



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

工學博士 學位論文

전기추진 선박의 소비 전력
예측 방법 비교 연구

A Comparison Study on Power Consumption Predicting of
the Electric Propulsion Vessel

The seal of Korea Maritime & Ocean University is a circular emblem. It features a central stylized figure resembling a person or a structure, with wavy lines below representing water. The text 'KOREA MARITIME & OCEAN UNIVERSITY' is written in an arc at the top, and '1945' is at the bottom. The Korean text '한국해양대학교' is also present in a circular arrangement.

指導教授 吳 珍 錫

2020 年 8 月

韓國海洋大學校 大學院

기관공학과

金 支 潤

本 論文을 金志潤의 工學博士 學位論文으로 認准함

委員長 柳 熙 漢



委 員 蘇 明 玉



委 員 吳 珍 錫



委 員 吳 世 駿



委 員 郭 俊 浩



2020 年 07 月 20 日

韓國海洋大學校 大 學 院

목 차

Nomenclature	iii
List of Tables	v
List of Figures	vii
Abstract	x
1. 서 론	
1.1 연구 배경	1
1.2 연구 동향	3
1.3 연구 내용 및 구성	6
2. 전기추진 선박 데이터 처리 기법	
2.1 선박 운항 데이터	9
2.1.1 선박제원	9
2.1.2 수집 데이터 개요	10
2.1.3 데이터 변환	12
2.1.4 선박 운항 모드 분석	14
2.2 데이터 처리	16
2.2.1 데이터 분석	16
2.2.2 데이터 산출 및 보완	22
2.2.3 데이터 변환	23
3. 전기추진 선박 부하 예측 모델 설계 및 구현	
3.1 이론적 배경 및 실험 절차	27
3.1.1 모델의 이론적 배경	27
3.1.2 예측 모델의 실험 절차	36
3.2 예측 모델 설계 및 구현	40
3.2.1 LSTM	41
3.2.2 bidirectional LSTM	42
3.2.3 CNN-LSTM (direct)	43
3.2.4 CNN-bidirectional LSTM (direct)	45

3.2.5 CNN-LSTM (parallel)	47
3.2.6 CNN-bidirectional LSTM (parallel)	49
3.2.7 LSTM auto encoder	51
3.2.8 ANN	53
3.2.9 DNN	54
3.2.10 CNN	55
4. 제안 모델 평가	
4.1 모델 평가 기준	58
4.2 실험 환경	59
4.3 실험 결과	60
4.2.1 LSTM	60
4.2.2 bidirectional LSTM	62
4.2.3 CNN-LSTM (direct)	64
4.2.4 CNN-bidirectional LSTM (direct)	66
4.2.5 CNN-LSTM (parallel)	68
4.2.6 CNN-bidirectional LSTM (parallel)	70
4.2.7 LSTM auto encoder	72
4.2.8 ANN	74
4.2.9 DNN	76
4.2.10 CNN	78
5. 모델 비교 분석 및 고찰	
5.1 모델 비교 분석	80
5.2 연구의 고찰	86
6. 결론	87
감사의 글	89
참고문헌	90

Nomenclatures

P_{EPS}	: 전기추진 선박의 추진부하 [kW]
L_{MPS}	: 기계식 추진 선박의 추진부하 [kW]
η_g	: 발전기 효율
η_{MS}	: 주배전반 효율
η_T	: 변압기 효율
η_{FC}	: 주파수 변환기 효율
η_{EPM}	: 전기추진 선박의 추진 전동기 효율
X_t	: 현재 시간(t) 에서의 입력
h_t	: C_t 레이어의 출력
h_{t-1}	: 이전 시간($t-1$)에서 히든 레이어의 결과 값
Y_t	: 현재 시간 (t)에서의 출력
\hat{C}_t	: 새로운 후보값 벡터
C_t	: 새로운 셀 상태
C_{t-1}	: 이전 셀 상태
F_t	: 망각여부를 결정하기 위한 상수 변수
* \hat{C}_t	: 이전 셀 상태와 망각여부를 결정하기 위한 상수 변수의 곱
W_f	: bidirectional LSTM 입력, 은닉층과 연결되는 가중치
W_i	: bidirectional LSTM 입력, 은닉층과 연결되는 가중치
W_o	: bidirectional LSTM 입력, 은닉층과 연결되는 가중치
W_c	: bidirectional LSTM 입력, 은닉층과 연결되는 가중치
U_f	: bidirectional LSTM 게이트, 출력 셀과 연결되는 가중치
U_i	: bidirectional LSTM 게이트, 출력 셀과 연결되는 가중치
U_o	: bidirectional LSTM 게이트, 출력 셀과 연결되는 가중치
U_c	: bidirectional LSTM 게이트, 출력 셀과 연결되는 가중치
b_f	: bidirectional LSTM 바이어스 벡터
b_i	: bidirectional LSTM 바이어스 벡터
b_o	: bidirectional LSTM 바이어스 벡터
b_c	: bidirectional LSTM 바이어스 벡터
σ_g	: bidirectional LSTM 활성화 함수

t : 각 시간의 변화
 y_i : i 번째의 실제 값
 \hat{y}_i : i 번째의 예측 값
 σ : 시그모이드 함수



List of Tables

Table 2.1	Specification of target ship	9
Table 2.2	List of acquisition data	10
Table 2.3	List of selected data	16
Table 2.4	Missing variables from data	20
Table 3.1	Code of time series data	38
Table 3.2	Code of RMSE	39
Table 3.3	Code of training function	39
Table 3.4	Summary of model structure	40
Table 3.5	Code of LSTM model	41
Table 3.6	Code of bidirectional LSTM model	43
Table 3.7	Code of CNN-LSTM (direct) model	44
Table 3.8	Code of CNN-bidirectional LSTM (direct) model	46
Table 3.9	Code of CNN-LSTM (parallel) model	48
Table 3.10	Code of CNN-bidirectional LSTM (parallel) model	50
Table 3.11	Code of LSTM auto encoder model	52
Table 3.12	Code of ANN model	53
Table 3.13	Code of DNN model	55
Table 3.14	Code of CNN model	56
Table 4.1	Environment information	59
Table 4.2	RMSE result of LSTM model experiment	60
Table 4.3	RMSE' s statistical information of LSTM model experiment result	60
Table 4.4	RMSE result of bidirectional LSTM model experiment	62
Table 4.5	RMSE' s statistical information of bidirectional LSTM model experiment results	62

Table 4.6 RMSE result of CNN-LSTM (direct) model experiment	64
Table 4.7 RMSE' s statistical information of CNN-LSTM (direct) model experiment results	64
Table 4.8 RMSE result of CNN-bidirectional LSTM (direct) model experiment	66
Table 4.9 RMSE' s statistical information of CNN-bidirectional LSTM (direct) model experiment results	66
Table 4.10 RMSE result of CNN-LSTM (parallel) model experiment	68
Table 4.11 RMSE' s statistical information of CNN-LSTM (parallel) model experiment results	68
Table 4.12 RMSE result of CNN-LSTM (Parallel) model experiment	70
Table 4.13 RMSE' s statistical information of CNN-LSTM (parallel) model experiment results	70
Table 4.14 RMSE result of LSTM auto encoder model experiment	72
Table 4.15 RMSE' s statistical information of LSTM Auto encoder model experiment results	72
Table 4.16 RMSE result of ANN model experiment	74
Table 4.17 RMSE' s statistical information of ANN model experiment results	74
Table 4.18 RMSE result of DNN model experiment	76
Table 4.19 RMSE' s statistical information of DNN model experiment results	76
Table 4.20 RMSE result of CNN model experiment	78
Table 4.21 RMSE' s statistical information of CNN model experiment results	78
Table 5.1 RMSE' s statistical information of experiment results	80

List of Figures

Fig. 1.1 IMO emission control	1
Fig. 1.2 Concept image of electrical propulsion smart vessel	2
Fig. 1.3 Hybrid ferry M/V Prinsesse Benedikte	4
Fig. 1.4 Contents of research	7
Fig. 2.1 Energy conversion efficiency of electric propulsion system	12
Fig. 2.2 Mechanical propulsion ship M/E power meter shaft power	13
Fig. 2.3 Mechanical propulsion ship general load consumption	14
Fig. 2.4 Electric propulsion ship load consumption	14
Fig. 2.5 Container vessel's load consumption pattern in voyage	15
Fig. 2.6 Load consumption in arrival & departure stand by	15
Fig. 2.7 Distribution of environment data	18
Fig. 2.8 Distribution of ship related data (1)	19
Fig. 2.9 Distribution of ship related data (2)	20
Fig. 2.10 Matrix for visualization of missing values	21
Fig. 2.11 Visualization of missing values imputation result	23
Fig. 2.12 Error data of propulsion load data	24
Fig. 2.13 Propulsion load data after conversion	24
Fig. 2.14 Propulsion load data after delete false data	25
Fig. 2.15 Propulsion load data after conversion	25
Fig. 2.16 Error data of water depth data	26
Fig. 2.17 Water depth data after conversion	26
Fig. 3.1 Type of RNN structures according to input and output	27
Fig. 3.2 Structure of RNN	28
Fig. 3.3 Long term dependency problems caused by RNN structure	29

Fig. 3.4	Structure of LSTM block cell	30
Fig. 3.5	Bidirectional LSTM structure	32
Fig. 3.6	Auto encoder structure	34
Fig. 3.7	CNN structure	35
Fig. 3.8	Separate data with train and test data set	36
Fig. 3.9	Training procedure	37
Fig. 3.10	The feature of time step	38
Fig. 3.11	Construction of LSTM	41
Fig. 3.12	Construction of bidirectional LSTM	42
Fig. 3.13	Construction of CNN-LSTM (direct)	44
Fig. 3.14	Construction of CNN-bidirectional LSTM (direct)	46
Fig. 3.15	Construction of CNN-LSTM (parallel)	48
Fig. 3.16	Construction of CNN-bidirectional LSTM (parallel)	50
Fig. 3.17	Construction of LSTM auto encoder	51
Fig. 3.18	Construction of ANN	53
Fig. 3.19	Construction of DNN	54
Fig. 3.20	Construction of CNN	56
Fig. 4.1	Result of LSTM model experiment	61
Fig. 4.2	Result of bidirectional LSTM model experiment	63
Fig. 4.3	Result of CNN-LSTM (direct) model experiment	65
Fig. 4.4	Result of CNN-bidirectional LSTM (direct) model experiment	67
Fig. 4.5	Result of CNN-LSTM (parallel) model experiment	69
Fig. 4.6	Result of CNN-bidirectional LSTM (parallel) model experiment	71
Fig. 4.7	Result of LSTM auto encoder model experiment	73
Fig. 4.8	Result of ANN model experiment	75
Fig. 4.9	Result of DNN model experiment	77
Fig. 4.10	Result of CNN model experiment	79
Fig. 5.1	Comparition of model's RMSE value	81

Fig. 5.2 Comparison of CNN-LSTM combine model's RMSE value 82
Fig. 5.3 Comparison of model's RMSE using box plot 83
Fig. 5.4 Comparison of model's RMSE using box plot 85



A Comparison Study on Power Consumption Predicting of the Electric Propulsion Vessel

Kim, JiYoon

Department of Marine Engineering

Graduate School of Korea Maritime and Ocean University

Abstract

In recent years, concern for the energy efficiency of vessels has increased due to the influence of environmental pollution. Electric propulsion systems have more energy flexibility than mechanical propulsion systems. So, it can improve power efficiency by using batteries. For the utilization of batteries, it is necessary to predict the power consumption of electric propulsion vessels.

In addition, researchers are studying multivariate time-series data for prediction. In contrast preceded studies were non applicability to electric propulsion systems for power consumption predicting. Because, limitation of accessible to vessel's data and the previous prediction researches were considerably studied to small ranged electrical data. According to these reasons, The research of models that capable a wide range of vessel load data is essential.

In this paper, aims to predict the electricity consumption of a vessel, using real vessel data and convert to electric propulsion vessel data, and select variables that affecting vessel's electricity consumption using heuristic. The converted data includes missing values, this can cause of weakens model's accuracy, therefore multiple imputation algorithm was used for cover it. After data preprocessing, several models are created to predict time-series data. This consists of single models for comparison criteria : LSTM(Long

Short-term Memory models), CNN(Convolutional Neural Network), ANN(Artificial Neural Network), DNN(Deep Neural Network), bidirectional LSTM, and conjunction models : CNN-LSTM (direct), CNN-bidirectional LSTM (direct), CNN-LSTM (parallel), CNN-bidirectional LSTM (parallel).

After models creation, the experiment method was decided, considered by clear comparison. that was composed of repeat test for the model's performance validation and utilized the widely used accuracy metric : RMSE.

KEY WORDS: Smart Vessel, Smart Vessel, Energy Management, Predict of Power, RNN, LSTM, Bidirectional LSTM



제 1 장 서 론

1.1 연구 배경

해양환경 오염 방지를 위해 시행된 IMO(International Maritime Organization) 2020은 선박의 배출가스 규제를 강화하였다[1]. 특히 ECA(Emission Control Area)에서는 0.1%의 저유황유 사용을 강제하는 등, 배출 가스 기준에 대한 엄격한 제한을 요구한다. 운항 중인 선박에서 배출가스 규제를 만족하기 위해서는 스크러버의 설치, 선택적 촉매 환원 장치(SCR, Selective Catalytic Reduction) 및 저-유황유(low sulfur fuel oil)를 사용해야 한다. 그러나 스크러버 및 저-유황유의 사용은 배출가스의 규제를 만족하지만, 높은 가격으로 인해 선주 측의 부담으로 돌아온다. 그러므로 기존의 선박을 대체할 수 있는 친환경 선박에 대한 연구가 진행되고 있으며, 대표적으로 전기추진 선박에 대한 연구가 수행되고 있다. 기존의 선박은 주기관을 이용하여 기계적으로 추진 동력을 발생시키는 반면, 전기추진 선박은 전동기를 이용하여 추진 동력을 발생시키므로 높은 용량의 발전기가 필수적이다.

전력 에너지는 기계식 에너지에 비하여 에너지의 저장 및 변환이 용이하므로 전기추진 선박은 전력 에너지 관리 알고리즘을 활용하여 에너지 효율을 높일 수 있는 장점을 보유하고 있다. Fig 1.1은 배출가스 통제 지역을 나타내고 있다.

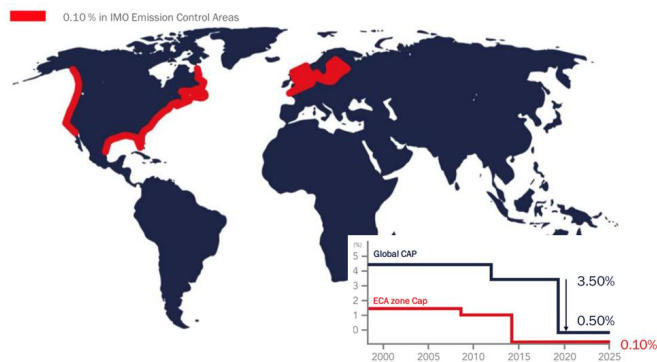


Fig. 1.1 IMO emission control

또한 전기추진 선박은 커패시터 혹은 배터리를 에너지 저장장치로 활용할 수 있으며 일반적으로 상선에서는 배터리를 주로 사용한다.

전기추진 선박의 연구와 병행하여 고효율 선박인 전기추진 스마트 선박에 대한 연구도 함께 진행 중에 있다. 전기추진 스마트 선박은 전기추진 선박에 4차 산업혁명 요소 기술인 ICT(Information & Communication Technology) 기술과 IoT(Internet of Things)·센서 기술 그리고 인공지능 기술을 활용하는 특징을 가지고 있다. 또한 최근 전기추진 선박에 장착되는 배터리의 효과적인 충·방전을 제어하는 부하 제어 시스템 (LCS, Load Control System) 알고리즘과 같이 에너지 효율성을 높이기 위한 연구 등이 중점적으로 수행되고 있다. Fig. 1.2는 전기추진 스마트 선박의 컨셉을 나타내었다.



Fig. 1.2 Concept image of electrical propulsion smart vessel

최근 기술 발전에 따라 전기추진 선박의 전력부하 예측 기술이 요구되나 이와 연관된 연구는 미흡한 상태이다. 특히 기존에는 기계식 추진 선박을 기반으로 정확도 높은 전력부하 예측모델의 연구가 수행되었다. 그러나 전기추진 선박 부하에 대한 예측 연구는 전기추진 선박과 연관된 다양한 요소를 포함하지 않았으며, 선박부하의 높은 변동성을 반영할 수 없는 단점이 존재하였다[2].

그러므로 전기추진 선박 연구에 활용할 수 있는 정확도 높은 “전기추진 선박 전력부하 예측 모델”에 대한 연구가 필요하다.

1.2 연구 동향

환경규제의 강화 추세와 노후선박의 교체 필요성과 연계되어, 유럽과 북미를 중심으로 전기추진 선박의 상용화를 위한 연구가 진행되고 있다[3-6].

- EU(European Union)는 CO_2 배출이 없는 E-Ferry Project를 추진하고 있으며, 최대 4MW의 충전 전력 용량을 사용하는 특징이 있다. 그러나 E-Ferry의 전력부하 데이터는 대형 전기추진 선박의 연구에는 활용하기 어려운 문제점이 있다.
- 노르웨이는 친환경 선박제조를 위하여 2030년까지 CO_2 배출 저감을 위한 the green shipping programme을 운영하고 있다. 위 프로그램 중 barrier study는 배터리와 배터리 충전 기술에 20억(NOK) 이상의 추가 투자 중에 있다. 또한 리튬이온 배터리로만 구동되는 최초의 전기추진 여객선 암페어호를 도입하였다. 그러나 5.6km의 짧은 운항거리를 가지고 있는 단점을 보유하고 있다.
- 스웨덴의 그린 시티 파리사는 슈퍼 충전지로 움직이는 모비츠 여객선을 건조하여 운항하고 있으며, 10분간 충전 이후 한시간 동안 9노트로 운항할 수 있는 특징이 있다.
- 덴마크는 2003년 하이브리드 디젤 전기추진 여객선 프린세스 베네딕트호를 운항하고 있으며, 디젤 선박이 하이브리드 선박으로 개조된 가장 큰 선박으로 1,140명의 승객과 364대의 차량을 운반할 수 있다. Fig. 1.3은 해당 선박을 나타내고 있다.
- 대한민국은 2010년부터 연구선 이사부호를 시작으로 2015년 연료전지 선박의 연구를 수행하였으며, 2019년 축 발전기 국산화에 성공하여 배출가스 규제에 대응을 위한 연구 및 상용화에 투자를 진행 중이다. 그러나 관련 연구 경험 부족으로 연구에 활용할 수 있는 전력부하 데이터 등의 부족으로 핵심기술을 해외에 의존하는 특징이 있다.

위와 같이 현재 전기추진 선박에 관한 연구는 소형 여객선과 레저보트를

중심으로 연구되는 특징을 보유하고 있다. 그러나 400MW 이상의 전력부하를 이용하는 대형 전기추진 선박의 연구에는 기존의 소형 전기추진 선박의 데이터와 전력부하 제어 알고리즘의 사용이 어려운 문제가 존재한다.


Scandlines		M/V Prinsesse Benedikte and M/V Prins Richard	
			
MAIN PARTICULARS		CARGO CAPACITIES	
Length over all	142.00 m	Cars	364 pcu
Breadth over all	25.40 m	Trucks	580 m
Depth moulded to weather deck	20.60 m	Train track	118 m / 2 tracks
Depth moulded to main deck	8.50 m		
Design draft	5.80 m		
Dead weight	2,247 t		
Lightweight	6,592 t	OVERSIZED CARGO	
Gross tonnage	14,822	(l x h x w)	25 m x 4.7 m x 4.5 m
Net tonnage	4,446		
Service speed	18.5 kn	DECK LOAD	
		Deck 2 max. load trucks	1.4 t/m ²
		(2 axles, axle distance 1.50m)	16 t/axle
		(3 axles, axle distance 1.35m)	13 t/axle
		Deck 4 max. load cars	0.3 t/m ²
		(each axle 2.0 t; load distance 2,5 m)	
CLASS		DIMENSIONS CAR DECK	
LR	100 A1 Double-ended RoRo ferry for passengers, vehicles and train for service between PUTTGÅRDEN & RØDBY, Ice Class 1C, LMC, UMS	Free height train deck	4.80 m
		Free height car deck	3.20 m
TANK CAPACITIES		LIFESAVING EQUIPMENT	
MDO & HFO tanks	144.3 m ³	Life boats semi-enclosed	2 x 150 pers.
Fresh water tanks	89.7 m ³	Rescue boat	1 x 6 pers.
Ballast tanks	725.6 m ³	Fast rescue boat	1 x 10 pers.
Heeling tanks	508.6 m ³	MES	2 x 650 pers.
Waste tanks	341.0 m ³	Life rafts	12 x 100 pers.
Closed-loop scrubber circ. tank	11.6 m ³		
PASSENGER CAPACITY		ENGINE PLANT	
Max. number of passengers	1,140 pers.	Prinsesse Benedikte:	
		GenSet MAN 6L32/44CR	1 x 3,600 kW
		GenSet MAK 8M32	2 x 3,520 kW
		GenSet MAN 8L32/44CR	1 x 4,800 kW
		Prins Richard:	
		GenSet MAN 9M32C	1 x 4,500 kW
		GenSet MAK 8M32	3 x 3,520 kW
		ESS capacity*	1,600 kWh
		ESS output*	3,500 kW
		*) ESS: Energy Storage System	
		PROPULSION	
		AquaMaster CRP	4 x 3,000 kW
		EMERGENCY GENERATOR	
		GenSet MAN D2840LE	1 x 357 kW
		Shipyard: Ørskov Staalskibsværft, Frederikshavn, Denmark	
		Port of Registry: Rødbyhavn, Denmark	
		Hull no.: 193 & 194	
		February 2018	

Fig. 1.3 Hybrid ferry M/V prinsesse benedikte

최근 전력 부하 예측과 관련된 다양한 모델의 연구가 수행 중에 있다. 그러나 전기추진 선박과 관련된 연구는 제한된 환경요소가 반영된 화이트박스 기반의 모델링을 연구하였으며, AI를 활용한 모델링 연구에서는 전력 소비량 예측에 관한 연구가 진행되었다. 그러나 부하 변동성이 심하며, 선박과 관련된 다양한 특징이 고려된 연구는 미흡한 상황이다. 다음은 최근 연구 중인 전력 부하 예측과 관련된 연구를 나타낸다.

