공학박사학위논문

딥러닝을 이용한 정상상태 데이터 설계 방법 개발: 디젤엔진의 과도상태 질소산화물 예측

Development of a Methodology for Designing a Steady-state Dataset using Deep Learning: An Application to Predict NOx Emissions under Transient Conditions of a Diesel Engine

2021 년 8 월

서울대학교 대학원

기계항공공학부

신 승 협

딥러닝을 이용한
정상상태 데이터 설계 방법 개발:
디젤엔진의 과도상태 질소산화물 예측

Development of a Methodology for Designing a Steady-state Dataset using Deep Learning: An Application to Predict NOx Emissions under Transient Conditions of a Diesel Engine

지도교수 민 경 덕
이 논문을 공학박사 학위논문으로 제출함

2021 년 4 월

서울대학교 대학원
기계항공공학부
신 승 협

신승협의 공학박사 학위논문을 인준함

2021 년 6 월

위 원 장 송 한 호

부위원장 민 경 덕

위 원 도 형 록

위 원 황 원 태

위 원 이 상 열

Abstract

Development of a Methodology

for Designing a Steady-state Dataset using Deep Learning:

An Application to Predict NOx Emissions

under Transient Conditions of a Diesel Engine

Seunghyup Shin

Department of Mechanical and Aerospace Engineering

The Graduate School

Seoul National University

Recently, the development of deep learning technology has been leading the rising of artificial intelligence. Deep learning technology is a different approach from conventional modeling methodologies and presents high prediction accuracy in various research fields, which attracted attention from many researchers. In automotive and internal combustion engine research, studies using deep learning are beginning to be actively performed. The conventional methodologies simplified engine phenomena and predicted target phenomena with several equations, which causes errors, and is a limitation of the conventional methodologies. Deep learning predicts phenomena by learning the internal relationship between variables of the data, and it can catch complex relationships that researchers could not recognize.

In this study, nitrogen oxides (NO_x) under transient conditions in a diesel engine were predicted through deep learning, which had limitations in prediction accuracy

using conventional modeling methodologies. This study included the entire process for constructing a deep learning model such as hyperparameter optimization, algorithm comparison, and dataset design for domain transfer.

Hyperparameters, the structure of the deep learning model, were automatically optimized by combining the Bayesian optimization and hidden-node determination logic. This optimization process could be conducted regardless of the type or number of data inputs and outputs. The hyperparameters to be optimized were the learning rate, learning rate decay, batch size, number of hidden layers, and number of nodes in 1st hidden layer. Because the Bayesian optimization method utilized the previous information based on the Bayesian rule during the optimization process, it was more effective and more accurate than the conventional grid search and random sampling. The hidden-node determination logic used the number of hidden layers and the number of nodes in 1st hidden layer to arrange the node sequence of hidden layers in an arithmetical sequence. It was possible to prevent information loss due to sudden changes in nodes, and control the number of exponentially increased iterations when each layer and node were individually optimized.

A study on the structure of a deep learning model suitable to predict NO_x in transient conditions was also conducted. The accuracy and computation time of the deep neural network (DNN) model and long short-term memory (LSTM) model were evaluated from the viewpoint of real-time prediction. The LSTM model presented higher prediction accuracy than that of the DNN model, but it needed more calculation time, which is not suitable for real-time prediction. Through data preprocessing, the accuracy of the DNN model increased to a level similar to that of the LSTM model, and its advantage of the calculation speed is maintained, which is suitable to predict NO_x emissions under transient conditions through quasi-stationary.

Finally, a study was performed to overcome the domain constraint to predict NO_x using deep learning. Previously, the domains of both training data and target data should be identical. In order to predict transient NO_x emissions using the model trained by the steady-state data, the experimental dataset under the steady-state conditions was designed and used for model training. From the analysis of the engine behaviors under transient conditions, the intake air mass, intake pressure, injection pressure, and main injection timing were set as swing variables, and data were acquired by experiments in certain ranges for operating points. This allowed the steady-state data to be extended to the transient prediction. Temperature swing experiments were additionally designed and performed to consider the effects of intake and coolant temperature in transient conditions. This process was proposed as a method for designing steady-state experimental conditions for predicting transient conditions, and the results were also provided. By overcoming the domain constraint of deep learning models using this method, a deep learning model for the NO_x emission prediction can be used at the overall development stages of engines including design, evaluation, and prediction of whether real engines for measurement of transient NO_x emissions exist.

In addition, the proposed procedure of the steady-state experimental design for predicting transient NO_x emissions could be applied to real driving conditions and other mechanical systems.

Keywords: Deep learning, Nitrogen oxides, Transient prediction, Diesel engine, Hyperparameter optimization, Design for steady-state dataset

Student Number: 2016-31756

Contents

List of Tables

List of Figures

Acronym

1D	One-dimensional
AI	Artificial intelligence
ANN	Artificial neural networks
BMEP	Brake mean effective pressure
BSFC	Brake specific fuel consumption
CO	Carbon oxide
DNN	Deep neural networks
ECU	Engine control unit
EGR	Exhaust gas recirculation
ELU	Exponential linear unit
ICE	Internal combustion engine
MFB50	Mass fuel burned 50%
MSE	Mean squared error
LSTM	Long short-term memory
NO_x	Nitrogen oxides
PM	Particulate matter
ppm	Parts per million
R^2	Coefficients of determination
RDE	Real driving emission
ReLU	Rectified linear unit
RMSE	Root mean squared error
RNN	Recurrent neural networks
THC	Total hydrocarbon
WLTP	Worldwide harmonized light vehicles test procedure

Chapter 1. Introduction

1.1 Background and motivation

1.1.1 Status overview of internal combustion engines

At present, environmental problem has been one of the major global concerns. Internal combustion engine (ICE) vehicles came to a crisis of their existence because they were pointed out of the cause of air pollution, and other energies for transportation such as electricity and hydrogen are rising as alternatives to fossil fuels.

National governments in Europe had announced phase-out plans for ICE vehicles [1]. According to the plan, conventional cars will be replaced by other types of energy from 2030 to 2040. As a response to policies, car manufacturers established their future portfolio for electrification until the end of 2030.

The Korean Society of Automotive Engineers announced their predictions of the market share of vehicle energy sources in 2030. They predicted approximately 65% of vehicles will be internal combustion vehicles, 28% for hybrid including plug-in hybrid, and 7% for the battery [2]. It indicates that the ICE will remain the main power source for the mid-term future. Research for the emission reduction technology in the internal combustion engine should be simultaneously conducted with the development of technologies related to battery or hydrogen power sources.

The profitability of the electric vehicle would reduce as decease of the government subsidy for it in the future. However, the manufacturers cannot stop the

investment in the new power source. Profits from the sales of the conventional vehicle will play a role as a main cash cow for the manufacturers. That's why it is important to conduct research for the technology of the internal combustion engines.

1.1.2 Issues of the internal combustion engines

Reducing both fuel consumption and emission have been major issues in the development of ICE technologies for a long time. The regulations of fuel efficiency and emission are becoming stricter than in the past. The regulation of fuel efficiency imposes a penalty on the manufacturers when they cannot meet the regulation, but the emission regulation prohibits vehicle sales on manufacturers, which is more critical for the companies.

With viewpoints of the test standard, the regulations have become harsher. Test cycles were updated to reflect real driving conditions which consist of complex operating profiles or on-road situations of vehicle driving. In 2017, the new regulation cycles were introduced by the EU government, which were the worldwide harmonized light vehicles test procedure (WLTP) and real driving emission (RDE) [3] as shown in Figure 1.1. The WLTP and RDE have longer and more complex driving profiles than the previous test cycles like the new European driving cycle or EPA federal test procedure known as FTP-75. Therefore, it is difficult to satisfy regulations and predict emission under these test cycles.

The emission regulation mainly defines the quantities of nitrogen oxides (NO_x) and particulate matter (PM) from ICE vehicles. NO_x plays a role as a precursor of the fine dust, and PM directly contributes to the dust.

Among them, various approaches for NO_x reduction have been attracted by many research groups because PM can be reduced by the diesel particulate filter [4]. There are two strategies to reduce NO_x emissions divided into combustion and post-combustion. In the viewpoint of combustion, the combustion temperature which higher causes more NO_x formation was decreased by exhaust gas recirculation (EGR) [5]. EGR is the technique that a portion of exhaust gas is recirculated into the intake port to increase the heat capacity of the intake air, and it decreases maximum in-cylinder temperature during the combustion process. For the post-combustion approach, lean NO_x trap and selective catalyst reduction are adopted as after-treatment systems to reduce system-out emission [6, 7]. These after-treatment systems convert NO_x emissions to harmless gases with high efficiency; however, it needs additives such as urea which maintaining cost makes increase. For this reason, reducing engine-out NO_x is still significant.

1.1.3 Rising of deep learning

Recently, deep learning has been attracted by researchers and the public as a symbol of artificial intelligence (AI). Deep learning is leading the rising of AI, and its accuracy is much higher than that of the conventional approaches in various research topics especially image and natural language processing [8]. In 2016, AlphaGo [9], a deep learning model, won the Go game against Lee Se-dol who is a top human player, and this event aroused public interest in deep learning and triggered the beginning of the third spring of AI.

The root of deep learning was started in the 1950s by developing perceptron which was the first neural network model. In 1969, Minskey et al., mathematically proved a perceptron is impossible to solve the XOR problem [10]. The authors insisted that the multi-layer perceptron could be a solution to the XOR problem, but there was no way to train the multi-layer perceptron. As all the researchers accepted the claim, researches on deep learning had stopped from late 1960, which was the first ice age of AI.

In 1986, the advent of the backpropagation method [11] finished the first ice age and the research on the artificial neural network was booming in the late 1980 and the beginning of 1990. Until then, every coefficient of the models was determined along the forward direction. However, the forward update could not be complex models such as neural networks. With the backpropagation method, the error between the result and truth is utilized to tune the neural network along a backward direction. The research on the artificial neural network was booming in the late 1980 and the beginning of 1990.

However, the boom of neural networks had ended in late 1990, the second ice age. There were two reasons for the second ice age. First was the vanishing gradient problem [12]. If the network has many layers, the error from the backend has no effect on the tuning of the upper layers. As the error goes step by step from the end, the impact of the error reduces because the other errors accumulate. The second was the lack of a theoretical base. Researchers had knowhows to build neural networks, but there was little theoretical basis. Also, other machine learning techniques such as support vector machines and random forest appeared at that time.

About 10 years later, at the end of the second ice age, deep learning was started with a pre-training method to solve the vanishing gradient problem [13, 14]. Deep neural networks could be trained with better initial values of the model obtained by the pre-training of each layer.

In 2012, the result of the ImageNet Challenge was a breakthrough that deep learning got public attention. ImageNet was an open-source data for image processing including 14 million images, and 20 thousand categories [15]. Deep learning showed much higher accuracy rather than the previous image processing techniques in 2012. In 2017, deep learning had an error of less than 3%. Researchers defined the human index at 5% that a Stanford student trains images, then he classifies the images with a 5% error. As a result, deep learning classified the images better than a Stanford student.

Nowadays, researches on natural language processing are recognized that the results achieved a breakthrough [16]. OpenAI GPT-3 consists of 175 billion model parameters and its performance overwhelms the previous language processing models.

1.2 Literature review

1.2.1 Conventional models for NO_x emission prediction

NO_x emission prediction by means of simulation models was conducted at the development stage of commercial engines to evaluate the feasibility of the engine for satisfying the regulation. The embedded NO_x model, coupled with the calibration multiplier maps by GT-Power, was applied during the development of the 12.7 L heavy-duty engine [17], and the mass fuel burned 50 (MFB50) based model was applied to the 11.0 L diesel engine evaluation[18]. A virtual engine model was developed to calibrate combustion and emission by simulation before conducting engine experiments [19].

Prediction of NO_x emissions via models has been studied by many research groups because it can facilitate anticipative control for engine-out NO_x emission reduction. However, many physical and chemical processes should be considered to predict NO_x emissions, and they are challenging and vital to predict NO_x emissions in order to reduce them. Egnell introduced a multizone approach to calculate NO [20]. Temperature and chemical species were calculated based on the lambda value of each zone. A sensitivity study was performed to identify the main parameters of NO_x estimation [21]. In-cylinder air mass and total mass were recognized as essential parameters of NO_x emissions. A semi-physical modeling approach was studied under steady-state conditions [22]. An empirical formula consisting of engine power, EGR rate, rail pressure, fuel injection quantity, and air mass was developed, whose coefficient of determination (R^2) value was 0.91. The cycle-to-cycle phenomena of NO_x emissions were investigated based on a crank-angle-based NO_x emission

prediction model [23]. Leach et al. combined NO and NO_x measurement data over crank angles and one-dimensional (1D) exhaust flow data for the model. They compared the maximum cylinder pressure and NO_x emissions to understand cycle-based NO_x emissions. However, the investigation was conducted for only four experimental cases.

Finesso et al. predicted NO_x formation under steady-state and transient conditions [24]. A combustion submodel was utilized to calculate the heat-release rate and in-cylinder pressure, and a thermodynamic submodel was used to compute the temperature value, which was a source term of in-cylinder NO_x emission concentration. The R coefficient under the new European driving cycle was between 0.90 and 0.97. A study for simplification of a NO_x emission model to increase calculation speed was conducted by extracting fundamental physical phenomena [25]. However, multiple injections or fundamental changes in the combustion characteristics were not considered by the model. Additionally, a model capable of predicting NO_x emissions using the composition ratio and maximum combustion temperature, was developed, which was then validated by means of computational fluid dynamics results [26]. The model was applied in real time and produced results with high accuracy ($R^2 = 0.98$). However, the EGR model in the ECU resulted in an error. Lee et al. introduced fast-response thermocouples to compensate for this limitation [27]. Thermocouples were installed at the intake manifold, in front of the EGR valve, and behind the intercooler. Nevertheless, the tip, which is the measuring portion of the thermocouple, was very narrow, and suffered from durability, and was not appropriate for practical uses at vehicles or long-term experiment. A mean value NO_x emission model was also developed focusing on ECU installation [28]. A look-up table was used to realize the mapping between the NO output and correction

coefficients, and the transient conditions were included by means of dynamic factors. Although the model operated quickly and its shape was suitable for ECU application, the methodology did not significantly differ from manual calibration that is widely adopted and recognized as inefficient. A real-time virtual NO_x sensor was developed for light-duty diesel engines [29]. The sensor employed physical and chemical phenomena to predict NO_x emissions and covered both steady and transient conditions. However, it required a cylinder pressure sensor to gather additional information not captured by the ECU.

NO_x formation was a result of the combination of complex physical and chemical phenomena. Previous studies adopted equation-based approaches for NO_x emission prediction. Phenomena were simplified to build a prediction model, which inevitably limited the accuracy. Additional sensors were required to compensate for the decrease in accuracy, however, they rendered the models difficult to be employed for commercial uses.

1.2.2 Deep learning research for prediction of ICE

Before the emergence of deep learning, various machine learning algorithms were emerged and utilized to realize AI. Machine learning has been studied for a long time (since the 1950s), and it has been applied to research in the fields of combustion and other engine research, especially problems that are difficult to solve by the conventional modeling approach. Schaffernicht et al. adopted a machine learning algorithm to optimize the combustion process of a hard-coal power plant [30]. They developed a control strategy based on statistics and non-linearity. The machine

learning algorithm was utilized to quantify uncertainties in turbulence and combustion through a stochastic model [31]. Researchers attempted to predict the anisotropy of turbulence in a non-equilibrium boundary layer flow. Furthermore, there was an attempt to predict the homogeneous charge compression ignition (HCCI) combustion phase using a machine learning algorithm [32]. R^2 of the model was not enough for the accurate prediction, however, the accuracy was better than in previous research. Deep belief network, a machine learning technique, was applied to diagnose faults in airplane engine components [33]. Several fault scenarios were defined by researchers and trained by the model. Hanuschkin et al. conducted feature extraction, including in-cylinder flow and tumble features, to classify high indicated mean effective pressure (IMEP) cycles using a machine learning approach [34].

Recently, deep learning has been recognized as a mainstream of AI, because deep learning has shown much higher accuracy than other machine learning approaches. In the automotive research field, several studies adopting deep learning algorithms have been conducted. Fuel economy and emissions of a hybrid vehicle in transient cycles have been predicted with artificial neural networks (ANN) [35]. Researchers have attempted to predict vertical and horizontal tire contact forces between vehicle tires and roads [36]. Lock-up clutch control of transmission was also a target subject using deep learning, and a deep convolutional neural network was utilized to classify time-series transmission measurement data [37]. Seo et al. studied ANN for low-friction road estimation to determine a road's status without additional sensors [38]. ANN coupled with a 1D simulation tool (Amesim) was studied to predict the cooling system, indoor humidity, and air conditioning system of the vehicle [39]. Long short-term memory (LSTM) was applied to predict time-related data of vehicles. The effect of the real-world signals on the prediction window was performed to predict vehicles'

velocity using LSTM [40]. Kim et al. adopted LSTM to predict hybrid electric bus speed [41]. The algorithms of machine learning and deep learning were compared to predict the vehicle velocity, and the results were that the LSTM algorithm showed the highest accuracy among the algorithms [42].

For internal combustion engines, several studies using deep learning have been conducted in the last 20 years. In 1999, a basic study was introduced to estimate cylinder pressure and engine torque using a neural network called 'neural estimator' [43]. Multi-layer perceptron was divided into regions and tuned systematically, then it was combined with a conventional modeling approach. In 2007, an ANN was used to predict the performance and emissions of a gasoline engine [44]. Target outputs were brake specific fuel consumption (BSFC), brake thermal efficiency, as well as exhaust temperature and emissions. Operating ranges of data for the training and test were 200 rpm intervals between 1,000 and 2,400 rpm and 20, 30, and 40 Nm torque, but they did not cover the overall operating conditions of the gasoline engine. In 2010, modeling of transient NO_x emission prediction was introduced using recurrent neural networks (RNN) [45]. The authors concluded that the prediction error in NO (in grams) throughout cycles was lower than 2%, but the gap between trajectories of both measured and predicted was significant.

Some research groups actively studied ANN for diverse fuel type engines, such as diesel engines using waste cooking biodiesel [46], CNG-diesel engine [47], and spark ignition engines fueled by alcohol-gasoline blends [48]. Their common limitation was that the number of datasets they used was under 100, and thus too small to train and validate the neural networks.

In 2016, a review paper on the potential and application of ANN was a turning point to broaden the perspective of engine researchers [49]. The authors stated that ANN can be used for virtual engines, control strategy, on-board diagnostics, etc. Although they focused on the utilization of ANN, it would be valid if the terminology of ANN in this paper broadly contained DNN.

Since 2017, meaningful trials for engine status prediction using neural networks have been disclosed. Finesso et al. controlled brake mean effective pressure (BMEP) and MFB 50 of a diesel engine in real-time [50]. The input parameters were engine speed, rail pressure, intake pressure/temperature, intake O_2 concentration, exhaust temperature, injection timing, and quantity of each pulse. The target outputs were BMEP and MFB50. Rayavalasa et al. predicted part-load in-cylinder pressure of a gasoline engine [51]. They trained coefficients of Wiebe function at full-load points, then estimated coefficients at part-load points. The concept of expansion of prediction range using limited data was valuable, but there was an unclear observation that burn rate coefficient at both full-load and part-load were the same when other conditions like engine speed, throttle, and spark timing were identical. A heavy duty engine model predicted engine parameters and emissions, but emission accuracy was limited [52]. A learning rate adaptive algorithm and cross-validation were employed for volumetric efficiency modeling, but the R^2 of the model was approximately 0.85 [53]. Models for performance and emissions including BSFC, Torque, NO_x, carbon oxide (CO), total hydrocarbon (THC), and CH_4 of hydrogen enriched compressed natural gas (HCNG) engine were built using ANN [54]. The network structure of each output was individually considered, but models could be combined with some model groups if DNN was used. Alcan et al. studied the NO_x emission model for steady and transient cycles via sigmoid based nonlinear autoregressive with exogenous input (NARX) [55].

The level of accuracy could be increased as accuracy at steady conditions was approximately 80% – 90%, and accuracy at transient conditions was somewhat lower (75% – 85%). A neural network was also applied to improve a 1D turbocharger model with the training of turbine outlet temperature error [56]. Dynamic of gasoline direct injector was one of the target systems of ANN, and injector lift was predicted by injector differential voltage signal analysis [57]. Knock, performance, combustion, and emissions of a gasoline engine were predicted simultaneously by adopting the DNN structure [58]. The single model covered combustion results including maximum cylinder pressure, maximum pressure rise rate, and crank angle at maximum pressure rise rate, performance results such as BMEP, BSFC, and emissions such as brake specific nitrogen oxides, and brake specific carbon oxide. However, the results were based on steady-state conditions, not transient conditions which reflected real-driving situations. The EGR rate of a diesel engine was predicted under steady-state conditions using deep learning [59] to complement the limitation of [26, 27]. However, accurate EGR prediction achieved by [59] did not mean that NO_x emissions could be predicted accurately by [26] because the EGR rate is one of the major factors to affect NO_x emissions. ANN models were studied for the prediction of performance, emission, and vibration in a compressed ignition engine using alumina nano-catalyst added to diesel-biodiesel blends [60]. The accuracy results of these prediction models were high, but they were required to validate under transient conditions because models were based on steady-state conditions data.

Most previous studies in academia related to the engine field utilized ANN for neural network structure. ANN consists of 1 input layer, 1 output layer, and 1 hidden layer. It can display accurate performance but is limited in terms of dimension expandability and complexity. Therefore, DNN is the right solution instead of ANN,

as DNN contains many hidden layers and theoretically has no dimension and complexity limitation.

Furthermore, most previous studies studied machine learning, and deep learning specifically, with MATLAB. Although MATLAB is easy to use, it has limitations of expandability and standardization of deep learning algorithms. Tensorflow and Keras are the mainstream platforms for deep learning research, and MATLAB is not preferred by deep learning researchers [61]. Keras is a high-level API for deep learning, and it covers various AI engines such as Tensorflow, Theano, CNTK, MXNET, etc., as common backend functions. Here, Keras (using a Tensorflow backend) was utilized as a tool for deep learning.

1.2.3 Domain discordance between steady-state and transient conditions

Researchers calibrate the engine maps at the individual operating points under steady-state conditions, and the engine interpolates the values between the operating points if it operates at the arbitrary points. Then, the engine is operated under transient conditions to evaluate the satisfaction of the regulations defined by the transient cycles. During this process, the domain discordance between the steady-state and transient conditions causes difficulties for the researchers to calibrate engine maps and predict the results of the target cycles.

The previous NO_x prediction approaches had the same limitations mentioned above. The conventional approaches for NO_x emission prediction were the utilization of engine maps. Previous studies adopted correction processes to use steady-state

maps of an engine for transient conditions. Pelkmans et al. developed the quasi-stationary method by a combination of steady-state maps and dynamic behavior [62]. They introduced four parameters to add dynamic factors to steady maps, i.e., steady-state emission rate, jump fraction, a time constant, and transient emission. Quasi-stationary models included transient corrections and delay models for converting steady-state maps to the transient applications [63]. It considered a delayed air mass flow caused by turbocharger lag. Other correction methodology combined instantaneous engine speed and torque, recent past history (hysteresis), and engine operating temperature [64]. Predicting NO_x emissions using engine maps was simple, and applicable with minimal effort for modeling. However, the engine map approach needed additional considerations and calibration processes [65] to use the map of the steady-state to predict the transient conditions. The correction factors for the domain transfer should be defined by the research, so it was not reliable because human factors could be involved. Moreover, this approach was not valid when parameter values of the engine control unit (ECU) were changed to vary operating strategies of the engine because the values of engine maps were fixed.

Most deep learning studies including previous works mentioned in section 1.2 were applied to predict engine results for the same domain with training data. It means that deep learning models predicted phenomena under steady-state conditions using data obtained under steady-state conditions. Also, target values under transient conditions were trained with the data of transient conditions such as studies performed in Chapters 4 and 5, which predicted transient NO_x emissions via training with transient data.

This domain limitation, in which transient data was required to predict transient phenomena, restricts the usage of prediction models because the model can predict

engine phenomena only when real engines are built and prepared to be operated. However, if the prediction model for transient phenomena can be trained by steady-state data, the model can be used at a development state when engines are not built, and engine data can be obtained by simulation under steady-state conditions. It will expand the model availability throughout engine development stages from design to real driving.

Unfortunately, there were few studies of deep learning on different domains for the engine application.

1.3 Objectives

In this study, the engine-out NO_x emissions under transient conditions of a diesel engine were predicted via deep learning methods. NO_x is one of the main emissions from diesel engines, but there were rarely accurate models predicting NO_x emission under transient conditions. Conventional modeling approaches simplified the internal relationship of the input parameters while organizing equations based on the physical and chemical fundamentals. In contrast, deep learning methods can extract all internal meanings from the data, therefore, the accuracy of deep learning models would be achieved higher than that of the equation-based models especially under transient conditions.

To overcome the limitation of domain discordance between calibration based on engine map under steady-state conditions and operation under transient conditions, the main objective of this study is to develop a methodology for the prediction of transient NO_x emissions using the steady-state data. This domain transfer enables deep learning models to predict NO_x emissions under transient conditions whether engines are installed in a test dynamometer.

Eventually, the steady-state dataset was designed to train a deep learning model to predict transient NO_x emissions. Before that, studies regarding hyperparameter optimization, algorithms, and data preprocessing were conducted to develop essential methodologies to apply to the design for the steady-state dataset. These methodologies were developed and evaluated under transient conditions, then developed methodologies were adopted to study on design of the steady-state dataset for domain transfer.

Main studies to develop a deep learning model for the prediction of engine-out NO_x emissions under transient conditions in a diesel engine are:

1. Hyperparameter optimization: development of hyperparameter optimization of DNN models via Bayesian optimization and hidden-node determination logic to optimize structures of deep learning model for automatic, effective, and robust ways (transient conditions).

2. Algorithm and data preprocessing: comparison to accuracy and calculation time between DNN and LSTM models for transient NO_x emissions and evaluation of the effects of time-series data preprocessing to DNN and LSTM models (transient conditions).

3. Design of steady-state dataset: analysis of engine behaviors under transient conditions and suggestion of the design methodology of steady-state experiments to predict transient NO_x emissions (domain transfer).

This dissertation consists of seven chapters; Chapter 1 is an introduction, Chapter 2 describes deep learning algorithms utilized in this study, Chapter 3 introduces experimental setup, Chapter 4 contains hyperparameter optimization method, Chapter 5 studies comparison between DNN and LSTM, and time-series data preprocessing, Chapter 6 covers a design methodology of the steady-state experiments for transient NO_x emission prediction, and Chapter 7 is a conclusion.

Some portions of the content in this dissertation were already published by the author in Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering [58], [66], and Engineering Applications of Artificial

Intelligence [67]. Chapter 1 was partly based on [58], [66] and [67], Chapter 4 is based on [67], and Chapters 2 and 5 are based on [66].

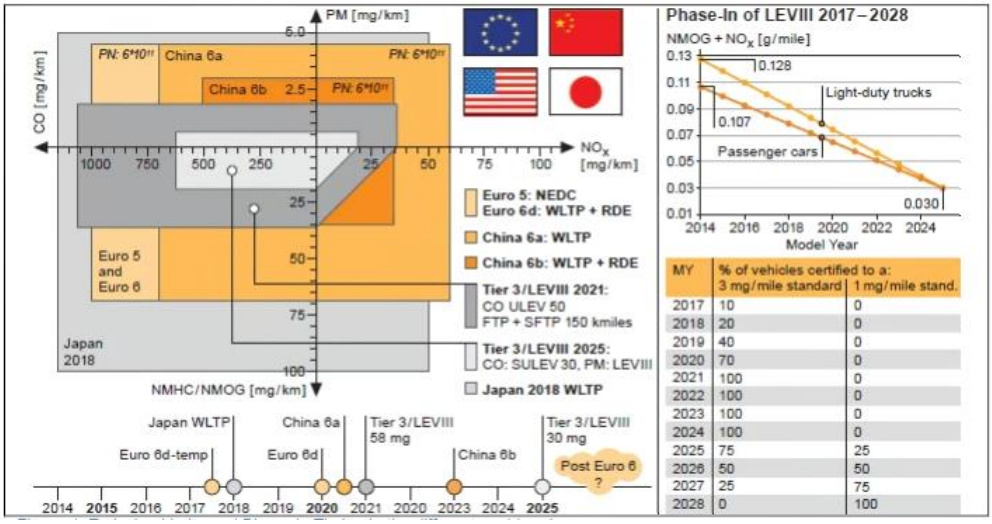


Figure 1.1 Emission limits and phase-in timing in the different world regions [3].

Chapter 2. Deep learning algorithms

DNN and LSTM algorithms were introduced in this study.

DNN neglects the time effects of transient data with the assumption that the result of a transient condition is the sum of individual results of the steady-state condition. In contrast, LSTM considers time-related information while predicting NO_x emissions. This implies that the present result is a combination of past results and present conditions.

DNN constitutes a base algorithm within deep learning and is simply defined as a multi-layered structure of ANN. A DNN comprises two or more hidden layers with one input layer and one output layer. Figure 1.2 shows the single neuron, and deep learning models consist of numbers of neurons arranged in several layers with multiple in a line. Predicted result $\hat{y}$ is derived by updated weights (w_i) and bias (b_i) of each node and sum of them as Eq. 1.1, and the activation function is located before $\hat{y}$ to insert nonlinearity into the predicted value [68].

$$\hat{y} = Activation(\sum_{i=1}^{k} w_i x_i + b_i) \tag{1.1}$$

Here, x_i represent a node input.

Each weight and bias were optimized during the training process to minimize the target objective function for the training process, which was given as

$$O(x) = \underset{w_i, b_i}{argmin}\, \mathbb{E}(y, \hat{y}) \tag{1.2}$$

where *O(x)* represents the objective function for training, and $\mathbb{E}(y, \hat{y})$ represents an error function between target values and predicted values defined differently by the problems. In this study, $\mathbb{E}(y, \hat{y})$ was the mean squared error (MSE), as indicated by Eq. 1.3 [69], because NO_x emission prediction is a regression problem.

$$\mathbb{E}(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y - \hat{y})^2 \tag{1.3}$$

The rectified linear unit (ReLU) function [70] is widely used as an activation function to organize DNN and is given by Eq. 1.4. The ReLU solves the vanishing-gradient problem; however, it was recently reported that there is a dying ReLU problem at negative inputs because ReLU neglects ouputs from negative x values.

$$f(x) = \begin{cases} x & , \ if\ x > 0 \\ 0 & , \ if\ x \leq 0 \end{cases} \tag{1.4}$$

In this study, to overcome this limitation, an exponential linear unit (ELU) function was introduced [71]. Eq. 1.5 gives the ELU function; the outputs of this function at negative x values differed from those of the ReLU function.

$$f(x) = \begin{cases} x & , \ if\ x > 0 \\ \alpha(exp(x) - 1), & if\ x \leq 0\ (\alpha < 0) \end{cases} \tag{1.5}$$

where α represents a hyperparameter of the ELU function.

In contrast to the DNN, LSTM is a type of RNN. RNNs are used to calculate and predict time-series data and utilize information from the past, which results in outputs of hidden layers becoming inputs. LSTM was first suggested by Hochreiter et al. [72] and can be employed to solve problems associated with the RNN involving long-term dependencies [73].

Figure 2.2 presents a diagram of the LSTM structure. As shown, there are three gates inside the LSTM module and one cell state, which is represented by the upper line in the LSTM module. Past information flows along with the cell state and plays a vital role in overcoming long-term dependency related problems associated with the RNN.

