/C 2 H 6 hydrate particle–surface interactions. Physical Chemistry Chemical Physics 19, 13307-13315.

[84] Zhang, F., Wang, X., Wang, B., Lou, X., Lipiński, W., 2022. Experimental and numerical analysis of CO2 and CH4 hydrate formation kinetics in microparticles: A comparative study based on shrinking core model. Chemical Engineering Journal, 137247.

[85] B. C. Juricek, D. E. Seborg, and W. E. Larimore, “Identification of the tennessee eastman challenge process with subspace methods,” *Control Engineering Practice*, vol. 9, no. 12, pp. 1337–1351, 2001.

[86] G. Li, S. J. Qin, Y. Ji, and D. Zhou, “Reconstruction based fault prognosis for continuous processes,” *Control Engineering Practice*, vol. 18, no. 10, pp. 1211–1219, 2010.

[87] M.G.DonandF.Khan,“Dynamicprocessfaultdetectionanddiagnosis based on a combined approach of hidden markov and bayesian network model,” *Chemical Engineering Science*, vol. 201, pp. 82–96, 2019.

[88] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, “Deep learning for anomaly detection: A review,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.

[89] C. Doersch, “Tutorial on variational autoencoders,” *arXiv preprint arXiv:1606.05908*, 2016.

[90] M.Hasan,J.Choi,J.Neumann,A.K.Roy-Chowdhury,andL.S.Davis, “Learning temporal regularity in video sequences,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 733–742.

[91] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, “A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 1409–1416.

[92] T. Schlegl, P. Seebo ̈ck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, “Unsupervised anomaly detection with generative adversarial networks to guide marker discovery,” in *International conference on information processing in medical imaging*. Springer, 2017, pp. 146– 157.

[93] W. Liu, W. Luo, D. Lian, and S. Gao, "Future frame prediction for anomaly detection–a new baseline," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6536– 6545.

[94] I. Golan and R. El-Yaniv, "Deep anomaly detection using geometric transformations," *Advances in neural information processing systems*, vol. 31, 2018.

[95] Z. Han, J. Zhao, H. Leung, K. F. Ma, and W. Wang, "A review of deep learning models for time series prediction," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 7833–7848, 2019.

[96] M. Galagedarage Don and F. Khan, "Process fault prognosis using hidden markov model–bayesian networks hybrid model," *Industrial & Engineering Chemistry Research*, vol. 58, no. 27, pp. 12 041–12 053, 2019.

[97] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[98] M. V. Garc ́ıa and J. L. Aznarte, "Shapley additive explanations for no2 forecasting," *Ecological Informatics*, vol. 56, p. 101039, 2020.

[99] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[100] R.Arunthavanathan,F.Khan,S.Ahmed,andS.Imtiaz,"Adeeplearning model for process fault prognosis," *Process Safety and Environmental Protection*, vol. 154, pp. 467–479, 2021.

[101] T. Kourti and J. F. MacGregor, "Multivariate spc methods for process and product monitoring," *Journal of quality technology*, vol. 28, no. 4, pp. 409–428, 1996.

[102] D. Sahoo, Q. Pham, J. Lu, and S. C. Hoi, “Online deep learning: Learn- ing deep neural networks on the fly,” *arXiv preprint arXiv:1711.03705*, 2017.

[103] D. T. Nguyen, S. Joty, M. Imran, H. Sajjad, and P. Mitra, “Applications of online deep learning for crisis response using social media informa- tion,” *arXiv preprint arXiv:1610.01030*, 2016.

[104] A. Saxena, J. Celaya, B. Saha, S. Saha, and K. Goebel, “Metrics for offline evaluation of prognostic performance,” *International Journal of Prognostics and health management*, vol. 1, no. 1, pp. 4–23, 2010.

[105] J. Z. Sikorska, M. Hodkiewicz, and L. Ma, “Prognostic modelling options for remaining useful life estimation by industry,” *Mechanical systems and signal processing*, vol. 25, no. 5, pp. 1803–1836, 2011.

[106] J. Z. Sikorska, M. Hodkiewicz, and L. Ma, “Prognostic modelling options for remaining useful life estimation by industry,” *Mechanical systems and signal processing*, vol. 25, no. 5, pp. 1803–1836, 2011.

[107] F.AhmadzadehandJ.Lundberg,“Remainingusefullifeestimation,”*In- ternational Journal of System Assurance Engineering and Management*, vol. 5, no. 4, pp. 461–474, 2014.

[108] X.-S. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, “Remaining useful life estimation–a review on the statistical data driven approaches,” *European journal of operational research*, vol. 213, no. 1, pp. 1–14, 2011.

[109] D. Abati, A. Porrello, S. Calderara, and R. Cucchiara, “Latent space autoregression for novelty detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 481– 490.

[110] K. E. S. Pilario and Y. Cao, “Canonical variate dissimilarity analysis for process incipient fault detection,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 12, pp. 5308–5315, 2018.