- (Kim, et al., 2019) 연구는 주거 에너지 장-단기 전력 소비량예측에 관한 연구를 진행하였으며, CRBMs, FCRBM, Seq2Seq 모델과 CNN-LSTM 모델을 비교 분석하였다. 전력 소비량의 변동성이 큰 데이터를 예측할 수 있는 장점을 가지고 있으나 추종하는 전력 소비량의 범위가 좁은 데이터를 활용하여, 본 연구에서 활용할 수 없는 단점이 존재한다[7].
- (Cao, et al., 2018)은 정수장 운영데이터의 관로, 수질, 사고 등 의 정보를 기반으로 정수시스템 안정성 확보를 위한 단기예측 및 장기예측 모델 연구를 진행하였다. 특히 예측모델에 다중 선형회귀를 시계열 데이터 예측 모델에 적용하여 예측 정확도를 높인 장점을 보유하고 있으나 다중 선형회귀가 불가능한 전기추진 선박의 전력부하에 해당 모델을 적용할 수 없는 단점을 지니고 있다[8].
- (Jung, et al., 2020)은 UMTC(UMass Trace Respositor)에서 제공하는 가정 단위 에너지 소비량 데이터를 2,976개의 시계열 데이터로 변환하여 전력 소비량 예측 연구를 수행하였다. 그러나 사용된 가정용 전력 데이터는 전기추진 선박의 데이터와 특성이 달라 전기추진 선박의 전력소비량 예측에 활용할 수 없다[9].
- (Lee, et al., 2019)에서 수행된 연구는 RNN을 활용한 시간단위 태양전지 전력 발전량에 대한 연구를 수행하였으며, ANN, ARIMA, S-ARIMA, DNN, LSTM 모델간의 비교를 통해 전력 소비량을 가장 잘 예측하는 모델을 제안 하였다. 해당 모델은 매우 높은 정확도로 태양전지 전력 발전량을 예측할 수 있는 장점을 가지고 있으나, 전력 부하의 변동성이 큰 전기추진 선박의 특성을 반영할 수 없다[10].
- (Kim, et al., 2020)에서는 XGBoost 알고리즘과 light GBM 알고리즘을 사용하여 실제 선박의 부하 예측을 진행하였다. 그러나 실제 선박의 중부하 예측 정확도가 낮은 문제점을 지니고 있으므로 본 연구에서 활용하는 전기추진 선박의 넓은 부하데이터의 예측을 수행할 수 없는 문제를 보유하고 있다[11].

- (Cui, et al., 2018)은 SBU-LSTM 신경망 모델을 제안하여, 복잡한 도로 교통을 분석하고 도로의 Loop Speed와 INRIX Speed 예측할 수 있는 연구를 진행하였다. 그러나 실제선박의 내·외부 환경을 고려하지 않는 특성을 지니고 있으므로 전기추진 선박 부하데이터 예측의 활용에 어려움이 있다[12].
- (Tang, et al., 2019)은 선박의 시계열 데이터를 활용하여, 선박의 항로 예측과 관련된 연구를 진행하였으며, EKF, BPNN과 LSTM 모델간의 예측 성능을 비교 분석하였다. 그러나 낮은 변동성과 짧은 데이터 길이로 인하여, 높은 변동성과 긴 데이터 길이를 보유한 전기추진 선박의 부하를 예측할 수 없는 단점이 있다[13].

1.3 연구 내용 및 구성

기존 전기추진 선박의 연구는 소형선박과 유람선을 대상으로 제한되어 있었으며, 상선 및 특수선 등 대형 전기추진 선박에 대한 연구는 미흡한 상황이다. 또한 전기추진 선박의 전력부하 예측 모델링 연구는 다양한 환경 변수를 활용하지 않았으며, 시계열 데이터 예측에 활용된 AI 모델링 연구는 낮은 변동성과 좁은 범위의 데이터를 예측하는 특징이 존재한다. 그러므로 실제 운항 중인 대형 선박의 데이터를 기반으로 다양한 환경변수를 고려하여, 전기추진 선박의 부하를 예측할 수 있는 모델이 필수적이다.

본 연구 목표는 장기간 운항한 실제 대형 선박에서 수집된 환경정보와 운항 정보를 포함한 데이터를 활용하여 정확도 높은 전기추진 선박 시계열 전력부하 예측 모델을 제안하는 것이다. 목표에 도달하기 위하여 다음과 같이 연구를 진행하였다.

- 수집된 선박 데이터를 분석하고 전기추진 선박 데이터로 변환한다.
- 모델의 정확도를 향상시키기 위하여, 수집된 선박 데이터 중 누락 데이터를 MICE 알고리즘을 활용하여 산출 및 보완한다.
- 전기추진 선박 시계열 데이터를 분석할 수 있는 모델을 구축하며, 모델

평가 기준을 수립한다.

- 구축된 모델을 활용하여, 전기추진 선박 시계열 데이터를 예측하고 모델별 학습 결과를 분석한다.
- 모델간 비교 분석을 진행하며, 고찰을 진행한다.

Fig 1.4는 본 연구의 진행 방법을 나타내었다.

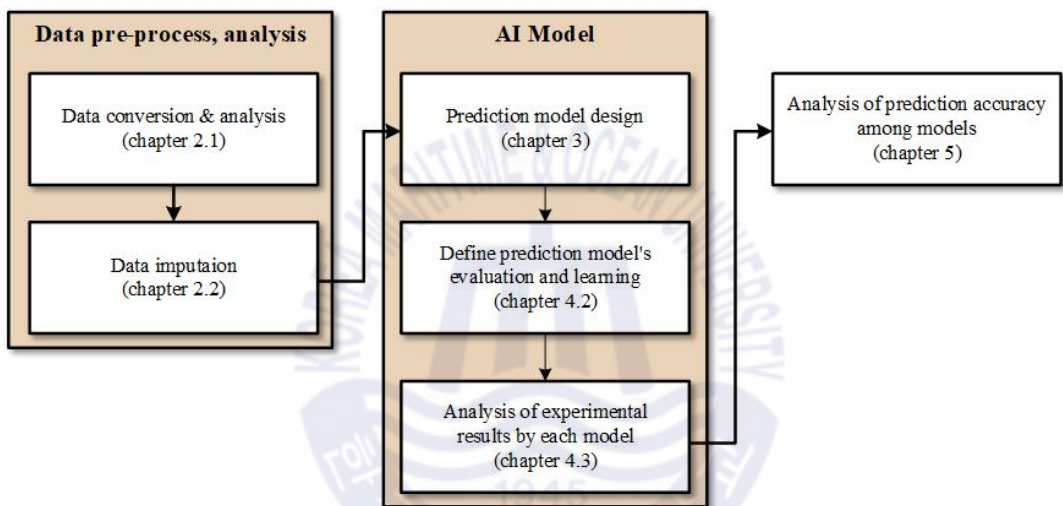


Fig. 1.4 Contents of research

제 1장은 연구 배경, 연구 목표, 방법, 범위 그리고 본 논문의 구성에 대하여 설명한다.

제 2장은 본 논문에 이용되는 전기추진 선박 데이터에 대한 설명과 더불어 모델 정확도를 향상시키기 위하여, 선박 데이터 수집 중 발생한 손실 데이터를 산출 후 입력을 진행한다.

제 3장은 전기추진 선박의 부하를 예측할 수 있는 모델 설계 및 구현에 대한 설명을 진행하며, 전기추진 선박 부하예측에 최적화된 모델을 찾기 위한 LSTM, bidirectional LSTM, CNN-LSTM (direct), CNN-bidirectional LSTM (direct), CNN-LSTM(parallel), CNN-bidirectional LSTM (parallel), ANN, DNN, CNN이

제시된다.

제 4장은 모델 평가기준, 실험환경을 설명하며, 3장에서 제안한 다양한 전기추진 선박 부하예측 모델의 실험 결과를 분석한다.

제 5장은 4장의 실험 결과를 토대로 모델간 성능 비교 분석을 진행하며, 고찰을 진행한다.

제 6장은 본 논문의 분석 결과와 연구의 의의 및 향후 과제에 관한 내용을 기술한다.



제 2 장 전기 추진 선박 데이터 처리 기법

본 장에서는 수집된 선박 운항 데이터를 활용하여, 선박의 운항모드를 분석한다. 또한 변수별 데이터 특징을 분석하고, 운항 데이터 중 소실된 데이터를 산출하여, 향후 AI 모델의 정확도를 높일 수 있도록 데이터를 보완하였다.

2.1 선박 운항 데이터

2.1.1 선박제원

본 연구에서 사용된 데이터 계측 대상 선박은 H사의 D호이며, 13,000 TEU(Twenty-foot Equivalent Unit)급 컨테이너 선박으로 주기관은 대형 2행정 디젤 엔진 1대, 발전용 보조기관은 3,480kW급 4행정 디젤 엔진 4대로 구성되어 있다. Table 2.1은 대상 선박의 제원이다.

Table 2.1 Specification of target ship

Type of vessel	Container Ship
Length	365m
Width	48m
Draft	10.8m
Engine power	79.106 BHP
Capacity of generator	3,480 kW * 4
Speed	23.0 knot
TEU	13,154 TEU

2.1.2 수집 데이터 개요

수집된 데이터는 10분 단위로 데이터를 계측하고 저장되었으며, 시간정보, 위치정보, 외부 환경정보, 선박 운항 정보로 구성된다. 시간정보는 협정세계시, 현지시간, 한국표준시로 구성되어 있으며, 위치정보는 위도, 북반구/남반구 구분, 경도로 구성되어 있다. 외부 환경 정보는 유속, 풍향, 풍속 정보를 마지막으로 선박 운항 정보는 선박의 선수 각도, 선박 방향키의 각도, 1~4번 발전기 관련 정보 (발전기 부하, 발전기 부하율, 발전기 전류 등), 주요 장비별 디젤유 사용 유무 (주기관, 발전기, 보일러), 선속, 주기관의 토크, 축 속도, 축 출력, 흘수정보 (선미, 선수, 좌현, 우현)로 구성되어 있다. Table 2.2는 수집된 데이터를 나타내었다.

Table 2.2 List of acquisition data

Name	Description	Unit
UTC	Coordinated Universal Time	
LT	Local Time	
KST	Korea Standard Time	
Latitude	Latitude	ϕ
N/S	Northern hemisphere/ Southern hemisphere	°
Longitude	Longitude	
Heading	Heading degree of the vessel	Degree (0~360) °
Rudder angle	Rudder angle degree of the vessel	Degree (-35~35) °
Water depth	Water depth	<i>m</i>
Water speed	Water speed	<i>m/s</i>
Wind angle	Wind angle	Degree (0~360) °

Wind speed	Wind speed	<i>m/s</i>
NO.1~4 gen. load	Generator load	<i>kW</i>
NO.1~4 gen. percent load	Generator load ratio	%
NO.1~4 gen. current	Generator current output	<i>A</i>
NO.1~4 gen. percent current	Generator current output ratio	%
NO.1~4 gen. voltage	Generator voltage output	<i>V</i>
NO.1~4 gen. frequency	Generator frequency	<i>Hz</i>
NO.1~4 gen. bus voltage	Generator bus voltage output	<i>V</i>
NO.1~4 gen. bus frequency	Generator bus frequency	<i>Hz</i>
M/E D.O TF	Main Engine Diesel Oil use or not (True/False)	[0,1]
G/E D.O TF	Generator Engine Diesel Oil use or not (True/False)	[0,1]
Boiler D.O TF	Boiler Diesel Oil use or not (True/False)	[0,1]
Vessel speed	Vessel speed	<i>Knot</i>
M/E torque	Main Engine torque	<i>Nm</i>
M/E shaft speed	Main Engine shaft speed	<i>rpm</i>
M/E shaft power	Main Engine shaft power output	<i>HP</i>
Draft(after)	After draft	<i>m</i>
Draft(forward)	Forward draft	<i>m</i>
Draft(port)	Port draft	<i>m</i>
Draft(starboard)	Starboard draft	<i>m</i>

2.1.3 데이터 변환

전기추진 선박의 전력 부하는 추진 전력 부하와 선내 전력 부하로 구성된다. 본 논문에서 활용할 전기추진 선박의 전력부하를 계산하기 위하여 추진 부하 데이터 변환 과정과 전기추진 선박 부하 데이터 합산 과정을 수행하여 전기추진 선박의 전력부하를 확보한다. 추진 부하 데이터 변환 과정은 기계식 추진 선박의 데이터 중 주기관 축 출력을 활용하여 전기추진 선박의 추진 전력 부하를 산출하는 과정을 수행한다. 추진 부하 데이터 변환 과정은 수집된 기계식 추진 선박의 데이터 중 발전기들의 발전 부하를 합산하여 전기추진 선박의 선내 전력 부하로 활용한다.

(1) 추진 부하 데이터 변환

기계식 추진 선박은 주기관의 회전에너지를 추진기로 직접적으로 전달하는 반면, 전기추진 선박은 발전기에서 생산된 전기에너지를 주배전반, 변압기, 주파수변환기, 전동기를 거쳐 추진기로 전달하는 과정을 거치며 전력 손실이 발생한다. Fig. 2.1은 선박의 각 기기 별 전력 변환 효율을 나타낸다[14-16].

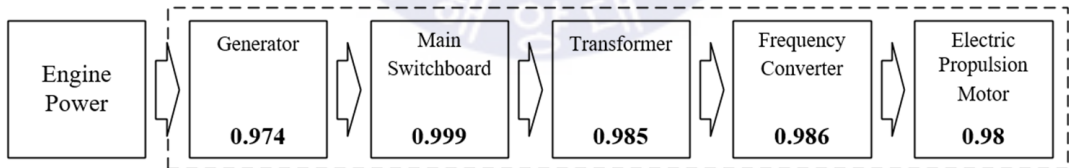


Fig. 2.1 Energy conversion efficiency of electric propulsion system

또한 기기별 효율을 활용하면 다음 식 (2.1)과 같이 기계식 추진 선박의 축 부하를 이용하여 전기추진 선박에서의 추진 전력 부하를 구할 수 있다.

$$P_{EPS} = L_{MPS} \div \eta_g \div \eta_{MS} \div \eta_T \div \eta_{FC} \div \eta_{EPM} \quad (2.1)$$

여기에서 P_{EPS} 전기추진 선박의 추진부하 (Load of electric propulsion ship [kW]), L_{MPS} 는 기계식 추진 선박의 주기관 축 출력(Load of mechanical propulsion ship), η_g 는 발전기 효율, η_{MS} 는 주배전반 (Main switch board) 효율, η_{EPM} 전동기(Electric propulsion motor) 효율을 각각 나타낸다.

(2) 전기추진 선박 부하 데이터 합산

전기추진 선박 부하데이터는 추진 전력 부하와 선내 전력부하로 구성된다. 그러므로 주기관의 축 마력 데이터를 식 (2.1)을 활용하여 전기추진 선박의 추진 전력 부하와 수집된 기계식 추진 선박의 데이터 중 4대의 발전기 부하 데이터를 합산하여 전기추진 선박의 선내 전력부하로 계산한다. 마지막으로 계산된 추진 전력 부하와 선내 전력부하를 합산하여 전기추진 선박 부하 데이터를 확보한다.

Fig 2.2는 기계식 추진 선박의 주기관 축 마력 데이터를 나타내며, Fig 2.3은 기계식 추진 선박의 선내 전력 부하를 나타낸다. 그리고 Fig 2.4는 합산된 전기추진 선박의 전체 부하 데이터이다.

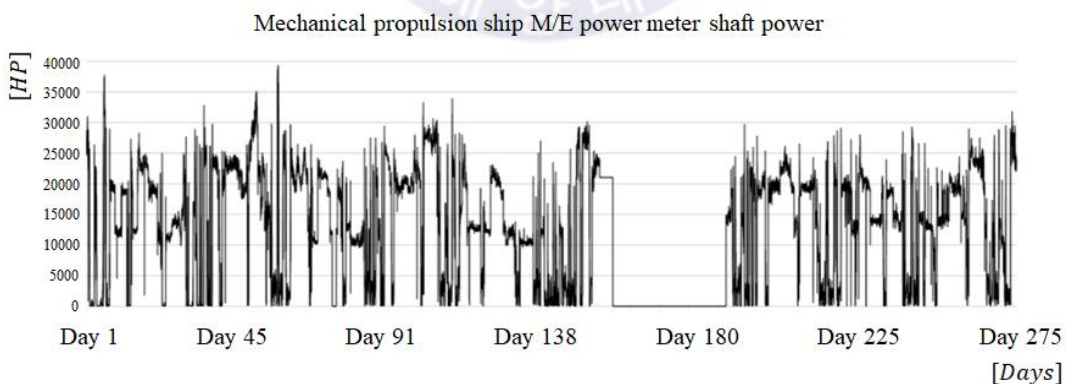


Fig. 2.2 Mechanical propulsion ship M/E power meter shaft power

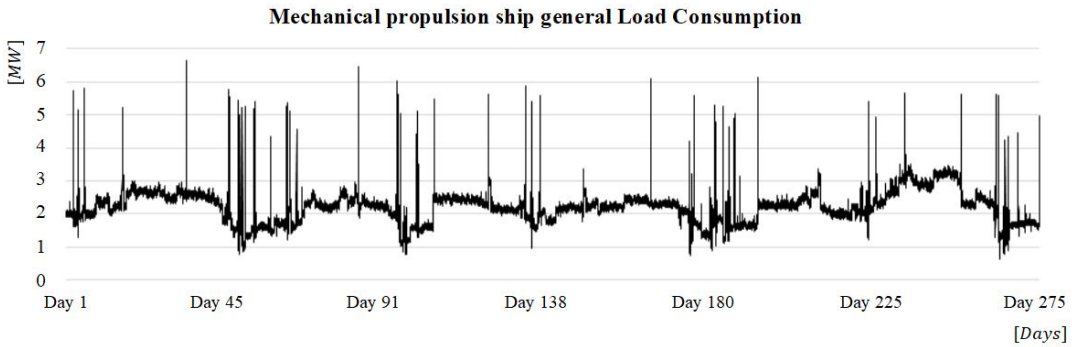


Fig. 2.3 Mechanical propulsion ship general load consumption

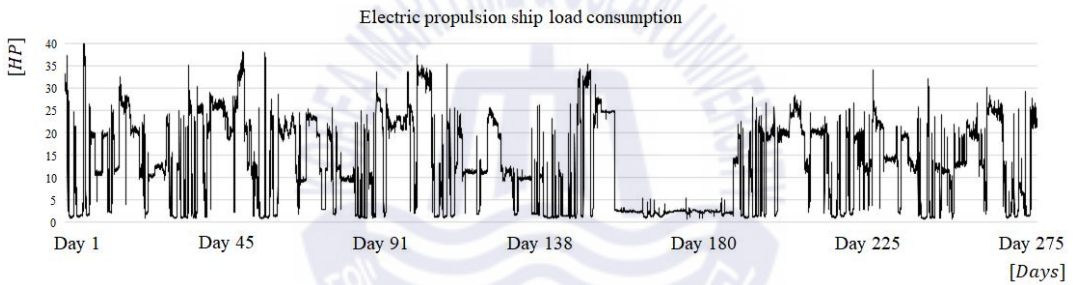


Fig. 2.4 Electric propulsion ship load consumption

2.1.4 선박 운항 모드 분석

선박은 크게 4가지의 상황에 따라서 모드로 정의하였다. 각 모드는 항해 모드(A/S : At Sea), 입항 모드(Arr' S/By : Arrival and Stand By), 정박 모드(in port), 출항 모드(Dep' S/By : Departure and Stand By)이다. 항해 모드에서는 선내의 전력 부하 변동 폭이 좁다. 입항 모드는 선박의 속도 변동 및 기타 보조 기기들의 사용으로 인해 총 소비 전력이 크게 변동한다. 정박 모드는 선박의 속력이 없으며, 기타 보기들의 사용이 타 모드에 비하여 적다. 그러므로 총 소비 전력이 낮고, 전력 변동 폭이 좁다. 출항 모드의 경우 총 소비 전력의 변동 폭이 가장 크다. Fig 2.5는 기존 컨테이너 선박을 기준으로 추진부하

데이터를 변환 과정과 선박부하 데이터 합산과정의 결과로 나온 가상 컨테이너 전기추진 선박의 모드 별 전력 패턴을 보여준다.