Eqs. 1.6 – 1.11 are used for the LSTM calculations performed by the module. Eq. 1.6 is associated with the forget gate layer and Eqs. 1.7 and 1.8 are related with the input gate layer. Eq. 1.9 is for updating the cell state, and Eqs. 1.10 and 1.11 are for the output gate layer. According to Eq. 1.6, the result of the forget gate (f_t) is a value between 0 and 1, which represents ignored information from the cell state. In the input gate, according to Eqs. 1.7 and 1.8, the sigmoid function determines which information from the input is updated to the cell state. Then, the tanh function generates a vector $\widetilde{C_t}$ for the following calculation. In Eq. 1.9, C_t, a new cell state, is updated using C_{t-1} and the results of the input gate. In the output gate, according to Eqs. 1.10 and 1.11, a sigmoid function determines which part of the cell state is to be output. This output is multiplied by the result of a tanh function of the cell state, i.e., tanh(C_t). Finally, the values of C_t resulting from the cell state update, and h_t resulting from the output gate transfer to the next LSTM module are output [41].

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{1.6}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{1.7}$$

$$\widetilde{C_t} = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{1.8}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \tag{1.9}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{1.10}$$

$$h_t = o_t * tanh(C_t) \tag{1.11}$$

Here, σ represents a sigmoid function, * represents pointwise multiplication, W represents the weight of each layer, and b represents a bias term.

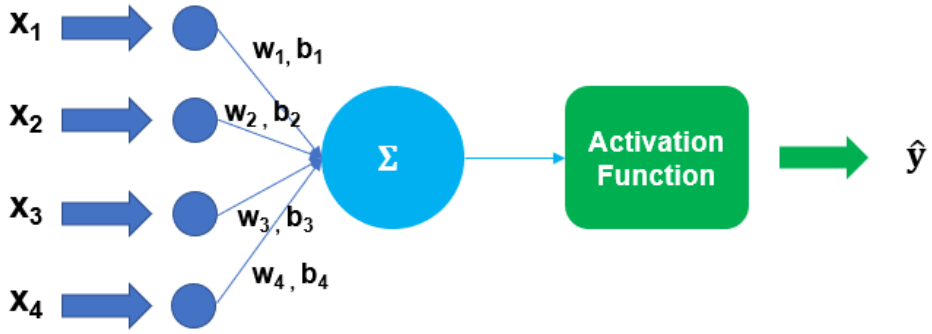


Figure 2.1. Structure of a single neuron.

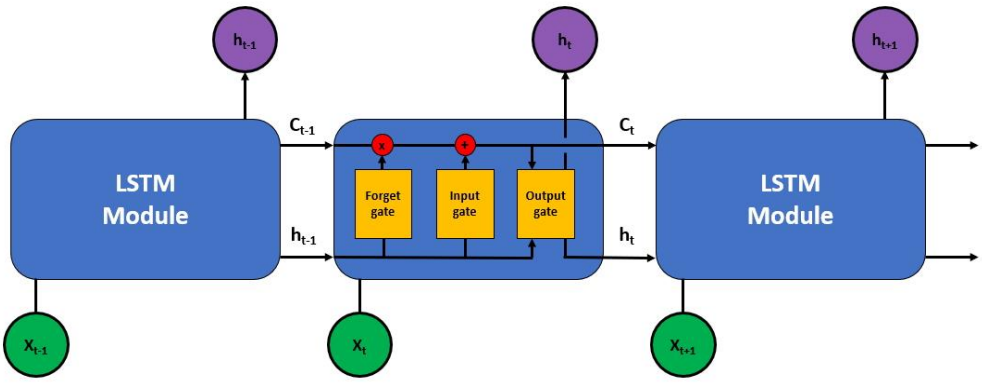


Figure 2.2 Diagram of a LSTM structure.

Chapter 3. Experimental setup

3.1 Hyperparameter optimization and deep learning algorithms

The experimental setup described in this section was applied to the studies performed in Chapters 4 and 5.

The detailed specifications of the engine used in this study are presented in Table 3.1. The experimental engine was a 1.6 L four-cylinder diesel engine with a compression ratio of 17.3. It was connected to a 340 kW alternating current dynamometer manufactured by AVL (Austria), which was able to operate under steady-state conditions and transient cycles (and specifically the WLTP cycle necessary for this study).

The experimental setup is shown in Figure 3.1. An ECU controlled the engine state and measured the engine data simultaneously. Moreover, an ETAS ES591 was used to connect the ECU and to a host personal computer. A Cambustion CLD500 Fast NO_x Analyzer was installed to measure the engine-out NO_x emissions in real time.

The WLTP cycle was employed as the representative transient cycle for deep learning. Figure 3.2 shows the vehicle speed profile of the WLTP cycle. The total time of the cycle was 1800 s, and the data of the ETK-ECU and CLD500 were measured every 0.1 s. Therefore, each cycle comprised 18,002 points, including start and end time steps. Four WLTP cycles, tested on different days, were employed to consider daily variations. Three types of WLTP experiments were conducted to prevent the DNN models from being overfitted and to provide various data results. Two cycles

were conducted according to the original ECU setting, which are termed “Normal 1” and “Normal 2” herein. The other two cycles were operated with a shifted air mass flow. The air mass flow of one cycle shifted by +3% compared with the normal cycle, and a –3% air mass flow was applied to the other cycle. They are termed “+3%” and “–3%,” respectively, in the following discussion. Figure 3.3 shows the NO_x measurement results at each WLTP cycle, and Table 3.2 presents data statistics of each WLTP cycle. As compared to the “Normal 2” cycle, NO_x emission level of the “+3%” cycle had a slightly higher and a cumulative NOx emission was also 8.9% higher because the EGR rate decreased as air mass flow increased. On the other hand, NOx emissions from the “–3%” cycle were lower than other cycles due to the increased EGR rate.

The datasets consisted of 14 columns including the data logging time, comprising ECU data, and engine-out NO_x data captured by the CLD500. The time interval of the data was 0.1 s, and 12 types of features were measured by the ECU, which were used as input features. Table 3.3 presents 12 features of the ECU. The engine-out NO_x was the target of the supervised learning and the output of the model.

The dataset contained 4 WLTP cycles, and the total number of data points was 72,008, which was randomly divided into three categories, i.e., training, validation, and test sets; 60% of the data, i.e., 43,205 points, were assigned to the training set, and the validation and test sets were each allotted 20% of the data. The training set was used during the model training; the validation set was used for the optimization process; the accuracy of the optimized model was evaluated using the test set. Table 3.4 shows the data assignment for the DNN model.

Type	Four-cylinder CI engine
Bore [mm] × Stroke [mm]	77.2 × 84.5
Displacement volume [L]	1.582
Compression ratio	17.3
Maximum power [kW]	94
Maximum torque [kg·m]	28.5

Table 3.1. Engine specifications.

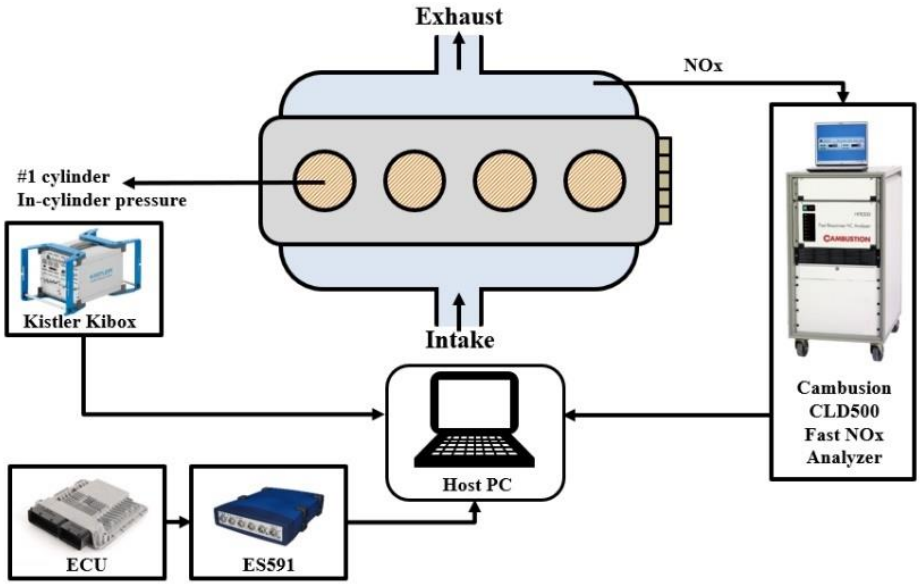


Figure 3.1. Diagram of experimental setup.

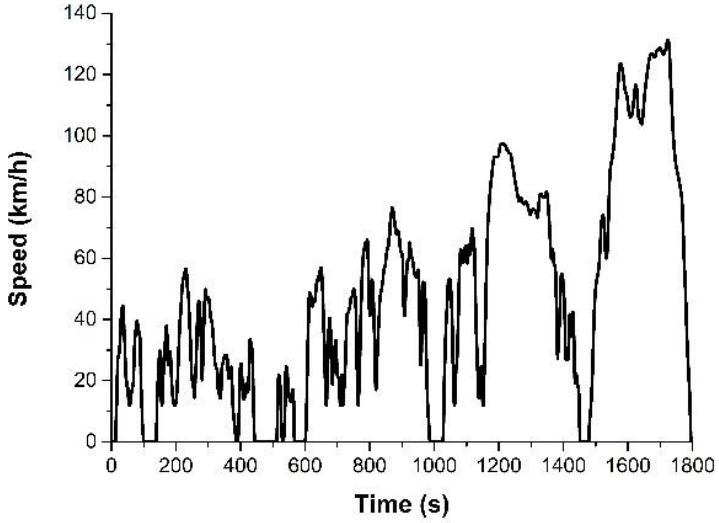
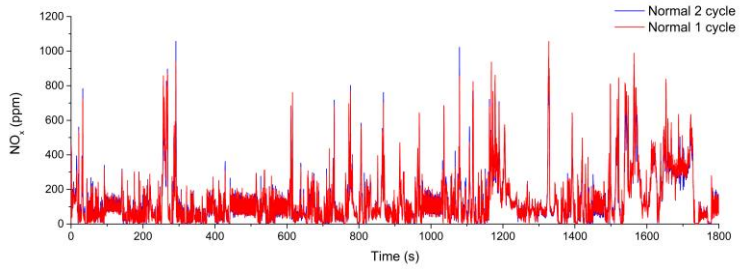
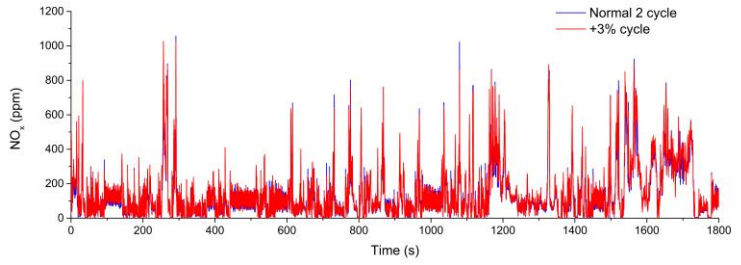


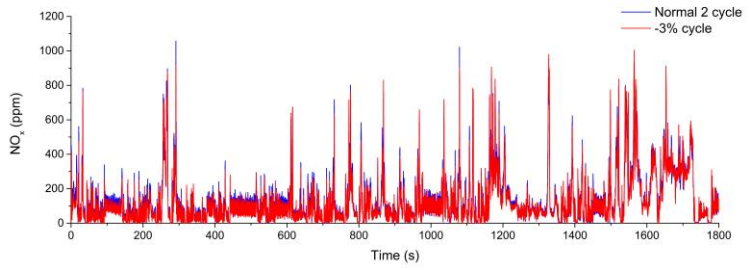
Figure 3.2. Vehicle speed profile of WLTP cycles [29].



(a)



(b)



(c)

Figure 3.3. NO_x measurement results of WLTP cycles: (a) Normal 1, (b) +3%, and (c) −3% cycles compared with Normal 2.

	Normal 1	Normal 2	+3%	–3%
Maximum NO_x [ppm]	1055.5	1057.0	1026.6	1003.8
Minimum NO_x [ppm]	0	0	0	0
Average NO_x [ppm]	131.5	129.8	138.5	117.2
Standard deviation [ppm]	143.2	137.9	146.0	133.5
Cumulative NO_x [normalized, compared to Normal 2]	1.022	1.000	1.089	0.909

Table 3.2. Data statistics of Normal 1, Normal 2, +3%, and –3% cycles.

Air mass per cylinder	EGR rate	Average engine speed
Counter segments of crankcase differential pressure signal	Main injection timing	Current injection quantity
Lambda value	Boost pressure	Pilot injection quantity
Pilot/main injection ratio	Rail pressure	Variable swirl valve position

Table 3.3. Features measured by the ECU.

WLTP Cycle	Normal 1	Normal 2	+3%	-3%
Number of points	18,002	18,002	18,002	18,002
Dataset	**Training (60%)**	**Validation (20%)**	**Test (20%)**	
Number of points	43,205	14,402	14,401	

Table 3.4. Data assignment for DNN model.

3.2 Steady-state experimental design

Table 3.5 presents the specifications of the engine used to acquire the steady-state and transient data for this study. The displacement volume of the engine was 2.151 L, and the compression ratio was 16.0.

The experimental setup is shown in Figure 3.4. The engine was controlled by an ECU, and a HORIBA MEXA 7100DEGR was utilized to measure the engine-out NO_x emissions, which were automatically logged by the dynamometer control software, AVL PUMA. In a steady-state experiment, the ECU data and dynamometer data, including the NO_x emissions, were acquired for 30 s, and the average values were saved to the host computer.

Four WLTP cycles were obtained from the transient experiment. The cycles were varied by controlling the temperatures of the dynamometer at the point after the intercooler, called the "Intake temperature" in this work, and at the coolant. Table 3.6 lists the setpoint for the intake and coolant temperatures. WLTP 1 – 3 were the cases in which the intake temperature changed while the coolant temperature remained the same. WLTP 4 had a different coolant temperature setpoint than WLTP 1. As the setpoint of the intake temperature decreased, more cooling was performed during the cycle. Figure 3.5 shows NOx measurement results and Table 3.7 presents data statistics of WLTP 1–4 cycles. Compared to WLTP 1, WLTP 2 cycle emitted 8.3% more cumulative NO_x mass caused by increased intake temperature. In cases of WLTP 3 and 4, cumulative NO_x masses were less because of lower intake temperature for WLTP 3 and lower coolant temperature for WLTP 4. Maximum NOx levels of WLTP 3 and 4 were smaller, and standard deviations were also smaller than those of WLTP 1.

WLTP 1 was utilized as the target variable regarding the results and discussion of the model trained with the map experimental data (Section 6.3.1) and designed experimental data including map and swing experiments (Section 6.3.2). WLTP 2 – 4 were used to evaluate the model accuracies according to the temperature, and design additional experimental points to reflect temperature in Section 6.3.3.

Basically, the dataset consisted of 10 input features and one target variable. The target variable was the NO_x emissions measured in the experiment, and it was the output result of the model. The input features are presented in Table 3.8. They were the intake air mass, EGR rate, engine speed, fuel quantity, main injection quantity, pilot injection quantity, main injection timing, injection pressure, intake pressure, and lambda. In Section 6.3.3, the intake temperature and coolant temperature were added to secure the high accuracies of the model under different temperature conditions.

Engine specifications	
Engine type	Four-cylinder CI
Bore [mm]	83.0
Stroke [mm]	99.4
Displacement volume [L]	2.151
Compression ratio	16.0
Maximum power [kW]	154.4
Maximum torque [kg·m]	45.0

Table 3.5. Engine specifications.

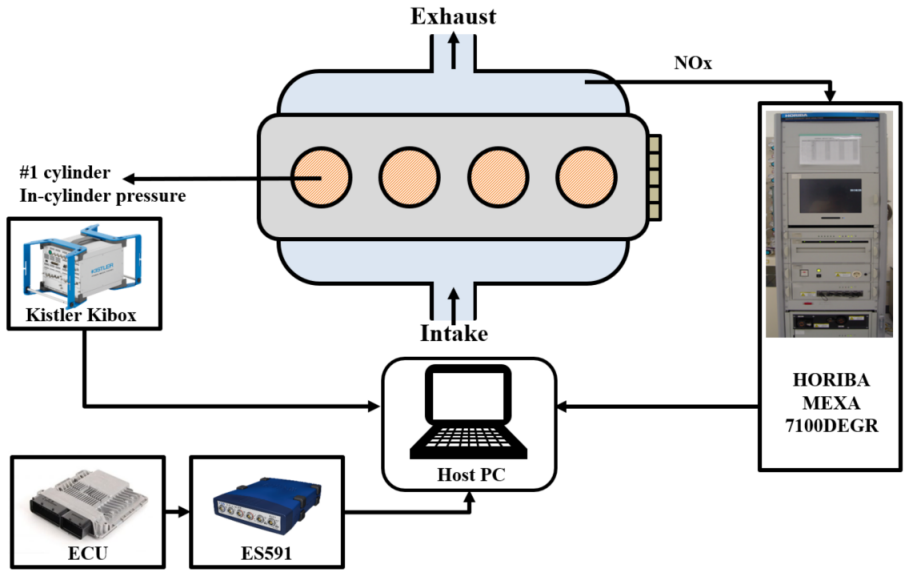
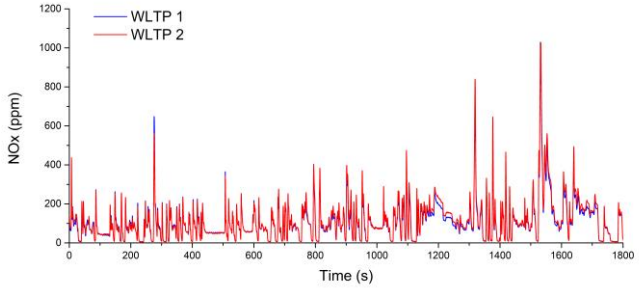


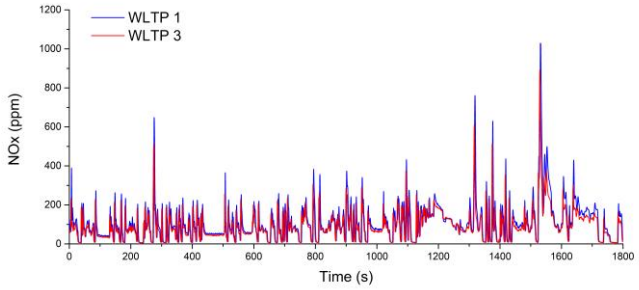
Figure 3.4. Experimental system.

	Intake temperature (°C)	Coolant temperature (°C)
WLTP 1	35	85
WLTP 2	45	85
WLTP 3	20	85
WLTP 4	35	40

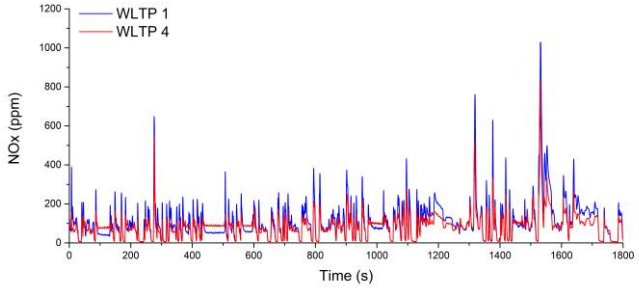
Table 3.6. Setpoint of dynamometers for transient experiments.



(a)



(b)



(c)

Figure 3.5. NO_x measurement results of WLTP cycles: (a) WLTP 2, (b) WLTP 3, and (c) WLTP 4 compared with WLTP 1.

	WLTP 1	WLTP 2	WLTP 3	WLTP 4
Maximum NO_x [ppm]	1028.9	1025.3	891.9	828.9
Minimum NO_x [ppm]	4.5	5.0	5.0	4.6
Average NO_x [ppm]	107.3	113.9	92.6	85.8
Standard deviation [ppm]	95.8	102.3	77.9	65.0
Cumulative NO_x [normalized, compared to WLTP 1]	1.000	1.083	0.855	0.768

Table 3.7. Data statistics of WLTP 1–4 cycles.

Intake air mass [mg]	EGR rate [%]
Engine speed [rpm]	Fuel quantity [mg]
Main injection quantity [mg]	Pilot injection quantity [mg]
Main injection timing [CA]	Injection pressure [hPa]
Intake pressure [hPa]	Lambda
Intake temperature [°C] (in Section 6.3.3)	Coolant temperature [°C] (in Section 6.3.3)

Table 3.8. Input variables of dataset.

3.3 Computing environment

The computing hardware configuration comprised a central processing unit (CPU, Intel® Core™ i7-9700K @ 3.60 GHz, 32 GB RAM) running Windows 10 OS. For hyperparameter optimization performed in the following sections was accelerated by graphical processing units (NVIDIA Tesla V100, 32 GB) adopted in Intel® Xeon® Gold 5120 @ 2.20 GHz and 180 GB RAM. The programming language was Python 3.70, and Keras using a Tensorflow backend was used as a deep learning library.

Chapter 4. Hyperparameter optimization using the Bayesian optimization and hidden-node determination logic

In this chapter, a hyperparameter optimization algorithm was developed to optimize the structure of the DNN model. Figure 4.1 shows the flowchart pertaining to this study; a DNN was employed to predict engine-out NO_x at transient conditions in WLTP cycles of a diesel engine. The target hyperparameters for the optimization were the number of hidden layers, number of nodes in the first hidden layer, learning rate, learning rate decay, and batch size. The Bayesian optimization method and hidden-node determination logic were combined to optimize the hyperparameters, and they were compared with grid search, random sampling, and the Bayesian optimization method, separately. Compared with other optimization methods, the Bayesian optimization method considers the results of previous iterations when the next iteration values are determined. Logical equations were developed using the number of hidden layers and the number of nodes in the first hidden layer to determine the number of nodes in every hidden layer. The accuracy of the hyperparameter optimized model was evaluated by indices such as R^2, RMSE, and MAE.

The contributions of this study are as follows:

1. The hyperparameter optimization method of the DNN can be used to determine the structure of models, regardless of the input features and output targets.

2. Deep learning can be used to predict emissions under transient conditions that are difficult to estimate by using conventional approaches. The accuracy of the model is comparable to that of a physical NO_x sensor.

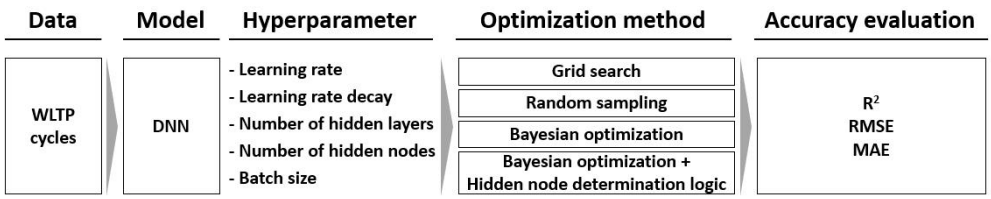


Figure 4.1. Flowchart of hyperparameter optimization.

4.1 Hyperparameter optimization methods

4.1.1 General optimization methods

The widely used optimization methods are manual search, grid search, and random sampling [74].

Manual search explores target objectives based on the human experience. Instinctively, the results of a manual search do not assure optimization because this method depends on chance.

Grid search divides searching ranges into the same interval and swings the values of each parameter independently. It is simple to implement, but the result is not optimum if the optimal point is located between grid points. In addition, the number of iterations increases exponentially as parameters are added. This is called the "curse of dimensionality" [75]. If n parameters exist for optimization and each parameter contains k searching values, the number of trials is k^n. Each searching value of the parameter λ is defined as shown in Eq. 4.1.

$$\lambda_i = \lambda_{min} + (i-1)\frac{(\lambda_{max} - \lambda_{min})}{k-1} \qquad (i = 1, 2, 3, \cdots, k) \qquad (4.1)$$

where λ_{min} and λ_{max} are the minimum and maximum searching values, respectively.

Random sampling selects the next iteration values based on a random possibility. It can reduce meaningless iterations compared with grid search; hence, it is reliable and effective in high-dimensional spaces [75]. Figure 4.2 presents a comparison of dimension effectiveness between grid search and random sampling. As nine trials

were applied to both methods, only three data were obtained along the x-axis (green function) by grid search, and nine data by random sampling. Random sampling introduces the searching probability, but prior results are ignored when the values for the next iteration are selected.

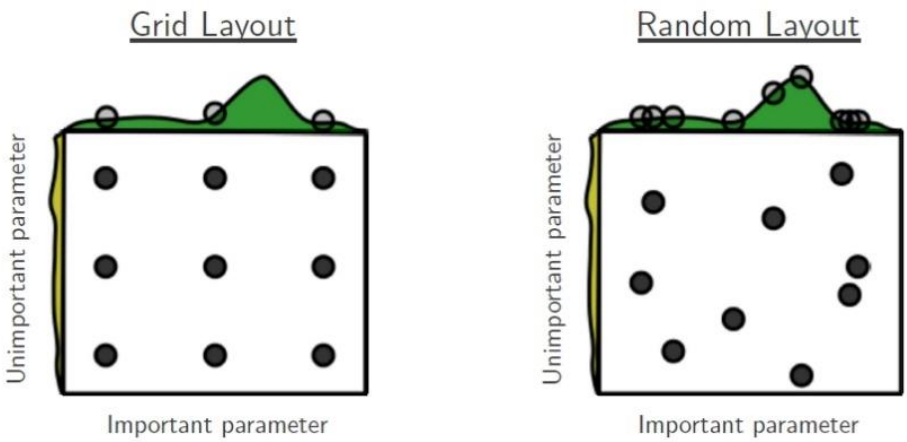


Figure 4.2. Comparison of dimension effectiveness between grid search and random sampling [75].

4.1.2 Bayesian optimization method

The Bayesian optimization method conducts optimization based on Bayes' rule of conditional probability. Bayes' rule employs prior knowledge to calculate the posterior possibility. Eq. 4.2 represents Bayes' rule [76].

$$p(w|D) = \frac{p(D|w)\, p(w)}{p(D)} \tag{4.2}$$

where *w* is an unobserved quantity, *p(w)* the prior distribution, *p(D|w)* the likelihood, *p(w|D)* the posterior distribution.

The Bayesian optimization method considers the results of previous iterations when selecting values of the next iteration because it is based on Bayes' rule. Hence, it can approach the optimal point more effectively than random sampling. Figure 4.3 shows a procedure example of the Bayesian optimization method.

The Bayesian optimization method can be implemented using two submodels; the surrogate and acquisition model. The surrogate model stochastically estimates the target function by using previous results based on the Gaussian process. In Figure 4.3, the upper lines of each graph are the results of the surrogate model. No uncertainties at the measured points, and uncertainty is greater as the distance from the measured point increase. The acquisition model recommends the next point for an iteration by using the results of the surrogate model. The acquisition model tends to increase near the maximum measured point, and the maximum posterior uncertainty is transferred from the surrogate model. The next value is determined from the relationship between the measured values and uncertainty, and the optimization point is determined by repeating this process and reducing uncertainty.

Brochu et al. provided detailed theoretical backgrounds and equations of the Bayesian optimization method [77]. As shown in Eq. 4.3, the Gaussian process (*GP*) over the function *f(x)* is specified by the mean function (*m*) and covariance function (*k*).

$$f(x) \sim GP(m(x), k(x_i, x_j)) \tag{4.3}$$

The acquisition function of the Bayesian optimization method depends on the previous observations, and it is maximized over iterations. As observations $D_{1:t} = [x_{1:t}, f(x_{1:t})]$ are accumulated, the prior distribution is combined with the likelihood function $P(D_{1:t}|f)$. A new trial point is selected to minimize the uncertainty and maximize the potential value of the function, as shown in Eq. 4.4.

$$x_{t+1} = \underset{x}{\operatorname{argmin}}\, \mathbb{E}(\| f_{t+1}(x) - f(x*) \| \, |D_{1:t})$$

$$= \underset{x}{\operatorname{argmin}} \int \| f_{t+1}(x) - f(x*) \| \, P(f_{t+1}|D_{t:1}) \, \mathrm{d}f_{t+1} \tag{4.4}$$

The improvement function shown in Eq. 4.5 was introduced by Mockus et al. [78] to solve Eq. 4.4 with respect to the expected improvement $f(x^+)$.

$$I(x) = max(0, f_{t+1}(x) - f(x^+)) \tag{4.5}$$

This means that *I(x)* is positive when the prediction is higher than the best value observed previous iteration; otherwise, *I(x)* is 0. The new query point is defined by the expected improvement as shown in Eq. 4.6.

$$x = \underset{x}{argmax}\, \mathbb{E}(max[0, f_{t+1}(x) - f(x^+)] \, |D_t) \tag{4.6}$$

The likelihood of improvement (I) can be calculated by the normal density function with $\mu(x)$ and $\sigma^2(x)$, and the expected improvement is the integral of the function. Eq. 4.7 expresses the analytic result of the expected improvement.

$$EI(x) = \begin{cases} \left(\mu(x) - x(x^+)\right)\Phi(Z) + \sigma(x)\phi(Z) if \sigma(x) > 0 \\ 0 if \sigma(x) = 0 \end{cases}$$

$$Z = \frac{\mu(x) - f(x^+)}{\sigma(x)} \tag{4.7}$$

where $\Phi(\cdot)$ is the standard normal distribution of a cumulative distribution function.

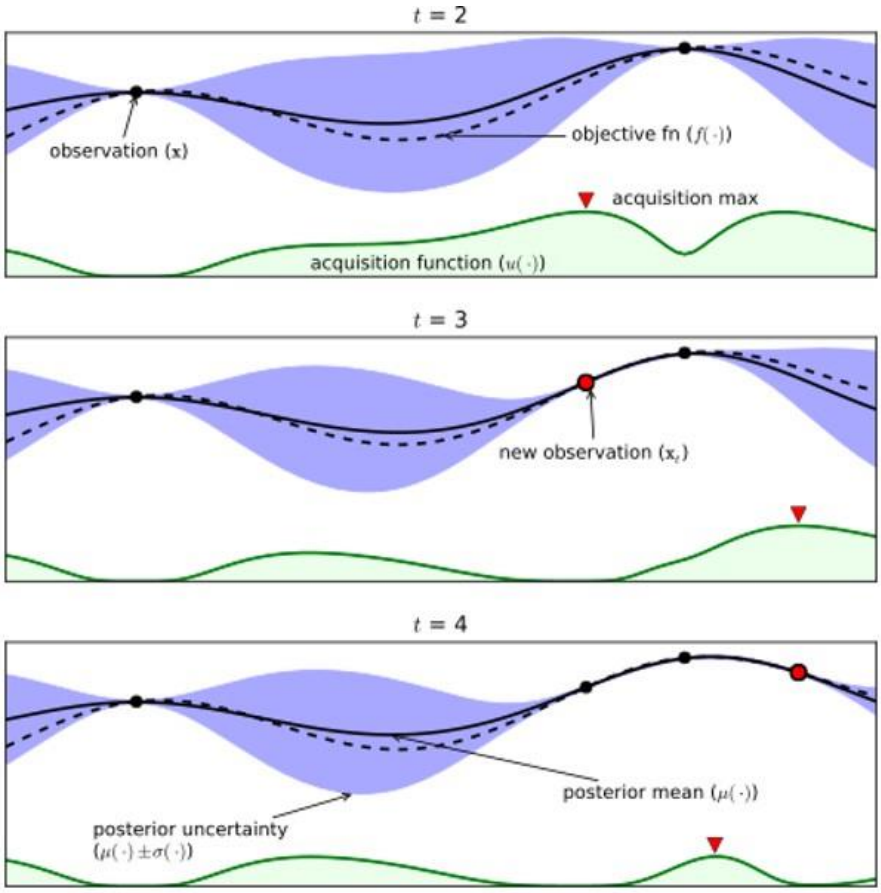


Figure 4.3. Procedure example of Bayesian optimization method according to iterations [77]. The dotted line indicates a real function, the solid line an estimated function, the shade around the line the standard deviation, and the lower green line the probability of the optimal point.