[111] J. J. Downs and E. F. Vogel, “A plant-wide industrial process control problem,” *Computers & chemical engineering*, vol. 17, no. 3, pp. 245– 255, 1993.

Abstract in Korean

최근 조선 해양 분야에서도 디지털화 (Digitalization)에 관한 관심이 점차 증대되고 있다. 특히 그 중에서도 디지털 트윈 (Digital Twin)은 가상의 모델에 실시간으로 실제 시스템의 데이터를 동기화 시켜서 모니터링할 수 있도록 하는 것으로, 디지털화와 관련된 여러 다양한 기술이 총집합된 주요 플랫폼이다. 디지털 트윈의 구성 요소는 크게 데이터, 통신, 모델 그리고 서비스 이렇게 네 가지로 구분된다. 이 중 데이터와 통신 부분은 사물 인터넷 (Internet of Things)과 빅데이터, 클라우드, 그리고 5G 등의 신기술의 동시적인 발달로 인해 많은 진전이 이루어졌으며 ISO 23247, ISO/IEC30172, 30173 등의 국제 표준까지 정립되었다. 반면 모델 및 서비스, 특히 서비스 부분에서는 상대적으로 발전이 더디게 일어나고 있다. 그 요인으로 기존의 시뮬레이션 모델 및 분석 기법으로는 실시간으로 수집되고 업데이트되는 센서 데이터를 다루고 처리할 수 있는 역량이 부족하다는 점, 또 최근 대두되고 있는 머신 러닝 및 데이터 기반 분석 기법이 해양 시스템에 바로 사용될 수 없어 추가적인 엔지니어링이 필요하다는 점 등을 꼽을 수 있다.

본 연구는 머신 러닝 모델을 기반으로한 이상 감지 모델, 센서 데이터 예측을 통한 위험 상황 감지 모델, 그리고 공정 예지 보전 모델을 해양 시스템에서 사용할 수 있도록 수정하고 또한 검증하였다.

첫째로, 공정 시스템에 적용할 수 있는 센서 간 연관도 기반 이상 감지 모델을 제안하였다. 본 모델은 MSCRED 모델을 수정한 것으로, 다변수

시계열 데이터 (Multivariate Time Series Data)를 이용하여 각 변수간 연관도를 계산한 2 차원 연관도 행렬을 시간에 따라 생성한 뒤 Conv-LSTM ED (Convolutional Long-short Term Memory Encoder Decoder) 모델을 이용하여 reconstruction 행렬을 얻어내어 입력값과의 차이를 통해 잔차 (residual) 행렬을 계산한다. 이렇게 계산된 잔차 행렬을 통해 이상 점수 (Anomaly score) 를 기존 방식과 다른 방식으로 계산하였다. 이는 해양 공정의 특성을 반영한 것으로 이상 상황이 아닐 때에도 공정 시스템 전체의 상황을 모니터링 할 수 있도록 하였다. 마지막으로 시계열로 된 이상 점수를 클러스터링 기법을 이용해 이상 케이스 (fault case) 분류를 수행할 수 있도록 하여 공정 상태를 모니터링 할 수 있도록 하였다. 제안된 이상 감지 모델을 검증하기 위하여 파일럿 스케일의 모노에틸렌글리콜 (Mono Etylene Glycol, MEG) 재생 공정을 이용하였으며, 4 가지의 정상 운전 데이터를 이용하여 모델을 학습시키고 1 가지의 운전 시작 상황, 4 가지의 이상 상황 데이터를 이용하여 모델을 검증하였다. 여러 정상 운전 데이터를 이용하여 학습을 진행한 결과 정확도가 높은 특정 정상 데이터를 기반으로 시계열 합성을 수행하여 모델의 성능을 높였다. 검증 결과 최고 88%에 가까운 정확도로 이상 탐지를 수행하였으며, 결과값으로 나온 이상 점수를 클러스터링한 결과 이상 상황별로 클러스터링이 이루어짐을 확인하였다.

둘째로, 다변수 시계열 센서 데이터를 기반으로 한 하이드레이트 생성 경향성 예측 및 transition, segregation 시점 예측을 수행하였다. 이는

일반적인 시계열 예측 모델인 Recurrent Neural Network를 이용하여 예측을 수행하되 PCA 기법을 이용하여 모델의 효율성을 높이고, 모델 간의 정확도 비교를 통해 가장 좋은 모델을 선택하였다. 본 프레임워크에서 적용된 특성 축소 및 전이 학습을 이용한 실험 데이터 학습을 이용함으로써, 다변수 센서 데이터를 활용한 실시간 예측이 가능함을 보였다.