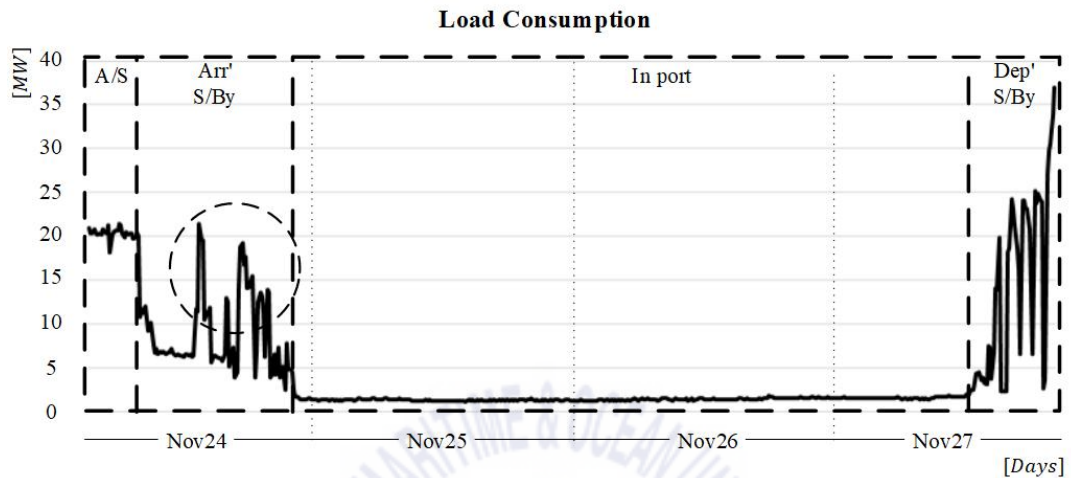


Fig. 2.5 Container vessel's load consumption pattern in voyage

전기추진 선박은 추진 부하에서 가장 많은 전력 소비가 발생한다. 그러므로 전기추진 선박의 전력 소비 전력은 추진 전력의 패턴에 가장 많은 영향을 받는 특징이 존재한다. Fig. 2.6은 입항 모드와 출항 모드에서 추진 동력과 선박내 총 전력을 나타낸다. 그래프를 보면 알 수 있듯이, 선박의 입·출항시 추진 동력의 패턴과 총 전력의 사용 패턴이 유사함을 알 수 있다.

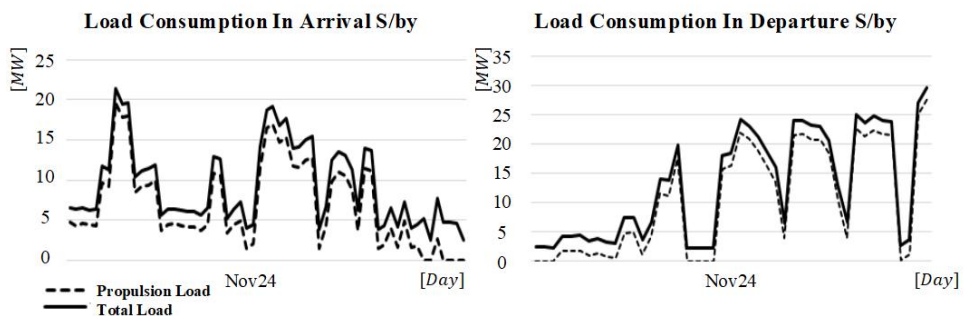


Fig. 2.6 Load consumption in arrival & departure stand by

2.2 데이터 처리

전기추진 선박 전력 부하 예측 모델의 성능향상을 위하여 수집된 데이터를 분석하고 데이터 수집 중 발생한 오류 데이터의 산출 및 보완을 수행하는 과정은 필수적이다. 본 장에서는 수집된 데이터 중 전기추진 선박의 부하 예측에 직접적으로 영향을 줄 수 있는 변수를 경험적 방법론에 근거하여 작업을 활용하여 선정 하였다. 또한 선정된 데이터의 변수별 정규분포를 분석하였다. 그리고 수집된 데이터 중 누락 데이터의 경우 알고리즘을 활용하여 데이터를 산출하고 보완 하였으며 오류 데이터의 경우 데이터 변환 과정을 수행하여 정상 데이터로 변환하였다.

2.2.1 데이터 분석

(1) 데이터 선정

전기추진 선박에서 전체 소비 전력 중 추진 전력이 차지하는 비율은 약 90%이다. 그러므로 전체 소비 전력을 예측하기 위하여, 추진 전력의 예측이 필수적이다. 추진 전력은 선박의 속력, 선체 저항 등과 연관이 있다. 그러므로 전기추진 선박 추진전력과 연관 있는 데이터를 선정하였다. Table 2.3은 본 연구에서 활용하는 선정된 데이터의 종류이다[17-18].

Table 2.3 List of selected data

Name	Unit	Remark
Electric load	kW	800~40,000
Propulsion load	HP	0~65,536
Heading angle	Degree	0~360 °
Rudder angle	Degree	-35~35 °
Water depth	m	0~836

Water speed	<i>m/s</i>	-4~6
Wind angle	Degree	0~360 °
Wind speed	<i>m/s</i>	0~47
Vessel speed	<i>Knot</i>	0~25
M/E RPM	<i>rpm</i>	0~76.3
Draft(after)	<i>m</i>	0~15.4
Draft(forward)	<i>m</i>	0~18.8
Draft(port)	<i>m</i>	0~15.6
Draft(starboard)	<i>m</i>	16.38

선박의 저항을 직접적으로 나타내는 데이터는 수심, 해수의 속도, 풍향, 풍속, 흘수 데이터이며, 간접적으로는 선박의 방향, 방향키 각도, 선속, 주기관의 회전수 이다.

(2) 데이터 분석

- 데이터 분포

수집된 데이터의 변수별 중앙값과 분포를 분석하였으며, 확률 밀도 함수 그래프를 활용하였다. Fig 2.7 ~ Fig 2.9는 환경데이터와 선박 운항 연관 정보 데이터의 분포를 나타낸다. Fig. 2.7는 선박에 영향을 끼칠 수 있는 외부 환경 데이터들(수심, 유속, 풍향, 풍속) 이다. 장기간 측정된 데이터의 특성상 선박의 상황에 따라 외부 환경 데이터의 분포가 매우 다양함을 확인할 수 있다.

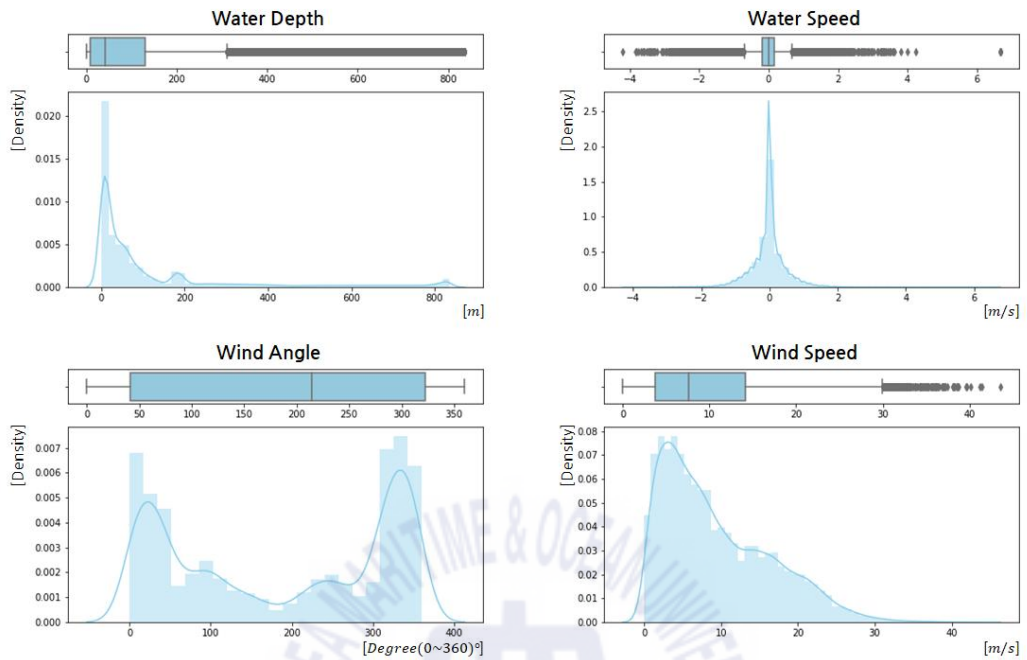


Fig. 2.7 Distribution of environment data

Fig 2.8은 선박 운항 연관 데이터로 전기추진 선박의 전력 부하는 분포가 매우 넓은 특징을 보이며, 추진 부하는 상대적으로 낮은 구간에서 많이 발생하는 것으로 확인되며 특히 추진 부하의 경우 60,000 이상의 오류 데이터가 수집되어 있음을 확인할 수 있다. 선수의 방향과 선박 방향키의 각도는 골고루 분포되어 있다. 선속은 정박 중인 데이터가 많이 파악되며, 10 ~ 20 노트 사이로 많이 운항되었음을 알 수 있다. 마지막으로 주기관의 회전수는 선속과 마찬가지로 정박 중인 데이터가 많이 파악되며, 40 ~ 75 RPM 사이에서 많이 사용되는 특징을 확인할 수 있다.

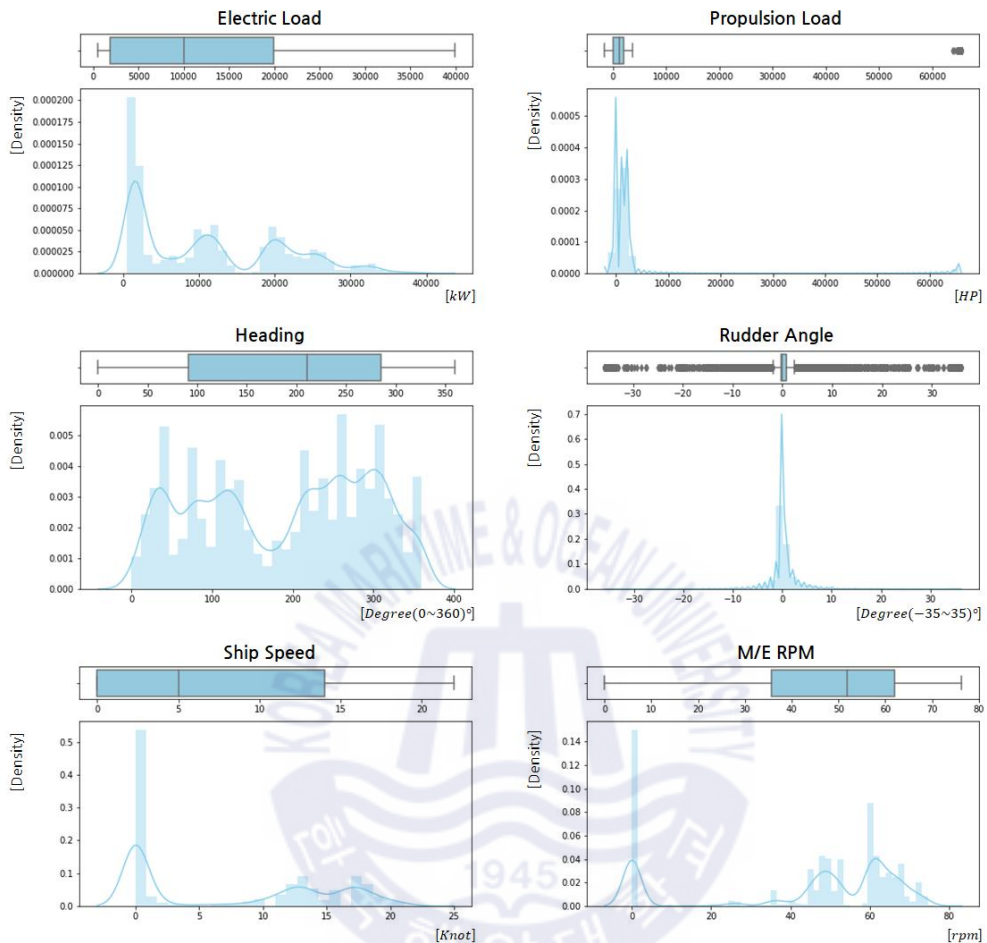


Fig. 2.8 Distribution of vessel related data (1)

Fig. 2.9는 선박의 흘수 데이터이다. 선미, 선수, 좌현, 우현 모두 유사한 데이터 분포를 보이며 주로 10m에서 14m 사이에 데이터가 가장 많이 분포되어 있음을 확인할 수 있다.

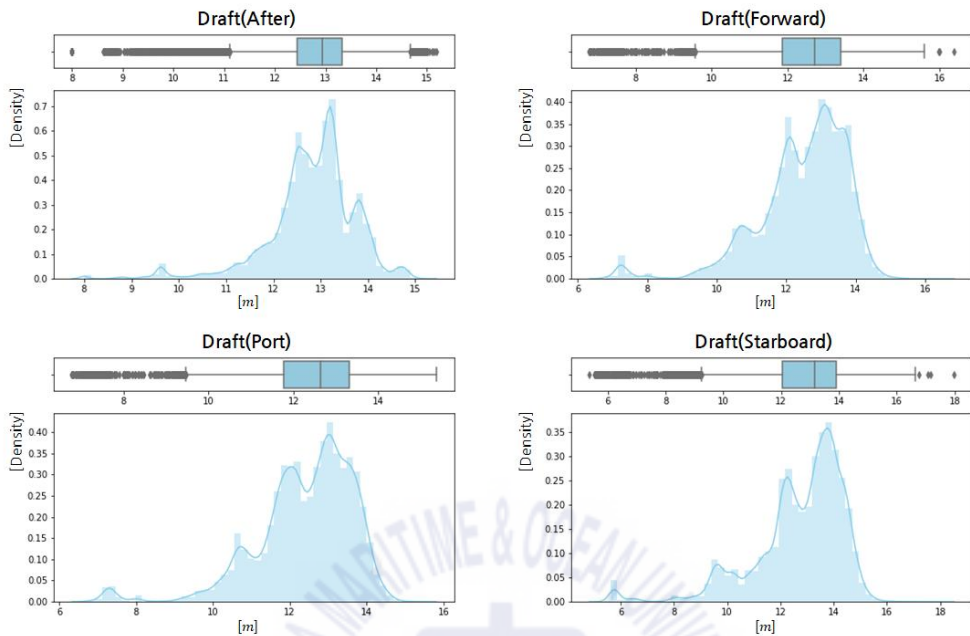


Fig. 2.9 Distribution of vessel related data (2)

- 누락 데이터 확인

선박에서 실시간으로 수집하는 데이터는 선박의 진동 또는 장비의 데이터 송·수신 과정 중 데이터 누락이 발생한다. Table. 2.4와 Fig. 2.10은 변수별 누락 데이터를 나타낸다.

Table 2.4 Missing variables from data

Name	Unit	Quantity
Electric load	kW	4
Propulsion load	$HP(\times 10)$	4
Heading	Degree (0~360) °	4
Rudder angle	Degree (-35~35) °	1,760
Water depth	m	21,633

Water speed	<i>m/s</i>	28
Wind angle	Degree (0~360) °	2
Wind speed	<i>m/s</i>	0
Vessel speed	<i>rpm</i>	0
M/E RPM	<i>rpm</i>	0
Draft(after)	<i>m</i>	0
Draft(forward)	<i>m</i>	0
Draft(port)	<i>m</i>	0
Draft(starboard)	<i>m</i>	0

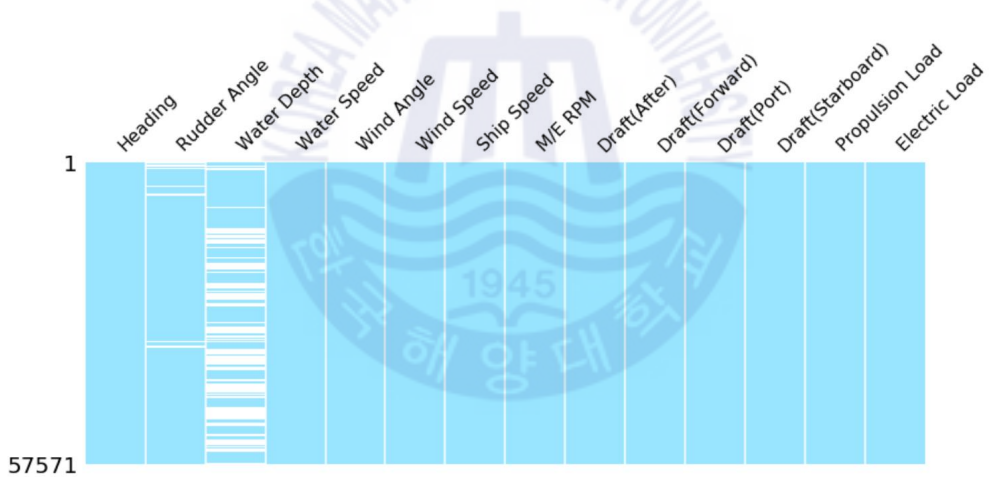


Fig. 2.10 Matrix for visualization of missing values

총 23,435개의 데이터가 누락된 것을 확인할 수 있으며, 수위 데이터와 선박 방향키 각도의 데이터가 가장 많이 누락되었음을 확인할 수 있다.

2.2.2 데이터 산출 및 보완

(1) 누락 데이터의 산출 방법

누락된 데이터를 산출하기 위하여, 본 논문에서는 다중 대입법(MICE, Multiple Imputation by Chained Equations)을 활용하였다. 다중 대입법은 연쇄 방정식에 의해 동작되며, 누락된 데이터를 처리하는데 높은 성능을 보여준다. 특히 연속 데이터의 묶음, 이진 데이터, 카테고리형 데이터를 산출할 수 있는 특징이 있다. 다중 대입법은 6단계를 활용하여 누락 데이터를 대입한다[19].

- 1 단계 : 평균 데이터 입력 등의 단순 대입법을 활용한 데이터 대입을 진행한다.
- 2 단계 : 1 단계에서 대입된 변수 중 특정 변수의 대입 값을 누락 데이터 상태로 변환한다.
- 3 단계 : 2 단계에서 누락 데이터 상태로 변환된 변수를 종속변수를 선정 후 나머지 변수를 활용하여 선형, 로지스틱, 푸아송 회귀 분석 등을 활용한 모델 생성 후 누락 데이터 상태로 변환된 변수에 데이터 입력 한다.
- 4 단계 : 2 단계에서 누락 데이터 상태로 변환된 변수를 다른 변수들의 분석을 위한 독립 변수로 활용한다.
- 5 단계 : 독립 변수를 제외한 나머지 변수를 활용하여 2단계부터 4단계까지의 작업을 반복한다. 한 번의 반복 작업이 완료된 후 기존의 누락 데이터 값은 모두 회귀분석을 활용한 예측 값으로 대체된다.
- 6 단계 : 여러 번의 반복 작업을 진행하며, 누락 데이터를 업데이트 해 나간다.

(2) 누락 데이터 산출 결과

데이터 산출을 활용하여 누락된 데이터를 입력한 결과 기존 누락되었던 데이터를 산출된 데이터로 보완하였다. Fig. 2.11은 Matrix를 활용하여 누락된

데이터가 없음을 나타낸다.

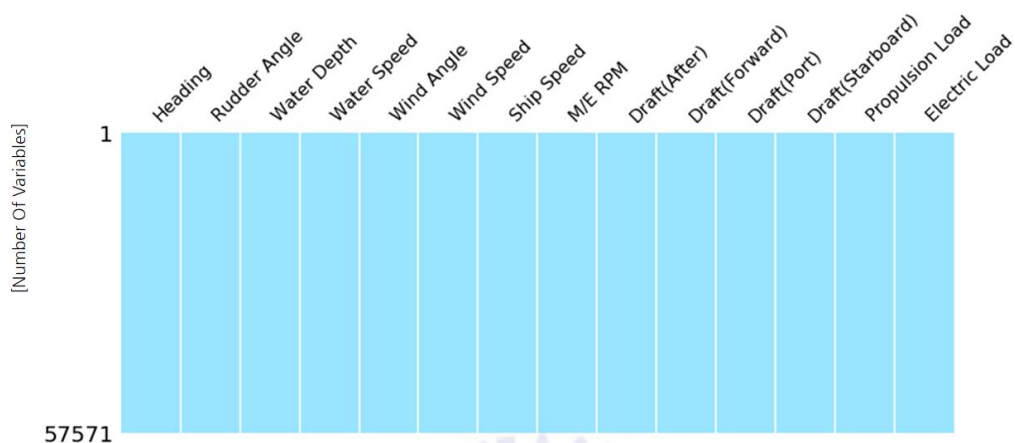


Fig. 2.11 Visualization of missing values imputation result

2.2.3 데이터 변환

데이터 산출 작업 후 2.2.1의 데이터 분석 과정 중 발견한 오류 데이터를 수정하였다. 데이터 변환 작업은 추진 부하, 수심 데이터를 대상으로 진행하였다.