4.1.3 Target hyperparameters and the hidden-node determination logic

A DNN model comprises one input layer, one output layer, and multiple hidden layers. The hyperparameters derived from this structure are the number of hidden layers and the number of nodes in each hidden layer. Other hyperparameters are the learning rate, learning rate decay, batch size, activation function, and dropout rate [74].

In this study, the Bayesian optimization method was introduced to optimize the hyperparameters of a DNN automatically. The target hyperparameters for optimization were the number of hidden layers, number of nodes in each hidden layer, learning rate, learning rate decay, and batch size. Other hyperparameters were fixed because their optimal values were known, which allows for more alternatives in the future. For example, the activation function was set as an ELU function [71] which is an improvement from the rectified linear unit function. The dropout rate was fixed at 0 by adopting batch normalization [79]. The Adam optimization algorithm [80] was used as an optimization function during training.

The number of nodes in each hidden layer was replaced by the number of nodes in the first hidden layer. If the number of nodes in each hidden layer is determined independently during the optimization process, the optimization parameters and the number of iterations will increase exponentially. This curse of dimensionality increases computational cost and time.

Therefore, logics were designed to determine the number of nodes in each hidden layer using the number of hidden layers and the number of nodes in the first hidden layer.

As shown in Table 4.1, the logic compared the number of nodes in the first hidden layer (n_1) and the input dimension (N_i).

The objective of the logic was to prevent an abrupt change in the number of nodes between the hidden layers, which is one of the causes of information loss within DNNs. After determining the number of hidden layers and number of nodes in the first hidden layer, an arithmetical sequence is the best option to arrange the nodes according to the hidden layers. This is because other approaches result in more significant changes than the arithmetical sequence in at least one hidden layer. Therefore, the DNN shapes derived by the logic were based on an arithmetical sequence.

In addition, the shape of the networks exhibited a continuous decrease or a decrease after an increase. The shape of a middle concave was not considered because the data information can be lost when the compressed data for decreasing nodes are increased again. This type of structure is employed to reduce dimensionality in an autoencoder [81].

If n_1 is smaller than N_i, then the logic determines whether n1 is a multiple of the number of hidden layers (h). Logic 1 was applied when n_1 was a multiple of h, and Logic 2 was operated otherwise. By contrast, if n_1 is larger than N_i, the logic is divaricated regardless of whether h is even. (Logic 3 was used for the answer “yes,” and Logic 4 for the answer “no”).

Table 4.2 lists detailed equations of the hidden-node determination logic; n_1 and h were selected by the optimization model. All logics were based on an arithmetical progression, and the prevention of a significant change in nodes causing information loss was considered. Logic 1 presented the simplest scenario, in which the number of hidden nodes decreased along an arithmetical sequence. If N_i is assumed to be 20, the output dimension is 1 for all scenarios to understand the logic. If n_1 is 16 and h is 4, the nodes of the hidden layers are arranged as 16-12-8-4 according to Logic 1. Logic 2 follows decreasing arithmetic as well, but one hidden layer is added to prevent a significant decrease in the last hidden layer by processing residual nodes. When n_1 is 18 and h is 4, the total number of hidden layers (N_h) is 5, which reduces abrupt node change at the last layer. The node sequence is 18-15-12-9-6 for this scenario.

For both Logics 3 and 4, the number of hidden layers decreasing by node ($N_{h,d}$) was larger than the number of hidden layers increasing by node ($N_{h,i}$) because the input dimension was larger than the output dimension, implying that considering a significant node change during a decrease was more important than that during an increase. During an increase, the number of increasing nodes (s_i) was defined by the difference between n_1 and N_i. During a decrease, the number of decreasing nodes (s_d) was defined as the maximum number of hidden nodes (n_m) divided by the value added with 1 and $N_{h,d}$. If 1 is not added to $N_{h,d}$, the number of nodes may be zero in some scenarios. For example, if n1 is 24 and h is 6, the nodes are arranged as 24-28-32-18-13-8 followed by Logic 3. When n1 is 24 and h is 7, the nodes are arranged as 24-28-32-26-20-14-8 according to Logic 4.

Input:

- Input dimension (N*i*)
- Number of nodes in the first hidden layer (n_1)
- Number of hidden layers (h)

Output:

- Number of hidden nodes in each hidden layer

def hidden-node determination logic (N_i, n_1, h):

if input dimension is smaller (or equal) than the number of nodes in the first hidden layer ($N_i \leq n_1$):

if number of nodes in the first hidden layer is divided by the number of hidden layers (n_1 % h == 0):

Logic 1

else:

Logic 2

else:

if number of hidden layers is divided by 2 (h % 2 == 0):

Logic 3

else:

Logic 4

return the number of hidden nodes in each hidden layer

Table 4.1. Pseudo code of hidden-node determination logic.

Logic 1	Logic 2
$N_h = h$ $s_d = n_1 / N_h$	$N_h = h + 1$ $s_d = n_1 / N_h$
Logic 3	**Logic 4**
$N_{h,i} = (h / 2) - 1$ $N_{h,d} = h - N_{h,i}$ $\therefore\ N_{h,i} + 2 = N_{h,d}$ $s_i = n_1 - N_i$ $n_m = n_1 + (N_{h,i} - 1) \times s_i$ $s_d = n_m / (N_{h,d} + 1)$	$N_{h,i} = (h / 2) - 0.5$ $N_{h,d} = h - N_{h,i}$ $\therefore\ N_{h,i} + 1 = N_{h,d}$ $s_i = n_1 - N_i$ $n_m = n_1 + (N_{h,i} - 1) \times s_i$ $s_d = n_m / (N_{h,d} + 1)$

Table 4.2. Hidden-node determination logic.

4.2 Optimization model setup

As mentioned in section 4.1.3, the target hyperparameters for optimization were the number of hidden layers, number of nodes in the first hidden layer, learning rate, learning rate decay, and batch size. Table 4.3 shows the optimization maximum and minimum bounds of each hyperparameter and whether the parameter was an integer or a real number. The maximum value of the batch size was equal to the total number of data points.

The improvement by the optimization methods was evaluated through grid search, random sampling, and the Bayesian optimization method for comparison. Hence, all nodes arranged in each hidden layer were the same as the number of nodes in the first hidden layer. In addition, the hidden-node determination logic was applied to the Bayesian optimization method to verify the effect of the logic. The grid search, random sampling, Bayesian optimization method (same nodes), and Bayesian optimization method with hidden-node determination logic are compared in the next section.

The iteration number of both random sampling and Bayesian optimization was 300, and that of grid search was 243 (three grids for every five hyperparameters, 3^5) because 1,024 iterations with four grids for the hyperparameters required significant computation time. Three grid points were set as the minimum, maximum, and median values of the hyperparameter ranges.

Each iteration was the result of 1,000 epochs because more time would be required if a higher epoch was set for each iteration. The objective function of the

optimization was the mean square error of the validation set, which is called the validation loss, and minimizing the validation loss was an optimization target.

To compare the model accuracies after the optimization, four models obtained separately by grid search, random sampling, Bayesian optimization method, and Bayesian optimization method with hidden-node determination logic were individually trained again to obtain the best models. In this process, the early stopping callback [82] was adopted to attain the highest accuracy in training regardless of the epoch setting. Figure 4.4 shows the concept of early stopping. The training loss decreased as the epochs increased because a model was continuously tuned to fit the training dataset. Therefore, the validation loss was the criterion to decide whether a model was overfitted or underfitted. A minimum validation loss implies that the model demonstrates the best performance at the epoch, called the "optimal epoch." If an epoch is smaller than the optimal epoch, then the model is underfitted and has the potential to be more accurate. However, if the epoch is larger than the optimal epoch, the model is overfitted. Without early stopping, the epoch number should be set as one of the hyperparameters to be optimized. Before training, an epoch number that is sufficiently large to be in an overfitting range is input as a model setting. While training, the early stopping callback stores the model's weights at the optimal epoch. The model further trains for the duration of the patience and then restores the weights when completing a run. The role of the patience is to prevent an abrupt stop by small fluctuations in the validation loss during training. Finally, the test accuracy of the model is assessed.

Hyperparameter	Minimum	Maximum	Type
Number of hidden layers	2	9	Integer
Number of nodes in the first hidden layer	5	25	Integer
Learning rate	10^{-7}	10^{-2}	Real number
Learning rate decay	10^{-9}	10^{-5}	Real number
Batch size	100	72,008	Integer

Table 4.3. Maximum and minimum bounds of hyperparameter optimization.

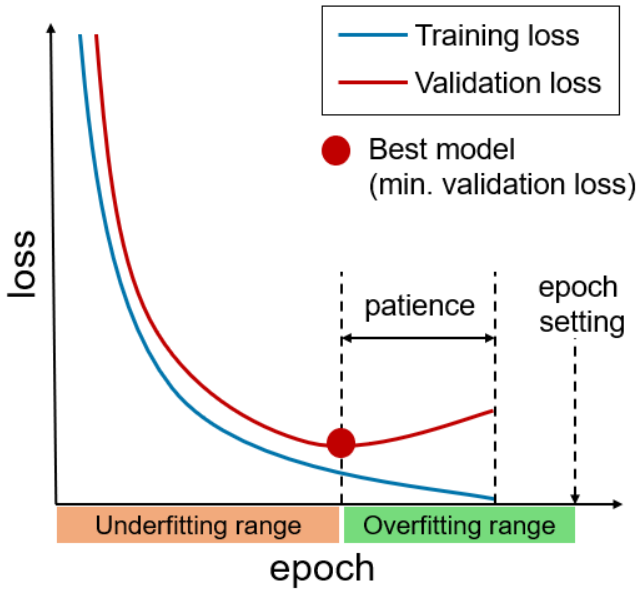


Figure 4.4. Concept of early stopping.

4.3 Results of the developed method

4.3.1 Optimization process

Optimization methods such as grid search, random sampling, Bayesian optimization method, and Bayesian optimization method with hidden-node determination logic were used in this study. Figures 4.5–4.8 show results of the validation loss (optimization target) and hyperparameters, including the batch size, number of nodes in the first hidden layer, number of hidden layers, learning rate, and learning rate decay, which depict the respective processes of each method.

The grid search was operated as a stepwise approach according to the hyperparameters. The optimization target was generally low, i.e., in the range of 10^{-2} of the learning rate; however, this process could not search the value between grids. The random sampling investigated various combinations of each value between the upper and lower limits of the hyperparameters. As shown in Figure 3-9(d), the sampling might be distributed unequally even though the random possibility was set as a uniform distribution between the limits. Random sampling is not a promising method because the opportunity of chance exists.

As shown in Figures 4.7 and 4.8, the Bayesian optimization method compensated for the limitations of previous methods, where various combinations of parameters were investigated, and all-around values were used between the upper and lower limits.

Table 4.4 shows a list of the best validation losses and hyperparameters of models from each method. The validation loss was the result of 1,000 epochs during the optimization process. Early stopping was employed to develop the best model

regardless of the epoch to compare the accuracies between the hyperparameter sets, as discussed below. Table 4.4 includes the information of the hidden node sequence. The first "12" indicates the inputs described in Table 3.4, and the final "1" is the output, i.e., the NO_x (ppm). Furthermore, batch normalization was not aimed for optimization; the ELU was used as an activation function, and Adam was used as an optimization function for the training process according to the epoch. This optimization differed from the hyperparameter optimization presented herein.

The calculation time can be theoretically analyzed between the optimization methods. Among the methods, grid search requires the longest calculation time because it performs the minimum batch points numerous times. Grid search mandatorily includes the minimum values of each hyperparameter, and repeatedly explores the points. Specifically, the batch size is one of the important factors affecting calculation time. A lower batch size means a much longer calculation time. Other methods do not explore the endpoint as frequently as the grid search. The calculation time of random sampling is decided randomly, and its calculation time cannot be guaranteed. In the worst-case scenario, a total of n! iterations are required to determine the optimal point for random sampling. However, the Bayesian optimization method can attain the optimal points faster than random sampling because it considers previous results to select the next point. Points unrelated to the optimal points are skipped by the Bayesian optimization method; therefore, the optimal path to the optimal point can be determined effectively.

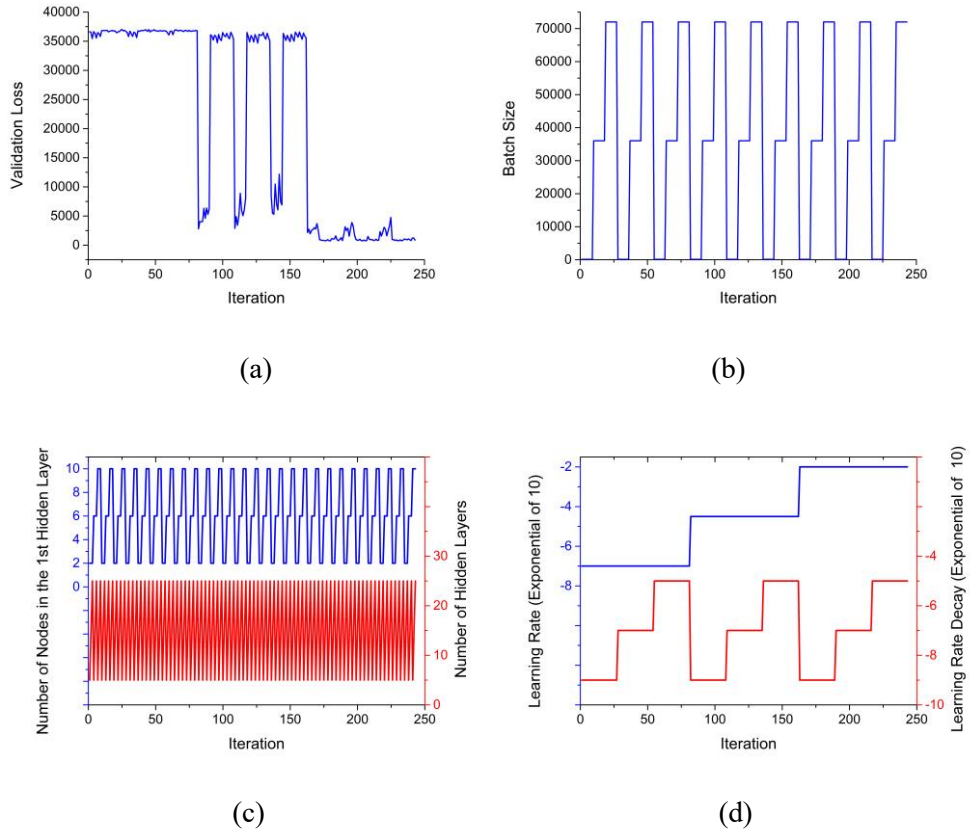


Figure 4.5. Grid search process: (a) validation loss (optimization target), (b) batch size, (c) number of nodes in the first hidden layer and number of hidden layers, (d) learning rate and learning rate decay (exponential of 10).

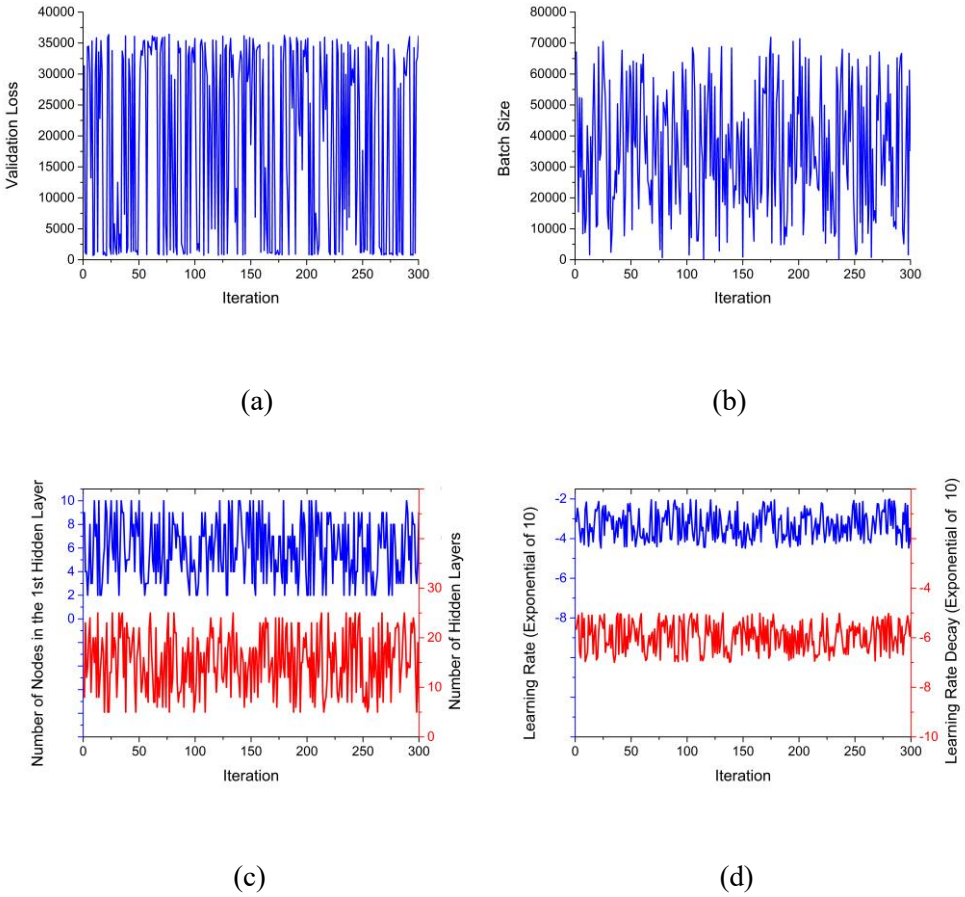


Figure 4.6. Random sampling process: (a) validation loss (optimization target), (b) batch size, (c) number of nodes in the first hidden layer and number of hidden layers, (d) learning rate and learning rate decay (exponential of 10).

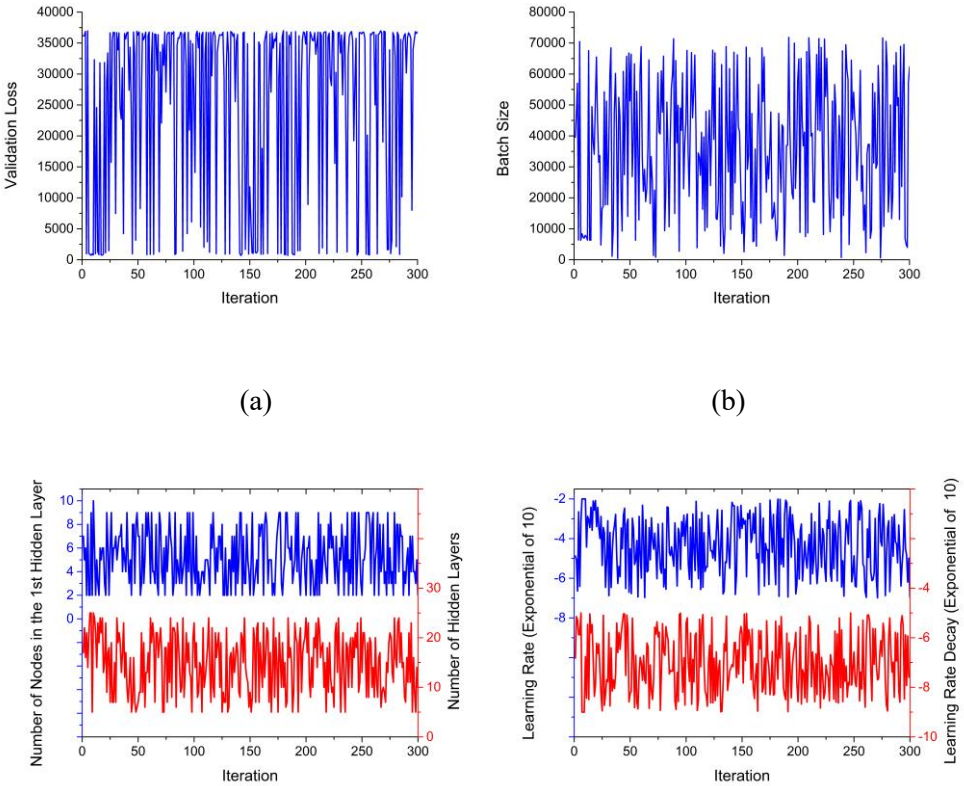


Figure 4.7. Process of Bayesian optimization method: (a) validation loss (optimization target), (b) batch size, (c) number of nodes in the first hidden layer, and number of hidden layers, (d) learning rate and learning rate decay (exponential of 10).

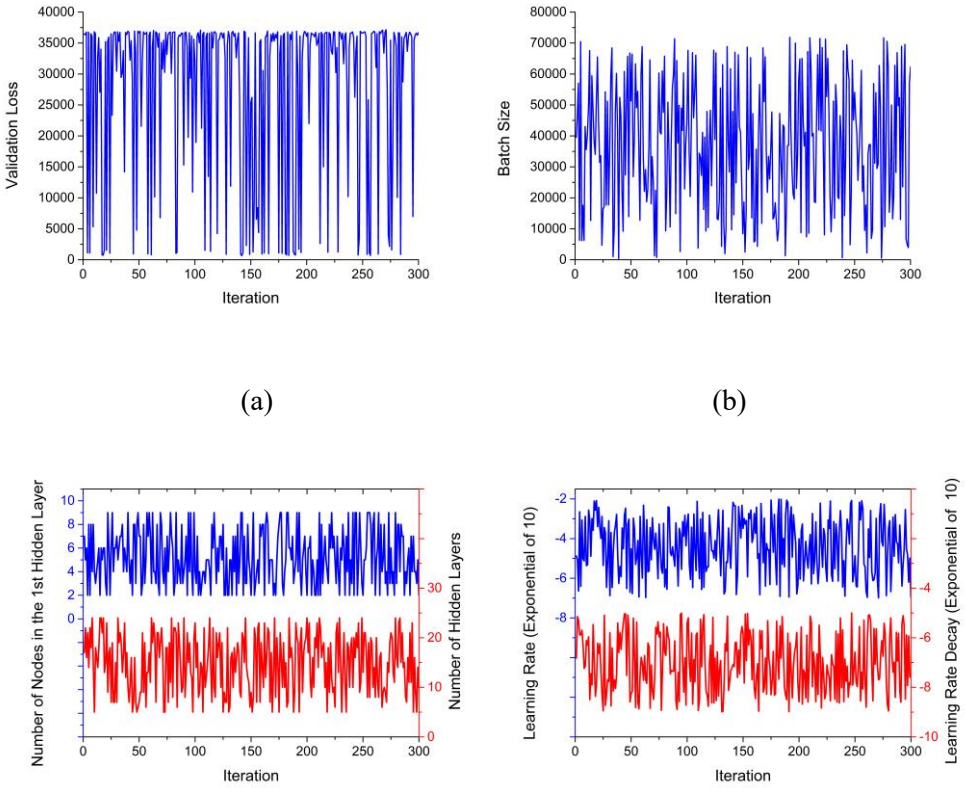


Figure 4.8. Process of Bayesian optimization method with hidden-node determination logic: (a) validation loss (optimization target), (b) batch size, (c) number of nodes in the first hidden layer, and number of hidden layers, (d) learning rate and learning rate decay (exponential of 10).

Hyperparameter	Grid search	Random sampling	Bayesian optimization method	Bayesian optimization method + ND
Validation loss	717.0	701.7	710.0	668.3
Learning rate	10^{-2}	$10^{-2.232}$	$10^{-2.241}$	$10^{-2.341}$
Learning rate decay	10^{-7}	$10^{-5.831}$	$10^{-7.165}$	$10^{-7.165}$
Batch size	36014	18143	14554	14554
Number of hidden layers	6	8	9	9
Number of nodes in the first hidden layer	25	24	21	21
Hidden node sequence	12-25-25-25-25-25-25-1	14-24-24-24-24-24-24-24-24-1	14-21-21-21-21-21-21-21-21-1	14-21-28-35-42-35-28-21-14-7-1
Batch Normalization	included			
Activation function	ELU			
Optimization function	Adam			

Table 4.4. Optimization results and selected hyperparameters of models from each method (ND: the hidden-node determination logic).

4.3.2 Accuracy analysis

Table 4.5 shows the results of an individual model with selected hyperparameters and early stopping callback. The test results were used to compare the accuracies of the models. Figure 4.9 indicates that changing the optimization method from grid search to Bayesian optimization increased the test accuracy. Subsequently, adopting the hidden-node determination logic in the Bayesian optimization method further improved the accuracy. This implies that the Bayesian optimization method and the hidden-node determination logic improved the model better than the previous methods. The R^2 value of the model based on the Bayesian optimization method and hidden-node determination logic, called the "final model," was 0.9674; this value is meaningful compared with those of previous studies because the model predicted NO_x emissions using only the ECU; therefore, it can be applied to the standard emission regulation procedure (WLTP). As mentioned in Section 1.2.1, the results of previous studies are achievable using some additional measurement equipment; in fact, NO_x emissions were predicted for only some steady-state points or simple transient conditions previously.

Figure 4.10 indicates that the results provided in Table 4.5 were not overfitted using the early stopping callback. Losses converged after approximately 200 epochs, but they oscillated as shown in a small-scale view. As the epochs increased, the training loss gradually decreased, which was natural because the models calibrated their internal weights to fit the training data. Overfitting was determined by the validation loss. The figures on the right in Figure 4.10 show the results after the models attained the minimum validation loss during training; the training was continued for 1,000 more epochs, which was the patience value of the early stopping

callback. After the optimal point, the validation losses maintained the same value or increased slightly. The early stopping callback yielded the best model after the training; hence, the models of Table 4.5 were not overfitted.

Figure 4.11 and Table 4.6 show the NO_x prediction results when the entire sequence of the WLTP data was input to the final model. The overall profiles of the NO_x results were similar between measured (reference) and predicted NO_x values, and the MAEs were all approximately 15–17 ppm. The table lists the standard deviation and maximum NO_x of the reference data to compare the level of MAEs. The average MAEs were 11.6% of the standard deviation and 1.6% of the maximum NO_x. The linearity of the Cambustion CLD500 Fast NO_x Analyzer used in this study was approximately 1% of the full scale (5000 ppm) [83]. The device is one of the fastest and most accurate measurement equipment; hence, the accuracy of the final model is comparable to that of the device.

Grid search	Training	Validation	Test
R^2	0.9710	0.9635	0.9634
RMSE [ppm]	24.0	27.2	27.1

Random sampling	Training	Validation	Test
R^2	0.9719	0.9659	0.9656
RMSE [ppm]	23.7	26.3	26.3

Bayesian optimization method	Training	Validation	Test
R^2	0.9720	0.9671	0.9665
RMSE [ppm]	23.6	25.7	25.8

Bayesian optimization method + ND	Training	Validation	Test
R^2	0.9755	0.9681	0.9674
RMSE [ppm]	22.0	25.3	25.4

Table 4.5. Result of models with selected hyperparameters and early stopping callback.

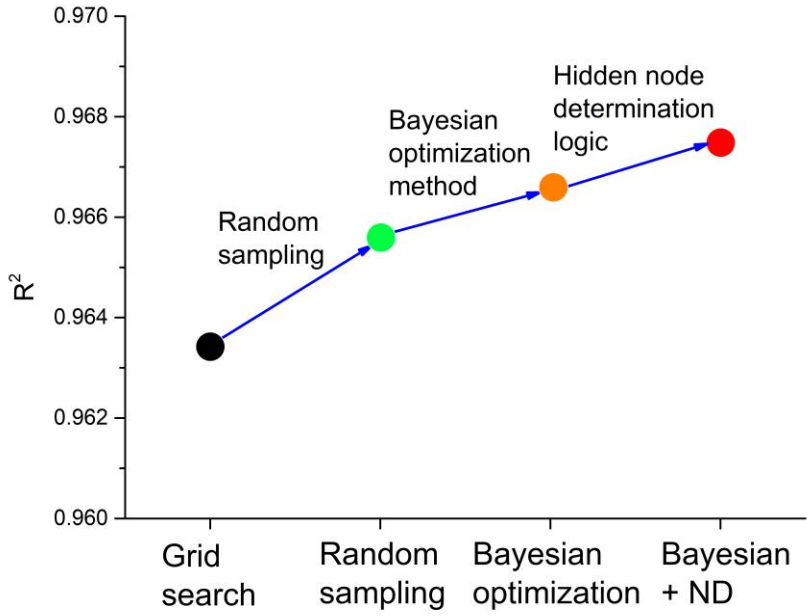
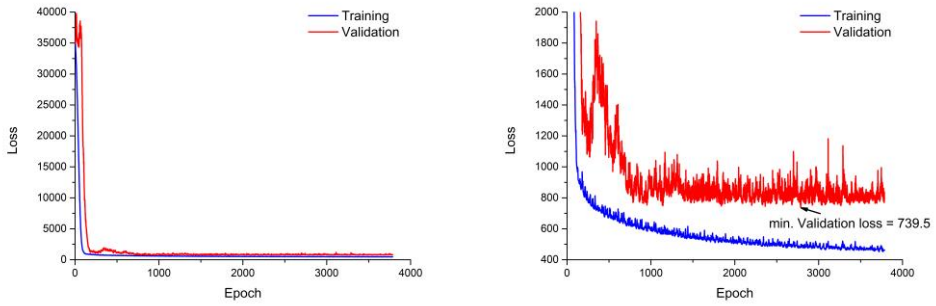
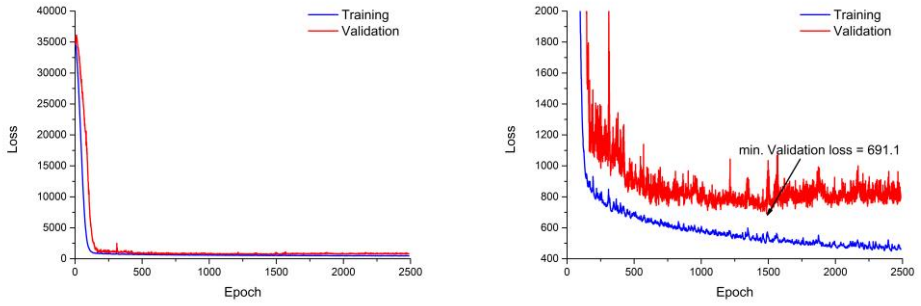


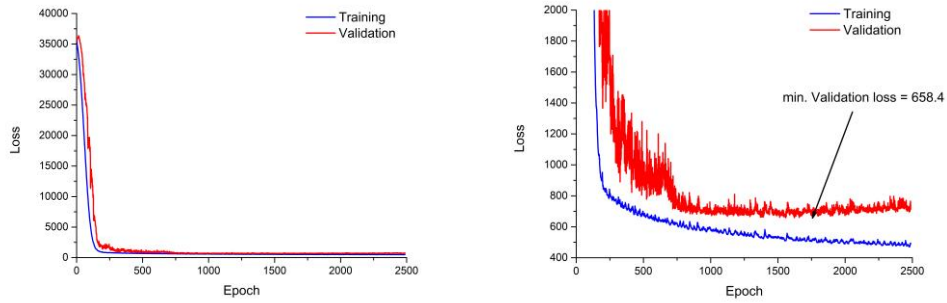
Figure 4.9. Accuracy improvement according to optimization method used.