셋째로, 이상 감지, 시계열 예측, 원인 분석 알고리즘을 조합하여 딥러닝 모델로만 구성된 예지 보전 프레임워크를 제안하였다. 본 연구에서는 전체 프레임워크를 딥러닝 기반의 모델로 구성함으로써 데이터의 활용도를 높이고자 하였다. 특히 시계열 예측의 경우 전이학습 기반의 Online Learning 을 적용하여 이상 상황이 발생하였을 때도 실시간으로 적응하여 학습할 수 있도록 함으로써 이상 상황이 흔하지 않은 해양 공정의 특성에 맞추어 모델을 구성하였다. 또한 Autoregressive Prediction을 통해 이상 상황이 감지되었을 때 언제쯤 이상 threshold에 도달하는지를 확인할 수 있도록 하였다.

Keyword : 디지털라이제이션, 디지털트윈, 머신러닝, 기계학습, 이상감지, 다변수시계열예측, 예지보전

Student Number : 2018-28756

Acknowledgment in Korean

저에게 있어 학위 과정은 한 명의 엔지니어로 성장하기 위한 과정이면서, 많은 사람들의 도움에 의지했던 시간이기도 했습니다. 1 인분을 할 수 있기 되기까지 얼마나 많은 사람들의 도움이 필요한지 자꾸 생각하게 되었습니다. 공부는 혼자 하는 것이라고들 말하지만 서로가 서로의 힘이 되어주는 사람들이 없었다면 저는 지금 이 자리에 없었을 것입니다. 먼저 제가 공부하고 연구할 수 있는 환경과 기회들을 마련해주신 서유택 교수님께 감사드립니다. 교수님 덕분에 새로운 과제들을 다양하게 접하고, 삶에서 마주치는 여러 일들을 다루고 견뎌내는 방법을 터득할 수 있었습니다. 바쁘신 와중에도 부족한 저를 지도해주시고 연구적으로 많이 도움을 주신 나종걸 교수님께도 감사드립니다. 또한 임영섭 교수님, 강상규 교수님, 우종훈 교수님께도 감사드립니다. 항상 시간을 내어 저를 가르쳐주시고 도와주신 김현호 박사님과 박기흠 박사님, 손영훈 박사님, 김자경 박사님께는 제가 큰 감사를 전하고 싶습니다. 항상 모범적이고 성실하며 따뜻한 선배님들 덕분에 연구실 생활을 열심히 해 나갈 수 있었습니다. 비슷한 시기에 연구실에 들어와서 힘든 시간을 함께한 종연, 새로운 것들이나 자주 알려주신 승만 오빠, 연구실 생활을 함께 할 수 있어서 정말 즐거웠습니다. 조용하지만 누구보다 든든하고 명석한 건우, 새로운 것을 배우기를 두려워하지 않는 진관, 육아와 학업을 병행하면서도 배움의 즐거움을 잃지 않는 준엽 오빠 모두 고맙습니다.

가족들에게도 저는 큰 빚을 지고 있습니다. 힘든 일들을 항상 도맡아 해온 엄마, 항상 고민을 나누고 서로 든든하게 지켜준 동생 승아, 철없는 누나와 지내느라 고생 많았던 막내 성민, 제가 길을 잃을 때마다 저를 항상 다잡아주는 하늘에 있는 아빠에게 사랑과 감사의 인사를 보냅니다. 힘든 시간이었지만, 모두가 있어 무너지지 않고 여기까지 올 수 있었습니다. 항상 아껴주고 걱정해준 이모와 이모부, 유선 언니와 유미 언니에게도 감사 드립니다. 언제나 안부를 물어봐주시고 돌봐주신 외할아버지께도 감사드립니다. 오랜 시간동안 음악을 통해 공동체가 되어준 밴드 친구들인 윤하, 영주, 승재, 구연에게도 고마움을 전합니다. 대학원 생활의 힘든 순간들에 누군가와 음악을 통해 마음을 나눌 수 있었던 것이 얼마나 큰 힘이 되었는지 모릅니다. 유정언니와 태영오빠, 지현언니, 의훈오빠, 해찬오빠에게도 감사인사를 드립니다. 오랜 시간 동안 옆에서 보면서 본받고 싶은 선배님들을 만나는 행운을 만나는 것은 쉬운 일이 아니지만 저에게는 그런 행운이 있었습니다. 멋지고 성실한 삶을 사시는 선배님들 덕에 저는 이전에는 몰랐던 새로운 세상을 많이 볼 수 있었습니다. 삶과 학업의 고민을 여럿 해소해주신 경서 선배와 서영 언니께도 감사합니다.

제가 미처 여기에 적지 못한 다른 많은 분들께도 감사 인사를 전합니다. 보잘것없는 성취이지만, 혼자였으면 불가능했을 일이라는 점을 시간이 갈수록 더 크게 느낍니다. 제 안에 여러분이 함께 있다는 것을 잊지 않고 앞으로도 바르게 나아가겠습니다.

감사합니다.

2023 년 02 월

이나영 올림

.