(1) 추진 부하

추진 부하 데이터 중 일부 데이터에 오류가 있음을 발견하였다. 데이터 분석 결과 해당 오류는 선박 데이터 수집 장치의 MSB(Most Significant Bit) 값의 오류가 발생하였거나, 양수 데이터가 음수 데이터로 입력된 경우로 분석되었다. 다음 Fig. 2.12는 추진 부하 데이터를 나타낸다. 이 문제를 해결하기 위하여, 데이터 중 2^{16} 이상의 데이터는 2^{16} 을 제거하고 음수 데이터는 양수 데이터로 변환하였다.

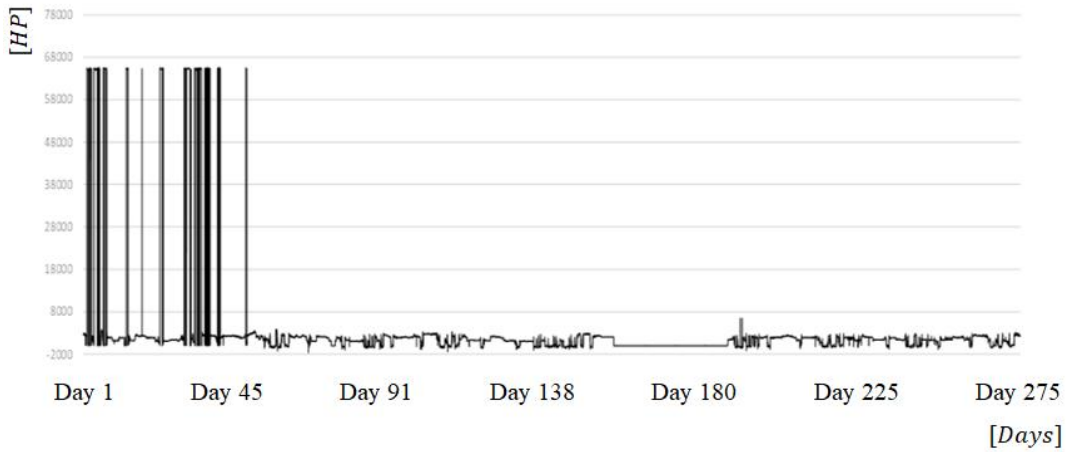


Fig. 2.12 Error data of propulsion load data

Fig. 2.13은 오류 데이터가 수정된 추진 부하를 나타낸다. 그러나 일부 구간에 데이터 수집 오류를 발견하였음을 알 수 있었다. 해당 데이터는 데이터 산출 등이 효과가 없다고 판단하여, 삭제를 진행하였다.

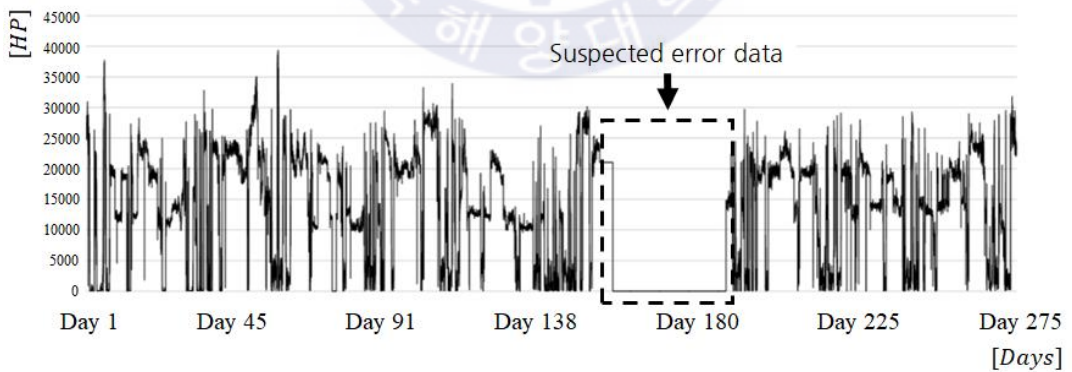


Fig. 2.13 Propulsion load data after conversion

Fig. 2.14은 수집 오류 데이터를 삭제한 후의 추진 부하를 나타낸다.

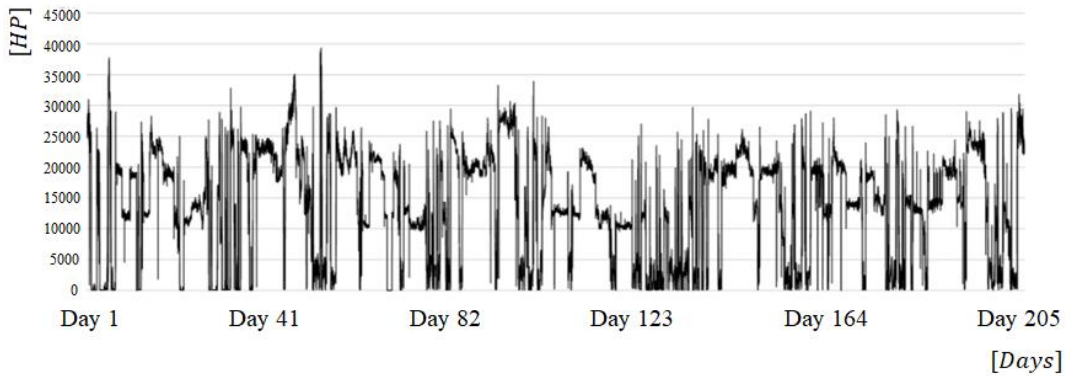


Fig. 2.14 Propulsion load data after delete false data

Fig. 2.15는 데이터 변환 작업이 완료된 추진 부하의 분포를 나타내었다. 데이터 변환 작업 전의 분포와는 다르게 이상치 (outlier) 값이 사라진 것을 확인할 수 있다.

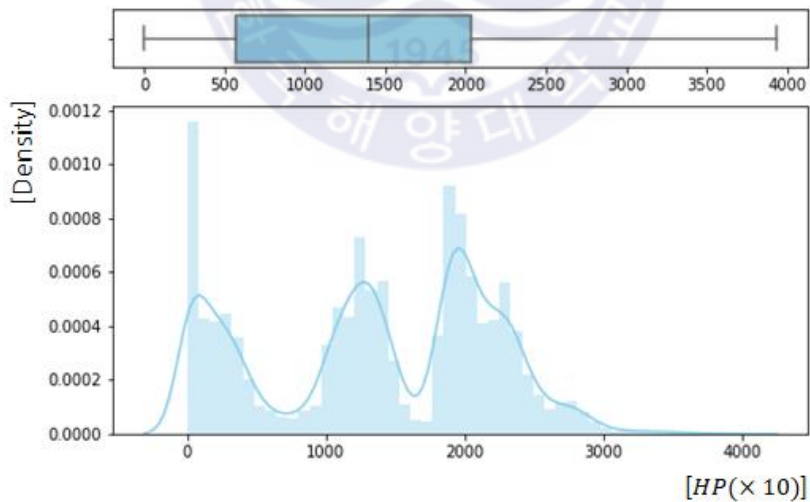


Fig. 2.15 Propulsion load data after conversion

(2) 수심

Fig. 2.16은 수집된 수심데이터를 나타낸다. 수집된 데이터 중 일부 데이터가 음수로 입력되어 있는 것을 확인할 수 있었다. 그러므로 수심데이터 중 오차 데이터로 판단되는 음수 데이터를 양수 데이터로 전환하였다.

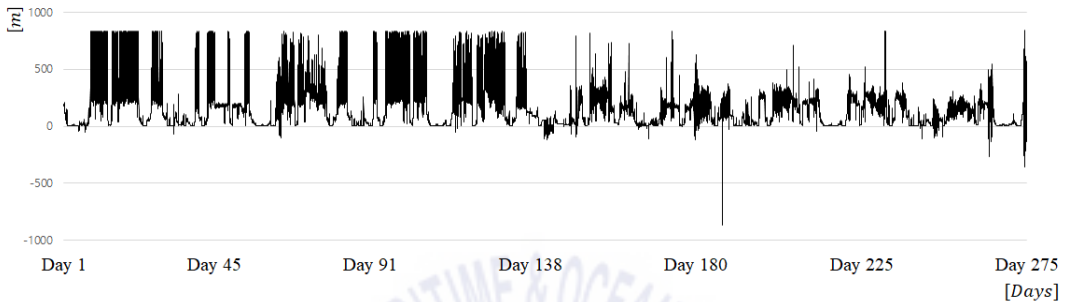


Fig. 2.16 Error data of water depth data

Fig. 2.16은 데이터 변환 작업 후의 수심 데이터 분포를 나타내었다. 특히 Fig 2.9의 수심 데이터와 비교하였을 경우, 이상치 (outlier) 값이 개선되었으며, 또한 변수별 밀도 값이 낮아진 것을 확인할 수 있다.

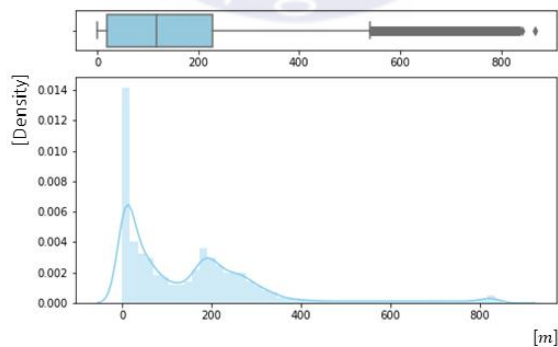


Fig. 2.16 Water depth data after conversion

제 3 장 전기추진 선박 부하 예측 모델 설계 및 구현

본 장에서는 전처리가 완료된 데이터를 활용하여, 전기추진 선박의 부하 예측을 할 수 있는 모델의 설계 및 구현을 진행하였다.

3.1 이론적 배경 및 실험절차

3.1.1 모델의 이론적 배경

(1) RNN (Recurrent Neural Networks)

RNN은 시계열 데이터를 활용하는 모델로써 일반적인 신경망 또는 분류를 위한 분류 전용 신경망과는 달리 입력 데이터와 출력 데이터 사이의 신경망이 상호 참조하는 구조를 지니고 있다. 또한 입력 데이터의 순서를 기반으로 데이터를 처리할 수 있는 모델링의 특징이 있으므로 지난 정보를 기억하고 이를 기반으로 새로운 작업을 처리할 수 있는 특징이 있다[20]. RNN 모델링은 입력 데이터와 출력 데이터의 특징에 따라 다양한 구조로 만들 수 있다. 다음 Fig. 3.1은 입력과 출력에 대한 관계에 따른 다양한 RNN의 구조를 나타낸다.

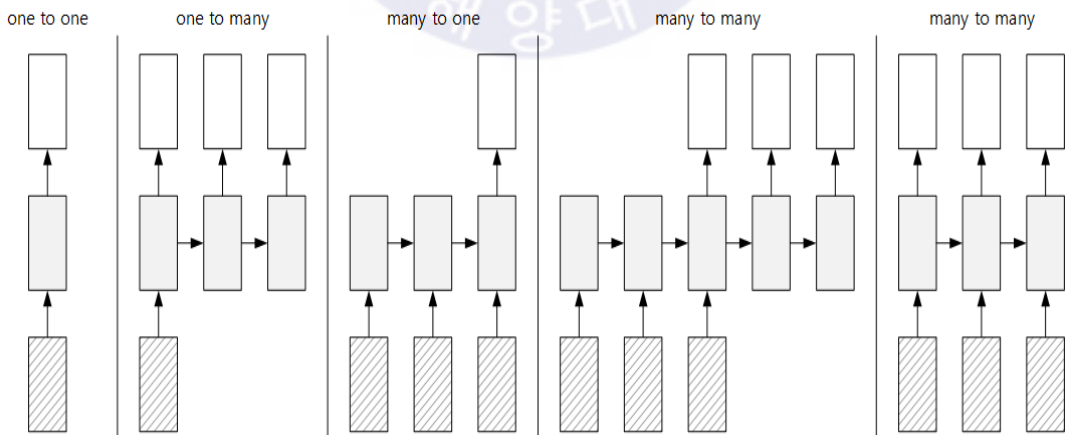


Fig. 3.1 Type of RNN structures according to input and output

Fig 3.1의 입력은 빗금으로 표시되어 있으며, 출력은 흰색 표시되어 있다. 마지막으로 회색은 표시된 영역은 신경망을 의미하며, 각 모델의 구조는 다음의 경우에 활용된다.

- one to one : 입력 데이터와 출력 데이터 모두 시계열 데이터가 아니라 일반적인 형태의 단순 신경망 모델이며 이미지 분류와 같은 작업에 활용되는 모델이다.
- one to many : 입력 데이터가 벡터 형식이며, 출력 데이터가 시계열 데이터 형식이며, 이미지 캡셔닝(image captioning) 등에 활용되는 모델이다.
- many to one : 입력 데이터가 시계열 데이터 이며 출력 데이터는 벡터이며, 감성분석 모델과 같은 경우에 활용되는 모델이다.
- many to many : 입력과 출력 데이터 모두 시계열 데이터 형식이며, 주로 번역기에 사용되는 모델이다.

Fig. 3.2는 RNN의 구조를 나타낸다. RNN은 과거의 값이 미래에 영향을 끼칠 수 있는 구조를 지니고 있다.

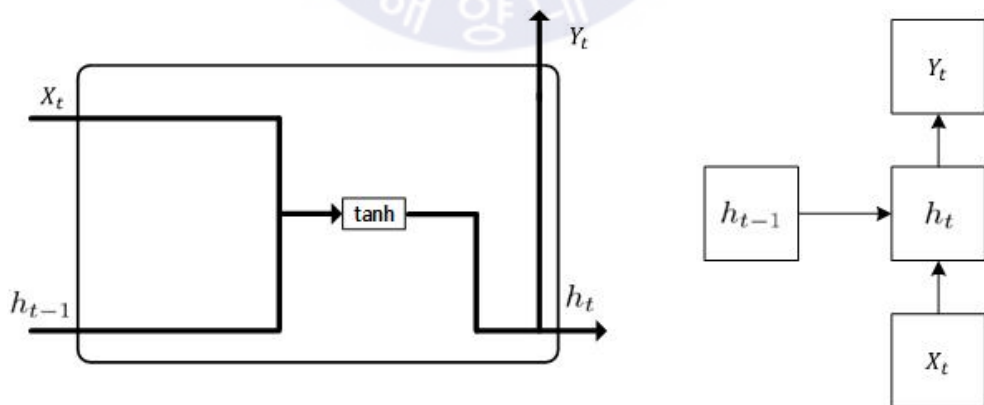


Fig. 3.2 Structure of RNN

X_t 는 현재시간 (t)에서의 입력을 나타내며. h_{t-1} 는 이전 시간($t-1$)의 은닉층의 결과 값을 나타낸다. Y_t 는 식 (3.1)을 사용하여 출력되는 결과 값으로 현재시간 (t)에서의 출력을 의미한다. 또한 가중치 값 W_{hy} 와 식 (3.2)에 의해 결정되는 출력인 h_t 그리고 편향 값인 b_y 를 사용한다. 또한 식 (3.2)는 비선형 함수인 \tanh 를 사용하며, W_{hh} , W_{xh} 는 가중치를 나타내며, b_y , b_h 는 편향을 나타낸다.

$$Y_t = W_{hy}h_t + b_y \quad (3.1)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}X_t + b_h) \quad (3.2)$$

위와 같이 RNN은 이전의 정보를 현재의 문제에 활용할 수 있는 장점이 있다. 그러나 RNN은 현재 시간 (t)부터 이전 시간 ($t-1$)까지의 데이터 처리만을 사용하는 한계가 발생하여 은닉층이 가진 정보가 마지막 은닉층까지 전달되지 못하는 장기 의존성 문제(long-term dependency)가 발생하게 된다. 이러한 문제를 해결하기 위하여 LSTM이 연구되었다. Fig. 3.3은 장기 의존성 문제를 일으킬 수 있는 RNN의 구조를 나타내었다.

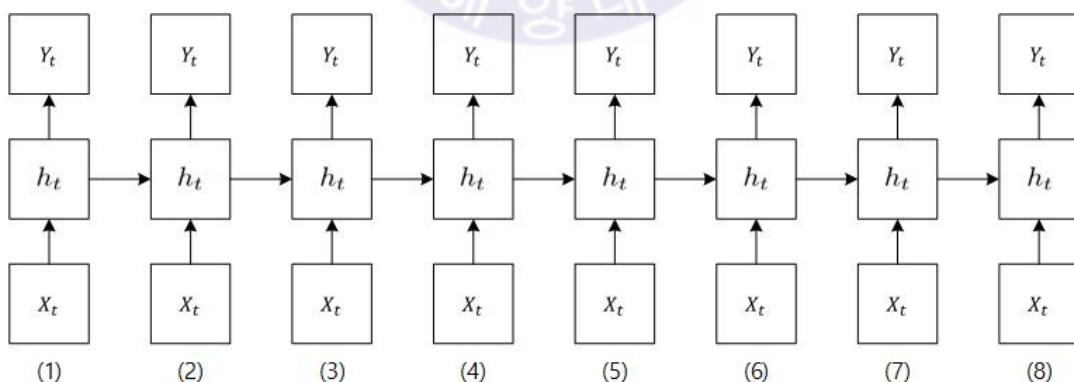


Fig. 3.3 Long term dependency problems caused by RNN Structure

(2) LSTM (Long Short-Term Memory models)

LSTM은 block cell이라고 불리는 block으로 구성된다. LSTM의 기본 개념은 장기 및 단기 순차 메모리를 활용하여 현재 및 과거 입력 값을 기반으로 중간 출력을 총체적으로 결정하는 것이다. 블록 셀의 입력에는 현재 관측치와 일부 매개 변수가 포함되며, 출력값은 장기적인 의존성에 의해 결정된다[21-25]. Fig. 3.4는 LSTM의 블록 셀을 나타낸다.

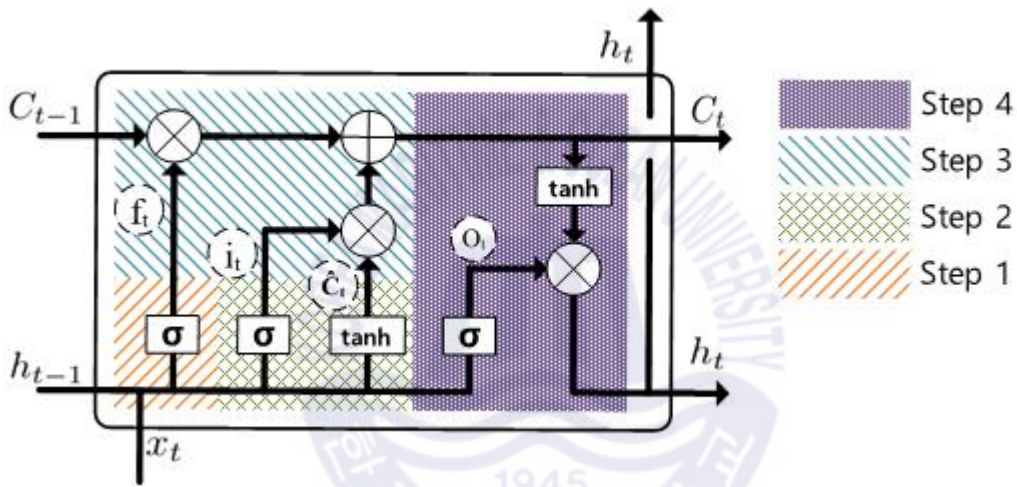


Fig. 3.4 Structure of LSTM block cell

LSTM의 셀은 입력, 망각, 출력 으로 구성되어 있다. 또한 모든 반복 신경망은 신경망의 반복 모듈 체인 형태를 가진다. 표준 RNN 에서 이 반복 모듈은 단일 \tanh 층과 같은 매우 간단한 구조를 갖는다. LSTM도 체인 구조를 갖지만 반복 모듈은 다른 구조를 갖는다. 하나의 신경망 계층을 갖는 대신에 매우 특별한 방식으로 상호 작용하는 4단계가 있으며, 셀 상태를 보호하고 제어하기 위해 사용된다

첫 번째 단계는 셀 상태에서 어떤 정보를 버릴 것인지 결정한다. 결정은 forget gate layer라는 시그모이드(sigmoid) 층에 의해 이루어진다. h_{t-1} 과 x_t 를

보고 셀 상태 C_{t-1} 의 각 숫자에 대해 0과 1 사이의 숫자를 출력한다. 1은 완전한 유지를 나타내고 0은 완전히 제거를 나타낸다. 다음 식 (3.3)은 첫 번째 단계의 연산과정 나타낸다.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (3.3)$$

두 번째 단계는 셀 상태에서 어떤 새로운 정보를 저장할 것인지 결정한다. 두 부분으로 이루어져 있다. 첫 부분은 input gate layer라는 시그모이드 레이어가 업데이트 할 값을 결정한다. 다음 부분인 tanh layer는 상태에 추가 될 수 있는 새로운 후보값 \hat{C}_t 의 벡터를 생성한다. 마지막으로 이 두가지를 결합하여 상태에 대한 업데이트를 만든다. 다음 식 (3.4)와 식 (3.5)는 두 번째 단계의 연산 과정을 나타낸다.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (3.4)$$

$$\hat{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (3.5)$$

세 번째 단계에서는 이전 셀 상태 C_{t-1} 을 새 셀 상태 C_t 으로 업데이트 한다. 이전 상태에 F_t 를 곱하여 이전에 잊어야 할 데이터에 대하여 잊는다. 그리고 그것을 * \hat{C}_t 라고 표현한다. 이 값은 새로운 후보 값으로 각 상태 값을 얼마나 업데이트하기로 결정했는지에 따라 조정된다. 다음 식 (3.6)은 세 번째 단계에서의 연산과정을 나타낸다.