Training
Validation
Loss
Epoch
min. Validation loss = 739.5

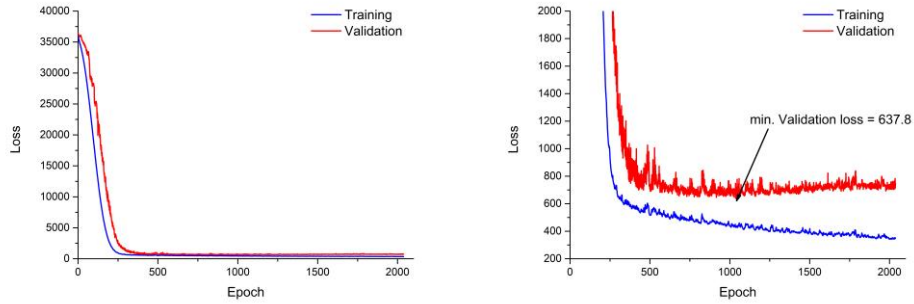
(a)


Training
Validation
Loss
Epoch
min. Validation loss = 691.1

(b)



(c)



(d)

Figure 4.10. Training and validation losses according to epoch during training: the model of (a) grid search, (b) random sampling, (c) Bayesian optimization method, and (d) Bayesian optimization method with hidden-node determination logic.

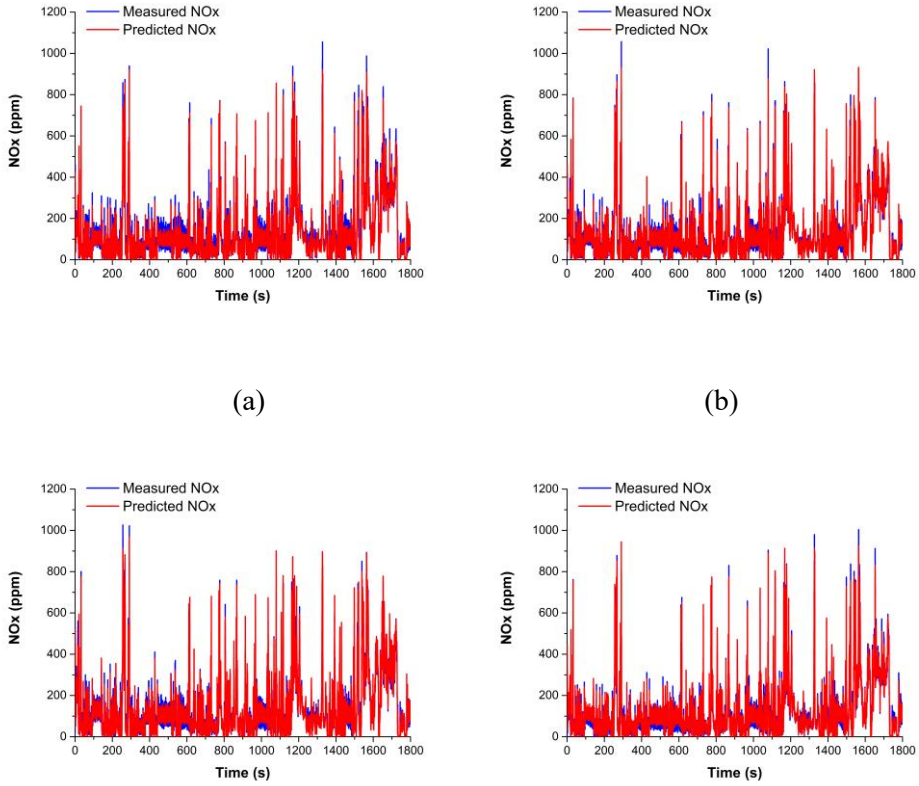


Figure 4.11. NO_x prediction results of the final model for (a) WLTP 1, (b) WLTP 2, (c) WLTP 3, and (d) WLTP 4.

	Normal 1	Normal 2	+3%	-3%
MAE [ppm]	16.9	15.5	16.1	16.7
Standard deviation of reference data [ppm]	142.2	137.9	146.0	133.4
Maximum NO_x of reference data [ppm]	1055.5	1057.0	1026.6	1003.8

Table 4.6. The statistical result of Final model for total WLTP cycles.

Chapter 5. Deep learning algorithms and time-series data preprocessing

In this chapter, the DNN and LSTM algorithms were used to predict the engine-out NO_x of WLTP cycles in a diesel engine under transient conditions to evaluate the accuracies and calculation times of the DNN and LSTM models from the viewpoint of a virtual sensor for real-time prediction.

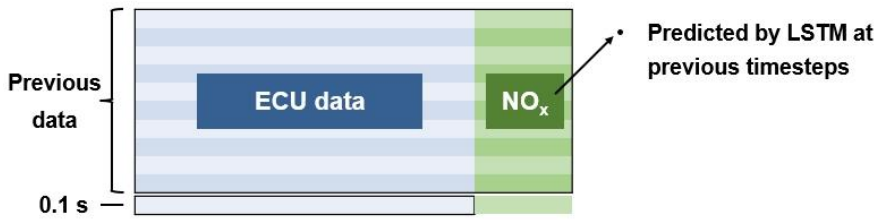
This study revealed that simple data preprocessing can increase the accuracy of the DNN model, and the proposed method can enhance the accuracy of the DNN for high-calculation-speed NO_x emission prediction under transient conditions to a level comparable to that of LSTM.

5.1 Methodology

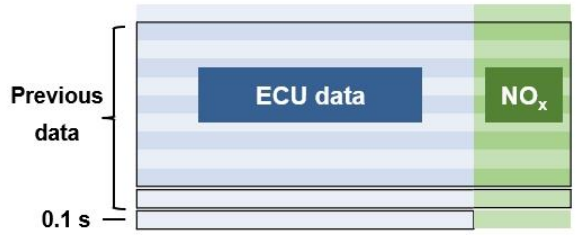
5.1.1 NO_x emission prediction sequence of LSTM

Figure 5.1 presents the NO_x emission prediction sequence of the LSTM model used in this study. The data contained multiple input features measured by the ECU and one target output, i.e., the NO_x value. The NO_x emission data were generated by accumulating values previously predicted by the LSTM. The data were sampled every 0.1 s.

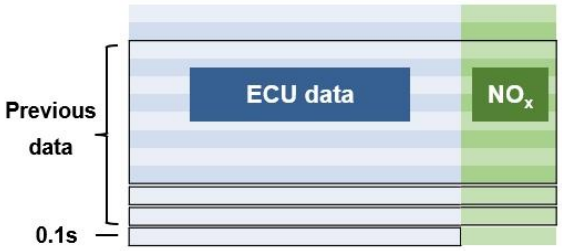
As shown in Figure 5.1(a), the LSTM NO_x emission prediction was based on the ECU data of the previous state and the NO_x emission value predicted by the LSTM in the previous state. When the ECU datum for the present 0.1 s was entered as the input value, the model predicted the NO_x value. After 0.1 s, as shown in Figure 5.1(b), the first datum among the past stored information was deleted, and the NO_x value predicted at the previous timestep was included in the historical information. In this state, when a new 0.1 s interval was entered, the ECU data were received, and a new NO_x value was predicted according to the updated historical NO_x and ECU values. As shown in Figure 5.1(c), the process of appending the predicted NO_x value to the past data, discarding the old information, and predicting the current NO_x value was continuously repeated.



(a)



(b)



(c)

Figure 5.1. LSTM NO_x emission prediction sequence: (a) first timestep; (b) second timestep; and (c) third timestep.

5.1.2 Data preprocessing

The measured ECU and NO_x data were used to evaluate the performances of the DNN and LSTM models. Then, data preprocessing was performed to insert time-related information into the data.

The data preprocessing was based on the fact that transient data are affected by past data. Data related to the past timestep were calculated as the weighted average by using the data of the current timestep, and the past information was reflected in the present prediction. The data preprocessing was applied to the ECU data only, and the NO_x value was used as-is. This is attributed to the fact that the results—considering both the ECU information indicating the state of the engine and the influence of time—indicated the current NO_x value, which was the output of the supervised learning model. Therefore, the NO_x value, i.e., the target output, was excluded from the data preprocessing.

Eq. 5.1 shows the ECU data preprocessing.

$$x_t^i = \frac{w_{t-1} \cdot x_{raw,t-1}^i - w_t \cdot x_{raw,t}^i}{10} \tag{5.1}$$

where x_t^i represents the preprocessed data of the i^{th} feature at timestep t, and $x_{raw,t}^i$ represents the raw data of the i^{th} feature at timestep t. w_{t-1} and w_t represent the weighting factors for the data preprocessing ratio, corresponding to the previous and current timesteps, respectively.

The preprocessing was performed using several ratios of the previous-timestep data to the current-timestep data, which were then used in the experiment, as detailed

in Section 5.2. The five ratios were as follows: 7:3 (7-3 avg.), 6:4 (6-4 avg.), 5:5 (5-5 avg.), 4:6 (4-6 avg.), and 3:7 (3-7 avg.).

Data standardization, i.e., Eq. 5.2, was also applied to both the DNN and LSTM models. There were 12 inputs in the dataset of this study described in Section 3.1, and the features had different means and standard deviations. During the training process, the features having large scale values affected the training more significantly than others, presenting an obstacle. After data standardization, the data distribution had a mean value of 0 and a standard deviation of 1; thus, the scaling effect was removed.

$$S(x) = (x - \bar{x})/\sigma_x \tag{5.2}$$

Here, *S(x)* represents the standardization of feature x, x is a raw input, $\bar{x}$ represents the mean of x, and σ_x represents the standard deviation of x.

5.1.3 Model hyperparameters

Several hyperparameters for the DNN and LSTM models exist. In this study, a few hyperparameters were optimized via the Bayesian optimization method and the hidden-node determination logic described in Chapter 4, whereas the other hyperparameters were set at specific values.

Table 5.1 presents the list of hyperparameters for optimization and their minimum and maximum values. For the optimization, the number of epochs in each iteration was set as 1000 and 15 for the DNN and LSTM, respectively. The number of iterations was set as 300 and 50 for the DNN and LSTM, respectively. The target objective function for the optimization was the minimization of the validation loss, i.e., the MSE.

The target optimized hyperparameters of the DNN model were the learning rate, learning-rate decay, number of hidden layers, number of 1st hidden nodes, and batch size. The number of 1st hidden nodes was set as one of the optimizing hyperparameters instead of the number of each hidden node to avoid the curse of dimensionality during the optimization process. If the number of each hidden node was set as the hyperparameter for optimization, the number of iterations to be calculated for obtaining the optimized set of hyperparameters would increase exponentially. The node arrangements of the hidden layers were determined by the hidden-node determination logic equations proposed in Section 4.3. The maximum batch size was 72,008 and was equal to the total number of data points comprising 4 WLTP cycles (each WLTP cycle comprised 18,002 data points). A detailed data description is provided in Section 3.1.

For the LSTM model, the optimized hyperparameters were the learning rate, learning-rate decay, number of LSTM layers, and window size. Because the number of LSTM nodes was equal to the number of input features (12 in this study), only the number of LSTM layers was used as a representative hyperparameter for the model structure. The window size referred to the length of the previous data, which was stored and used to predict the current NO_x value.

Other hyperparameters omitted from Table 5.1 included the activation function, batch normalization, and an optimization function for training. The ELU activation function [71] was used for the DNN, and the tanh function was used for LSTM [72].

Batch normalization [79] was applied to the hidden layers of the DNN. Batch normalization reduces the internal covariant shift, which causes gradient vanishing or gradient explosion during the training process. The algorithm is described by Eqs. 5.2–5.5. As indicated by Eqs. 5.2 and 5.3, the mean (μ) and variance (σ) of each mini-batch (B) with size m were calculated using individual features. As indicated by Eq. 5.4, the features were normalized using the mean and variance of the mini-batch. Finally, the batch normalization ($BN_{\gamma,\beta}(x_i)$) was completed via a linear transform, where a scale factor (γ) and shift factor (β) were applied to the normalized features ($\hat{x}_i$) to reflect different sample distributions of each mini-batch.

$$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} x_i \qquad // \, mini - batch \, mean \tag{5.2}$$

$$\sigma_{\mathcal{B}}^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad // \, mini - batch \, variance \tag{5.3}$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \qquad // \, normalize \tag{5.4}$$

$$y_i = \gamma \hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i) \quad // \, scale \, and \, shift \tag{5.5}$$

Here, ϵ is a constant added for numerical stability to prevent a denominator of 0.

The Adam optimizer [80] was used to train the DNN and LSTM models. This optimizer combined the concepts of stochastic gradient descent and moment to overcome the limitation of the single gradient descent optimization at the local minima. As indicated by Eq. 5.6, the gradients at timestep t (g_t) were calculated in accordance with the stochastic objective. Eqs. 5.7 and 5.8 indicate that the first moment (m_t), which was an exponential moving average of the gradient, and the second raw moment (v_t), i.e., the squared gradient, were updated using the gradients (g_t) and moments of the previous epoch (m_{t-1}, v_{t-1}). The updated parameter (θ_t) resulting from the Adam optimizer was derived using the parameter of the previous timestep (θ_{t-1}) and the first and second moments via Eq. 5.9.

$$g_t = \nabla_\theta f_t(\theta_{t-1}) \tag{5.6}$$

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \tag{5.7}$$

$$v_t = \beta_2 \cdot m_{t-1} + (1 - \beta_2) \cdot g_t^2 \tag{5.8}$$

$$\theta_t = \theta_{t-1} - \alpha \cdot m_t / (\sqrt{v_t + \epsilon}) \tag{5.9}$$

Here, g_t^2 represents the elementwise square operator $g_t \odot g_t$; $f(\theta)$ represents a stochastic objective function of parameter θ; β_1 and β_2 represent the exponential decay

rates for the moment, which are 0.9 and 0.999, respectively; $\alpha = 0.001$ represents the step size of one training epoch; and $\epsilon = 10^{-8}$ is a model constant.

After the hyperparameter optimization, the model was rerun to obtain its optimal performance for comparison. During this operation, the early-stopping callback [82] was introduced for the DNN and LSTM to prevent overfitting and attain the best model regardless of the number of epochs setting. In this study, the patience number of the early-stopping callback for DNN was set as 1,000, which implies that the model stopped training after 1,000 epochs when it reached the minimum validation loss, and the best model was restored at the minimum validation loss. For LSTM, the patience number was set as 20.

DNN			LSTM		
Hyperparameter	Min.	Max.	Hyperparameter	Min.	Max.
Learning rate	10^{-7}	10^{-2}	Learning rate	10^{-6}	10^{-2}
Learning-rate decay	10^{-9}	10^{-5}	Learning-rate decay	10^{-9}	10^{-5}
Number of hidden layers	2	10	Number of LSTM layers	1	5
Number of 1st hidden nodes	5	25	Window size	1	50
Batch size	20	72,008			

Table 5.1. Minimum and maximum values for the Bayesian hyperparameter optimization.

5.1.4 Dataset assignment

The experimental setup for this section was identical to it described in section 4.3.

A total of 4 WLTP cycles, which comprised 72,008 data points, were employed for the research. The data units differed between the DNN and LSTM. While DNN used one data point as a data sample, LSTM utilized each cycle as a data sample. Therefore, the data subset for the DNN comprised data points, and that for LSTM comprised WLTP cycles.

The data were distributed among training, validation, and test sets. For the DNN, 60 % of the data, i.e., 43,205 data points, were assigned to the training set, and the remaining 40 %, i.e., 28,804 data points, were split equally between the validation and test sets (14,402 data points each). For LSTM, two cycles, i.e., "+3%" and "–3%," were assigned to the training set; "Normal 1" was assigned to the validation set, and "Normal 2" was assigned to the test set. In this study, model training was performed using the training set. In contrast, the validation set was used for hyperparameter optimization and to determine whether overfitting occurred, and the final model accuracy was evaluated against the test set. Tables 5.2 and 5.3 present the data configuration and data assignment, respectively, for the DNN and LSTM.

WLTP Cycle	Normal 1	Normal 2	+3%	–3%
Number of points	18,002	18,002	18,002	18,002

Table 5.2. Dataset configuration of the WLTP cycles.

NN Dataset	Training (60%)	Validation (20%)	Test (20%)
Number of points	43,205	14,402	14,402
LSTM Dataset	Training (50%)	Validation (25%)	Test (25%)
Number of cycles	2 (+3%, –3%)	1 (Normal 1)	1 (Normal 2)

Table 5.3. Data assignment for the DNN and LSTM.

5.2 Results

5.2.1 Comparison between DNN and LSTM

The hyperparameters for each model were optimized using Bayesian optimization and hidden-node determination logic with the search range presented in Table 5.4 to compare the performance, including accuracy and calculation time, of the DNN and LSTM models. The data used in this section were measured using the WLTP cycles, without data preprocessing. The optimized structures of both the DNN and LSTM models are presented in Figure 5.2. The DNN model consisted of eight hidden layers, with the following hidden-node arrangement: 21-30-39-33-27-21-15-9. In Figure 5.2(a), the first "12" of the DNN's node sequence refers to the number of input features, and the last "1" refers to the model output, which was the NO_x value in this study. Figure 5.2(b) presents the LSTM model with a window size of 5 and two LSTM layers.

For the hyperparameter optimization process, the number of epochs in each iteration was set as 1,000 for the DNN and 15 for LSTM. The models were built with optimized hyperparameters and trained to optimize their performance. For the DNN, the number of epochs was set as 20,000, and the patience number of the early-stopping callback was set as 1,000 to optimize accuracy and prevent overfitting. For LSTM, the number of epochs was set as 300, and the patience number of the early-stopping callback was set as 20. Figure 5.3 shows the training and validation losses of the DNN and LSTM models according to the epoch. The DNN model attained the minimum validation loss at the 1,595th epoch and trained 1,000 more epochs owing to the set patience number. The model finished training at the 2,595th epoch, and the 1,595th

epoch was stored as the final DNN model. The validation loss was minimized at the 1,595th epoch and increased after this epoch. Thus, at the 1,595th epoch, this model had the highest accuracy and was not overfitted. The validation loss of LSTM was minimized at the 81st epoch, and training was performed for 20 additional epochs, which was the set patience number. With these procedures, the final models were not overfitted, and their accuracies were optimized.

Table 5.5 presents the accuracy results for the DNN and LSTM models with the training, validation, and test sets. The model accuracy should be compared using the test-set results shown in Figure 5.4. The LSTM model had a larger R^2 value for the test set than the DNN model.

Two hypotheses are proposed regarding the transient conditions. The first is that the result of a transient condition is the sum of all individual results of the steady-state condition and is valid for DNNs. The second is that the present result of the transient condition is the combination of the past results and the present condition. Thus, the time effects, as in the case of LSTM, should be considered. Regarding the accuracy results, consideration of the time effect increased the prediction accuracy for the engine-out NO_x emissions in WLTP cycles. Even though the operating conditions of the engine at different times were identical, the results were different from the trajectory for which the engine had been operated. From this viewpoint, the LSTM algorithm was better suited than the DNN algorithm for the WLTP cycle.

The LSTM model achieved higher accuracy than the DNN model, even though it was disadvantaged with regard to the data distribution. The DNN model utilized 60 %, 20 %, and 20 % of the data for the training, validation, test sets, respectively. The LSTM model used 50 %, 25 %, and 25 % of the data for the training, validation,

and test sets, respectively. The DNN model can use a single operating point as a data point. Therefore, the DNN model exploited 18,002 data per WLTP cycle. The data for the LSTM model should be time-series data; therefore, the LSTM model recognized only one data point for each WLTP cycle. The total number of measured WLTP cycles was four, and they were distributed as follows: two cycles for training, one cycle for validation, and one cycle for testing. Deep-learning models generally exhibit a higher test accuracy when the model has a higher training set ratio, and a lower test set data ratio results from increasing the training set ratio. The results for the data distribution and accuracy indicate that the second hypothesis was applicable to the engine-out NO_x emission prediction for the WLTP cycle.

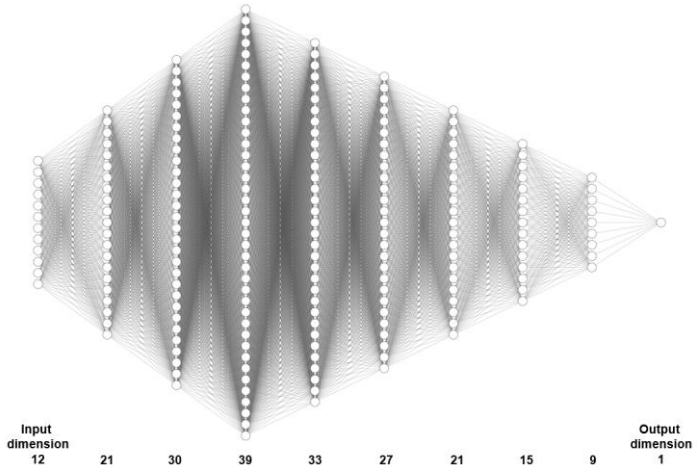
The calculation times required for the test set were compared between the DNN and LSTM models. Given that the trained model was employed for predicting the NO_x emissions, the calculation time for the test set was more important than those for the training and validation sets. For the computing environment described in Section 3.3., the DNN and LSTM models took 0.36 and 1381.0 s, respectively, to calculate the test set.

According to the calculation-time results, the DNN is a better algorithm than LSTM. As a virtual sensor for real-time prediction and control, the DNN model outperforms the LSTM model. If the DNN model is applied to real-time operations, active control can be achieved through sequences such as prediction of the current condition, parameter control for better results, and prediction of the updated condition during one engine cycle. The LSTM can only predict the results of given conditions owing to its calculation speed and ECU hardware; it is significantly slower than the CPU of a personal computer. For real-time prediction, users should accept the inferior accuracy of the DNN owing to the higher calculation speed.

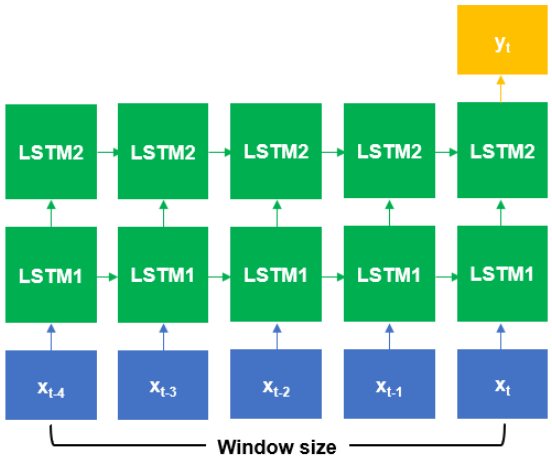
Nonetheless, the disadvantage of the DNN with regard to accuracy can be overcome via data preprocessing. The data-preprocessing results are presented in the next section.

DNN		LSTM	
Learning rate	$10^{-2.341}$	Learning rate	$10^{-3.547}$
Learning-rate decay	$10^{-7.165}$	Learning-rate decay	$10^{-6.712}$
Batch size	14,490	Window size	5
Number of hidden layers	8	Number of LSTM layers	2
Node sequence	12-21-30-39-33-27-21-15-9-1		

Table 5.4 Optimized hyperparameters of the DNN and LSTM models using the measured data.

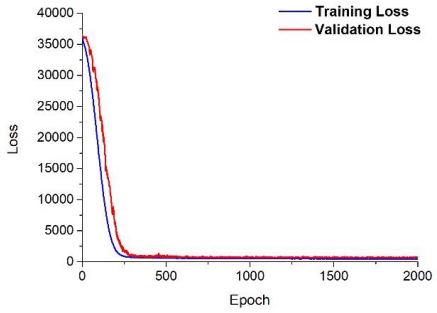


(a)

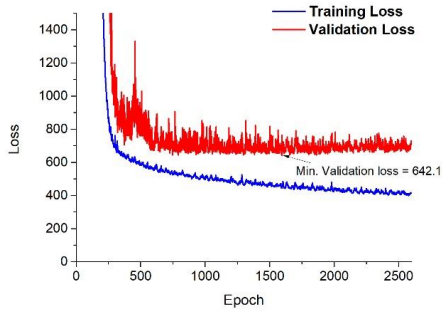


(b)

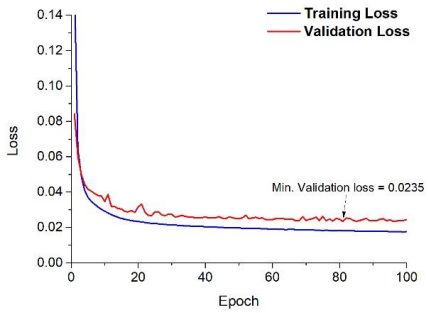
Figure 5.2. Structures of the optimized models: (a) DNN [84] ; (b) LSTM.



(a)



(b)

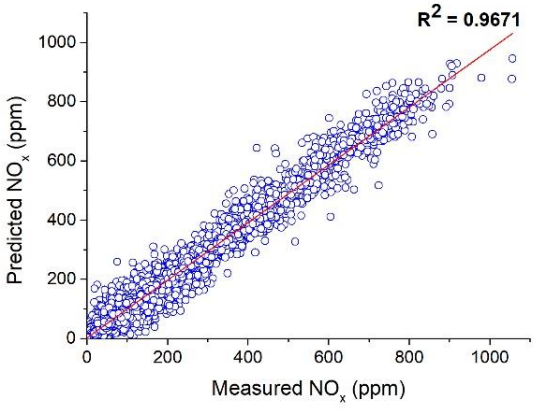


(c)

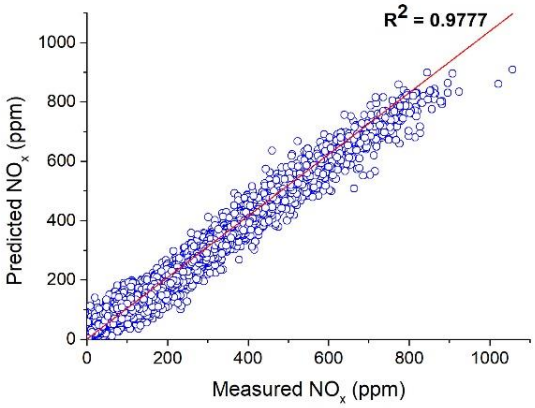
Figure 5.3. Training and validation losses according to the epoch: (a) DNN model; (b) DNN model (enlarged graph); (c) LSTM model.

DNN model				LSTM model			
	Training	Validation	Test		Training	Validation	Test
R^2	0.9744	0.9679	0.9671	R^2	0.9825	0.9776	0.9777
RMSE	22.4	25.3	25.5	RMSE	18.6	21.5	20.6

Table 5.5. Accuracy results for the DNN and LSTM models.



(a)



(b)

Figure 5.4. Results for the R^2 value: (a) DNN model; (b) LSTM model.

5.2.2 Data-preprocessing results

Data preprocessing was performed to contain the time effect on the data. Five preprocessing ratios of the previous-timestep data to the current-timestep data were used: 7:3 (7-3 avg.), 6:4 (6-4 avg.), 5:5 (5-5 avg.), 4:6 (4-6 avg.), and 3:7 (3-7 avg.). The hyperparameters of the model with the preprocessed data were identical to those presented in Table 5.6—optimized for the measured data. The effect of the data preprocessing was investigated using the same hyperparameters.

Tables 5.6 and 5.7 present the accuracy results for the training, validation, and test sets according to the preprocessing ratios of the DNN and LSTM models.

Figure 4.5 presents a comparison of the R^2 values of the preprocessing ratio obtained using the DNN and LSTM models. The LSTM model, with the measured data, had the highest accuracy. The accuracy of the DNN model increased owing to the preprocessing and was maximized at the preprocessing ratio of 7:3. The data preprocessing reflected the previous-timestep data in the current timestep. It played a positive role in supplementing the limitation of the DNN model, which could not consider the time effect on the data. Originally, the DNN recognized the data, regardless of the previous trajectory of the time-series data. Data preprocessing allowed each dataset to include time information, thus allowing the DNN model to utilize time information for the prediction.

However, the LSTM model did not experience any positive effect on the data preprocessing. All the results with the data preprocessing were lower than the measured data. This is because the LSTM already considered time information during

the training using the measured data. The data preprocessing was ineffective and a disturbance for the LSTM model.

As mentioned in Section 5.2.1, the DNN could be used for real-time application, and its accuracy was increased by the data preprocessing. The accuracy of the 7-3 avg. DNN model exhibited differences of 0.0036 (R^2 value) and 2.2 ppm (RMSE) relative to the LSTM model with measured data. Thus, the DNN model with data preprocessing was better with regard to both the accuracy and the calculation time.

Figure 5.6 shows the "Normal 2" cycle prediction results for the best DNN and LSTM models, i.e., the 7-3 avg. DNN model, and the LSTM model with measured data. The "Normal 2" cycle was the test set of the LSTM model. Thus, this cycle was selected for comparison with the entire cycle prediction result. The overall profiles followed the measurement data. The mean absolute error was 14.2 ppm for the DNN model, and 14.1 ppm for the LSTM model.

Figure 5.7 presents the results for specific time windows highlighted in Figure 5.6. The NO_x variability was low over the period of 500–550 s, as shown in Figure 5.7(a). The results for both the models exhibited trends similar to that of the measured values. Figure 5.7(b) highlights the range of high variability at 1600–1700 s. The DNN results exhibited larger fluctuations than the LSTM results around the 1610–1615 s period. The LSTM results stably followed the measurements, and the excessive changes from 1660 to 1680 s were smoothed.

One of the maximum values was observed between 1050–1100 s, as shown in Figure 5.7(c). Figure 5.7(d) presents results specifically focused on time from 1077–1080 s. At 1077.6 s, the first peak was observed, as shown in Figure 5.7(d). Both models predicted the maximum values and their associated times. Around the first

peak, the LSTM slightly overestimated the NO_x values, and the DNN underestimated the values. At the second peak, as shown in Figure 5.7(d), both models predicted the timing of the maximum value later than the measured data, and the maximum NO_x values were slightly lower than the measured value.

Overall, the LSTM stably followed the trends of the NO_x more than the DNN. The DNN had the advantage of predicting the high-frequency fluctuation range. However, the difference between the DNN and LSTM models was insignificant with regard to detailed trajectory analysis.

The data preprocessing increased the accuracy of the DNN model compared with the LSTM model, while the advantage over the LSTM model regarding the calculation speed was maintained.

DNN	7-3 avg.			6-4 avg.		
	Train	Val.	Test	Training	Val.	Test
R^2	0.9791	0.9755	0.9741	0.9785	0.9737	0.9730
RMSE	20.5	22.3	22.8	20.6	22.9	23.2

DNN	5-5 avg.			4-6 avg.		
	Train	Val.	Train	Val.	Train	Val.
R^2	0.9736	0.9702	0.9736	0.9702	0.9736	0.9702
RMSE	23.0	24.6	23.0	24.6	23.0	24.6

DNN	3-7 avg.		
	Train	Train	Train
R^2	0.9720	0.9720	0.9720
RMSE	23.5	23.5	23.5

Table 5.6 Accuracy results according to the preprocessing ratio of the DNN (Train: training set; Val: validation set; Test: test set).