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \quad (3.6)$$

네 번째 단계에서는 출력할 내용을 결정한다. 이 출력은 셀 상태를 기반으로 하지만 필터링 된 버전이 된다. 먼저, 시그모이드 레이어를 실행하여 셀 상태에서 출력할 부분을 결정한다. 그런 다음 셀 상태를 tanh를 통해

시그모이드 게이트의 출력에 곱하여 결정한 부분만 출력한다. 다음 식 (3.7)과 식 (3.8)은 네 번째 단계에서의 연산과정을 나타낸다.

$$O_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{3.7}$$

$$h_t = o_t * \tanh(C_t) \tag{3.8}$$

(3) bidirectional LSTM

기존의 LSTM은 데이터의 입력을 과거에서 미래의 순서로 입력하여 계산 결과를 생성한다. 그러나 bidirectional LSTM은 기존의 LSTM의 구조에 데이터의 입력을 미래에서 과거의 순서로 입력하여 계산하는 영역을 추가하여, 계산을 진행한다[26]. 세부 계산은 같으나, Fig. 3.5와 같이 값을 기존 방향의 LSTM 결과 값과 반대 방향의 LSTM 결과 값을 합하여 준다.

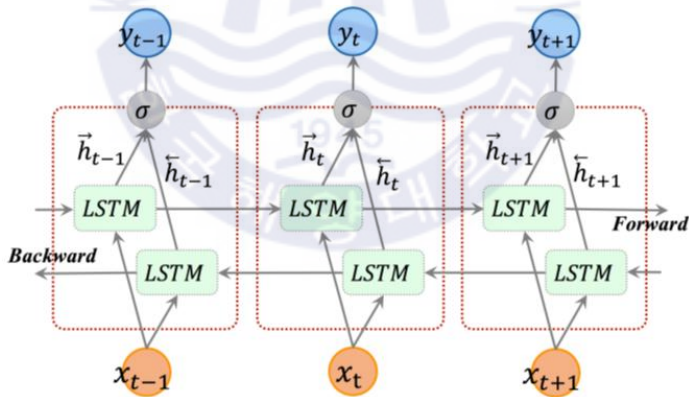


Fig. 3.5 bidirectional LSTM structure

다음 식 (3.9), (3.10), (3.11), (3.12)는 bidirectional LSTM의 식을 나타낸다. W_f , W_i , W_o 그리고 W_c 는 세개의 게이트에서 입력되는 가중치 행렬이며, 히든 레이어 입력 셀이다. U_f , U_i , U_o 그리고 U_c 는 이전 셀과 연결되며, 출력 셀과 연결되는 가중치 행렬이다. b_f , b_i , b_o 그리고 b_c 네 개의 바이어스

벡터이다. σ_g 는 게이트의 활성화 함수이며, 일반적으로 시그모이드 함수가 동작된다.

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \quad (3.9)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \quad (3.10)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \quad (3.11)$$

$$\hat{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (3.12)$$

위의 4개 식을 기반으로 각 시간의 변화를 t , 셀 출력 상태를 C_t , 레이어의 출력을 h_t 라고 하였을 때 식 (3.13), (3.14)와 같이 계산된다.

$$C_t = f_t \times C_{t-1} + i_t \times \hat{C}_t \quad (3.13)$$

$$h_t = o_t \times \tanh(C_t) \quad (3.14)$$

(4) LSTM auto encoder

auto encoder는 입력한 데이터를 압축하여, 데이터의 특징을 추출하는 장점을 가지고 있다. 입력된 데이터는 encoder 레이어를 활용하여 데이터를 압축하고, decoder 레이어를 활용하여 압축해제를 진행한다. 이때 압축 해제된 데이터는 입력된 데이터는 특징을 보유하고 있다. auto encoder는 기존 수학적 무손실 압축 알고리즘과 달리, 입력된 데이터의 압축 과정을 통해 기존 데이터의 특징이 일부 손실이 발생하는 단점을 보유하고 있다. 그러나 데이터의 특징을 자동 학습하는 장점을 보유하는 장점이 존재하여, 영상 처리와 같이 데이터의 특징을 자동으로 분석할 수 있는 알고리즘이 필요한 연구 영역에서 활용 중이다. 또한, 시계열 데이터 예측과 시계열 데이터 중 비정상 데이터의 구별에 많이 활용된다.

다음 Fig. 3.6은 auto encoder를 나타낸다. encoder는 입력 데이터를 출력 데이터로 복사하는 특성을 가지고 있으며, decoder는 encoder를 지나 압축된 데이터를 다시 원래 형태의 데이터로 변형하는 작업을 수행한다.

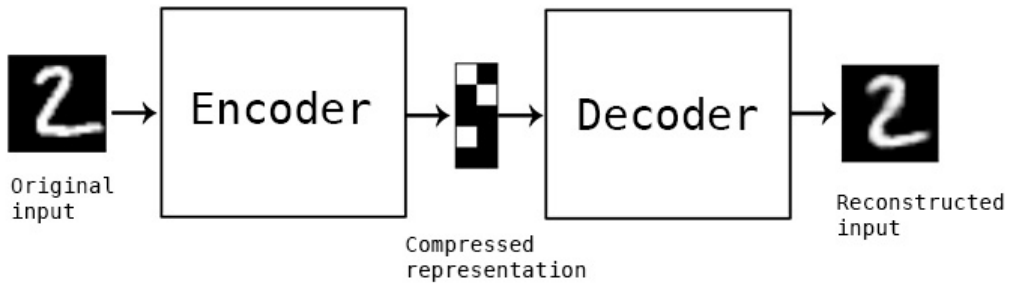


Fig. 3.6 Auto encoder structure

(5) CNN (Convolutional Neural Network)

CNN[27]은 convolution layer와 pooling layer를 사용하여 특징을 추출하는데 사용되어진다. 입력 데이터를 필터가 순회하며 합성곱을 계산하고, 결과를 이용하여 feature map을 만든다. feature map은 subsampled를 통해서 차원을 줄여주는 효과를 가지게 된다. convolution layer는 filter 크기, stride, padding 적용여부, max pooling의 크기에 따라서 출력 데이터의 모양이 결정된다.

convolution Layer는 입력데이터로부터 특징을 추출하는 역할을 한다. 컨볼루션 레이어는 필터와 값을 비선형 값으로 바꾸어주는 활성화 함수로 이루어져 있다. 또한, 입력 데이터는 일반적으로 3차원으로 구성되나, 본 연구에서는 데이터 특성을 고려하여 CNN을 1차원 입력 데이터로 사용한다. 필터는 그 특징이 데이터에 있는지 없는지를 검출해주는 함수이다. 필터는 입력받은 데이터에서 그 특성을 가지고 있으면 큰 값이 나오고, 특성이 없으면 결과값이 0에 가까운 값이 나오면서 해당 특성이 있는지를 파악하게 해준다. 필터는 filter number, stride, padding, activation function으로 구성된다.

pooling layer는 컨볼루션 레이어의 출력데이터를 입력으로 받아서 출력 데이터의 크기를 줄이거나 특정 데이터를 강조하는 용도로 사용된다. 일반적으로 더 작은 차원의 매트릭스가 구성된다. 풀링 레이어의 종류에는 max pooling, average pooling, min pooling이 있다. 풀링 레이어 들은 컨볼루션 레이어와 비교하여 세가지 특징을 가진다. 학습대상의 파라미터가 없으며, 풀링레이어를 통과하면 행렬의 크기가 감소하고, pooling layer를 통해서 채널의

수가 변경되지 않는다는 특징을 가진다.

fully connected layer는 전 결합층이라 불리며, CNN에서 연산된 데이터의 결과를 1차원 배열로 정렬하는 역할을 한다. 통상적으로 분류하고자 하는 데이터의 결과 종류에 맞추어 길이를 사용자가 지정하며, drop out 등의 기법을 활용하여 데이터에 학습이 편향되는 역할을 방지하는 역할을 할 수 있다. padding은 convolution layer와 pooling layer에서 입력 데이터를 감싸는 것처럼 픽셀을 배치하는 기법을 의미한다. padding의 목적은 convolution 처리를 하더라도 데이터의 크기가 변하지 않게 하기 위해서 이다. stride는 convolution filter가 이동하는 간격을 의미하며, stride가 커지면 filter의 이동 간격도 커지기 때문에 생성되는 데이터의 크기가 작어지는 특징이 있다.

Fig. 3.7은 2D CNN과 1D CNN의 구조를 나타내었다.

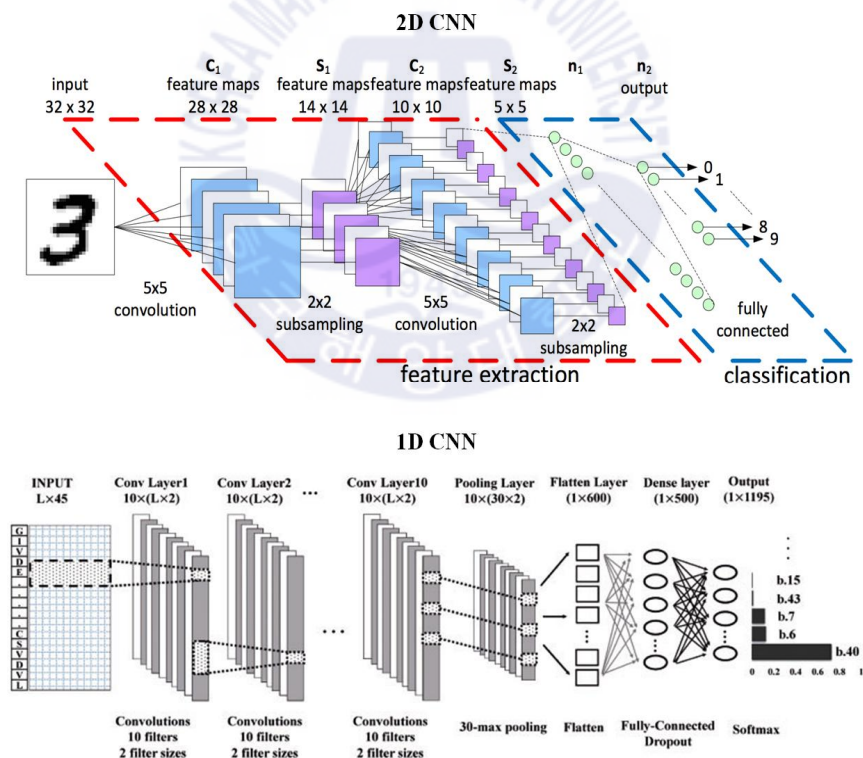


Fig. 3.7 CNN structure

위와 같이 CNN은 일반적인 신경망과 동일하게 역전파를 활용하는 학습을 진행하며, convolution layer는 filter가 학습에 의해 최적화 되는 특징이 있다.

3.1.2 예측 모델의 실험 절차

데이터 세트는 현재 운항 중인 컨테이너 선박에서 10분마다 기록된 데이터를 사용하였으며, 약 205일간 저장 되었다. 저장된 데이터 중 선박 부하에 영향을 끼치는 데이터를 선택하여, 14개의 변수로 데이터를 취합 하였다. 또한 학습, 훈련, 검증에 데이터 표준화를 적용하였다. 여기에서 훈련 데이터는 모델 훈련에 사용할 데이터를 의미하며, 테스트 데이터는 훈련의 정확도 평가에 이용된다. 그리고 검증 데이터는 훈련 데이터를 활용하여 학습되는 모델의 성능 향상과 과적합(overfitting) 문제를 해결할 수 있게 사용된다. 그러므로 본 연구에서는 예측 모델의 과적합 문제를 해결하고자 기준모델 테스트 결과로 얻어진 학습, 훈련, 검증 데이터 비율을 선정하였다[28]. 학습과 훈련의 데이터 비율은 7:3을 사용하였으며 훈련과 검증 데이터 셋의 비율도 7:3을 사용하였다. 그러므로 학습 데이터 세트는 100일간의 데이터를 사용하였다. 또한 검증 데이터 세트는 43일간의 데이터를 사용하였다. 마지막으로 훈련 데이터 세트는 61일간의 데이터를 사용하였다. Fig 3.8은 훈련과 학습 데이터의 비율을 나타내었다.

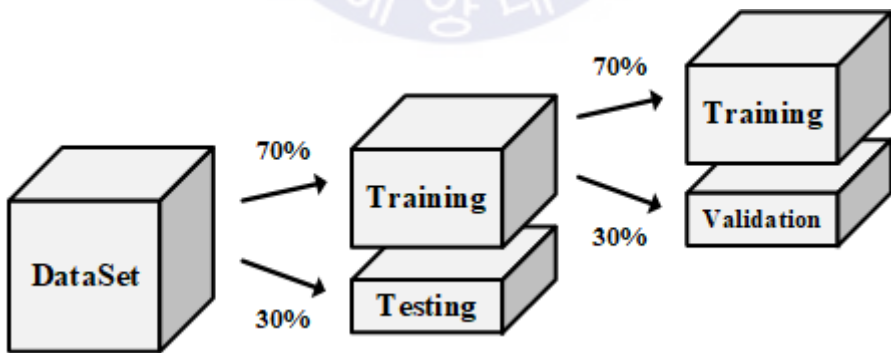


Fig. 3.8 Separate data with train and test data set

훈련의 속도 향상을 위하여 mini-batch gradient descent을 사용하였다. 배치 크기는 512를 사용하였고, learning rate는 0.001, 그리고 epoch는 500을 주었으며, 학습 시 오버피팅을 방지하기 위하여, early stopping을 설정하였다. 마지막으로 early stopping의 patience를 100으로 설정하여, 모델이 최적의 값을 학습할 수 있도록 구성하였다. 다음 Fig. 3.9는 학습 절차를 나타낸다.

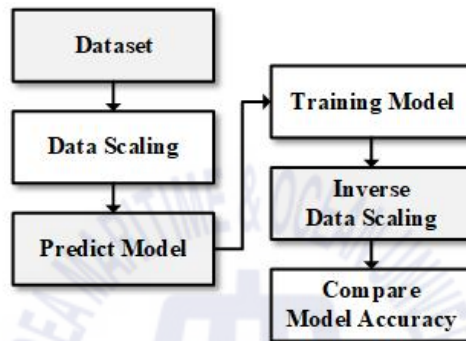


Fig. 3.9 Training procedure

정확한 데이터 학습을 위하여, 데이터 셋을 스케일링한다. 스케일링 된 데이터는 2.2장에 제시된 각 모델로 입력되고 학습이 진행된다. 학습이 완료된 모델에서 예측하는 예측 값들을 역 스케일링(inverse scaling)하여, 실제 값과 비교한다. 실제 값과의 비교를 평가하는 값은 평균 제곱근 오차(RMSE:root mean square error)를 사용하며, 각 모델별 실제 값과 모델의 예측값을 비교하여 가장 성능이 좋은 모델을 선별한다.

- 시계열 데이터 생성

시계열 데이터를 생성하기 위하여, time step을 선정하였다. time step은 예측 프로그램 구성시 데이터의 몇 회분을 통하여 예측하는지에 대한 수치이며, time step의 값은 활용하고자 하는 이전시간 값의 변화에 따라 값의

크기 설정이 가능하다. 통상적으로 time step이 길어질 경우, 경사 사라짐 현상(vanishing gradient problem)이 발생하는 문제가 발생한다. 본 논문에서 제시하는 데이터는 10분 단위로 계측된 데이터이므로 time step을 1로 설정할 경우 10분전 데이터 한 세트를 활용하여 다음 10분 뒤의 전기추진 선박의 부하를 예측함을 의미한다. 본 연구에서는 지난 50분 간의 전기추진 선박데이터 변화에 따른 10분 후의 전기추진 선박 부하를 예측하고 하였다. 그러므로 time step을 5로 설정하였다.

Table 3.1은 시계열 데이터 생성 함수를 나타내었으며, Fig. 3.10은 본 연구에서 활용한 시계열 데이터의 time step을 나타낸다.

Table 3.1 Code of time series data

```
time_steps = 5

def create_dataset(X, y, time_steps=1):
    Xs, ys = [], []
    for i in range(len(X) - time_steps):
        v = X.iloc[i : i + time_steps].values
        Xs.append(v)
        ys.append(y.iloc[i + time_steps])
    return np.array(Xs), np.array(ys)
```

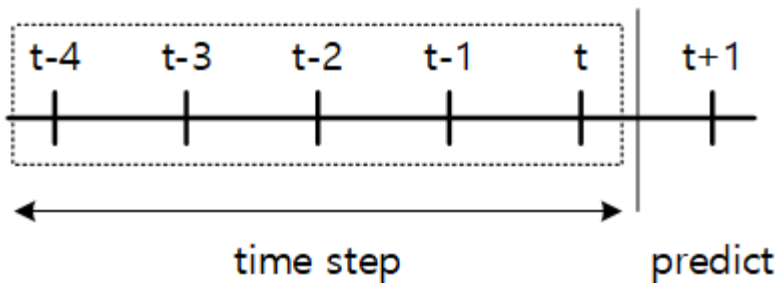


Fig. 3.10 The feature of time step

- 예측모델 평가 함수

table 3.2는 예측모델 평가 함수를 나타내며, 모델이 예측한 값과 실제 값을 비교하였다. 비교에는 평균 제곱근 오차 (RMSE)가 활용되었으며 식 (3.15)는 평균 제곱근 오차의 수식이다. 여기서 n 은 데이터의 개수를 의미한다. y_i 는 i 번째의 실제 값을 의미한다. \hat{y}_i 는 i 번째의 예측값을 의미한다.

Table 3.2 Code of RMSE

```
def rmse(y_true, y_pred):
    return backend.sqrt(backend.mean(backend.square(y_pred - y_true), axis=-1))
```

$$\sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \tag{3.15}$$

- 학습 실행 함수

여러 모델의 학습을 편리하게 할 수 있도록 학습 실행 함수를 활용하였다. table 3.3은 학습 실행 함수를 나타낸다. 학습 실행함수에는 훈련 데이터, 결과 데이터, 모델정보, 훈련 횟수를 저장한다.

Table 3.3 Code of training function

```
def callHistory(model, X_train, y_train, epoc) :
    history = model.fit(
        X_train, y_train,
        epochs=epoc,
        batch_size=512,
        validation_split=0.3,
        shuffle=False,
        callbacks=[early_stop]
    )
    plot_history(history)
```

3.2 예측 모델 설계 및 구현

본 논문은 다양한 구조의 예측모델을 적용하여 전기추진 선박 전력 부하의 예측 정확성을 확인하고자 한다. 그러나 기존 전력부하 예측에 활용된 AI 모형은 상대적으로 적은 범위(0~100)의 전력 값을 예측하는데 이용되었으며, 전력부하의 변동성이 낮은 특징이 있다. 그러나 전기추진 선박의 경우 예측 전력부하의 범위가 상대적으로 넓은 특징이 존재한다(0~40,000). 본 연구에서는 전기추진 선박의 전력부하의 범위와 변동성을 가장 잘 예측할 수 있는 모델이 필요하다. 그러므로 다양한 형태의 모델을 활용하여, 전기추진 선박 부하예측 성능이 가장 높은 모델을 선정하는 것에 목적이 있다. Table 3.4는 설계된 모델의 적용을 나타내었다.

Table 3.4 Summary of model structure

번호	모델 구조	조합	구성
1	LSTM	-	LSTM : (1 x 512) - (1 x 256) - (128)
2	Bidirectional LSTM	-	Bidirectional LSTM : (1 x 512) - (1 x 256) - (128)
3	CNN-LSTM	Direct	CNN : (5 x 128) - (2 x 128) - (2 x 64) - (1 x 64) - (64) LSTM : (1 x 512) - (1 x 256) - (128)
4	CNN-Bidirectional LSTM	Direct	CNN : (5 x 128) - (2 x 128) - (2 x 64) - (1 x 64) - (64) Bidirectional LSTM : (1 x 512) - (1 x 256) - (128)
5	CNN-LSTM	Parallel	CNN : (5 x 128) - (2 x 128) - (2 x 64) - (1 x 64) - (64) LSTM : (1 x 512) - (1 x 256) - (128)
6	CNN-Bidirectional LSTM	Parallel	CNN : (5 x 128) - (2 x 128) - (2 x 64) - (1 x 64) - (64) Bidirectional LSTM : (1 x 512) - (1 x 256) - (128)
7	ANN	-	ANN : (5 x 13) - (5 x 1024) - (5120)
8	DNN	-	DNN : (5 x 13) - (5 x 1024) - (5 x 512) - (5 x 64) - (320)
9	CNN	-	CNN : (5 x 128) - (2 x 128) - (2 x 64) - (1 x 64) - (64)

3.2.1 LSTM

일반적인 vanilla LSTM의 구조와 유사하나, 다층 LSTM레이어와 단일 신경망으로 구성하였으며, many to one 구조를 채택하였다. LSTM에서 학습된 가중치 및 활성화 함수를 이용하여 다음 값을 예측한다. 다음 Fig. 3.11은 LSTM 모델의 구성을 나타낸다[29-30].

본 구조의 핵심은 다층 LSTM을 활용하여 이전 시간에 존재하였던 전기추진 선박의 부하 데이터와 유사한 데이터를 기반으로 전기추진 선박의 부하를 예측하는 것이다.

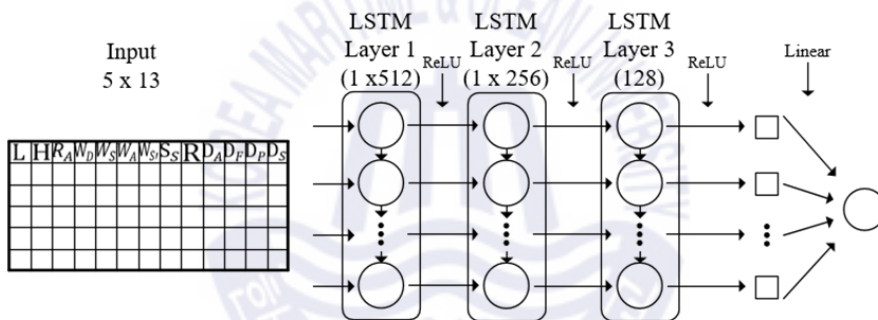


Fig. 3.11 Construction of LSTM

다음 Table 3.5는 LSTM 모델 구현을 나타내었다.

Table 3.5 Code of LSTM model

```
xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

xLstm_2 = LSTM(512, return_sequences=True, activation='relu')(xInput)
xLstm_2 = LSTM(256, return_sequences=True, activation='relu')(xLstm_2)
xLstm_2 = LSTM(128, activation='relu')(xLstm_2)

xOutput = Dense(1, activation='linear')(xLstm_2)
```

```

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse',optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]

```

3.2.2 bidirectional LSTM

bidirectional-LSTM 모델을 3개의 계층으로 구성하였으며, Many To One 구조를 채택하였다. bidirectional-LSTM에서 학습된 가중치 및 활성화 함수를 이용하여 다음 예측값을 출력한다. 본 구조의 핵심은 bidirectional-LSTM 구조의 장점인 은닉층 성능 향상과 데이터 길이와 층에 상관없이 이전 시간의 정보가 손실되지 않는 점을 활용하여, 전기추진 선박의 부하를 예측하는 것이다. Fig. 3.12는 bidirectional-LSTM 기반 모델의 구조이다[31].

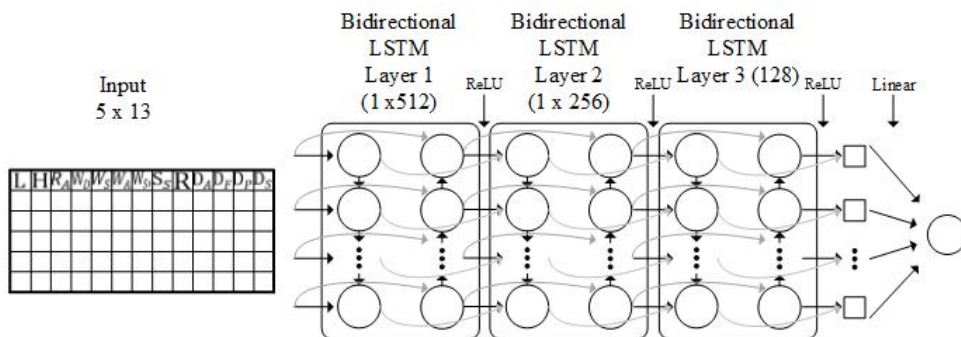


Fig. 3.12 Construction of bidirectional LSTM

다음 table 3.6은 bidirectional LSTM 모델 구현을 나타내었다.

Table 3.6 Code of bidirectional LSTM model

```
xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

xLstm_2 = Bidirectional(LSTM(512, return_sequences=True, activation='relu'))(xInput)
xLstm_2 = Bidirectional(LSTM(256, return_sequences=True, activation='relu'))(xLstm_2)
xLstm_2 = Bidirectional(LSTM(128, activation='relu'))(xLstm_2)

xOutput = Dense(1, activation='linear')(xLstm_2)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse', optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
```

3.2.3 CNN-LSTM (direct)

시계열 데이터는 1D CNN으로 입력되며, 1D CNN에서 추출한 시계열 데이터의 특징을 3개의 계층으로 구성된 LSTM으로 입력한다. 이후 LSTM에서 학습된 가중치 및 활성화 함수를 이용하여 다음 예측값을 출력한다. Fig. 3.13은 CNN-LSTM (direct) 기반 모델의 구조이다[32].

본 구조의 핵심은 CNN에 입력된 전기추진 선박 데이터의 특징을 추출하고 이전 시간대에 존재하는 데이터를 분류하여, LSTM의 시계열 데이터 예측 성능

항상에 활용하고자 하는 것이다. 우선 시계열로 준비된 전기추진 선박의 데이터를 CNN 모델에 입력한다. CNN 모델에서 분류된 결과 값을 LSTM에 입력하기 위하여 1차원 데이터로 CNN의 데이터를 변환하는 flatten 레이어로 입력한다. flatten 레이어의 데이터를 LSTM 모델에 입력하고 전기추진 선박의 부하 예측 값을 출력하게 된다.

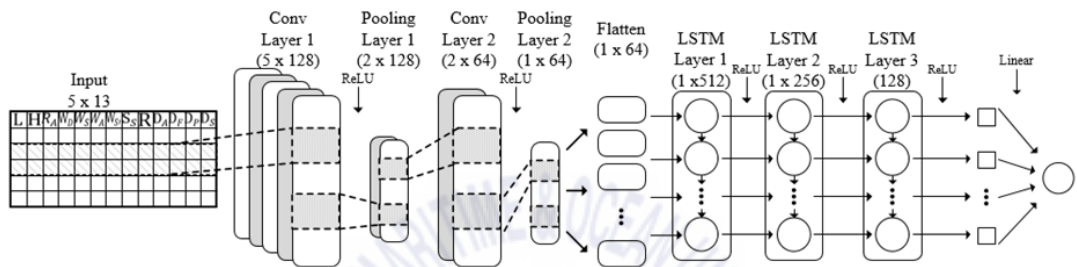


Fig. 3.13 Construction of CNN-LSTM (direct)

다음 Table 3.7은 CNN-LSTM (direct) 모델 구현을 나타내었다.

Table 3.7 Code of CNN-LSTM (direct) model

```

xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

CNN = Conv1D(128, 14, padding='same', activation='relu', strides=1)(xInput)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = Conv1D(64, 7, padding='same', activation='relu', strides=1)(CNN)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = TimeDistributed(Flatten())(CNN)

xLstm_2 = LSTM(512, return_sequences=True, activation='relu')(CNN)
xLstm_2 = LSTM(256, return_sequences=True, activation='relu')(xLstm_2)
xLstm_2 = LSTM(128, activation='relu')(xLstm_2)
    
```

```

xOutput = Dense(1, activation='linear')(xLstm_2)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse', optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
final_df.to_csv(result_name, index=False)

```

3.2.4 CNN-bidirectional LSTM (direct)

시계열 데이터는 1D CNN으로 입력되며, 1D CNN에서 추출한 시계열 데이터의 특징을 3개의 계층으로 구성된 bidirectional-LSTM으로 입력한다. 이후 bidirectional-LSTM에서 학습된 가중치 및 활성화 함수를 이용하여 다음 예측값을 출력한다. Fig. 3.14는 CNN-LSTM (direct) 기반 모델의 구조이다.

본 구조의 핵심은 CNN에 입력된 전기추진 선박 데이터의 특징을 추출하고 이전 시간대에 존재하는 데이터를 분류하여, bidirectional-LSTM 구조의 장점인 은닉층 성능 향상과 데이터 길이와 층에 상관없이 이전 시간의 정보가 손실되지 않는 점을 활용하여, 데이터 예측 성능 향상에 활용하고자 하는 것이다. 즉 CNN의 분류 결과에 bidirectional-LSTM의 예측을 결합한 형태로 구성하였다.

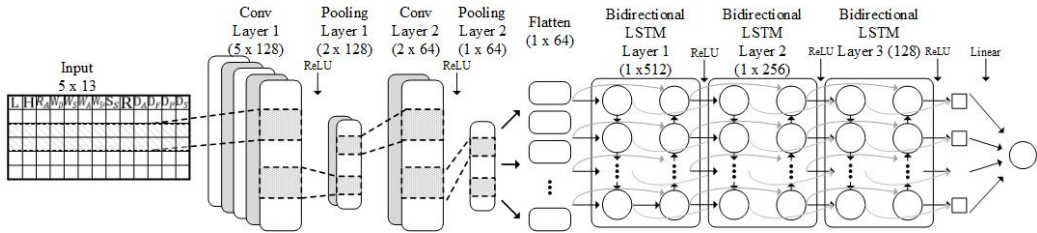


Fig. 3.14 Construction of CNN-bidirectional LSTM (direct)

Table 3.8은 CNN-bidirectional LSTM (direct) 모델 구현을 나타내었다.

Table 3.8 Code of CNN-bidirectional LSTM (direct) model

```

xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

CNN = Conv1D(128, 14, padding='same', activation='relu', strides=1)(xInput)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = Conv1D(64, 7, padding='same', activation='relu', strides=1)(CNN)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = TimeDistributed(Flatten())(CNN)

xLstm_2 = Bidirectional(LSTM(512, return_sequences=True, activation='relu'))(CNN)
xLstm_2 = Bidirectional(LSTM(256, return_sequences=True, activation='relu'))(xLstm_2)
xLstm_2 = Bidirectional(LSTM(128, activation='relu'))(xLstm_2)

xOutput = Dense(1, activation='linear')(xLstm_2)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse', optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

```

```

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
final_df.to_csv(result_name, index=False)

```

3.2.5 CNN-LSTM (parallel)

CNN에서 학습된 시계열 예측 데이터와 LSTM에서 학습된 시계열 예측 데이터를 병렬 구조로 결합하여, 시계열 예측을 진행할 수 있게 구성하였다. CNN과 LSTM에서 학습된 가중치 및 활성화 함수를 이용하여 다음 예측값을 출력한다. Fig. 3.15는 CNN-LSTM (parallel) 기반 모델의 구조이다.

본 구조의 핵심은 CNN에 입력된 전기추진 선박 데이터의 특징을 추출하고 이전 시간대에 존재하는 데이터를 분류된 결과의 시계열 데이터의 예측값과 LSTM에서 학습된 시계열 데이터예측을 합성하여 출력을 나타내는 것이다. 본 구조는 3.2.3에서 제안된 CNN-LSTM (direct) 구조와는 다른 방법을 활용하여 전기추진 선박의 예측 부하 값을 출력한다. 모델의 학습을 위하여 준비된 전기추진 선박의 시계열 데이터를 활용하여 CNN 모델과 LSTM 모델로 입력한다. CNN 모델은 전기추진 선박의 데이터 특성을 분류하고 분류된 결과 값을 이차원 값으로 변환한다. 그리고 LSTM 모델은 시계열 데이터를 분석하여 전기추진 선박의 예측 부하 값을 출력한다. 각 모델에서 도출된 CNN 모델의 결과 데이터와 LSTM 모델에서 도출된 전기추진 선박 예측 부하 값을 합성하기 위하여, concatenate 레이어를 활용하여 딥러닝 모델간 병합을 수행한다. 위와 같이 모델간 합성을 활용하여

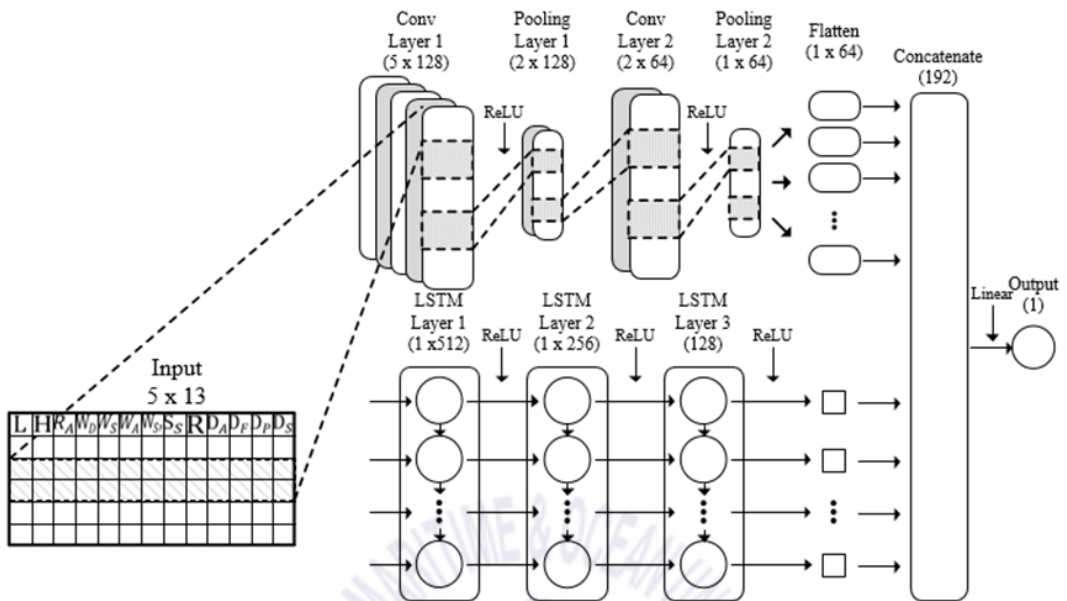


Fig. 3.15 Construction of CNN-LSTM (parallel)

Table 3.9는 CNN-LSTM (parallel) 모델 구현을 나타내었다.

Table 3.9 Code of CNN-LSTM (parallel) model

```

xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

CNN = Conv1D(128, 14, padding='same', activation='relu', strides=1)(xInput)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = Conv1D(64, 7, padding='same', activation='relu', strides=1)(CNN)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = TimeDistributed(Flatten())(CNN)

xLstm_2 = Bidirectional(LSTM(512, return_sequences=True, activation='relu'))(CNN)
xLstm_2 = Bidirectional(LSTM(256, return_sequences=True, activation='relu'))(xLstm_2)
xLstm_2 = Bidirectional(LSTM(128, activation='relu'))(xLstm_2)

```

```

xOutput = Dense(1, activation='linear')(xLstm_2)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse',optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
final_df.to_csv(result_name, index=False)

```

3.2.6 CNN-bidirectional LSTM (parallel)

CNN에서 학습된 데이터 특징 분석 결과와 bidirectional-LSTM에서 학습된 시계열 예측 데이터를 병렬 구조로 결합하여, 시계열 예측을 진행할 수 있게 구성하였다. CNN과 bidirectional-LSTM에서 학습된 가중치 및 활성화 함수를 이용하여 다음 예측값을 출력한다. Fig. 3.16은 CNN-bidirectional LSTM (parallel) 기반 모델의 구조이다.

본 구조의 핵심은 CNN에 입력된 전기추진 선박 데이터의 특징을 추출하고 이전 시간대에 존재하는 데이터를 분류된 결과의 시계열 데이터의 예측 값과 bidirectional LSTM에서 학습된 은닉층 성능 향상과 데이터 길이와 층에 상관없이 이전 시간의 정보가 손실되지 않는 시계열 데이터예측을 합성하여 출력을 나타내는 것이다.

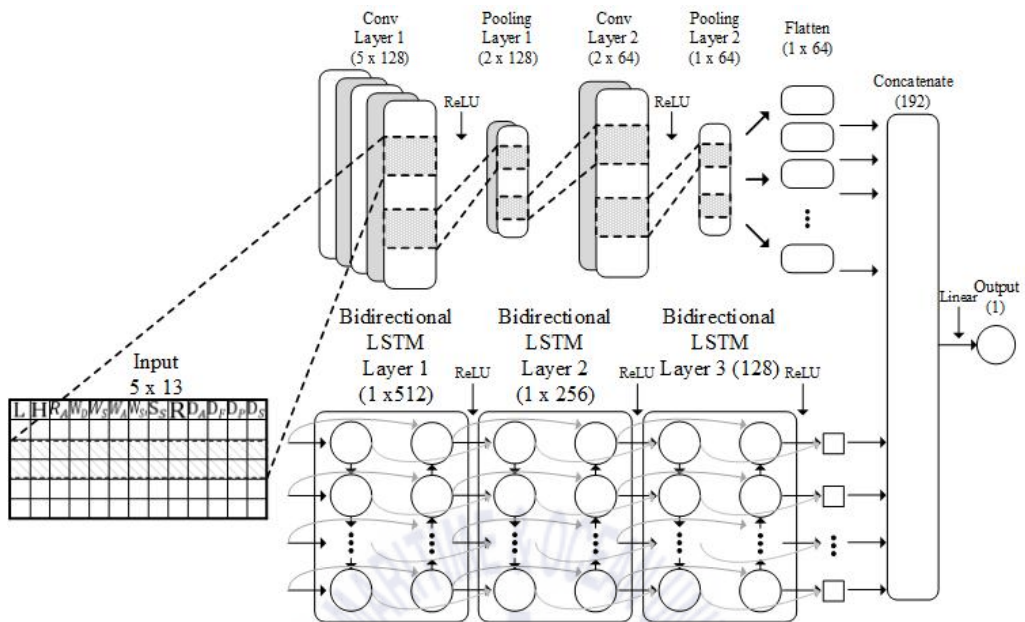


Fig. 3.16 Construction of CNN-bidirectional LSTM (parallel)

Table 3.10은 CNN-bidirectional LSTM (parallel) 모델 구현을 나타내었다.

Table 3.10 Code of CNN-bidirectional LSTM (parallel) model

```

xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

CNN = Conv1D(128, 14, padding='same', activation='relu', strides=1)(xInput)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = Conv1D(64, 7, padding='same', activation='relu', strides=1)(CNN)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = Flatten()(CNN)

xLstm_2 = Bidirectional(LSTM(512, return_sequences=True, activation='relu'))(xInput)
xLstm_2 = Bidirectional(LSTM(256, return_sequences=True, activation='relu'))(xLstm_2)
xLstm_2 = Bidirectional(LSTM(128, activation='relu'))(xLstm_2)

```



```

conc = concatenate([CNN, xLstm_2])

xOutput = Dense(1, activation='linear')(conc)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse',optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
final_df.to_csv(result_name, index=False)

```

3.2.7 LSTM auto encoder

LSTM auto encoder 기반 모델에서 입력된 시계열 데이터의 특징을 압축하고 압축된 특징을 다시 원복하는 방식으로 데이터를 예측한다. 다음 Fig 3.17은 LSTM auto encoder의 모델 구조를 나타내었다.

본 구조의 핵심은 LSTM auto encoder의 특징을 활용하여, CNN을 대체하는 것이다.

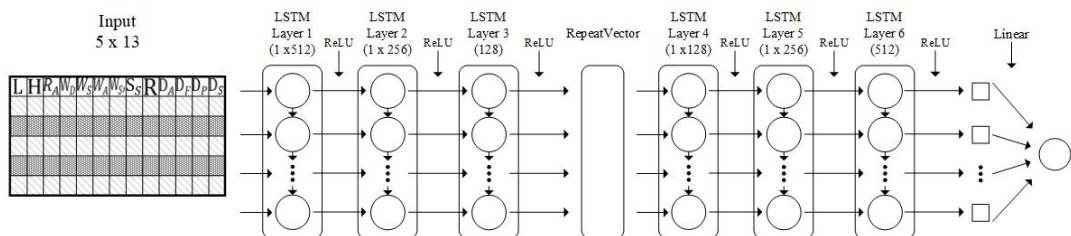


Fig. 3.17 Construction of LSTM auto encoder

다음 Table 3.11은 LSTM auto encoder 모델 구현을 나타내었다.

Table 3.11 Code of LSTM auto encoder model

```
xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

xLstm_2 = LSTM(512, return_sequences=True, activation='relu')(xInput)
xLstm_2 = LSTM(256, return_sequences=True, activation='relu')(xLstm_2)
xLstm_2 = LSTM(164, activation='relu')(xLstm_2)

xLstm_2 = RepeatVector(time_steps)(xLstm_2)

xLstm_2 = LSTM(164, return_sequences=True, activation='relu')(xLstm_2)
xLstm_2 = LSTM(256, return_sequences=True, activation='relu')(xLstm_2)
xLstm_2 = LSTM(512, activation='relu')(xLstm_2)

xOutput = Dense(1, activation='linear')(xLstm_2)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.0001)
model.compile(loss='mse', optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
final_df.to_csv(result_name, index=False)
```

3.2.8 ANN

ANN 기반 모델에서 각 계층의 노드에 대한 입력값으로 사용되는 출력을 생성한다. 학습된 가중치 및 활성화 함수를 이용하여 다음 예측 값을 출력한다. 다음 Fig. 3.18은 ANN 기반 모델의 구조이다[33-34].