LSTM	7-3 avg.			6-4 avg.		
	Train	Val.	Test	Training	Val.	Test
R^2	0.9757	0.9689	0.9685	0.9780	0.9722	0.9721
RMSE	21.9	25.3	25.0	20.9	23.9	23.5

LSTM	5-5 avg.			4-6 avg.		
	Train	Val.	Train	Val.	Train	Val.
R^2	0.9802	0.9743	0.9748	0.9818	0.9758	0.9764
RMSE	20.0	23.0	22.6	19.0	22.3	21.7

LSTM	3-7 avg.		
	Train	Train	Train
R^2	0.9834	0.9774	0.9768
RMSE	18.1	21.6	21.6

Table 5.7. Accuracy results according to the preprocessing ratio of the LSTM (Train: training set; Val: validation set; Test: test set).

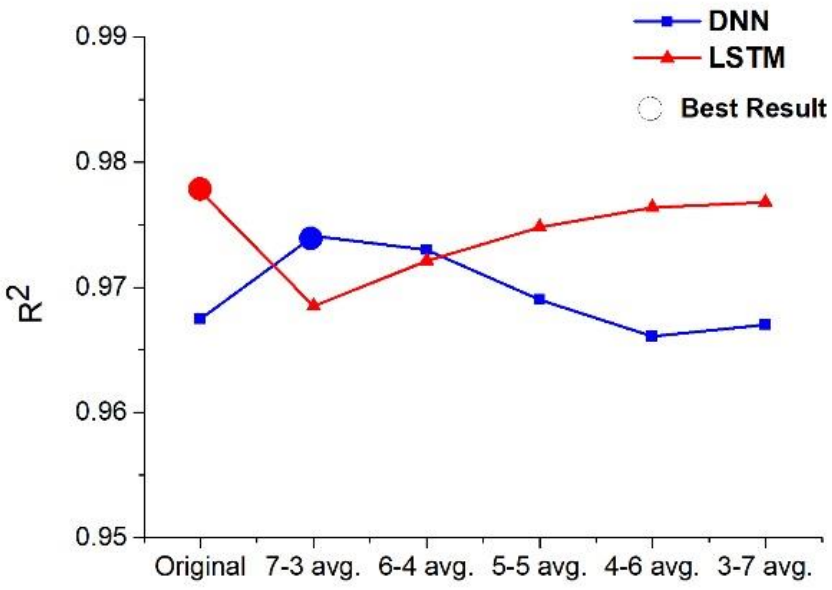


Figure 5.5. Comparison of the R^2 results for the DNN and LSTM models regarding the preprocessing ratio.

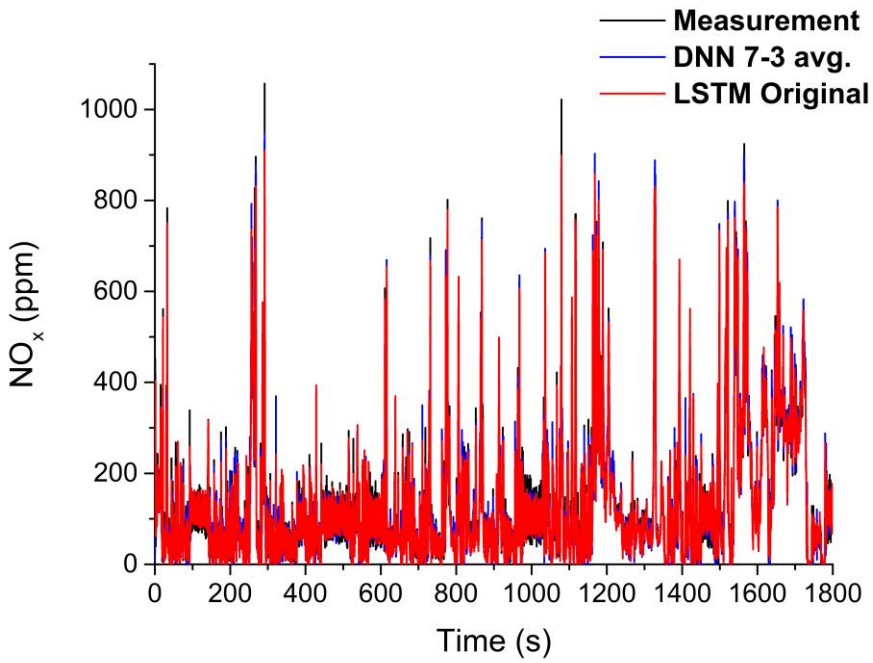


Figure 5.6. “Normal 2” cycle prediction results.

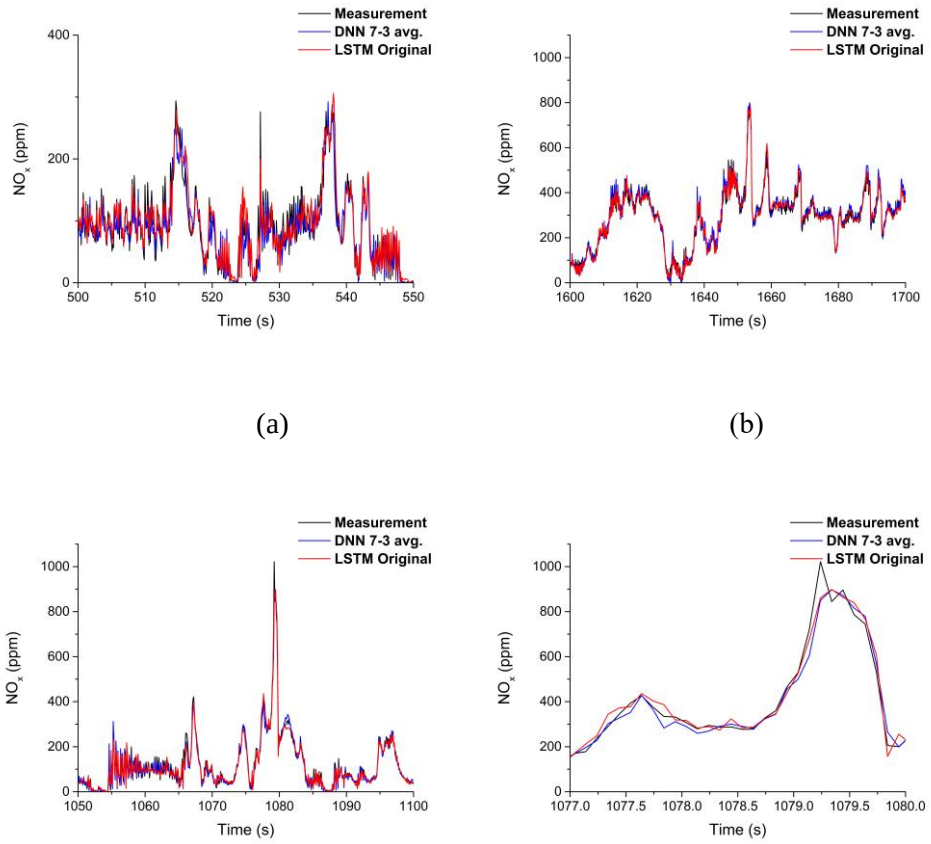


Figure 5.7. “Normal 2” cycle results of the DNN model with 7-3 avg. and LSTM model with measured data for (a) 550–550 s, (b) 1600–1700 s, (c) 1050–1100 s, and (d) 1077–1080 s.

Chapter 6. Steady-state experimental design for prediction of transient NO_x emissions

6.1 Overview and necessity of steady-state experimental design

In this chapter, the transient NO_x emissions were predicted using steady-state data. Steady-state experiments had to be designed to train the model with steady-state data for NO_x prediction under transient conditions. Eventually, a design methodology for the steady-state experiments was suggested. Figure 6.1 summarizes the objectives of this chapter.

In previous chapters, data acquired under transient conditions were utilized to predict NO_x emissions under transient conditions. The DNN model predicted NO_x under transient conditions with an R^2 value of 0.97, which meant transient conditions could be approached by a quasi-stationary concept when deep learning models predicted them.

Basically, if steady-state data were used as a training dataset to predict the NO_x emissions under transient conditions, an imbalance in the data distribution could occur. It is usually recommended that the largest portion of the data be used for the training set to construct a model with stable accuracy [85]. The common ratios used for datasets are 6:2:2, 7:1.5:1.5, and 8:1:1 in the order of the training, validation, and test datasets. However, a training set configured with steady-state data is inevitably smaller than a test set consisting of transient data. This is one of the difficulties in predicting transient NO_x emissions with a model trained using steady-state data.

Furthermore, there are local variations in some engine parameters when the engine operates under transient conditions. These are caused by mechanical and electrical delays, and their effects on other engine parameters. Because steady-state experiments cannot cover this variability, simple steady-state experiments such as map-based tests are not appropriate for predicting transient NO_x emissions. Additional experiments should be performed to determine the variation tendencies caused by the transient conditions in the model during the training process.

In this study, the behaviors of engine parameters under transient conditions were analyzed and additional engine experimental conditions were designed based on this analysis. The intake air mass, intake pressure, main injection timing, and injection pressure were considered as swing parameters of steady-state experiments.

After establishing the dataset for transient NO_x prediction, an additional temperature swing experiment was performed to predict the effects of the intake and coolant temperatures on the NO_x emissions.

The results of the NO_x emission profiles according to the time and cumulative NO_x mass of WLTP cycles proved that well-designed experiments made it possible to predict the transient NO_x emissions using steady-state data. Additional temperature experiments enabled the model to deal with the temperature change.

This study investigated an experimental procedure for designing a steady-state experimental conditions to predict the NO_x emissions under transient conditions using a deep learning approach.

[당위성 추가]

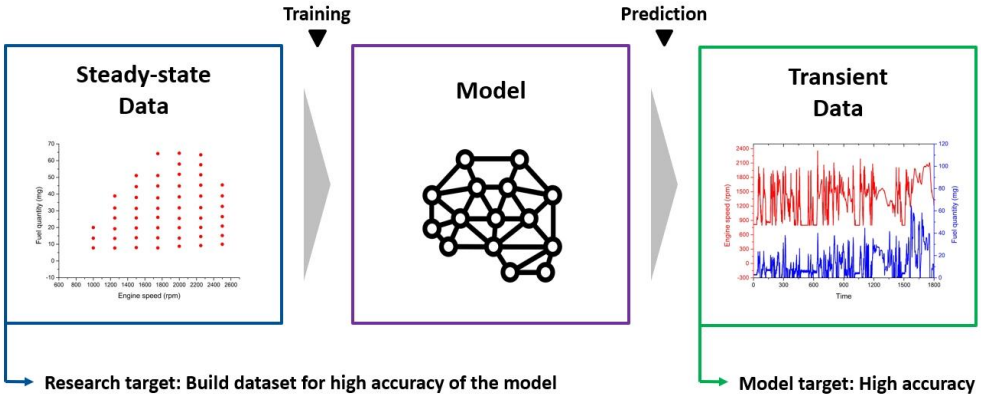


Figure 6.1. Diagram for objective of this chapter.

6.2 DNN model setup

Figure 6.2 shows the training process for a deep learning model. A DNN model was trained with steady-state data. The accuracy of the model was repeatedly evaluated by comparing measured WLTP NO_x and predicted WLTP NO_x of the model. The training algorithm assessed whether the prediction accuracy was the best. If the accuracy could be increased via more training, the training, evaluation, and assessment loop would be repeated until the maximum accuracy of the model was achieved.

The algorithm of the deep learning model in this study was the DNN. Some hyperparameters such as the learning rate, learning rate decay, number of hidden layers, number of 1^{st} hidden nodes, and batch size were optimized by the Bayesian optimization. The minimum and maximum values for the Bayesian hyperparameter optimization were listed in Table 6.1. The arrangement of the hidden nodes in each hidden layer was determined with the number of hidden layers and number of 1^{st} hidden nodes according to the hidden-node determination logic. The Bayesian optimization and the hidden-node determination logic were explained in Chapter 4. The maximum batch size was varied by the size of the training dataset which would be introduced in the following sections. The maximum batch size was 53 for the map experiment of Section 6.3.1, 251 for the map and swing experiment of Section 6.3.2, and 300 for the steady-state dataset including the temperature swing experiment of Section 6.3.3. The iteration number of the Bayesian optimization was 350, and each iteration was stopped when the model for the iteration achieved the best accuracy decided by the early stopping callback [82] introduced in Section 4.3.

Other hyperparameters excluded in Table 6.1 were fixed at the specific value. The activation function was set at the ELU activation function [71], and the optimization function for the training of neural networks was Adam optimizer [80]. In addition, batch normalization [79] was applied to the networks. These hyperparameters were specifically introduced in Chapter 2 and Section 4.1.3.

After the process of hyperparameter optimization, the model accuracy was obtained by running the model with optimized hyperparameters. Early stopping callback was also applied to this step. The prediction accuracy of the model was evaluated by RMSE of NO_x emission profiles over time and a cumulative NO_x mass [mg] of WLTP cycles. The comparison of the cumulative NO_x results was based on the normalized value. The result of the measured cycle was 1.0 as the reference value. NO_x emissions were measured every 0.1 seconds; therefore, the cumulative NO_x mass was derived by the summation of NO_x mass for 1800 seconds.

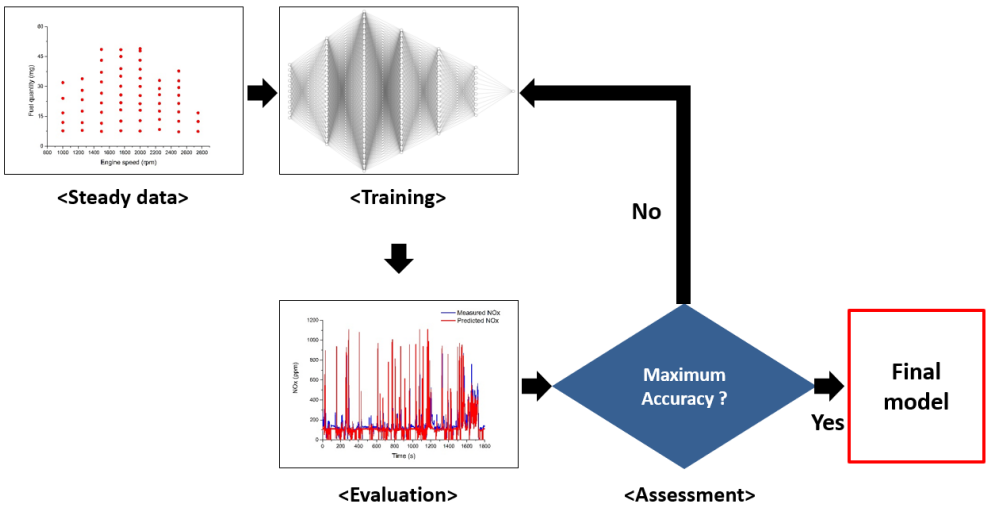


Figure 6.2. Training process for a deep learning model.

Hyperparameter	Min.	Max.
Learning rate	10^{-7}	10^{-2}
Learning rate decay	10^{-9}	10^{-5}
Number of hidden layers	2	10
Number of 1st hidden nodes	5	20
Batch size	20	Number of data points of the steady-state experiment
Dropout rate	0	0.5

Table 6.1. Minimum and maximum values for the Bayesian hyperparameter optimization.

6.3 Results and discussion

6.3.1 Results of model trained with map experimental data

The NO_x emissions of the WLTP cycle were predicted by the model trained using map data. The map experiment was conducted at 250 rpm (engine speed) and 2 bar (BMEP) intervals, and the total number of map points was 53. Figure 6.3 shows the operating points of the map experiment and the WLTP cycle. The red dots represent the steady-state operating points of the map experiment, and the black dots represent the operating points of WLTP cycle for every 0.1 seconds. Table 6.2 lists the engine speed and BMEP of the map experimental data points.

The optimized hyperparameters for the model trained using the map data are presented in Table 6.3. The model trained by map data predicted NO_x emissions with an RMSE of 98 ppm as shown in Figure 6.4(a). As shown in Figure 6.4(b), the normalized cumulative NO_x mass was 1.144, which meant that the model predicted a 14.4% larger NO_x mass for WLTP cycle than that measured for the WLTP cycle.

As shown in Figure 6.4(a), the predicted NO_x values were limited to the given range. The maximum value of the measured NO_x was 1029 ppm, and the minimum NO_x was 5 ppm. However, the model did not output values lower than 77 ppm, and the maximum predicted value was 664 ppm. The model could not predict the NO_x emissions of the WLTP cycle.

The reason for this result was the limitation of the map experimental data. The maximum NO_x value of the map experimental data was 666 ppm and the minimum value was 64 ppm. Therefore, the predicted values were distributed between the upper

and lower limits of the map experimental data because the model did not learn any values outside the limits during the training process.

The other reason was that the map experimental data obtained by steady-state experiments did not contain the variability of the transient conditions. Figure 6.5 presents the operating profiles of the entire WLTP cycle and the operating points at 1250 rpm, BMEP of 4 bar. The solid lines show the engine speed in red, and fuel quantity in blue. Red circles show the points at an engine speed of 1250±20 rpm, and the blue squares indicate a fuel quantity of 13.44±1 mg. At 1250 rpm, a fuel quantity of 13.44 mg was obtained at BMEP of 4 bar. During the WLTP cycle, the engine operated 16 times, passing through 1250 rpm, BMEP of 4 bar and its adjacent points.

Figure 6.6 shows comparison between the steady-state and transient operating conditions of the intake air mass, intake pressure, injection pressure, and main injection timing at 1250 rpm, BMEP of 4 bar. The black dots represent the values of each parameter at 1250 rpm, BMEP of 4 bar. The red dotted line represents the map value of each parameter acquired in the steady-state experiment. The blue dotted lines represent the limit values for the points at 1250 rpm, BMEP of 4 bar of the WLTP cycle.

Even though the operating points, an engine speed of 1250±20 rpm and a fuel quantity of 13.44±1 mg were similar, the parameter values were not the same. The values of the parameters such as the intake air mass, intake pressure, and injection pressure as shown in Figure 6.6 were spread within certain ranges. The upper deviation of the intake air mass was 10.5% compared to the map value, and the lower deviation was 4.6%. The minimum value of the intake pressure was 3.8% smaller than the map value, and the values of the injection pressure were distributed from the –3.9%

to +7.4% range compared to the map value. In this case, the deviation of the main injection timing values was not observed. That is, all values of the main injection timing at 1250 rpm, BMEP of 4 bar were identical during the WLTP cycles.

Figure 6.7 shows the operating profiles of the WLTP cycle, and operating points at 1500 rpm, BMEP of 4 bar (an engine speed of 1500±20 rpm and a fuel quantity of 13.87±1 mg). The engine passed through 1500 rpm, BMEP of 4 bar, and its adjacent points 60 times during the WLTP cycle.

At 1500 rpm, BMEP of 4 bar as presented in Figure 6.8, the deviations of the intake air mass, intake pressure, injection pressure, and main injection timing for the WLTP cycle were larger than those at 1250 rpm, BMEP of 4 bar. The values of the intake air mass were distributed between +15.5% and –11.0% from the map value. The ranges were from –5.8% to +12.3% for the intake pressure, and from –8.5% to +8.8% for the injection pressure. Different from the main injection timing at 1250 rpm, BMEP of 4 bar, a deviation of the main injection timing existed at 1500 rpm, BMEP of 4 bar.

Figures 6.9 and 6.10 are the comparisons between steady-state and transient operating conditions of the parameters at 1500 rpm, BMEP of 8 bar, and 1750 rpm, BMEP of 8 bar.

As shown in Figures 6.6 and 6.8 – 6.10, the map values did not represent the transient conditions. The map value was one of the possible statuses when the engine was operated under transient conditions. This was because the map was defined under steady-state conditions, which excluded the deviation caused by real-time operation over time. However, various time effects are involved in transient conditions. These include mechanical delays that affect the air flow rate, EGR rate, etc., and electrical

delays such as injection variables. Therefore, the model trained using the map experimental data could not accurately predict the transient NO_x emissions because the data did not contain these variations.

Table 6.4 lists the map value, difference between the upper limit and map value (Max. diff.), and difference between the lower limit and map value (Min. diff.) of the intake air mass, intake pressure, injection pressure, and main injection timing at several operating points. Compared to the map value, the red color indicates that the "Max. diff." is larger than +5%, and the blue color means that the "Min. diff." is smaller than –5%.

As listed in Table 6.4, the intake air mass exhibited large deviation between the map and transient values at the various operating points. The intake pressure and injection pressure also exhibited deviations at some operating points. The deviation of the main injection timing were the smallest, and the map value was the same as the transient values at 1250 rpm, BMEP of 4 bar.

Based on the analysis, additional experimental points with the deviations under transient conditions should be designed to provide the model during the training process. In particular, the intake air mass, intake pressure, and injection pressure should be considered significant because of their deviations under transient conditions. The deviation of the main injection timing had a small range, which should be taken into account in the experimental design.

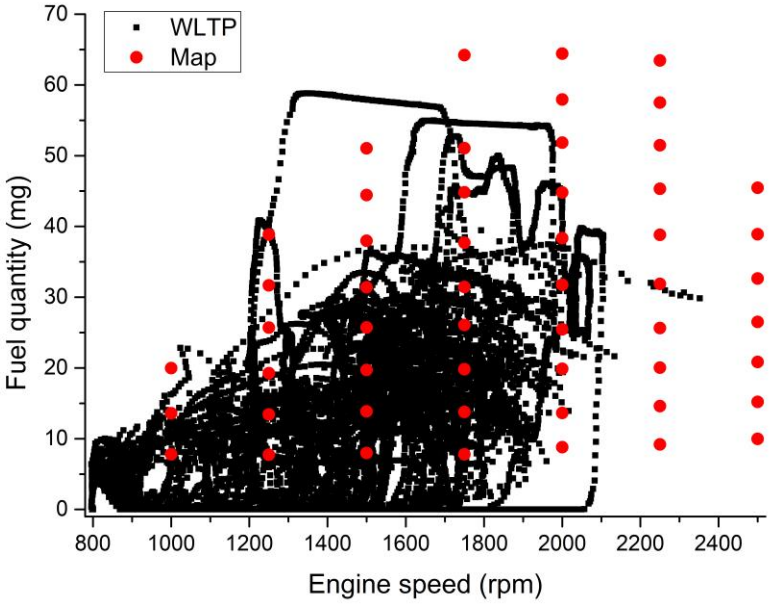


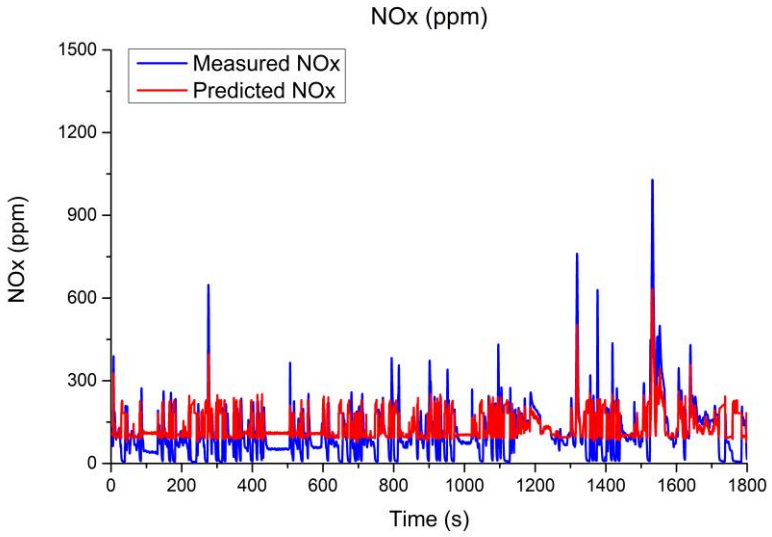
Figure 6.3. Operating points of map experiment and WLTP cycle.

Engine speed [rpm]	BMEP [bar]
1000	2, 4, 6
1250	2, 4, 6, 8, 10, 12
1500	2, 4, 6, 8, 10, 12, 14, 16
1750	2, 4, 6, 8, 10, 12, 14, 16, 20
2000	2, 4, 6, 8, 10, 12, 14, 16, 18, 20
2250	2, 4, 6, 8, 10, 12, 14, 16, 18, 20
2500	2, 4, 6, 8, 10, 12, 14

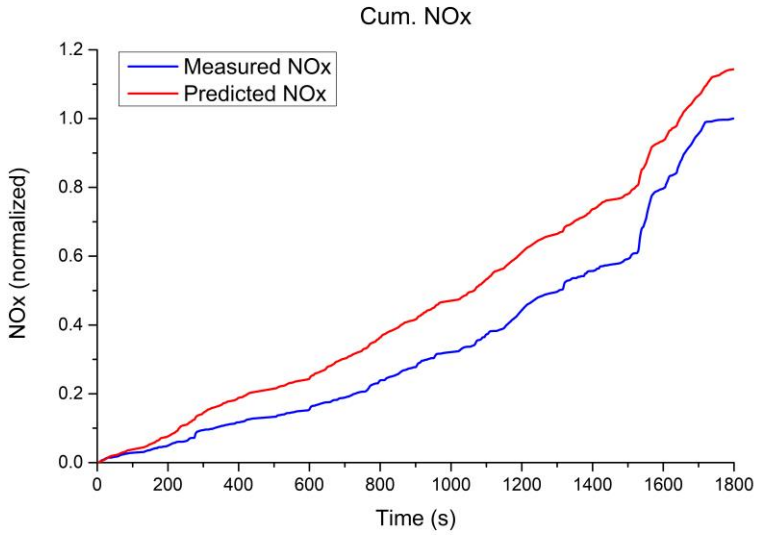
Table 6.2. Engine speed and BMEP of map experiment.

Hyperparameter	Optimized value
Learning rate	$10^{-2.64}$
Learning rate decay	$10^{-8.773}$
Batch size	34
Dropout rate	0.3
Number of hidden layers	2
Number of 1st hidden nodes	7
Node arrangement	10-7-4-1

Table 6.3. Optimized hyperparameters of model trained with map experiment.



(a)



(b)

Figure 6.4. Prediction results of model trained by map experimental data: (a) NO_x emission profile, and (b) cumulative NO_x mass (normalized).

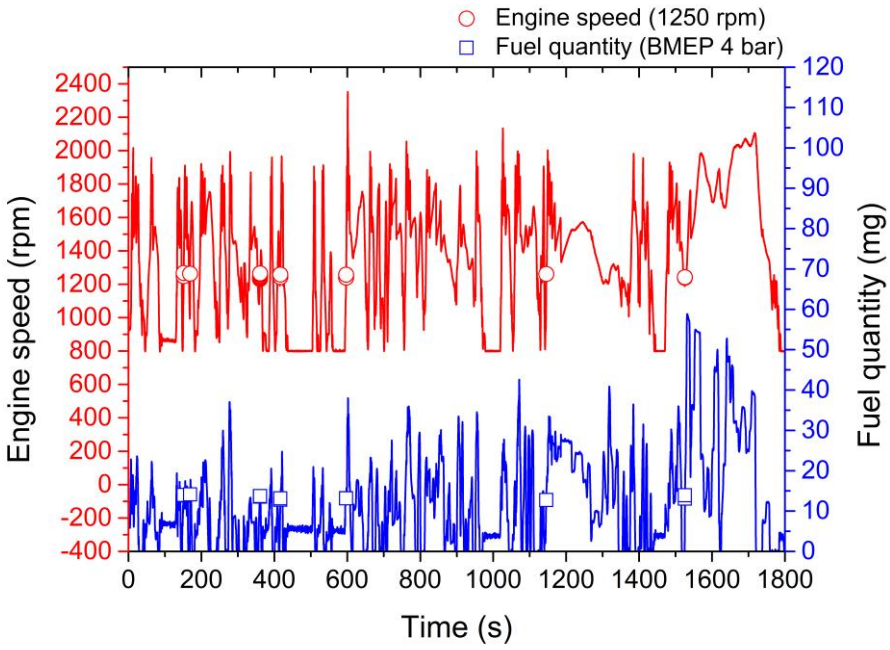


Figure 6.5. Operating points at 1250 rpm, BMEP of 4 bar for entire WLTP cycle.

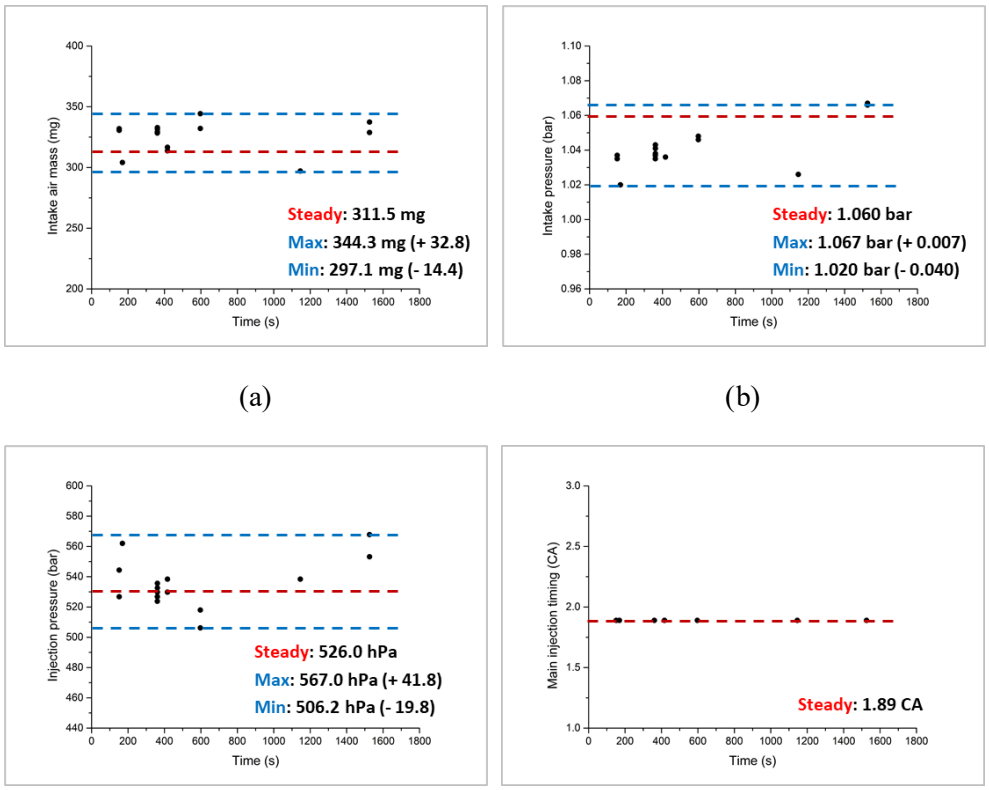


Figure 6.6. Comparisons between steady-state and transient operating conditions: steady value and transient range of (a) intake air mass, (b) intake pressure, (c) injection pressure, and (d) main injection timing at 1250 rpm, BMEP of 4 bar.