ANN 모델은 가장 은닉 층이 하나인 딥러닝 모델을 의미한다. ANN은 시계열 형태의 전기추진 선박 데이터를 하나의 뉴럴 네트워크 레이어로 입력하고, 뉴럴 네트워크에서 연산된 결과 값을 1차원 형태로 변환한 다음 전기추진 선박 예측 부하 값을 출력하는 형태를 취한다. 본 구조의 핵심은 가장 기본적인 형태의 ANN을 활용하여, 전기추진 선박 데이터의 예측 성능을 확인하는 것이다.

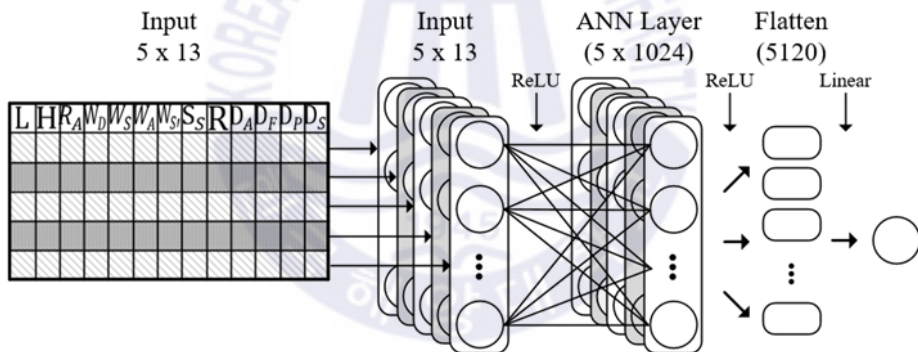


Fig. 3.18 Construction of ANN

Table 3.12는 ANN 모델 구현을 나타내었다.

Table 3.12 Code of ANN model

```
xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

DNN = Dense(9, activation='relu')(xInput)
DNN = Dense(1024, activation='relu')(DNN)
DNN = Flatten()(DNN)
```

```

xOutput = Dense(1, activation='linear')(DNN)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse', optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
final_df.to_csv(result_name, index=False)

```

3.2.9 DNN

DNN 기반 모델에서 각 계층의 노드에 대한 입력값으로 사용되는 출력을 생성한다. 학습된 가중치 및 활성화 함수를 이용하여 다음 예측값을 출력한다. 다음 Fig. 3.19는 DNN의 구조를 나타내었다[34-36].

본 구조의 핵심은 ANN의 단점을 극복하여 상대적으로 깊은 수준의 인공신경망인 DNN을 활용하여 시계열 데이터를 예측하고자 했다.

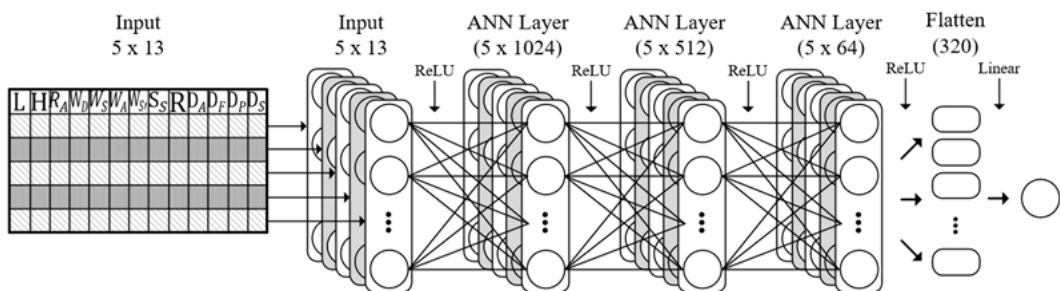


Fig. 3.19 Construction of DNN

Table 3.13은 DNN 모델 구현을 나타내었다.

Table 3.13 Code of DNN model

```
xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

DNN = Dense(14, activation='relu')(xInput)
DNN = Dense(1024, activation='relu')(DNN)
DNN = Dense(512, activation='relu')(DNN)
DNN = Dense(64, activation='relu')(DNN)
DNN = Flatten()(DNN)

xOutput = Dense(1, activation='linear')(DNN)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse', optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
final_df.to_csv(result_name, index=False)
```

3.2.10 CNN

CNN 기반 모델의 시계열 데이터 예측을 생성한다. 학습된 가중치 및 활성화 함수를 이용하여 다음 예측값을 출력한다. Fig. 3.20은 CNN의 구조를 나타내었다[37-38].

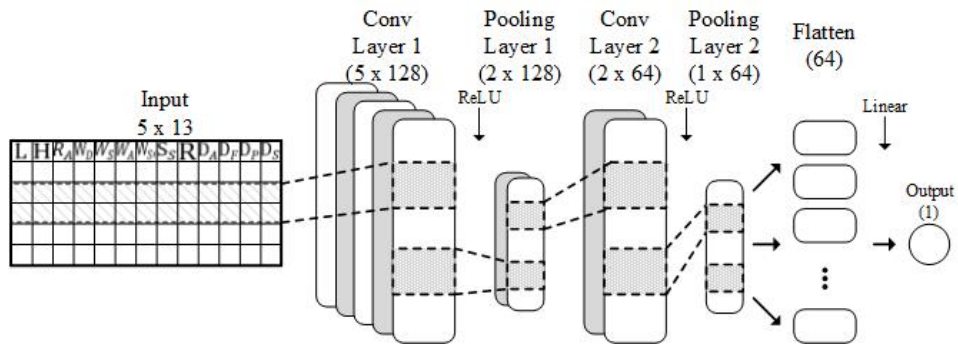


Fig. 3.20 Construction of CNN

본 구조의 핵심은 데이터의 특징 추출과 분류에 특화된 CNN 모델을 활용하여 전기추진 선박의 시계열 데이터를 예측하는 것이다.

Table 3.14는 CNN 모델 구현을 나타내었다.

Table 3.14 Code of CNN model

```

xInput = Input(shape=(X_train.shape[1], X_train.shape[2]))

CNN = Conv1D(128, 14, padding='same', activation='relu', strides=1)(xInput)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = Conv1D(64, 7, padding='same', activation='relu', strides=1)(CNN)
CNN = MaxPooling1D(pool_size=2, stride=2)(CNN)
CNN = Flatten()(CNN)

xOutput = Dense(1, activation='linear')(CNN)

model = Model(xInput, xOutput)

optimizer = optimizers.Adam(lr=0.001)
model.compile(loss='mse', optimizer=optimizer, metrics=[rmse])
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=100)

```

```
callHistory(model, X_train, y_train, 500)

test_predictions = model.predict(X_test).flatten()

final_df = pd.DataFrame({"realData" : [], "predData":[]})
final_df["realData"] = load_min_max_scaler.inverse_transform([y_test])[0, :]
final_df["predData"] = load_min_max_scaler.inverse_transform([test_predictions])[0, :]
final_df.to_csv(result_name, index=False)
```



제 4 장 제안 모델 평가

본 장에서는 3장에 기재한 전기추진 선박의 부하예측을 위한 모델들을 예측 모델의 실험 절차와 예측모델 평가 함수를 활용한다. 또한 모델의 정확도를 검증하기 위하여, 모델의 평가 기준을 정하였다. 이후 모델학습을 진행하였으며, 모델별 학습 결과를 평가하였다.

4.1 모델 평가 기준

본 논문에서는 모델간의 성능 평가는 RMSE를 사용한 방법을 사용하였다. 또한, 평가를 위하여 다음과 같은 평가 기준을 수립하였다.

- 학습과 평가에 사용되는 데이터는 모두 동일한 데이터를 사용하며 5번의 반복학습을 수행한다.
- 모델은 3장에서 제안한 구성을 사용하며, 조합에 따른 LSTM과 CNN의 하이퍼파라미터 구성은 모두 동일하게 구성하였으며 RMSE를 사용한다.
- RMSE를 평가할 때 기존에 연구되었던 관련 연구의 RMSE 값을 참고하여 모델의 정확도를 평가한다.
 - ✓ 전력부하의 변동성이 높은 주택 전력 부하 (0~11kW) 예측 모델의 경우 0.6의 RMSE 값을 도출하였다. [7]
 - ✓ 전력부하의 변동성이 낮은 태양전지 전력 부하 (0~25kW) 예측 모델의 경우 0.7의 RMSE 값을 도출하였다. [10]
- 모델간 성능 비교시 저장된 RMSE 데이터의 분포와 box plot을 활용한다.
- 개별모델 성능 평가시 반복학습으로 얻어진 RMSE 값을 활용하여 통계분석을 진행한다.
 - ✓ 평균값 : 성능을 나타낼 수 있는 대푯값으로 활용

- ✓ 표준 편차 : 산포도를 평가할 수 있는 지표로 활용
 - ✓ 최솟값 : 표준편차를 추정하는데 활용할 수 있는 값으로 활용
 - ✓ 4분위수 : 학습 결과 집합의 범위 활용
 - ✓ 최대값 : 표준편차를 추정하는데 활용할 수 있는 값으로 활용
- 학습에 사용된 데이터는 숫자형 데이터 이므로, 분류에 활용되는 precision 또는 recall과 같은 평가 함수는 고려하지 않으며, 예측 값이 실제 값과 얼마나 가까운지 고려한다.

4.2 실험 환경

모델의 학습을 위하여 사용된 하드웨어는 CPU는 Intel(R) Xeon(R) CPU @ 2.20GHz를 사용하였으며, 12GB RAM, Tesla P100-PCIE-16GB GPU를 사용하였다. 또한 python 3.6.9, tensorflow 1.15.2, keras 2.2.5를 사용하여 모델 학습을 수행하였다. 또한 공용 라이브러리들을 사용하였으며, numpy는 시계열 데이터 생성에 활용되었고 Pandas는 .csv 형태의 선박 데이터 읽기와 학습 결과 값을 dataframe 형태와 일반 파일 형태로 저장하는 것에 활용하였다. sklearn은 데이터의 normalization에 활용되었으며 마지막으로 matplotlib는 결과 시각화에 활용되었다. 장시간 GPU 학습을 위하여 Google 사의 Colab을 활용하였으며, 활용한 데이터는 2장에서 데이터 전처리를 거친 CSV 형태의 파일을 활용하였다. Table 4.1은 전기추진 선박의 부하 예측 모델 학습을 위한 실험 환경을 나타내었다.

Table 4.1 Environment information

HW	CPU : Intel(R) Xeon(R) CPU @ 2.30GHz Memory : 12 GB GPU : Tesla P100-PCIE-16GB
SW	OS : Ubuntu 18.04.3 LTS Tensorflow : 1.15.2 Keras : 2.2.5 Library : Numpy, Pandas, Sklearn, Matplotlib

4.3 실험 결과

4.3.1 LSTM

LSTM 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 Table 4.2와 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 1,792.1 이며, 가장 적은 RMSE 값은 1,709.8으로 모델이 평균적으로 동일한 성능을 내는 것을 확인할 수 있다.

Table 4.2 RMSE result of LSTM model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
1,758.6	1,713.0	1,709.8	1,792.1	1,765.0

Table 4.3은 LSTM 모델의 1차 실험부터 5차 실험 까지의 RMSE값 통계를 나타내었다. 평균값은 1,747.7이 나왔으며, 표준편차는 35.5으로 모델의 재현율이 높은 것을 확인할 수 있었다. 최솟값은 1,709.8, 제1 사분 위수(Q1)은 1,713, 제2 사분 위수(Q2)는 1,758.6, 제3 사분 위수 (Q3)는 1,765.1로 사분 위간 범위는 1,713~1,765.1로 확인되었다. 마지막으로 최대값은 1,792.1로 확인되었으며 모델의 성능이 상대적으로 낮은 것을 확인할 수 있다.

Table 4.3 RMSE' s statistical information of LSTM model experiment results

평균값	표준 편차	최솟값	4분 위수			최대값
			25%	50%	75%	
1,747.7	35.5	1,709.8	1,713	1,758.6	1,765.1	1,792.1

Fig. 4.1은 LSTM 모델의 실험 결과를 나타낸다.

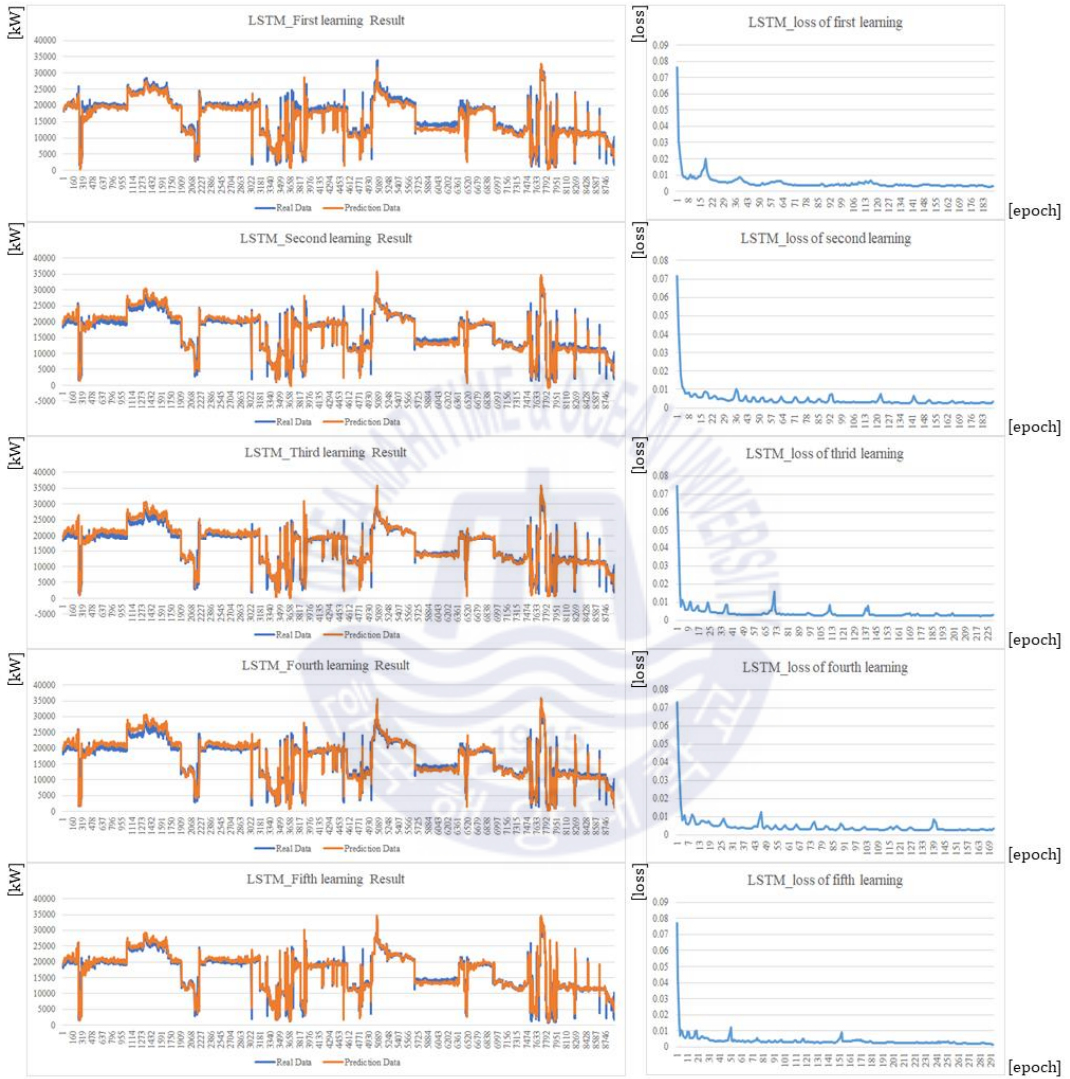


Fig. 4.1 Result of LSTM model experiment

4.3.2 bidirectional LSTM

bidirectional LSTM 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 Table 4.4와 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 1,748.8 이며, 가장 적은 RMSE 값은 1,602.8으로 모델의 성능 평균 분포가 상대적으로 큰 것을 확인할 수 있다.

Table 4.4 RMSE result of bidirectional LSTM model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
1,731.7	1,732.7	1,748.8	1,704.0	1,602.8

Table 4.5는 bidirectional LSTM 모델의 1차 실험부터 5차 실험까지의 RMSE값 통계를 나타내었다. 평균값은 1,704이 나왔으며, 표준편차는 58.8으로 모델의 재현율이 상대적으로 낮은 것을 확인할 수 있었다. 최솟값은 1,602.8 제1 사분위수(Q1)은 1,704, 제2 사분위수(Q2)는 1,731.1, 제3 사분위수(Q3)는 1,732.7로 사분위간 범위는 1,704~1,732.7로 확인되었다. 마지막으로 최대값은 1,748.8로 확인되었으며 모델의 성능이 상대적으로 낮은 것을 확인할 수 있다.

Table 4.5 RMSE's statistical information of bidirectional LSTM model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
1,704	58.8	1,602.8	1,704	1,731.7	1,732.7	1,748.8

다음 Fig. 4.2는 bidirectional LSTM 모델의 실험 결과를 나타낸다.

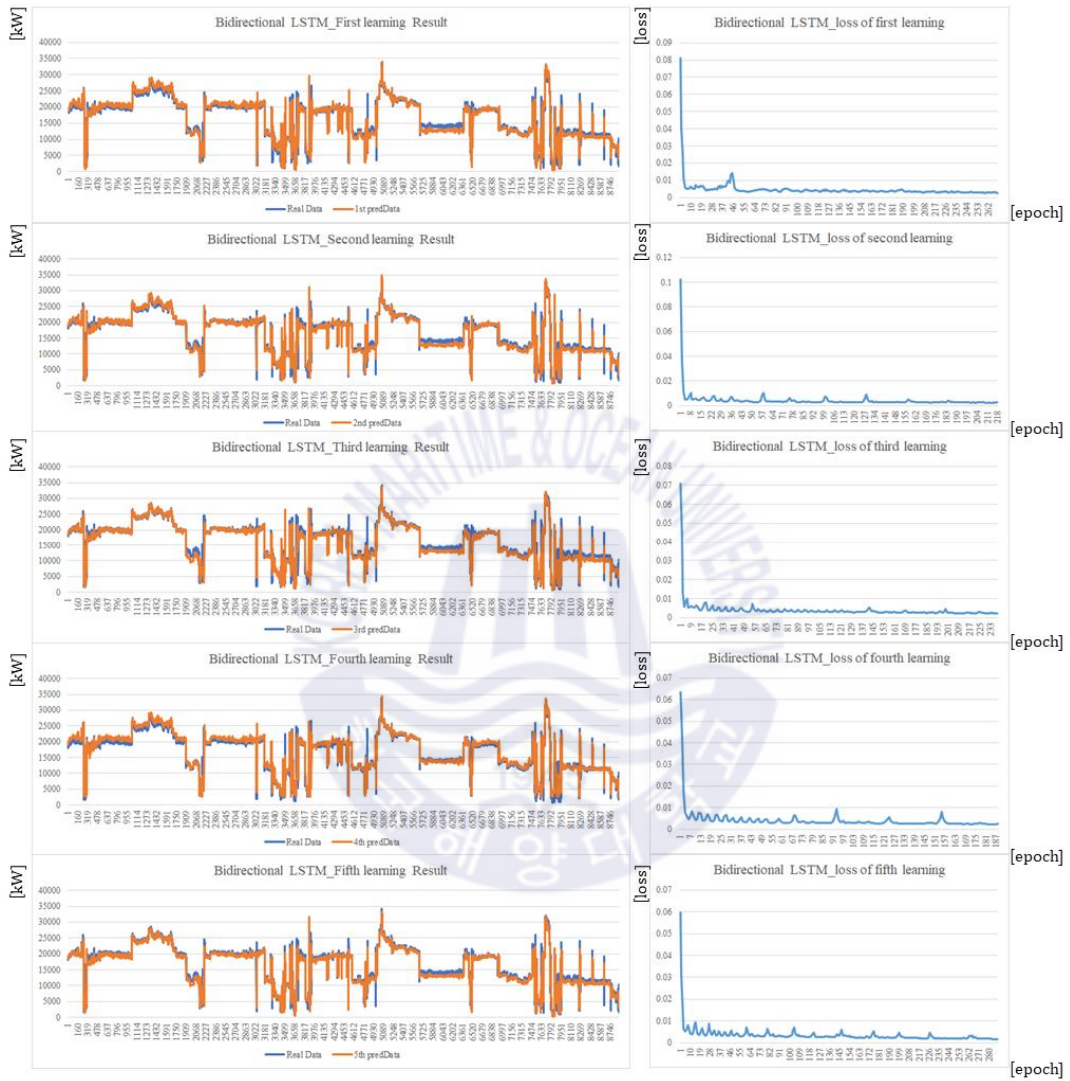


Fig. 4.2 Result of bidirectional LSTM model experiment

4.3.3 CNN-LSTM (direct)

CNN-LSTM (direct) 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 Table 4.6과 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 1,869.2이며, 가장 적은 RMSE 값은 1,523.6으로 모델의 성능 평균 분포가 큰 것을 확인할 수 있다.

Table 4.6 RMSE result of CNN-LSTM (direct) model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
1,752	1,679.3	1,869.2	1,806.6	1,523.6

Table 4.7은 CNN-LSTM (direct) 모델의 1차 실험부터 5차 실험까지의 RMSE값 통계를 나타내었다. 평균값은 1,726.1이 나왔으며, 표준편차는 133.0으로 모델의 재현율이 낮은 것을 확인할 수 있었다. 최솟값은 1,523.6 제1사분위수(Q1)은 1,679.3, 제2사분위수(Q2)는 1,752, 제3사분위수(Q3)는 1,806.6로 사분위간 범위는 1,679.3~1,806.6로 확인되었다. 마지막으로 최대값은 1,869.2로 확인되었으며 모델의 성능이 상대적으로 낮은 것을 확인할 수 있다.