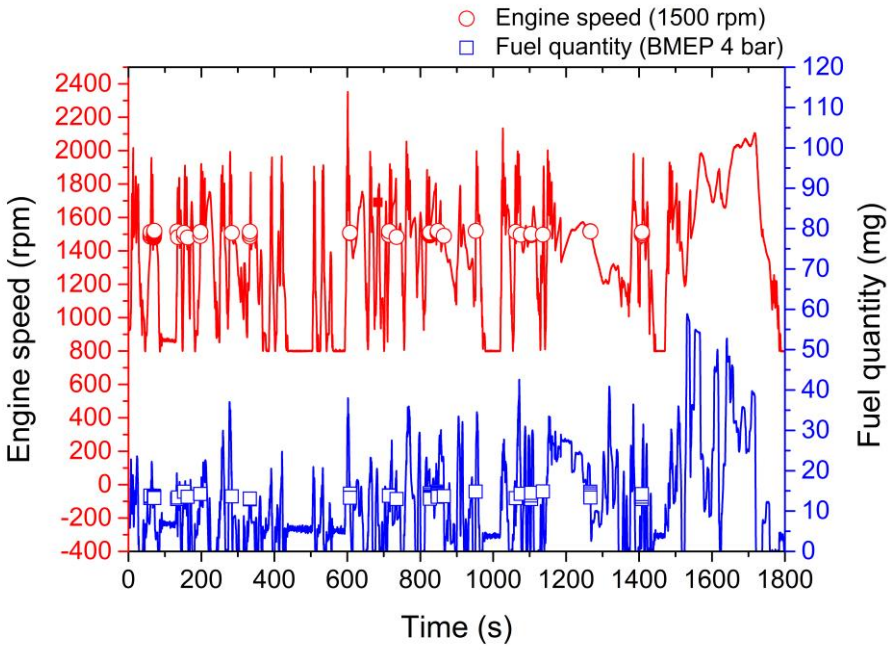
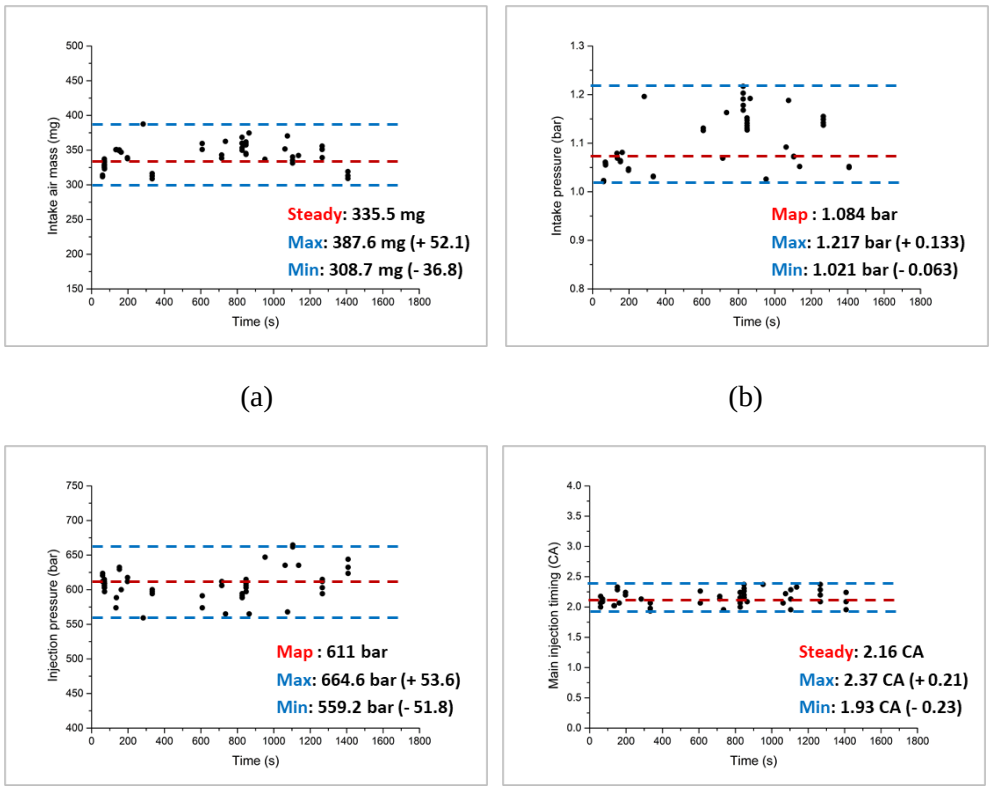


Figure 6.7. Operating points at 1500 rpm, BMEP of 4 bar for entire WLTP cycle.



(a) (b)

(c) (d)

Figure 6.8. Comparisons between steady-state and transient operating conditions: steady values and transient ranges of (a) intake air mass, (b) intake pressure, (c) injection pressure, and (d) main injection timing at 1500 rpm, BMEP of 4 bar.

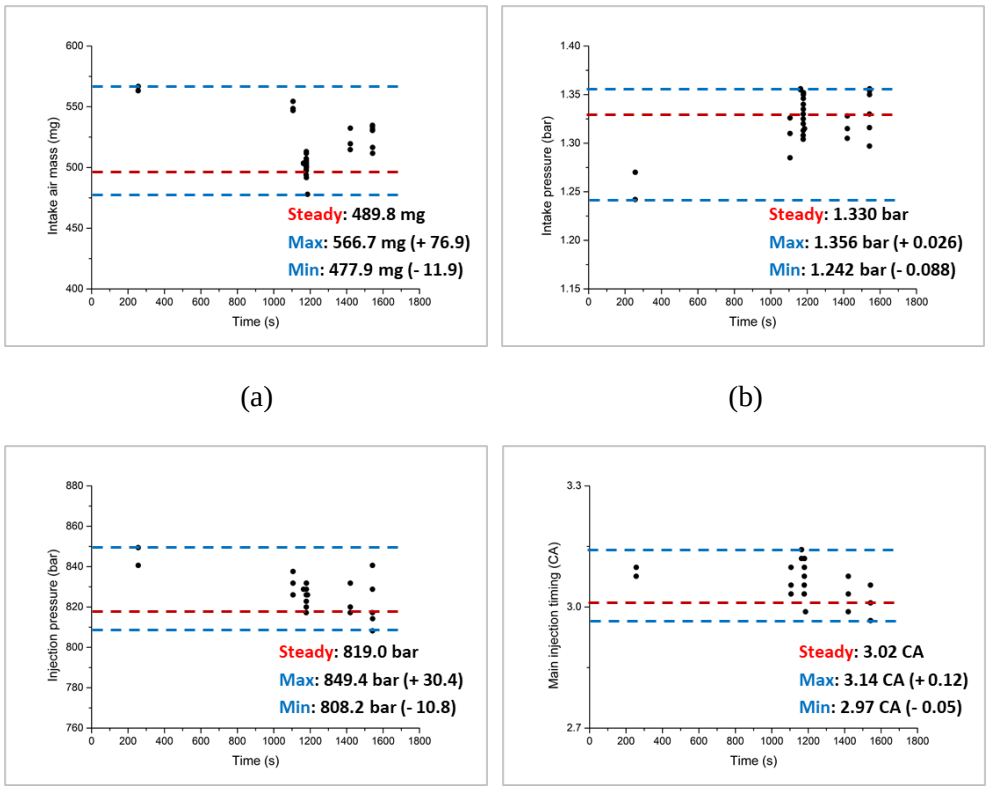


Figure 6.9. Comparisons between steady-state and transient operating conditions: steady values and the transient ranges of (a) intake air mass, (b) intake pressure, (c) injection pressure, and (d) main injection timing at 1500 rpm, BMEP of 8 bar.

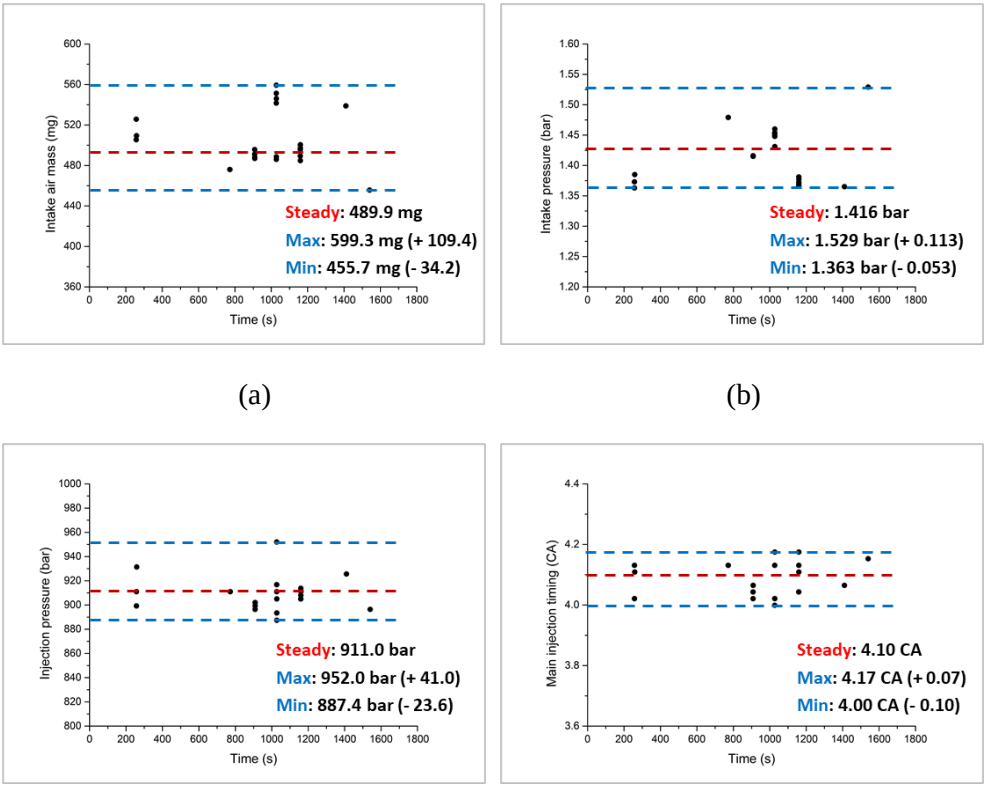


Figure 6.10. Comparisons between steady-state and transient operating conditions: steady values and transient ranges of (a) intake air mass, (b) intake pressure, (c) injection pressure, and (d) main injection timing at 1750 rpm, BMEP of 8 bar.

Operating point	Intake air mass [mg]			Intake pressure [bar]			Main injection timing [CA]			Injection pressure [bar]		
	Map	Max. diff.	Min. diff.	Map	Max. diff.	Min. diff.	Map	Max. diff.	Min. diff.	Map	Max. diff.	Min. diff.
1250 rpm 4 bar	311.5	+32.8	-14.4	1.060	+0.007	-0.040	1.89	-	-	526	+41.8	-19.8
1500 rpm 4 bar	335.5	+52.1	-36.8	1.084	+0.133	-0.063	2.16	+0.21	-0.23	611	+53.6	-51.8
1500 rpm 8 bar	489.8	+96.9	-11.9	1.330	+0.026	-0.088	3.02	+0.12	-0.05	819	+30.4	-10.8
1750 rpm 6 bar	424.8	+64.6	-19.7	1.285	+0.119	-0.124	4.05	+0.10	-0.29	845	+45.4	-107.0
1750 rpm 8 bar	489.9	+109.4	-34.2	1.416	+0.113	-0.053	4.10	+0.07	-0.1	911	+41.0	-23.6
2000 rpm 8 bar	485.2	+25.7	-45.4	1.488	+0.104	-0.013	5.49	+0.09	-0.19	999	+8.6	-44.0

Table 6.4. Map values, Max. diff. (difference between upper limit and map value), and Min. diff. (difference between lower limit and map value) of intake air mass, intake pressure, main injection timing, and injection pressure.

6.3.2 Design of steady-state experimental condition for prediction of transient cycle

This experiment was designed to improve the accuracy of the NO_x emission prediction by providing additional data on the transient variation in the model. The swing experiment was performed as the value of a certain parameter was changed, while the other parameters were fixed at the map values. The swing parameters were the intake air mass, intake pressure, injection pressure, and main injection timing, which were derived from the analysis discussed in the previous section.

Figure 6.11 shows the operating points of the WLTP cycle and steady-state experimental condition. The steady-state experimental condition included the ECU map data and swing conditions. The black dots represent the operating points of the WLTP cycle at 0.1 s interval. The red dots represent the operating points of the ECU map data condition. The swing experiment did not use these points. The green triangles, sky blue squares, and orange circles represent the points where the swing experiment was applied. At the sky-blue square points, a swing experiment was conducted for each intake air mass, intake pressure, and injection pressure. For the swing experiment at low speed and low load indicated by the green triangles, the intake air mass and injection pressure were swung. The main injection timing was also swung at seven operating points marked by the orange circles. These seven points were adopted from the major calibration points defined by the engine manufacturer. The points were derived by the frequency of use under transient conditions. Thus, the manufacturer primarily investigated these points to characterize the engine output under transient conditions.

The swing experiment was not conducted at all operating points. Seventeen points were selected for the swing experiment, and the points were in the area of high NO_x mass as shown in Figure 6.12. The contour of the NO_x emissions was depicted by the percentage of the NO_x emissions at each area over the WLTP cycle. The NO_x emitted mass at each timestep and the number of times that it passed through the point during the WLTP cycle were considered to produce the contour.

Table 6.5 summarizes the total of 251 experimental conditions of the steady-state experiment. Motoring data at five engine speeds were obtained to provide information of fuel zero status to the model. The idle and map conditions were the points operated according to the ECU setting. No parameter swing was conducted for the idle and map conditions.

At low-speed and low-load conditions, especially 800 rpm, BMEP of 4 bar, 1000 rpm, BMEP of 2 bar, and 1250 rpm, BMEP of 4 bar, the intake air mass and injection pressure were selected as the swing parameters. For the intake air mass swing, the intake air mass values were set at the map value and $\pm$25 mg compared to the map value. Injection pressure swing was conducted by setting the map value and $\pm$50 bar (= 50000 hPa). The intake pressure was not a swing parameter because it was not controlled by the ECU under low-speed and low-load conditions. Compared to the other conditions, the swing range of the air mass was smaller because its control range was limited under these conditions.

For other swing conditions, the intake air mass, intake pressure, and injection pressure were the swing parameters. The intake air mass swing was conducted at the map value and $\pm$50 mg, and the intake pressure swing was performed at the map value and $\pm$0.1 bar. The injection pressure was changed to a map value of $\pm$50 bar.

At 1500 rpm, BMEP of 4 bar, the range of the air mass swing was ±25 mg for the low-speed and low-load point. The range of the air mass swing was extended to ±100 mg to cover a broad variation range for the air mass at 1750 rpm, BMEP of 16 bar. The main injection swing was also performed at seven operating points, and its values were the map value ±2 CA.

Figures 6.13 and 6.15 show the coverage of the swing experimental data compared to the WLTP cycle at 1500 rpm, BMEP of 4 bar (with a fuel quantity of 13.87 mg), at 1500 rpm, BMEP of 8 bar (with a fuel quantity of 25.7 mg), and at 1750 rpm, BMEP of 16 bar (with a fuel quantity of 51.07 mg). The operating ranges of the WLTP to be compared were ±20 rpm for the engine speed and ±1 mg for the fuel quantity. As shown in the heatmap of the NO_x emissions (Figure 6.13 – 6.15 (a)) these three operating points to be analyzed were located at the high emitted area of NO_x. The data for the swing parameters had a normal distribution around the map values.

At 1500 rpm, BMEP of 4 bar, the swing experimental ranges of each swing parameter were ±25 mg for the intake air mass, ±0.1 bar for the intake pressure, ±50 bar for the intake pressure, and ±2 CA for the main injection timing. The light blue squares represent the ranges of the swing experiments.

As seen in Figures 6.13 (b) and (c), the coverage ranges of the intake air mass and intake pressure resulting from the swing experiments were 87% and 69% of the data distribution of the WLTP cycle, respectively. In the case of the injection pressure, as shown in Figure 6.13 (d), the swing experimental data covered 99.3% of the WLTP cycle at 1500 rpm, BMEP of 4 bar. The coverage of the main injection timing was 100%, as shown in Figure 6.13 (e). The coverage of the parameters related to the intake air was relatively low compared to the injection pressure and main injection

timing. At 1500 rpm, BMEP of 4 bar, which included a low-load operating range, the swing of the intake air parameters was limited to certain ranges because of the lack of airflow. At 1500 rpm, BMEP of 4 bar, the swing experiments for the intake parameters were designed to cover more than 50% ranges of the ranges of the WLTP data distribution, and they covered 87%, 69% of the intake air mass and intake pressure, respectively.

At 1500 rpm, BMEP of 8 bar, swing experiments were conducted for ±50 mg of the intake air mass, ±0.1 bar of the intake pressure, ±50 bar of the injection pressure, and ±2 CA of the main injection timing. In this case, the swing experimental data of the intake air mass covered 84% of the data distribution of the WLTP cycle as shown in Figure 6.14 (b). The data coverages of other swing parameters such as the intake pressure, injection pressure, and main injection timing were all 100% presented in Figure 6.14 (c) – (e).

At 1750 rpm, BMEP of 16 bar, swing experimental ranges were ±100 mg for the intake air mass, ±0.1 bar for the intake pressure, and ±50 bar for the injection pressure. A swing experiment for the main injection timing was not performed at this point. The swing value of the intake air mass, ±100 mg, was set larger than that of other operating points because the operating point was a high load condition. The data coverage of the swing experiments was 100% for the intake air mass and injection pressure, and 85% for the intake pressure, as shown in Figure 6.15.

Table 6.6 lists the optimized hyperparameters of the model trained with the steady-state experiment including the map and swing experimental data. The accuracy of NO_x emission prediction was an RMSE of 43.9 ppm, and the prediction result of a normalized cumulative NO_x was 0.976 as shown in Figure 6.16. The accuracy results

were improved compared to the results of the model with map experimental data in section 6.3.1.

The WLTP cycle consists of four parts: low (0 – 589 s), medium (589 – 1022 s), high (1022 – 1044 s), and extra-high (1477 – 1800 s). Figure 6.17 and Table 6.7 present the NO_x prediction results for WLTP parts. Among the four parts, the prediction accuracy of the extra-high part (RMSE of 52.0 ppm) and normalized cumulative mass of 0.928 were lower than those of the other parts.

NO_x emissions were mostly emitted in the extra-high region as shown in Figure 6.18 (a). Although the RMSE values were similar for all the parts, the error of the cumulative NO_x mass at the extra-high part was higher than at the other parts because the proportion of the NO_x emissions was the highest at this part. High speed and high load mainly composed the extra-high part as presented in Figure 6.18 (b). In this study, a relatively small dataset (steady-state dataset) compared to the target prediction (WLTP cycle) was used for training; therefore, the prediction accuracy was limited by the rapid changes in the engine speed and load at the extra-high part.

Figure 6.19 (a) indicates an increasing error in the normalized cumulative NO_x mass in the extra-high part. In particular, the gap between the measured NO_x and predicted NO_x mass widened at 1640 – 1720 s. As shown in Figure 6.19 (b), the error accumulated because the level of predicted NO_x emissions was lower than that of the measured NO_x emissions. Figure 6.20 shows a comparison between the EGR rates measured by the HORIBA emission bench and calculated by ECU under steady-state conditions. The EGR rate measured by HORIBA could be recognized as a true value, and the ECU EGR rate was derived using the EGR model adopted in the ECU. The R^2 value was 0.9264, and the regression equation was y=0.8352x+6.994. From the

relationship between the two values, the ECU EGR rate was higher than the HORIBA EGR rate where the EGR rate was lower than 35%. The period of 1640 – 1740 s was included in the high-load conditions, and the EGR rate during this period was lower than 35%. The EGR rate from the ECU was higher than the true value, and this was a source of the error in the extra-high part. The error caused by the ECU EGR model was inevitable because a constraint of this study was that the input data were limited to available data from the ECU, without using additional sensors.

In addition, the effect of the bias of the measurement equipment on the model was analyzed. The NO_x emission data used in this study was measured with the same equipment as both the training and the test data, so the equipment bias was ignored. Therefore, to evaluate the effect on the measurement equipment bias, it was assumed that the NO_x emissions of the training data and the test data were measured with different equipment. It means that the accuracy of the model was evaluated when there was +1% bias in the training data, and when there was +1% bias in the test data, respectively.

As shown in Figure 6.21, the +1% bias for training data and test data respectively contributed to a 1% drop in the accuracy of the cumulative NO_x emission prediction. In other words, the bias of the measured value had a proportional effect on the accuracy of the cumulative NO_x emission prediction, and from the results, it was found that securing the accuracy of the experimental data was an important factor for training and prediction of the model.

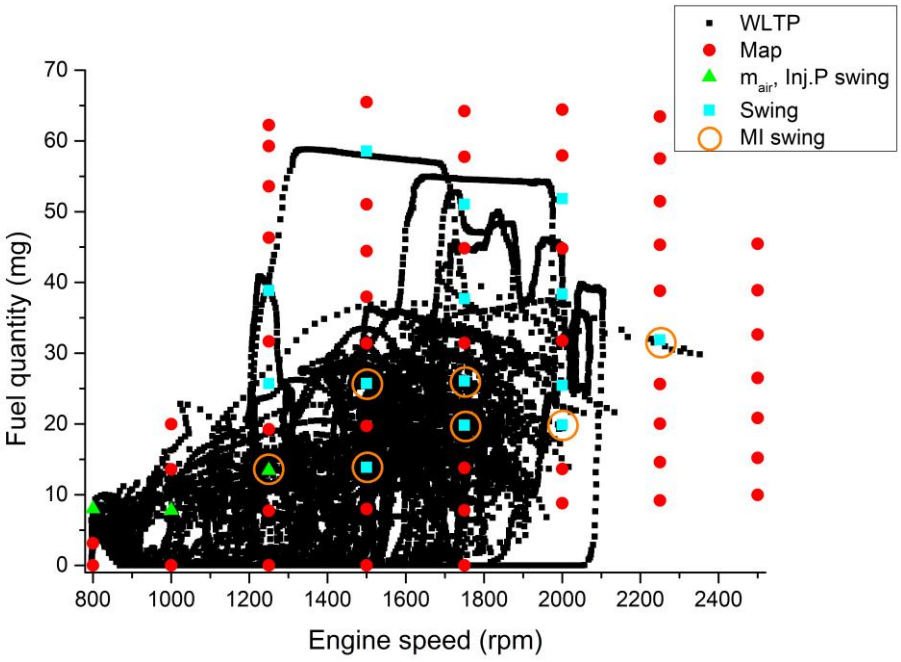


Figure 6.11. Operating points of the WLTP cycle and steady-state experiment.

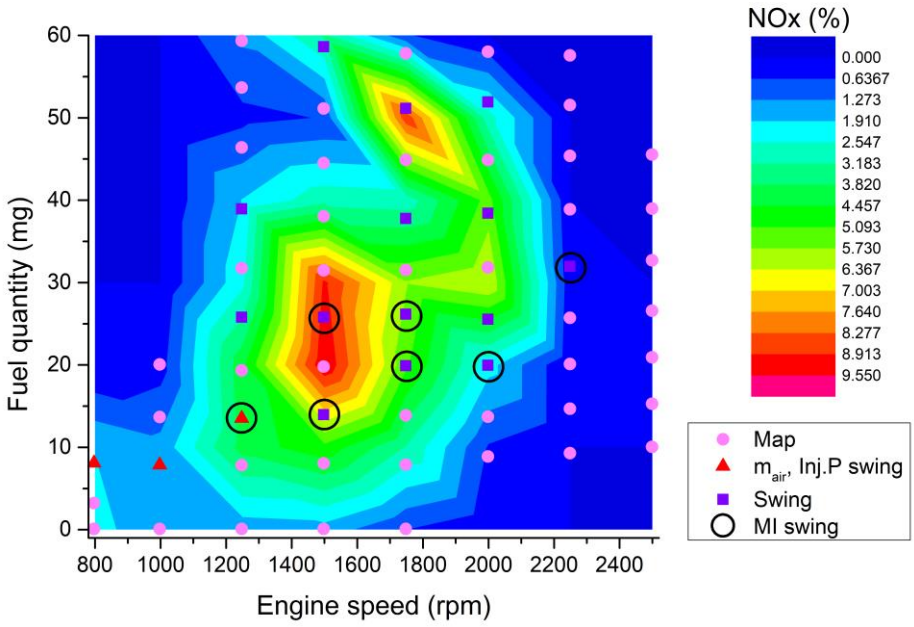
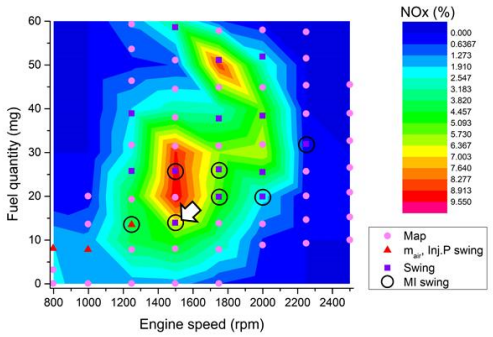


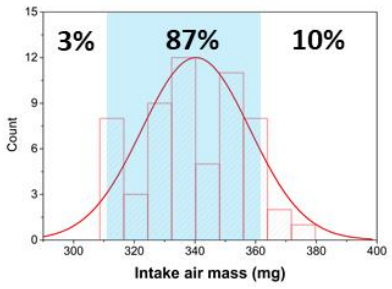
Figure 6.12. Contour of NO_x emissions and operating points of steady-state experiment.

Condition	Engine speed [rpm]	BMEP [bar]	Remark
Motoring	800, 1000, 1250, 1500, 1750	Fuel mass = 0	
Idle	800	0	
Map	1000	2, 4, 6	
	1250	2, 4, 6, 8, 10, 12, 14, 16, 18, 20	
	1500	2, 4, 6, 8, 10, 12, 14, 16, 18, 20	
	1750	2, 4, 6, 8, 10, 12, 14, 16, 18, 20	
	2000	2, 4, 6, 8, 10, 12, 14, 16, 18, 20	
	2250	2, 4, 6, 8, 10, 12, 14, 16, 18, 20	
	2500	2, 4, 6, 8, 10, 12, 14	
Swing - Air mass (±25 mg) - Injection pressure (±50 bar)	800	2	Low speed / low load
	1000	2	
	1250	4	
Swing - Air mass (±50 mg) - Intake pressure (±0.1 bar) - Injection pressure (±50 bar)	1250	8, 10	• 1500 – 4: Air mass (±25 mg) • 1750 – 16: Air mass (±100 mg)
	1500	4, 8, 18	
	1750	6, 8, 12, 16	
	2000	6, 8, 12, 16	
	2250	10	
Swing - Main injection timing (±2 CA)	1250 – 4, 1500 – 4 & 8, 1750 – 6 & 8, 2000 – 6, 2250 – 10		Additionally performed at major calibration points

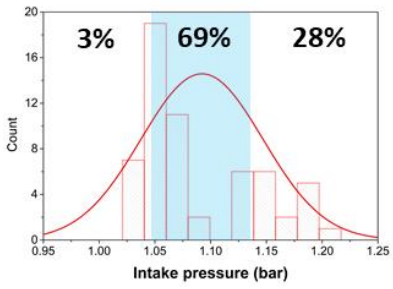
Table 6.5. Experimental conditions of steady-state experiment.



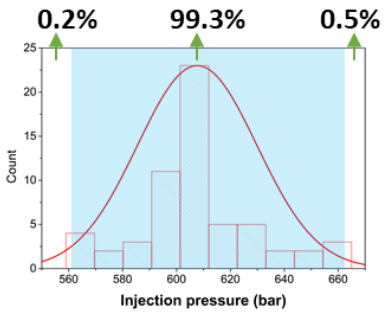
(a)



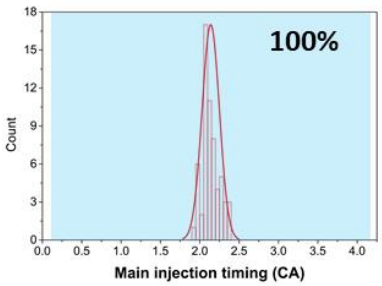
(b)



(c)

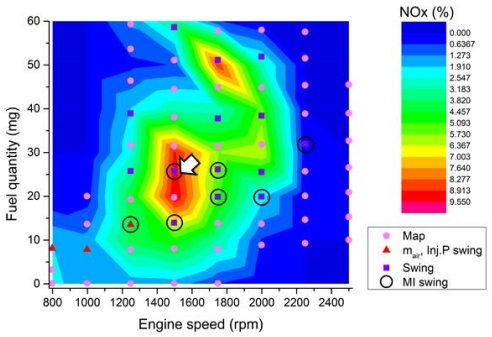


(d)

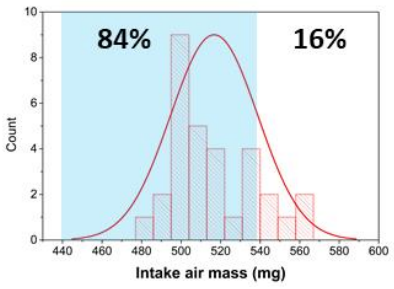


(e)

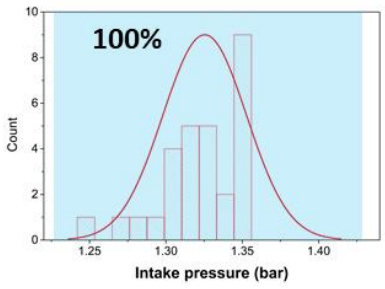
Figure 6.13. Coverage of swing experimental data at 1500 rpm, BMEP of 4 bar compared to WLTP cycle: (a) experimental point, (b) intake air mass, (c) intake pressure, (d) injection pressure, and (e) main injection timing.



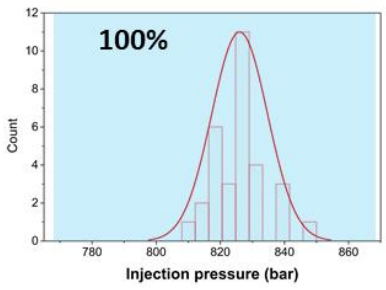
(a)



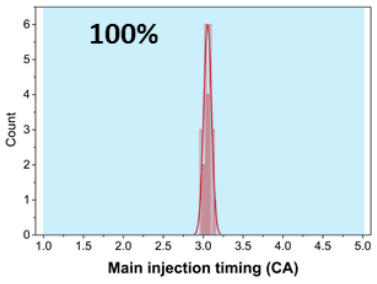
(b)



(c)

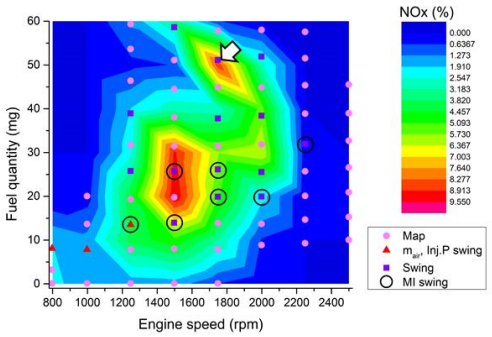


(d)

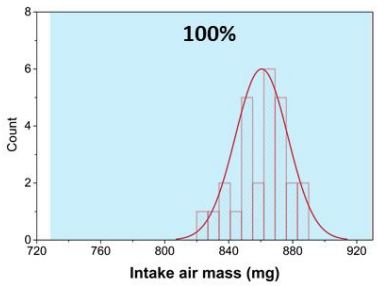


(e)

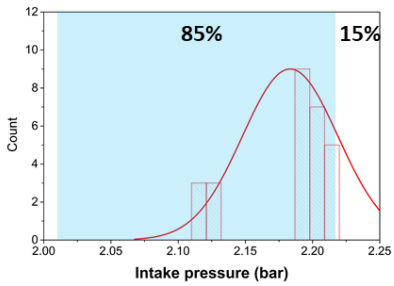
Figure 6.14. Coverage of swing experimental data at 1500 rpm, BMEP of 8 bar compared to WLTP cycle: (a) experimental point, (b) intake air mass, (c) intake pressure, (d) injection pressure, and (e) main injection timing.



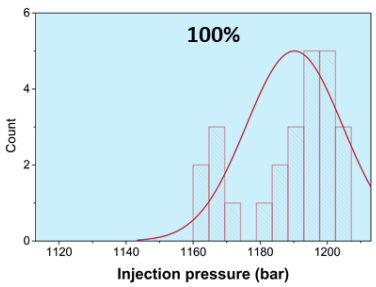
(a)



(b)



(c)

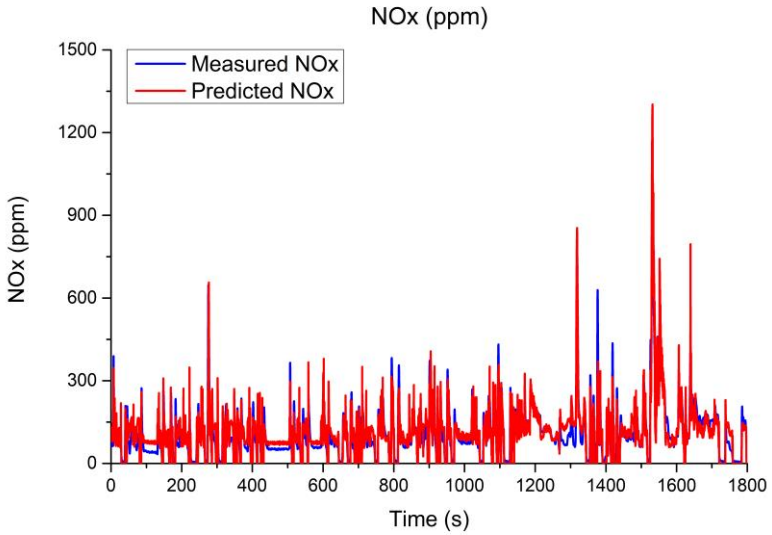


(d)

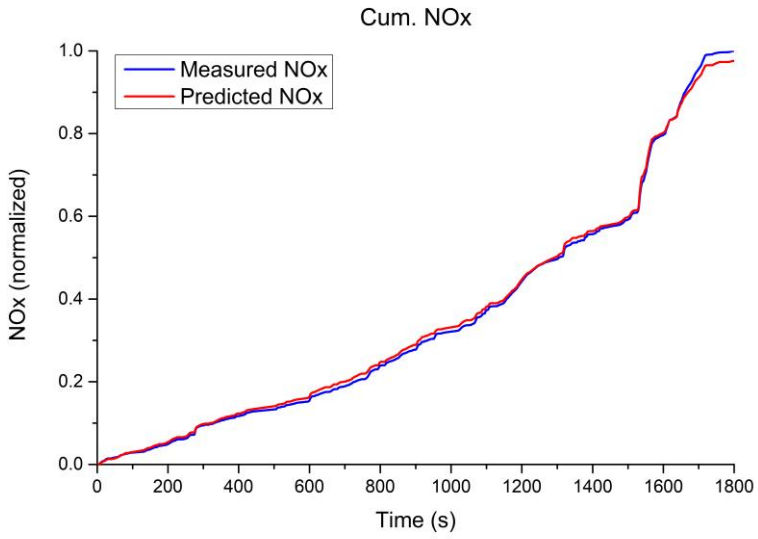
Figure 6.15. Coverage of swing experimental data at 1750 rpm, BMEP of 16 bar compared to WLTP cycle: (a) experimental point, (b) intake air mass, (c) intake pressure, and (d) injection pressure.

Hyperparameter	Optimized value
Learning rate	$10^{-2.724}$
Learning rate decay	$10^{-6.740}$
Batch size	45
Dropout rate	0.2
Number of hidden layers	6
Number of 1st hidden nodes	17
Node arrangement	10-17-24-20-16-12-8-1

Table 6.6. Optimized hyperparameters of model trained with steady-state experiment.

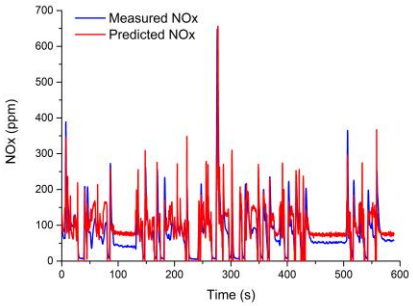


(a)

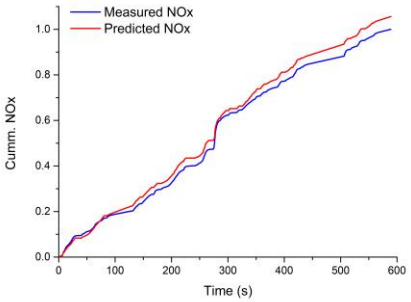


(b)

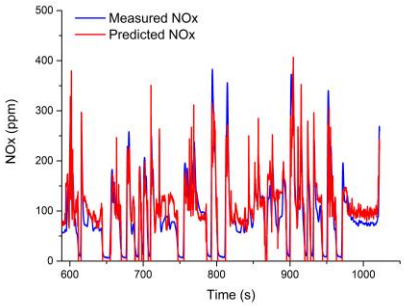
Figure 6.16. Prediction results of model trained by steady-state experimental data: (a) NO_x emission profile, and (b) cumulative NO_x mass (normalized).


Measured NOx
Predicted NOx
NOx (ppm)
Time (s)

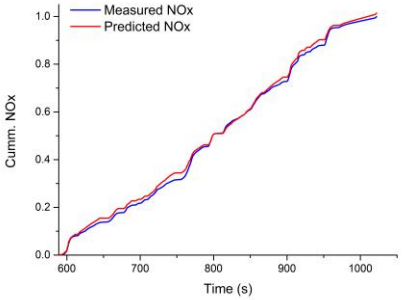
(a)


Measured NOx
Predicted NOx
Cumm. NOx
Time (s)

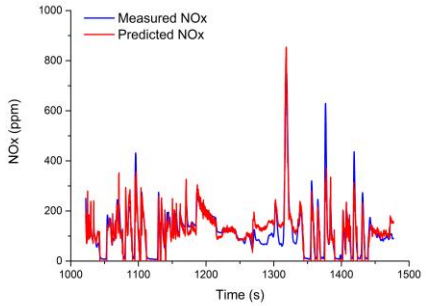
(b)


Measured NOx
Predicted NOx
NOx (ppm)
Time (s)

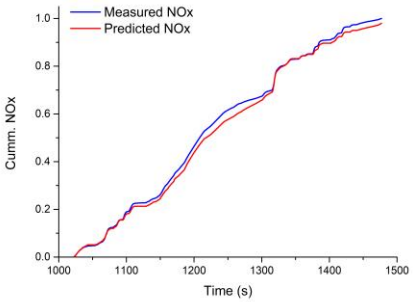
(c)


Measured NOx
Predicted NOx
Cumm. NOx
Time (s)

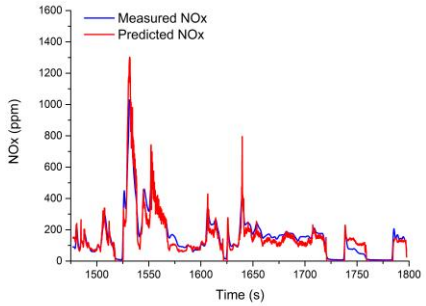
(d)



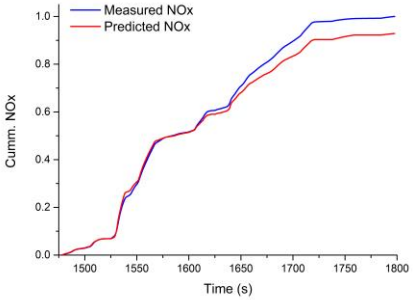
(e)



(f)



(g)

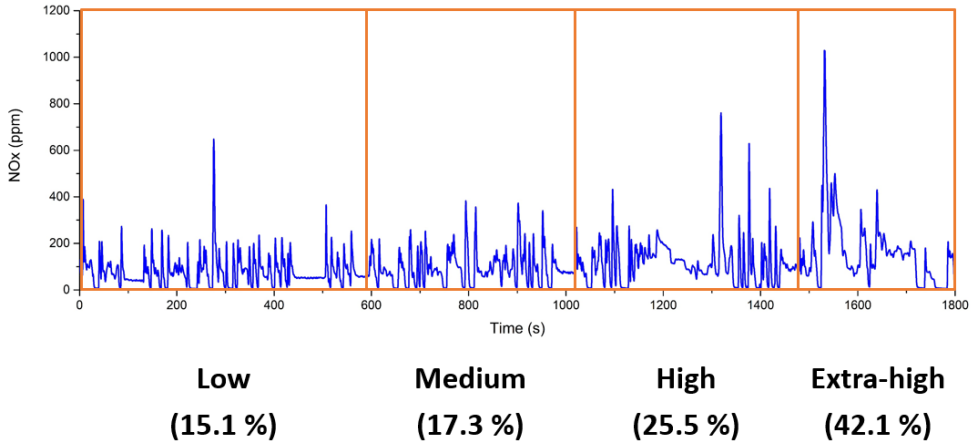


(h)

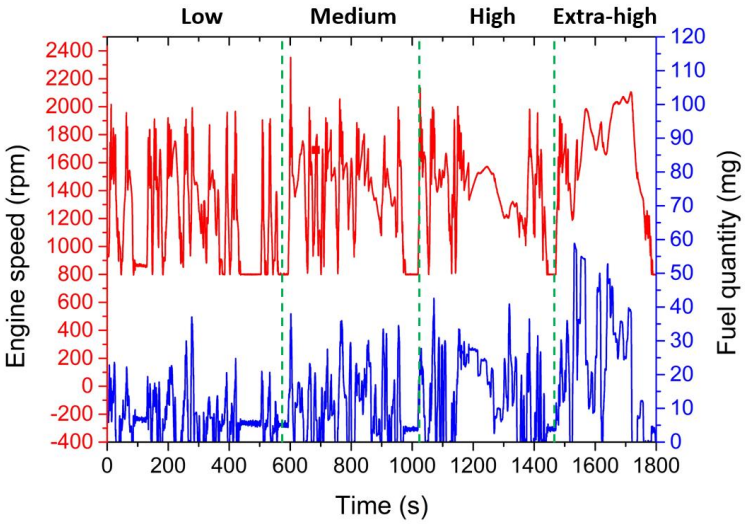
Figure 6.17. NO_x prediction results of WLTP parts: NO_x emission profile and cumulative NO_x mass (normalized) – (a)&(b) Low part, (c)&(d) Medium part, (e)&(f) High part, and (g)&(h) Extra-high part.

	Low part (0 – 589 s)	Medium part (589 – 1022 s)	High part (1022 – 1477 s)	Extra-high part (1477 – 1800 s)	Total (0 – 1800 s)
RMSE [ppm]	44.1	37.7	42.9	52.0	43.9
Cumulative mass [normalized]	1.056	1.014	0.980	0.928	0.976

Table 6.7. NO_x prediction accuracies of each WLTP part.

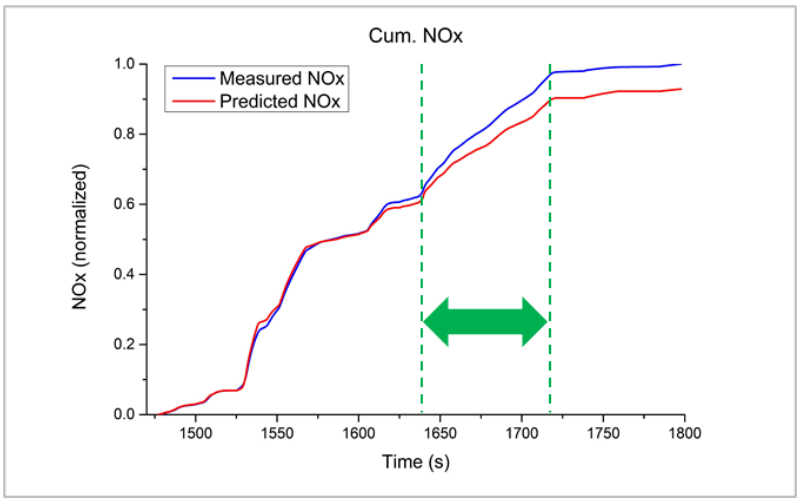


(a)

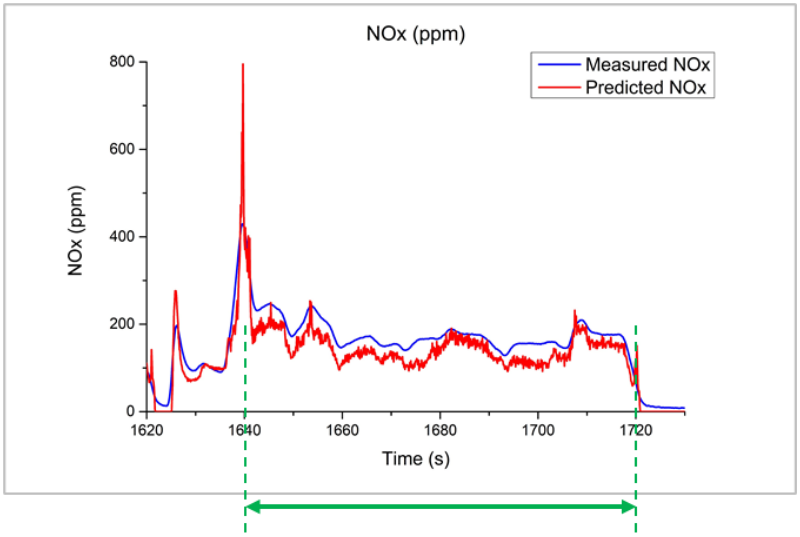


(b)

Figure 6.18. The profiles of the WLTP cycle: (a) portion of NO_x emissions by WLTP parts, and (b) engine speed – fuel quantity .



(a)



(b)

Figure 6.19. (a) Increasing error of normalized cumulative NO_x mass at 1640 – 1720 s, and (b) NO_x emission profile at 1640 – 1720 s.

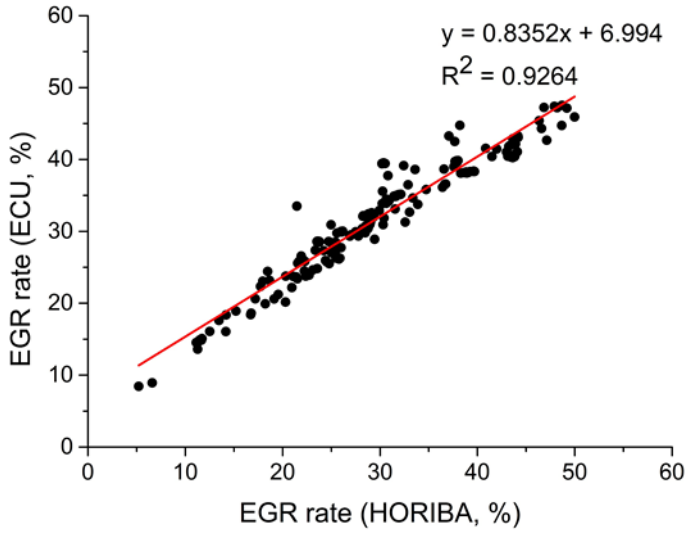


Figure 6.20. Comparison between EGR rates measured by HORIBA emission bench and calculated by ECU under steady-state conditions.

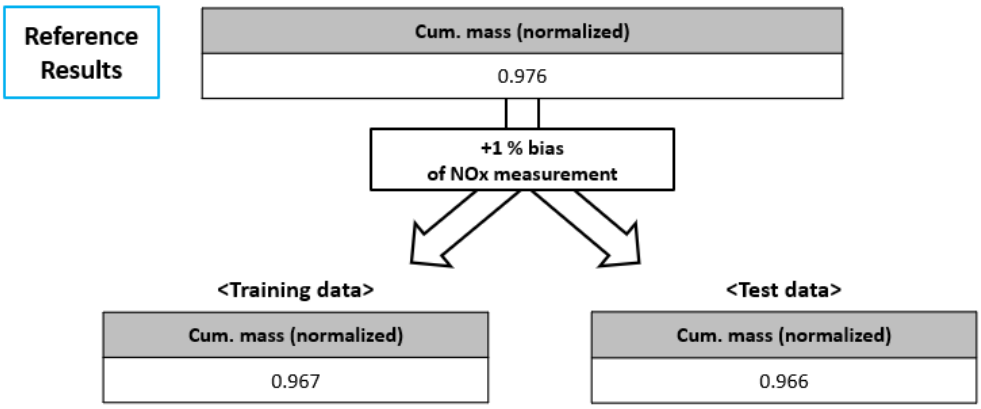


Figure 6.21. Effect of bias of measurement equipment on prediction accuracy of model.

6.3.3 Design of additional experimental points for considering intake and coolant temperatures

In the previous section, the model was trained by steady-state experimental data to predict NO_x emissions of the WLTP 1 cycle. As shown in Table 3.5, three more WLTP cycles (WLTP 2 – 4) were experimented with changing intake and coolant temperatures.

Table 6.8 and Figure 6.22 were the results of WLTP 2 – 4 predicted by the model of the previous section. The result of the WLTP 2 was similar to the accuracy result of the WLTP 1 (an RMSE of 44 ppm and a cumulative NO_x mass of 0.976). However, the accuracy results of WLTP 3 and 4 deteriorated. It was because the dataset of the steady-state experiment did not include the temperature information as listed input variables in Table 3.6.

As compared to the accuracy results and temperature profiles presented in Figure 6.23, the coolant temperature differences mainly influenced the accuracies. While the engine was operated by a dynamometer, the intake temperature and coolant temperature were influenced by the setpoints and from each other. For the WLTP 2 and 3, the setpoints of the intake temperature were different from that of WLTP 1. The coolant temperature profile of the WLTP 2 was similar to that of WLTP 1. It meant that the intake temperature difference between the WLTP 1 and 2 was not great to affect the coolant temperature. However, the coolant temperature profile of the WLTP 3 was distinguished from that of the WLTP 1 because the intake temperature difference affected the coolant temperature. The coolant temperature of the WLTP 4

was different from that of the WLTP 1 since the setting value of the coolant temperature in WLTP 4 was 40 °C.

As a result of the evaluation, the model trained in Section 5.3.2 accurately predicted the WLTP 2 which had a similar coolant temperature profile compared to the WLTP 1. However, the model could not predict the WLTP 3 and 4 which had different coolant temperature profiles caused by different intake temperatures or coolant temperatures.

Ideally, the steady-state experiment designed in the previous section (total 251 cases) should be performed with other temperature settings, e.g. 20 °C of the intake temperature and 85 °C of the coolant temperature for the WLTP 3. However, the repetition of the experiment for a certain temperature required much additional time and cost causing ineffectiveness.

In order to provide information on the intake and coolant temperatures to the model, two input variables were added to the dataset, and additional experimental conditions were designed.

The intake temperature and coolant temperature of the steady-state experimental data obtained in Section 6.3.2 were 35 °C and 85 °C, respectively. The swing experiments of the intake and coolant temperature were conducted at seven operating points which were the major calibration points of the engine manufacturer described in the previous section. Engine speed and BMEP conditions of the major calibration points were 1250 – 4, 1500 – 4, 1500 – 8, 1750 – 6, 1750 – 8, 2000 – 6, and 2250 – 10 as presented in Table 6.5.

For the intake temperature swing, the coolant temperature was maintained at 85 °C. Three intake temperatures were considered in the experiment, except at 1250 rpm, BMEP of 4 bar, and at 2250 rpm, BMEP of 10 bar. The minimum intake temperature was 35 °C, and the maximum temperature was determined to be the highest temperature at which the engine could be stably operated at each operating point. The last temperature was determined to be between the minimum and maximum temperatures. At 1250 rpm, BMEP of 4 bar, only two temperatures were explored because the point was a low-load condition. At 2250 rpm, BMEP of 10 bar, the data were measured at four temperatures because the gap between the minimum and maximum temperatures was larger than that for other points. For the coolant temperature swing, the coolant temperature was set at 40 °C, 55 °C, 70 °C, and 85 °C, while the intake temperature was maintained at 35 °C. The conditions of the temperature swing experiments are listed in Table 6.9.

The dataset combining the steady-state experiment (251 points, map + swing experiments) and the temperature swing experiment (49 points, intake + coolant temperature) was utilized to optimize hyperparameters, and the optimized set was listed in Table 6.10. The node arrangement was different from Table 6.6 because the first node was increased to 12 by containing the intake and coolant temperature for input variables.

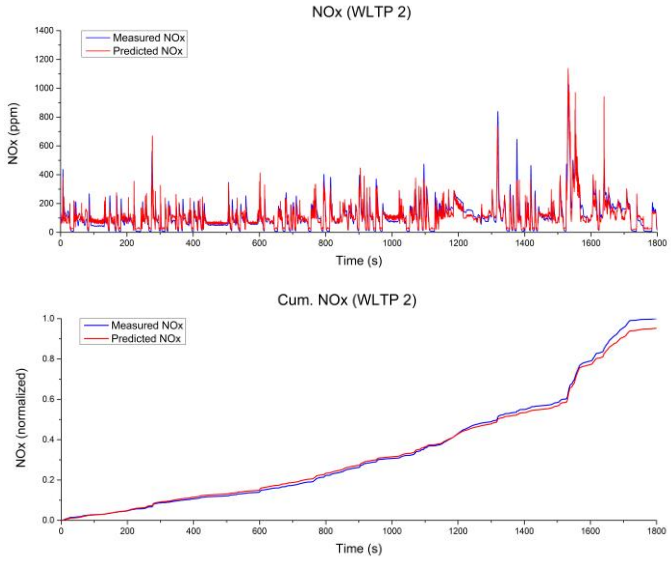
Table 6.11 and Figure 6.24 present the NO_x prediction accuracy of the WLTP 3 and 4 predicted by the model. The RMSE errors were 37 ppm and 40 ppm for WLTP 3 and 4, respectively. Normalized cumulative NO_x masses were 0.961 and 1.091 for WLTP 3 and 4. The accuracy results of the WLTP cycles trained with the data including the temperature swing experiment (WLTP 3 – 4 in Table 6.11) were

increased compared to the results of the model trained only by steady-state experimental data (WLTP 3 – 4 in Table 6.8).

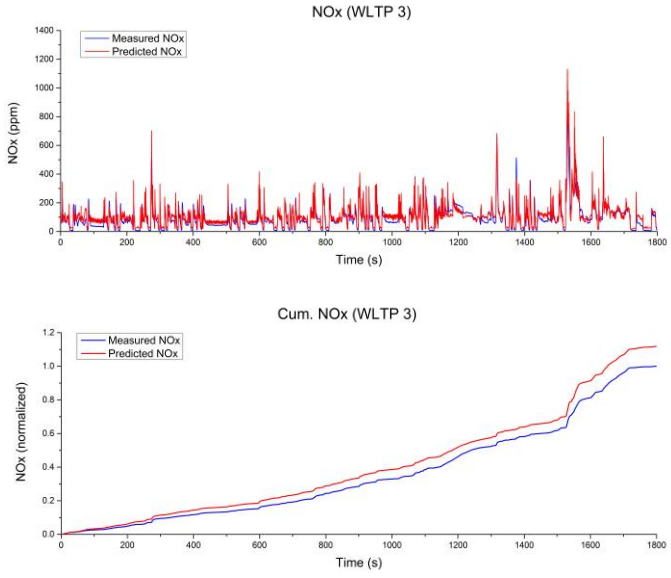
The results signified that some additional experimental cases enabled the model to predict the effects of the intake and coolant temperature on NO_x emissions. The number of additional data cases was not a multiple of 251, which would simply repeat the map and swing experiments at other temperatures. It was already possible for the model to predict transient conditions using the 251 cases of the steady-state experiment, and only 49 additional data points were required to include the effect of the temperature.

	WLTP 2	WLTP 3	WLTP 4
RMSE [ppm]	45.8	41.1	78.2
Cumulative mass [Normalized]	0.952	1.118	1.636

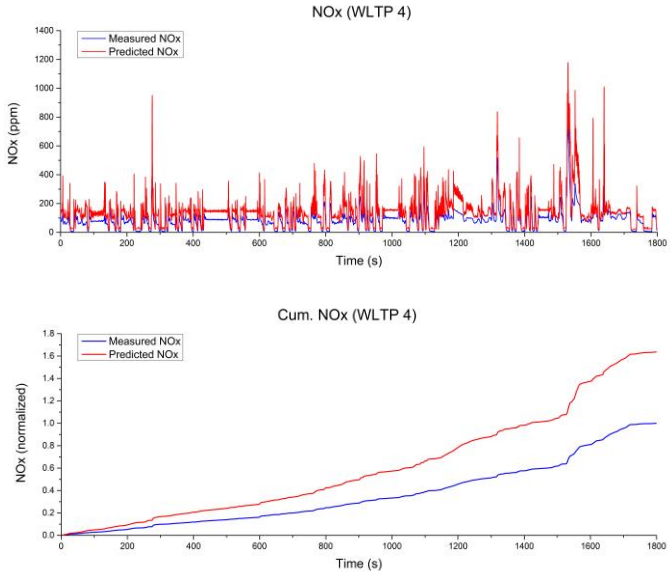
Table 6.8. NO_x prediction accuracy of the model trained by steady-state experimental data for WLTP 2 – 4.


NOx (WLTP 2)
Measured NOx
Predicted NOx
NOx (ppm)
Time (s)
Cum. NOx (WLTP 2)
Measured NOx
Predicted NOx
NOx (normalized)
Time (s)

(a)

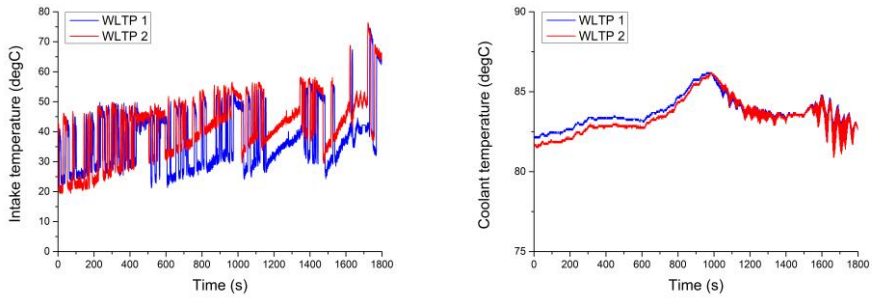

NOx (WLTP 3)
Measured NOx
Predicted NOx
NOx (ppm)
Time (s)
Cum. NOx (WLTP 3)
Measured NOx
Predicted NOx
NOx (normalized)
Time (s)

(b)

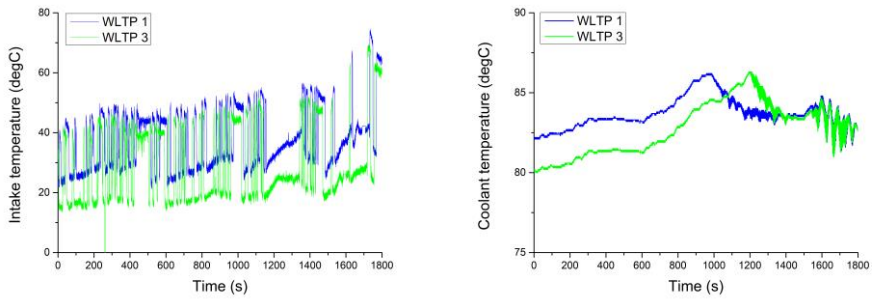


(c)

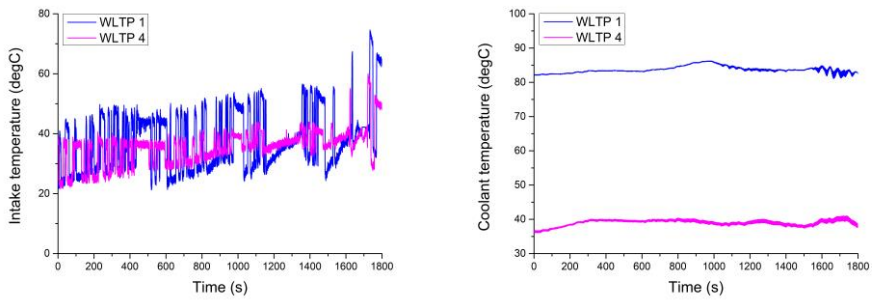
Figure 6.22. NO_x emission profiles (upper figures) and normalized cumulative NO_x mass (bottom figures) predicted by model trained with steady-state experimental data for (a) WLTP 2, (b) WLTP 3, and (c) WLTP 4.



(a)



(b)



(c)

Figure 6.23. Intake temperature (left figures) and coolant temperature (right figures): (a) WLTP 2, (b) WLTP 3, and (c) WLTP 4.

	Engine speed - BMEP [rpm – bar]	Intake temperature [°C]	Coolant temperature [°C]
Intake temperature swing (21 points)	1250 – 4	35, 55	85
	1500 – 4	35, 50, 65	
	1500 – 8	35, 70, 105	
	1750 – 6	35, 60, 90	
	1750 – 8	35, 70, 105	
	2000 – 6	35, 70, 105	
	2250 – 10	35, 60, 85, 140	
Coolant temperature swing (28 points)	1250 – 4	35	40, 55, 70, 85
	1500 – 4		
	1500 – 8		
	1750 – 6		
	1750 – 8		
	2000 – 6		
	2250 – 10		

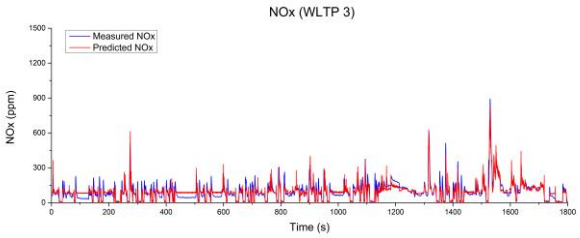
Table 6.9. Intake and coolant temperatures in temperature swing experiments.

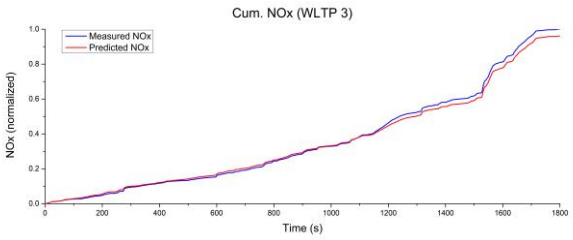
Hyperparameter	Optimized value
Learning rate	$10^{-2.724}$
Learning rate decay	$10^{-6.740}$
Batch size	45
Dropout rate	0.2
Number of hidden layers	6
Number of 1st hidden nodes	17
Node arrangement	12-17-22-18-14-10-6-1

Table 6.10. Optimized hyperparameters of model trained with data combined by steady-state and temperature swing experiments.

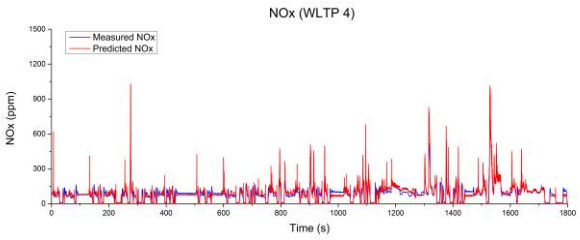
	WLTP 3	WLTP 4
RMSE [ppm]	37.4	39.9
Cumulative mass [Normalized]	0.961	1.091

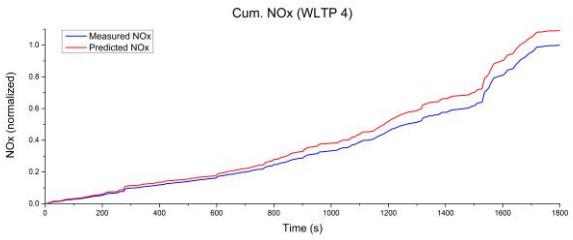
Table 6.11. NO_x prediction accuracy of WLTP 3 – 4 by model trained with data combined by steady-state and temperature swing experiment.





(a)





(b)

Figure 6.24. NO_x emission profiles (upper figures) and normalized cumulative NO_x mass (bottom figures) predicted by model trained with data combined by steady-state and temperature swing experiment for (a) WLTP 3, and (b) WLTP 4.

6.3.4 Case Study: Application of Dataset Designing Methodology to RDE

The case study for applying the dataset designing methodology developed in this study was conducted to predict RDE NO_x emissions. This case study provided implications to apply the methodology to other mechanical systems.

Figure 6.25 shows an experimental setup for RDE and Table 6.12 is test vehicle specifications. The vehicle was equipped with NOx sensors at the front and rear end of the selective catalytic reduction, which is for post-processing of NO_x emissions. Because the prediction target of this study was the engine-out NOx emissions, an additional NO_x sensor was installed in the front end of the lean NO_x trap in the vehicle to measure the engine-out NO_x emissions.

RDE was conducted on the route developed by the National Institute of Environmental Research [86] as shown in Figure 6.26, and data were measured for approximately 8300 seconds as shown in Figure 6.27.

Experimental considerations of designing steady-state experiments for RDE applications are largely divided into the analysis of operating area and ranges of swing variables.

Figure 6.28 (a) shows the steady-state operating points selected based on the WLTP cycle conducted in this study. When these operating points were substituted into the RDE data as shown in Figure 6.28 (b), it could be seen that the overall operating areas of the RDE were covered by the steady-state operating points.

An analysis of the ranges of swing variables should be carried out. In this study, the ranges of swing variables were determined to cover the WLTP variable range.

Through the analysis of variables from RDE data, the swing ranges for the variables of the steady-state experiment should be considered as presented in Figure 6.29.

The analysis results of operating points and ranges of swing variables were the prerequisite for the steady-state experiment to predict RDE conditions. After the process, the experiments provided in sections 6.3.2 and 6.3.3 could be performed.

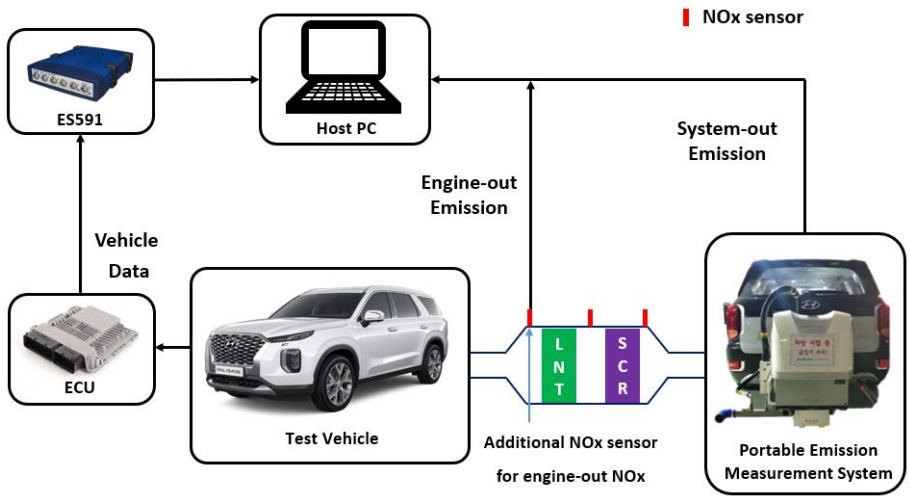


Figure 6.25. Experimental setup for measurement of engine-out NO_x emissions under RDE conditions.

Vehicle specifications	
Engine type	Diesel e-VGT
Displacement volume [L]	2.199
Maximum output [PS/rpm]	202 / 3,800
Maximum torque [kg.m/rpm]	45.0 / 1,750-2,750 rpm
Overall length [mm]	4,980
Overall width [mm]	1,975
Overall height [mm]	1,750

Table 6.12. Test vehicle specifications.

Figure 6.26. Map of KOR-NIER Route 1 [86].

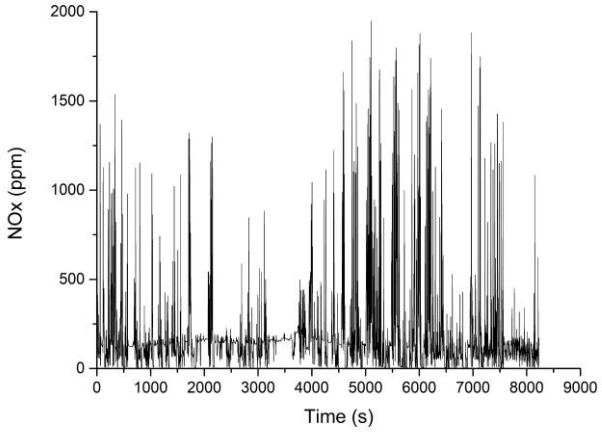
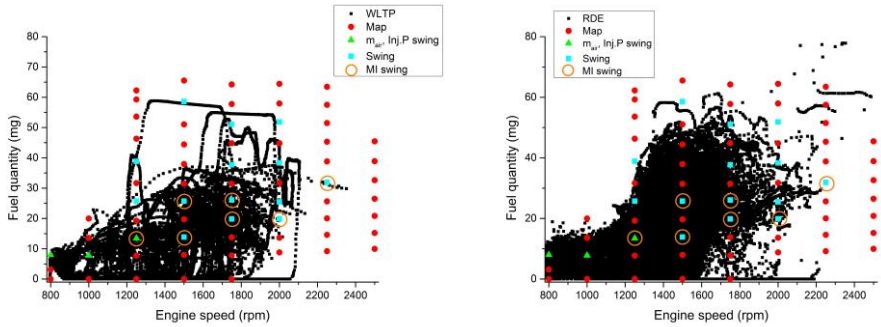


Figure 6.27. NO_x measurement data on RDE conditions.



(a) (b)

Figure 6.28. (a) Steady-state operating points selected based on WLTP cycle, and (b) substitution result of steady-state operating points to RDE data.

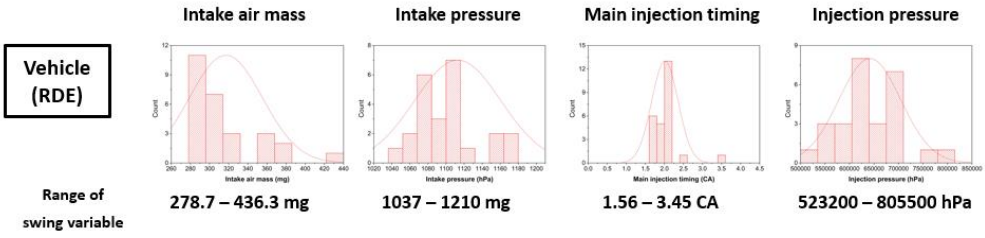


Figure 6.29. Ranges of swing variables under RDE conditions at 1500 rpm, BMEP of 4 bar.

Chapter 7. Conclusions

In this study, the engine-out NO_x emissions under transient conditions of a diesel engine were predicted via deep learning methods. Conventional modeling approaches simplified phenomena and extracted major variables to construct equations to predict emissions, and they were causes of error. However, the deep learning model considered all implicit meaning in data, so accuracy losses could be prevented compared to conventional approaches.

This study dealt with overall processes to establish deep learning models including hyperparameter optimization, algorithm comparison, and dataset design. Hyperparameters, structures of deep learning models, were optimized via Bayesian optimization and hidden-node determination logic. DNN and LSTM models were compared for the NO_x emission prediction based on the accuracy and calculation time, and data preprocessing was suggested to increase the accuracy of the DNN model. In addition, the experimental design under steady-state conditions was performed to predict NO_x emissions under transient conditions using the model trained by the steady-state dataset.

In conclusion, the methodology for designing a steady-state dataset was developed to predict transient NO_x emissions, which was for domain transfer, using a deep learning model. Transient behaviors were analyzed to design steady-state experiments. Before studying on domain transfer, hyperparameter optimization method and algorithm evaluation were performed under transient conditions to define essential elements of deep learning studies.

7.1 Hyperparameter optimization using the Bayesian optimization and hidden-node determination logic

A DNN model with optimized hyperparameters was developed to predict engine-out NO_x emissions at the WLTP cycle of a diesel engine. The Bayesian optimization method and a hidden-node determination logic were introduced and developed to determine the model's hyperparameters. The target hyperparameters for optimization were the number of hidden layers, number of hidden nodes in each hidden layer, learning rate, learning rate decay, and batch size.

The Bayesian optimization method considered previous results to select the next iteration values based on Bayes' rule. The hidden-node determination logic converted the number of hidden nodes in the first hidden layer and the number of hidden layers to the number of hidden nodes in each hidden layer.

The key findings of this study are as below.

1) Compared to grid search and random sampling, the accuracy of the model using the Bayesian optimization method increased, and the optimal points were obtained more effectively. The accuracy increased, as evident by an R^2 value of 0.9675, by adopting the hidden-node determination logic to the Bayesian optimization method.

2) As for the accuracy of cycle prediction, the MAE of the WLTP cycles was 16–17 ppm. The level of error was approximately 1.6% of the maximum NO_x at each cycle. The accuracy of the model was comparable to that of a physical NO_x measurement device, which demonstrated linearity that was 1% of the full scale (5,000 ppm).

3) The Bayesian optimization method and the hidden-node determination logic enabled the hyperparameters of the DNN to be optimized regardless of the input features and target outputs.

7.2 Deep learning algorithms and time-series data preprocessing

DNN and LSTM deep-learning models were developed to predict the engine-out NO_x emissions under WLTP cycles in a diesel engine. Two types of data—the measured data and preprocessed data—were applied to train the models, with several ratios of previous-timestep data to the current-timestep data. From the results, the following conclusions are drawn.

1) The accuracy of LSTM (R^2 = 0.9777), using the measured data was higher than that of the DNN (R^2 = 0.9671). However, the DNN model had a far shorter computation time (0.36 s) than the LSTM model (1381.0 s). As a virtual sensor with a real-time prediction capability, the DNN model is more appropriate owing to its calculation speed.

2) Data preprocessing was effective for increasing the accuracy of the DNN model but not for the LSTM model. Through the data preprocessing, the accuracy of the DNN model was increased at the ratio of 7:3. The achieved accuracy of the DNN model with data preprocessing was given by R^2 = 0.9741 and was only slightly lower (R^2 value 0.0036 smaller and RMSE 2.2 ppm larger) than that of the LSTM model.

Data preprocessing helped to improve the limitation of the DNN regarding the lack of time information.

3) There were no significant differences between the detailed trajectories of the DNN model with data preprocessing and the LSTM model with measured data.

Through the data preprocessing, the DNN model achieved an accuracy comparable to that of the LSTM model while maintaining the advantage regarding the computation time. The results of this study indicate that the DNN model with data preprocessing is suitable as a virtual sensor with a sufficient calculation speed for real-time predictions.

7.3 Steady-state experimental design for prediction of transient NOx emissions

The experiments under steady-state conditions were designed to predict transient NO_x emissions. To determine the experimental conditions, the behaviors of the engine parameters under transient conditions were analyzed. Two procedures were performed to achieve the predictability for transient NO_x emissions of the model by training with steady-state data.

1) ECU map data and swing experiments were performed to predict the NO_x emissions during the WLTP cycle. The operating points for the swing experiments were selected based on the NO_x emission mass. The intake air mass, intake pressure, injection pressure, and main injection timing were derived as swing parameters. A total of 251 cases were considered in the

experiments, and the accuracy results of the model were an RMSE of 43.9 ppm and a normalized cumulative NO_x mass of 0.976.

2) Temperature swing experiments were designed to provide information on the temperature changes in the model. Additional experimental cases included 21 cases for the intake temperature and 28 cases for the coolant temperature. When the data from the 251 steady-state experimental data and the 49 temperature experimental data were combined, the errors in the NO_x emissions were less than 10% of the cumulative NO_x mass for all the WLTP cycles.

The study suggests a design methodology for setting steady-state experimental conditions to predict transient NO_x emissions. By swinging some engine parameters and temperatures, a dataset was constructed to train the model to predict transient NO_x emissions. This design methodology could be applied to predict RDEs. Based on the engine behaviors during the RDE conditions, a steady-state experiment could be designed to predict the RDEs. Data cases could be changed for the RDEs, but the framework of the dataset would not be different from that of the dataset derived in this study.

By overcoming the domain constraint of deep learning models, the usage of a deep learning model for the NO_x emission prediction was expanded to overall development stages of engines including design, evaluation, and prediction.

Bibliography

1. Wappelhorst, S., *The end of the road? An overview of combustionengine car phase-out announcements across Europe*. 2020.05, The International Council on Clean Transportation.
2. Min, K., *2030 Powertrain Outlook*, in *Auto Journal*. 2019.08, The Korean Society of Automotive Engineers.
3. Continental-automotive, *Worldwide Emission Standards and Related Regulations - Passenger Cars / Light and Medium Duty Vehicles*. May 2019, CPT Group GmbH.
4. Caliskan, H. and K. Mori, *Environmental, enviroeconomic and enhanced thermodynamic analyses of a diesel engine with diesel oxidation catalyst (DOC) and diesel particulate filter (DPF) after treatment systems.* Energy, 2017. **128**: p. 128-144.
5. Zheng, M., G.T. Reader, and J.G. Hawley, *Diesel engine exhaust gas recirculation—a review on advanced and novel concepts.* Energy conversion and management, 2004. **45**(6): p. 883-900.
6. Johnson, T.V., *Diesel emission control in review.* SAE international journal of fuels and lubricants, 2009. **1**(1): p. 68-81.
7. Guan, B., et al., *Review of state of the art technologies of selective catalytic reduction of NOx from diesel engine exhaust.* Applied Thermal Engineering, 2014. **66**(1-2): p. 395-414.
8. LeCun, Y., Y. Bengio, and G. Hinton, *Deep learning.* nature, 2015. **521**(7553): p. 436.

9. Silver, D., et al., *Mastering the game of Go with deep neural networks and tree search.* nature, 2016. **529**(7587): p. 484.
10. Marvin, M. and A.P. Seymour, *Perceptrons*. 1969, MIT Press.
11. Rumelhart, D.E., G.E. Hinton, and R.J. Williams, *Learning representations by back-propagating errors.* nature, 1986. **323**(6088): p. 533-536.
12. Hochreiter, S., *Untersuchungen zu dynamischen neuronalen Netzen.* Diploma, Technische Universität München, 1991. **91**(1).
13. Hinton, G.E., S. Osindero, and Y.-W. Teh, *A fast learning algorithm for deep belief nets.* Neural computation, 2006. **18**(7): p. 1527-1554.
14. Bengio, Y., et al. *Greedy layer-wise training of deep networks*. in *Advances in neural information processing systems*. 2007.
15. Russakovsky, O., et al., *Imagenet large scale visual recognition challenge.* International journal of computer vision, 2015. **115**(3): p. 211-252.
16. Brown, T.B., et al., *Language models are few-shot learners.* arXiv preprint arXiv:2005.14165, 2020.
17. Özgül, E. and H. Bedir, *Fast NOx emission prediction methodology via one-dimensional engine performance tools in heavy-duty engines.* Advances in Mechanical Engineering, 2019. **11**(4): p. 1687814019845954.
18. Finesso, R., et al., *Real-time simulation of torque and nitrogen oxide emissions in an 11.0 L heavy-duty diesel engine for model-based combustion control.* Energies, 2019. **12**(3): p. 460.
19. Özgül, E., M. Şimşek, and H. Bedir, *Use of thermodynamical models with predictive combustion and emission capability in virtual calibration of heavy duty engines.* Fuel, 2020. **264**: p. 116744.
20. Egnell, R., *Combustion diagnostics by means of multizone heat release analysis and NO calculation.* SAE transactions, 1998: p. 691-710.

21. Arrègle, J., et al., *Sensitivity study of a NOx estimation model for on-board applications*. 2008, SAE Technical Paper.
22. DONMEZ, N. and O. OZENER, *Modelıng of NOx Emmıssıons in Internal Combustıon Engıne.* International Journal of Engineering Research and Advanced Technology (IJERAT), 2019. **5**(4).
23. Leach, F., M. Davy, and M. Peckham, *Cycle-to-Cycle NO and NOx Emissions From a HSDI Diesel Engine.* Journal of Engineering for Gas Turbines and Power, 2019. **141**(8).
24. Finesso, R. and E. Spessa, *Real-time predictive modeling of combustion and NO x formation in diesel engines under transient conditions*. 2012, SAE Technical Paper.
25. Asprion, J., O. Chinellato, and L. Guzzella, *A fast and accurate physics-based model for the NOx emissions of Diesel engines.* Applied energy, 2013. **103**: p. 221-233.
26. Lee, J., et al., *The development of real-time NOx estimation model and its application*. 2013, SAE Technical Paper.
27. Lee, S., et al., *Study on reduction of diesel engine out emission through closed loop control based on the in-cylinder pressure with EGR model*. 2013, SAE Technical Paper.
28. Guardiola, C., et al., *ECU-oriented models for NOx prediction. Part 1: a mean value engine model for NOx prediction.* Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, 2015. **229**(8): p. 992-1015.
29. Lee, S., et al., *Development of a real-time virtual nitric oxide sensor for light-duty diesel engines.* Energies, 2017. **10**(3): p. 284.

30. Schaffernicht, E., et al. *Machine learning techniques for selforganizing combustion control*. in *Annual Conference on Artificial Intelligence*. 2009. Springer.
31. Tracey, B., K. Duraisamy, and J. Alonso. *Application of supervised learning to quantify uncertainties in turbulence and combustion modeling*. in *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. 2013.
32. Vaughan, A. and S.V. Bohac, *An extreme learning machine approach to predicting near chaotic HCCI combustion phasing in real-time.* arXiv preprint arXiv:1310.3567, 2013.
33. Lin, X.S., B.W. Li, and X.Y. Yang. *Engine components fault diagnosis using an improved method of deep belief networks*. in *2016 7th International Conference on Mechanical and Aerospace Engineering (ICMAE)*. 2016. IEEE.
34. Hanuschkin, A., et al., *Machine learning–based analysis of in-cylinder flow fields to predict combustion engine performance.* International Journal of Engine Research, 2019: p. 1468087419833269.
35. Asher, Z.D., et al., *Economic and Efficient Hybrid Vehicle Fuel Economy and Emissions Modeling Using an Artificial Neural Network*. 2018, SAE Technical Paper.
36. McBride, S., et al., *Estimation of Vehicle Tire-Road Contact Forces: A Comparison between Artificial Neural Network and Observed Theory Approaches*. 2018, SAE Technical Paper.
37. Kawakami, T., et al., *Classification of Time Series Measurement Data for Lock-Up Clutch of Automatic Transmission of Vehicles Using Deep Convolutional Neural Networks*. 2018, SAE Technical Paper.

38. Seo, H., et al., *A Study of Low-Friction Road Estimation using an Artificial Neural-Network*. 2018, SAE Technical Paper.
39. Mirzabeygi, P. and S. Natarajan, *Artificial Neural Network Based Predictive Approach in Vehicle Thermal Systems Applications*. 2020, SAE Technical Paper.
40. Gaikwad, T., et al., *Vehicle Velocity Prediction Using Artificial Neural Network and Effect of Real World Signals on Prediction Window*. 2020, SAE Technical Paper.
41. Hwang, G., et al., *Prediction of Hybrid Electric Bus Speed Using Deep Learning Method*. 2020, SAE Technical Paper.
42. Liu, K., et al., *Vehicle Velocity Prediction and Energy Management Strategy Part 1: Deterministic and Stochastic Vehicle Velocity Prediction Using Machine Learning*. 2019, SAE Technical Paper.
43. Winsel, T., et al., *A neural estimator for cylinder pressure and engine torque.* SAE transactions, 1999: p. 1608-1624.
44. Sayin, C., et al., *Performance and exhaust emissions of a gasoline engine using artificial neural network.* Applied thermal engineering, 2007. **27**(1): p. 46-54.
45. Arsie, I., et al., *Real-time estimation of engine NOx emissions via recurrent neural networks.* IFAC Proceedings Volumes, 2010. **43**(7): p. 228-233.
46. Ghobadian, B., et al., *Diesel engine performance and exhaust emission analysis using waste cooking biodiesel fuel with an artificial neural network.* Renewable energy, 2009. **34**(4): p. 976-982.
47. Yusaf, T.F., et al., *CNG-diesel engine performance and exhaust emission analysis with the aid of artificial neural network.* Applied Energy, 2010. **87**(5): p. 1661-1669.

48. Kapusuz, M., H. Ozcan, and J.A. Yamin, *Research of performance on a spark ignition engine fueled by alcohol–gasoline blends using artificial neural networks.* Applied Thermal Engineering, 2015. **91**: p. 525-534.
49. Turkson, R.F., et al., *Artificial neural network applications in the calibration of spark-ignition engines: An overview.* Engineering Science and Technology, an International Journal, 2016. **19**(3): p. 1346-1359.
50. Finesso, R., et al., *Neural-Network Based Approach for Real-Time Control of BMEP and MFB50 in a Euro 6 Diesel Engine*. 2017, SAE Technical Paper.
51. Rayavalasa, S., et al., *Gasoline Engine Part Load Performance: Cylinder Pressure Curves Prediction using Neural Networks to Reduce the Dependency on Testing*, in *FISITA*. 2018, SAEINDIA: Chennai, India.
52. Sediako, A.D., et al., *Heavy Duty Diesel Engine Modeling with Layered Artificial Neural Network Structures*. 2018, SAE Technical Paper.
53. Luján, J.M., et al., *Volumetric efficiency modelling of internal combustion engines based on a novel adaptive learning algorithm of artificial neural networks.* Applied Thermal Engineering, 2017. **123**: p. 625-634.
54. Mehra, R.K., et al., *Experimental and artificial neural network (ANN) study of hydrogen enriched compressed natural gas (HCNG) engine under various ignition timings and excess air ratios.* Applied Energy, 2018. **228**: p. 736-754.
55. Alcan, G., et al., *Predicting NOx emissions in diesel engines via sigmoid NARX models using a new experiment design for combustion identification.* Measurement, 2019. **137**: p. 71-81.

56. Huang, L., et al., *Applying neural networks (NN) to the improvement of gasoline turbocharger heat transfer modeling.* Applied Thermal Engineering, 2018. **141**: p. 1080-1091.
57. Lucido, M. and J. Shibata, *Learning Gasoline Direct Injector Dynamics Using Artificial Neural Networks*. 2018, SAE Technical Paper.
58. Shin, S., et al., *Deep learning procedure for knock, performance and emission prediction at steady-state condition of a gasoline engine.* Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, 2020: p. 0954407020932690.
59. Lee, S., et al., *EGR Prediction of Diesel Engines in Steady-State Conditions Using Deep Learning Method.* International Journal of Automotive Technology, 2020. **21**(3): p. 571-578.
60. Hosseini, S.H., et al., *Artificial neural network modeling of performance, emission, and vibration of a CI engine using alumina nano-catalyst added to diesel-biodiesel blends.* Renewable Energy, 2020. **149**: p. 951-961.
61. Hale, J. *Deep Learning Framework Power Scores 2018*. 2018; Available from: https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a.
62. Pelkmans, L., et al., *Development of a simulation tool to calculate fuel consumption and emissions of vehicles operating in dynamic conditions*. 2004, SAE Technical Paper.
63. Ericson, C., B. Westerberg, and R. Egnell, *Transient emission predictions with quasi stationary models*. 2005, SAE Technical Paper.
64. Gao, Z., et al., *A proposed methodology for estimating transient engine-out temperature and emissions from steady-state maps.* International Journal of Engine Research, 2010. **11**(2): p. 137-151.

65. Kulkarni, C.V., D. Rathod, and V. Sharma, *NOx Model Calibration for BS VI Applications*. 2019, SAE International.

66. Shin, S., et al., *Predicting transient diesel engine NOx emissions using time-series data preprocessing with deep-learning models.* Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, 2021: p. 09544070211005570.

67. Shin, S., et al., *Deep neural network model with Bayesian hyperparameter optimization for prediction of NOx at transient conditions in a diesel engine.* Engineering Applications of Artificial Intelligence, 2020. **94**: p. 103761.

68. Kriesel, D. *A Brief Introduction to Neural Networks*. 2007; Available from: http://www.dkriesel.com.

69. Goodfellow, I., Y. Bengio, and A. Courville, *Deep learning*. 2016: MIT press.

70. Glorot, X., A. Bordes, and Y. Bengio. *Deep sparse rectifier neural networks*. in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 2011.

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

72. Hochreiter, S. and J. Schmidhuber, *Long short-term memory.* Neural computation, 1997. **9**(8): p. 1735-1780.

73. Hochreiter, S., et al., *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*. 2001, A field guide to dynamical recurrent neural networks. IEEE Press.

74. Bengio, Y., *Practical recommendations for gradient-based training of deep architectures*, in *Neural networks: Tricks of the trade*. 2012, Springer. p. 437-478.

75. Bergstra, J. and Y. Bengio, *Random search for hyper-parameter optimization.* Journal of Machine Learning Research, 2012. **13**(Feb): p. 281-305.

76. Shahriari, B., et al., *Taking the human out of the loop: A review of Bayesian optimization.* Proceedings of the IEEE, 2015. **104**(1): p. 148-175.

77. Brochu, E., V.M. Cora, and N. De Freitas, *A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning.* arXiv preprint arXiv:1012.2599, 2010.

78. Mockus, J., V. Tiesis, and A. Zilinskas, *The application of Bayesian methods for seeking the extremum.* Towards global optimization, 1978. **2**(117-129): p. 2.

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

80. Kingma, D.P. and J. Ba, *Adam: A method for stochastic optimization.* arXiv preprint arXiv:1412.6980, 2014.

81. Hinton, G.E. and R.R. Salakhutdinov, *Reducing the dimensionality of data with neural networks.* science, 2006. **313**(5786): p. 504-507.

82. *EarlyStopping*. Keras Documentation 2019 [cited 2021. 4. 26.]; Available from: https://keras.io/api/callbacks/early_stopping/.

83. *Cambustion Ultra fast-response gas analyers for transient HC, NOx, CO & CO2 exhaust, intake and in-cylinder applications*. Cambusion: United Kingdom.

84. *NN SVG*. Available from: http://alexlenail.me/NN-SVG/index.html.

85. Ng, A., *Machine learning yearning.* URL: http://www.mlyearning. org/(96), 2017.

86. Kang, G., et al., *Development of Korean RDE routes for on-road emissions measurement of light duty vehicles.* Transactions of the Korean Society of Automotive Engineers, 2017. **25**(3): p. 287-296.

국 문 초 록

최근 딥러닝 기술의 발전은 인공지능의 비약적인 도약을 이끌었다. 딥러닝 기술은 기존의 모델링 방법론과는 다른 접근 방법으로 다양한 분야에서 높은 정확도를 보여주고 있어, 많은 연구자들에게 주목받고 있다. 자동차 및 내연기관 연구 분야에도 이러한 딥러닝을 이용한 연구가 활발히 진행되기 시작하고 있다. 기존의 연구는 엔진의 현상을 단순화하여 주요한 수식으로 모델링하여 대상 현상을 예측하는 방식을 가진다. 이 경우, 현상을 단순화하는 과정에서 오차가 발생하여 예측 정확도에 한계가 있을 수밖에 없었다. 딥러닝은 이러한 단순화 과정 없이, 데이터에 내재된 변수 간의 관계를 학습하여 현상을 예측하기 때문에 오차가 발생할 여지가 적고, 연구자가 파악하기 어려운 관계까지 파악하여 기존보다 정확한 예측 정확도를 달성할 수 있다.

본 연구에서는 기존의 모델링 방법으로는 예측 정확도에 한계가 있었던 과도 상태에서의 디젤 엔진의 질소산화물을 딥러닝을 통해 예측하였다. 하이퍼파라미터 최적화, 알고리즘 비교, 도메인 전이를 위한 데이터 세트 설계를 포함한 딥러닝 모델을 구축하기 위한 전체 프로세스에 대한 연구를 진행하였다.

딥러닝 모델의 구조인 하이퍼파라미터는 베이지안 최적화와 은닉 노드 결정 로직을 결합하여 데이터의 입출력 종류나 개수와 관계없이 자동으로 최적화하였다. 최적화 대상 하이퍼파라미터는 학습률, 학습률의 감쇄율, 배치크기, 은닉 층의 개수, 첫 번째 은닉층의 노드 개수였다. 베이지안 최적화 방법은 최적화 과정에서 이전 정보를 활용하는 베이지안 룰을 기반으로 동작하므로, 기존의 그리드 검색, 랜덤 샘플링에 비해 효과적이고 정확도가 높았다. 은닉 노드 결정 로직은 은닉층의 개수와 첫

번째 은닉층의 노드 개수를 이용하여, 은닉층의 노드 배열을 등차수열로 배치한다. 이를 통해 노드 개수의 급격한 변화로 인한 정보 손실을 막고, 각 은닉층과 노드를 개별적으로 최적화할 경우 기하급수적으로 발산하는 최적화 횟수를 제어할 수 있었다.

과도 상태에서의 질소산화물 예측에 적합한 딥러닝 모델 구조에 관한 연구도 진행되었다. Deep neural network (DNN) 모델과 Long short-term memory (LSTM) 모델의 정확도 및 계산 시간을 실시간 예측 관점에서 평가하였다. LSTM 모델은 DNN 모델에 비해 예측 정확도가 높았으나, 계산에 더 많은 시간이 소요되어 실시간 예측으로의 적용에는 적절하지 않았다. 또한 데이터 전처리를 통해 DNN 모델의 빠른 계산 속도의 장점을 유지하면서, 계산 시간의 정확도를 LSTM 모델과 비슷한 수준으로 높일 수 있었다. 이는 딥러닝을 통해 과도 상태의 질소산화물을 예측할 시, 준 정상 상태를 통한 예측이 가능함을 의미하였다.

마지막으로, 딥러닝을 통해 질소산화물 예측 시, 기존의 학습데이터와 예측데이터의 상태가 동일해야 한다는 제약을 극복하기 위한 연구를 진행하였다. 이를 위해 정상 상태에서의 실험 데이터 세트를 설계하여 모델 학습에 사용함으로써 과도 상태에서의 질소산화물을 예측하였다. 과도 상태의 엔진 거동을 분석하여, 흡기 공기량, 흡기 압력, 분사 압력 및 주 분사 타이밍을 스윙 변수로 설정하여 각 운전점 별로 일정 범위 내에서 실험하여 데이터를 취득하였다. 이를 통해 정상 상태 데이터를 과도 상태 예측으로 확장할 수 있었다. 또한 과도 조건에서 흡기 온도와 냉각수 온도의 영향을 고려하기 위하여 온도 실험을 추가로 설계하고 수행하여, 온도가 다른 조건에서의 과도 상태 질소산화물을 예측하였다. 이러한 과정을 통해 과도 상태 예측을 위한 정상 상태 실험 조건 설계에 대한 방법을 제시하였고, 그 결과 역시 확인할 수 있었다. 이러한 방법으로 딥러닝 모델의 도메인 제한을 극복함으로써, 질소산화물 예측

모델을 엔진 설계, 검증, 예측 등의 개발 단계 전반과 과도 데이터를 측정할 수 있는 실제 엔진이 없는 경우에도 사용할 수 있게 되었다.

또한 본 연구에서 제시된 정상 상태 실험 계획 방법론은 향후 실도로 조건에서의 질소산화물 예측에 적용할 수 있을 것이며, 다른 기계 시스템의 예측에도 활용할 수 있다.

주요어: 딥러닝, 질소산화물, 과도상태 예측, 디젤 엔진, 하이퍼파라미터 최적화, 정상상태 실험설계 방법론

학번 : 2016-31756