Table 4.7 RMSE's statistical information of CNN-LSTM (direct) model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
1,726.1	133.0	1,523.6	1,679.3	1,752	1,806.6	1,869.2

다음 Fig. 4.3은 CNN-LSTM (direct) 모델의 실험 결과를 나타낸다.

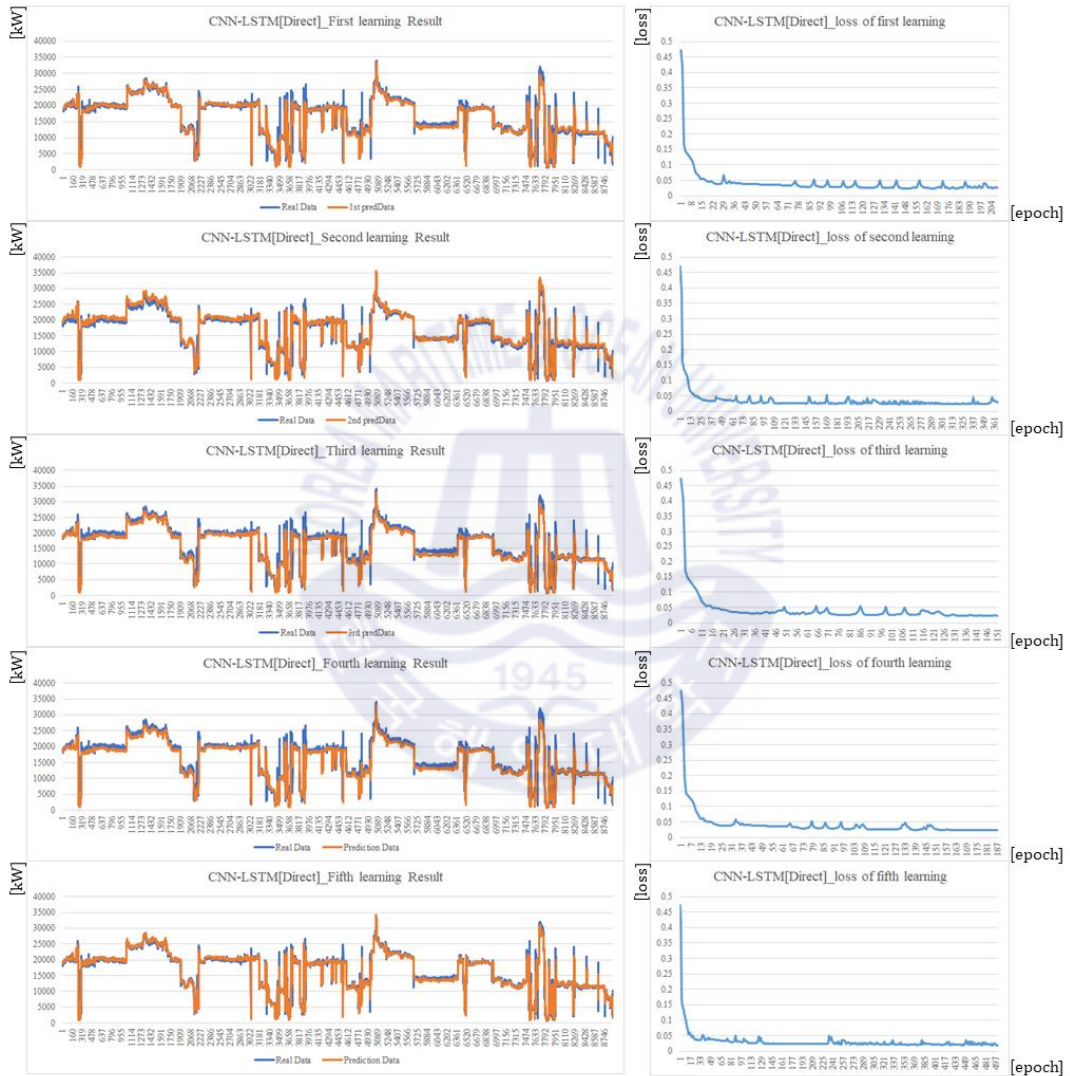


Fig. 4.3 Result of CNN-LSTM (direct) model experiment

4.3.4 CNN-bidirectional LSTM (direct)

CNN-bidirectional LSTM (direct) 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 다음 Table 4.8과 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 1,670.7이며, 가장 적은 RMSE 값은 1,501.7으로 모델의 성능 평균 분포가 낮은 것을 확인할 수 있다.

Table 4.8 RMSE result of CNN-bidirectional LSTM (direct) model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
1,670.7	1,534.4	1,501.7	1,536.3	1,520.2

Table 4.9은 CNN-bidirectional LSTM (direct) 모델의 1차 실험부터 5차 실험까지의 RMSE값 통계를 나타내었다. 평균값은 1,552.7이 나왔으며, 표준편차는 67.4으로 모델의 재현율이 상대적으로 높은 것을 확인할 수 있었다. 최솟값은 1,501.7 제1 사분 위수(Q1)은 1,520.2, 제2 사분 위수(Q2)는 1,534.4, 제3 사분 위수 (Q3)는 1,536.3로 사분위간 범위는 1,520.2~1,536.3로 확인되었다. 마지막으로 최대값은 1,670.7로 확인되었으며 모델의 성능이 상대적으로 높은 것을 확인할 수 있다

Table 4.9 RMSE' s statistical information of CNN-bidirectional LSTM (direct) model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
1,552.7	67.4	1,501.7	1,520.2	1,534.4	1,536.3	1,670.7

다음 Fig. 4.4는 CNN-bidirectional LSTM (direct) 모델의 실험 결과를 나타낸다.

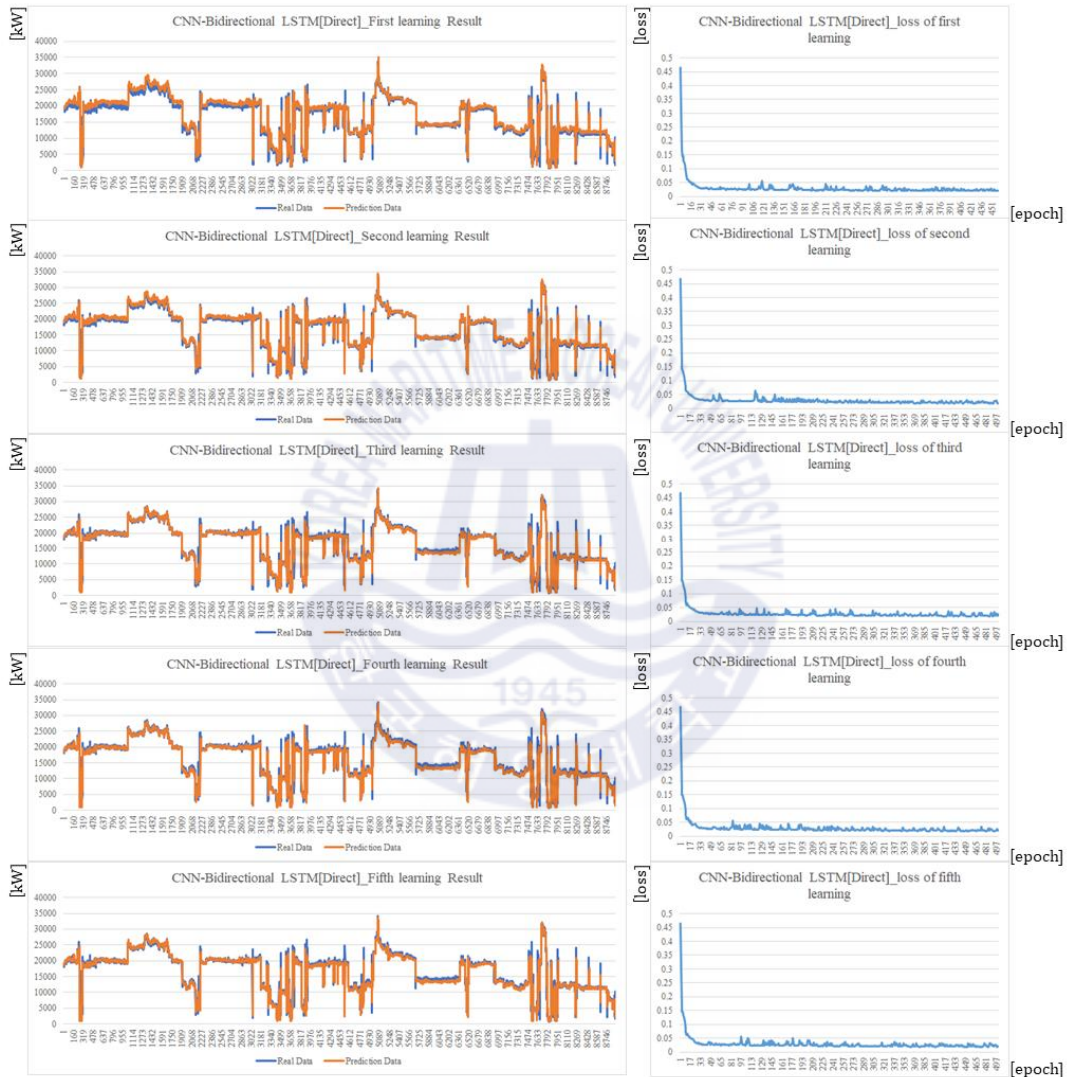


Fig. 4.4 Result of CNN-bidirectional LSTM (direct) model experiment

4.3.5 CNN-LSTM (parallel)

CNN-LSTM (parallel) 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 Table 4.10과 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 1,461이며, 가장 적은 RMSE 값은 1,552으로 모델의 성능 평균 분포가 낮은 것을 확인할 수 있다.

Table 4.10 RMSE result of CNN-LSTM (parallel) model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
1,461	1,552	1,476.6	1,512.9	1,507.6

Table 4.11은 CNN-LSTM (parallel) 모델의 1차 실험부터 5차 실험까지의 RMSE값 통계를 나타내었다. 평균값은 1,502이 나왔으며, 표준편차는 35.3으로 모델의 재현율이 높은 것을 확인할 수 있었다. 최솟값은 1,461 제1 사분위수(Q1)은 1,476.6, 제2 사분위수(Q2)는 1,507.6, 제3 사분위수(Q3)는 1,512.9로 사분위간 범위는 1,476.6~1,512.9로 확인되었다. 마지막으로 최대값은 1,552로 확인되었으며 모델의 성능이 높은 것을 확인할 수 있다

Table 4.11 RMSE's statistical information of CNN-LSTM (parallel) model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
1,502	35.3	1,461	1,476.6	1,507.6	1,512.9	1,552

다음 Fig. 4.5는 CNN-LSTM (parallel) 모델의 실험 결과를 나타낸다.

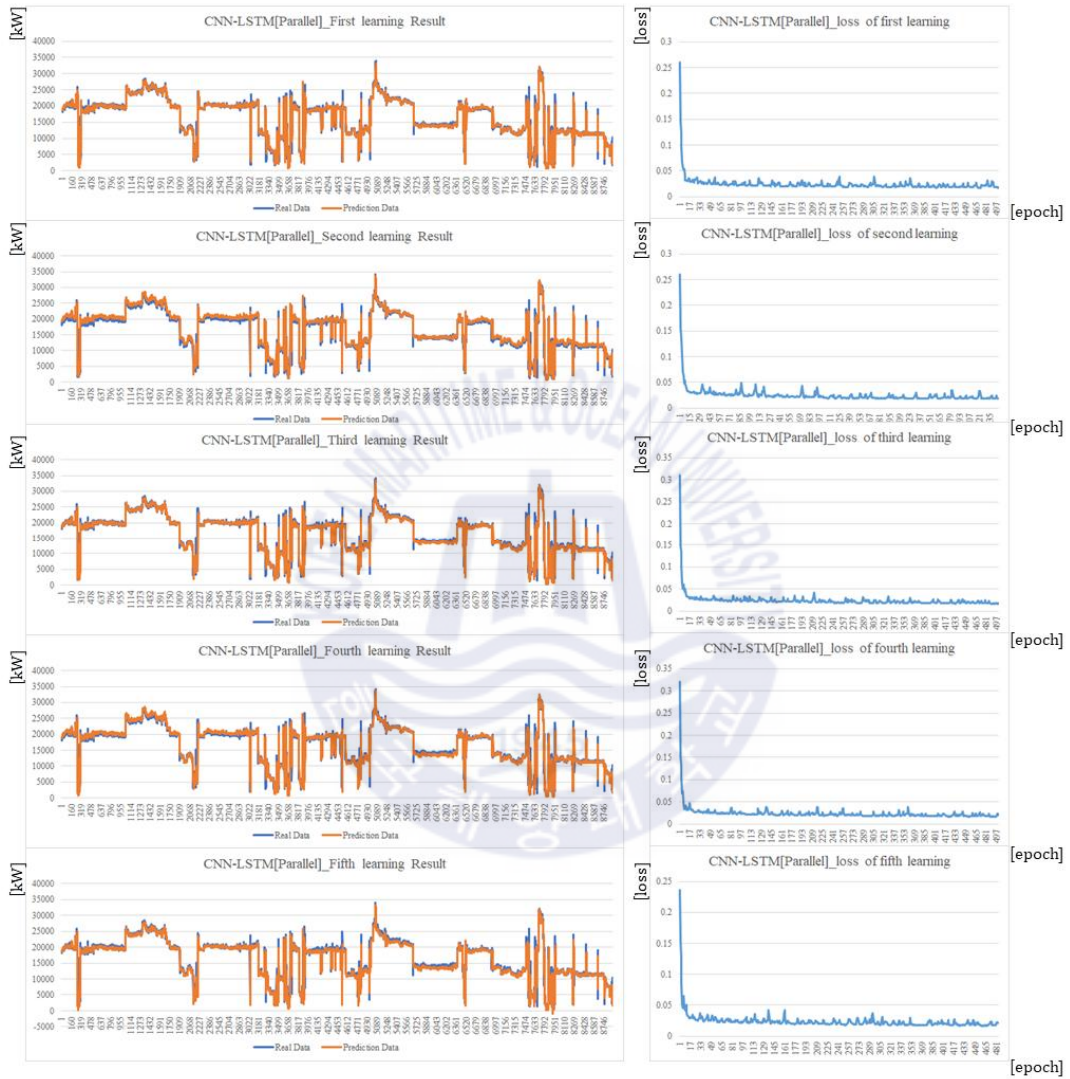


Fig. 4.5 Result of CNN-LSTM (parallel) model experiment

4.3.6 CNN-bidirectional LSTM (parallel)

CNN-bidirectional LSTM (parallel) 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 Table 4.12와 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 1,484이며, 가장 적은 RMSE 값은 1,579.1으로 모델의 성능 평균 분포가 낮은 것을 확인할 수 있다.

Table 4.12 RMSE result of CNN-LSTM (parallel) model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
1,579.1	1,514.4	1,501.9	1,509.1	1,484

Table 4.13은 CNN-bidirectional LSTM (parallel) 모델의 1차 실험부터 5차 실험 까지의 RMSE값 통계를 나타내었다. 평균값은 1,517.7이 나왔으며, 표준편차는 36.2으로 모델의 재현율이 높은 것을 확인할 수 있었다. 최솟값은 1,484 제1 사분 위수(Q1)은 1,501.9, 제2 사분 위수(Q2)는 1,509.1, 제3 사분 위수(Q3)는 1,514.4로 사분위간 범위는 1,501.9~1,514.4로 확인되었다. 마지막으로 최대값은 1,579.1로 확인되었으며 모델의 성능이 보통임을 확인할 수 있다.

Table 4.13 RMSE' s statistical information of CNN-LSTM (parallel) model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
1,517.7	36.2	1,484	1,501.9	1,509.1	1,514.4	1,579.1

다음 Fig. 4.6은 CNN-bidirectional LSTM (parallel) 모델의 실험 결과를 나타낸다.

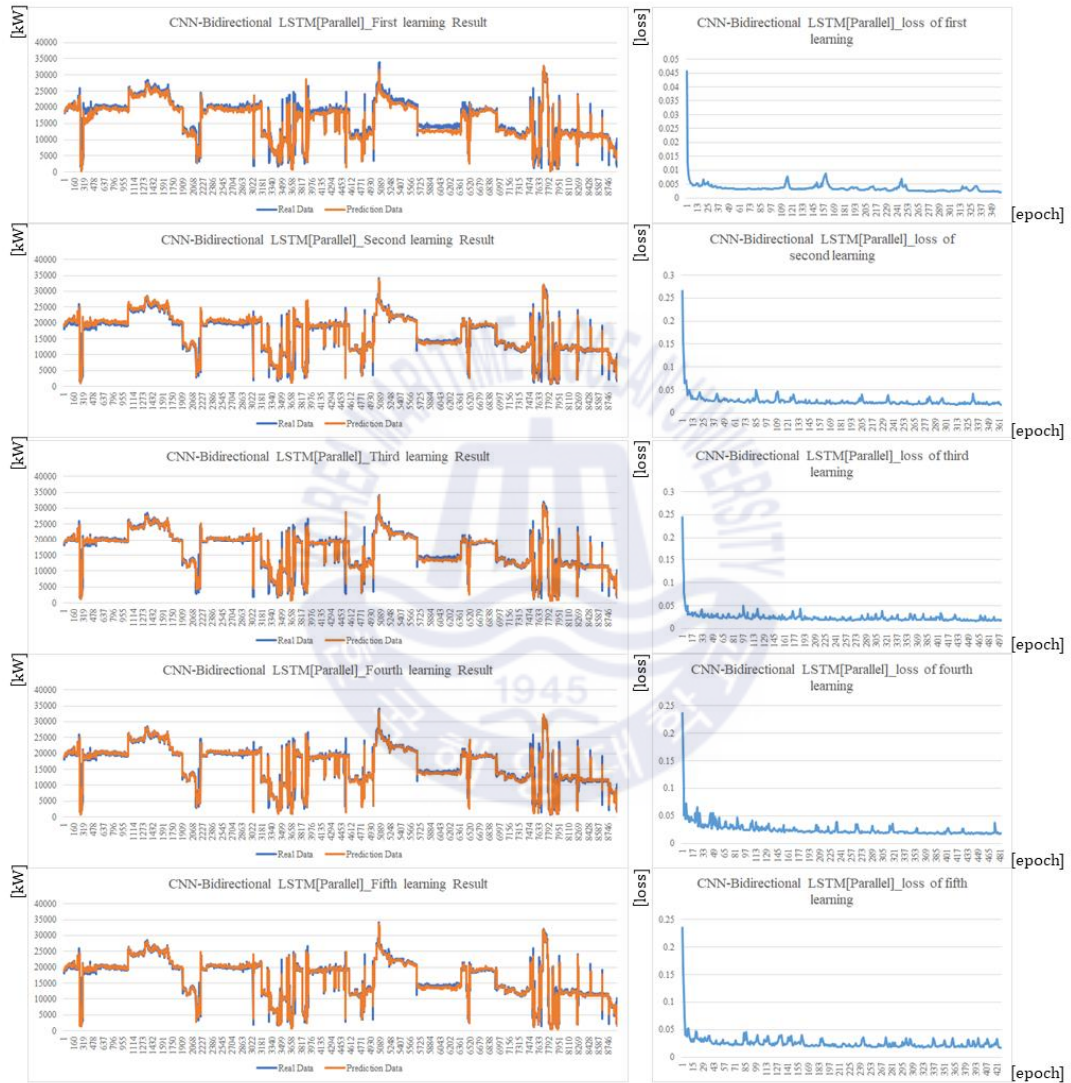


Fig. 4.6 Result of CNN-bidirectional LSTM (parallel) model experiment

4.3.7 LSTM auto encoder

LSTM auto encoder 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 다음 Table 4.14와 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 1,728.3이며, 가장 작은 RMSE 값은 1,656.2으로 모델의 성능 평균 분포가 낮은 것을 확인할 수 있다.

Table 4.14 RMSE result of LSTM auto encoder model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
1,710.7	1,660.6	1,728.3	1,677.5	1,656.2

Table 4.15는 LSTM auto encoder 모델의 1차 실험부터 5차 실험까지의 RMSE값 통계를 나타내었다. 평균값은 1,686.7이 나왔으며, 표준편차는 31.6으로 모델의 재현율이 높은 것을 확인할 수 있었다. 최솟값은 1,656.2 제1 사분위수(Q1)은 1,660.6, 제2 사분위수(Q2)는 1,677.5, 제3 사분위수(Q3)는 1,710.7로 사분위간 범위는 1,660.6~1,710.7로 확인되었다. 마지막으로 최대값은 1,728.3로 확인되었으며 모델의 성능이 상대적으로 낮은 것을 확인할 수 있다.

Table 4.15 RMSE's statistical information of LSTM auto encoder model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
1,686.7	31.6	1,656.2	1,660.6	1,677.5	1,710.7	1,728.3

다음 Fig. 4.7은 LSTM auto encoder 모델의 실험 결과를 나타낸다.

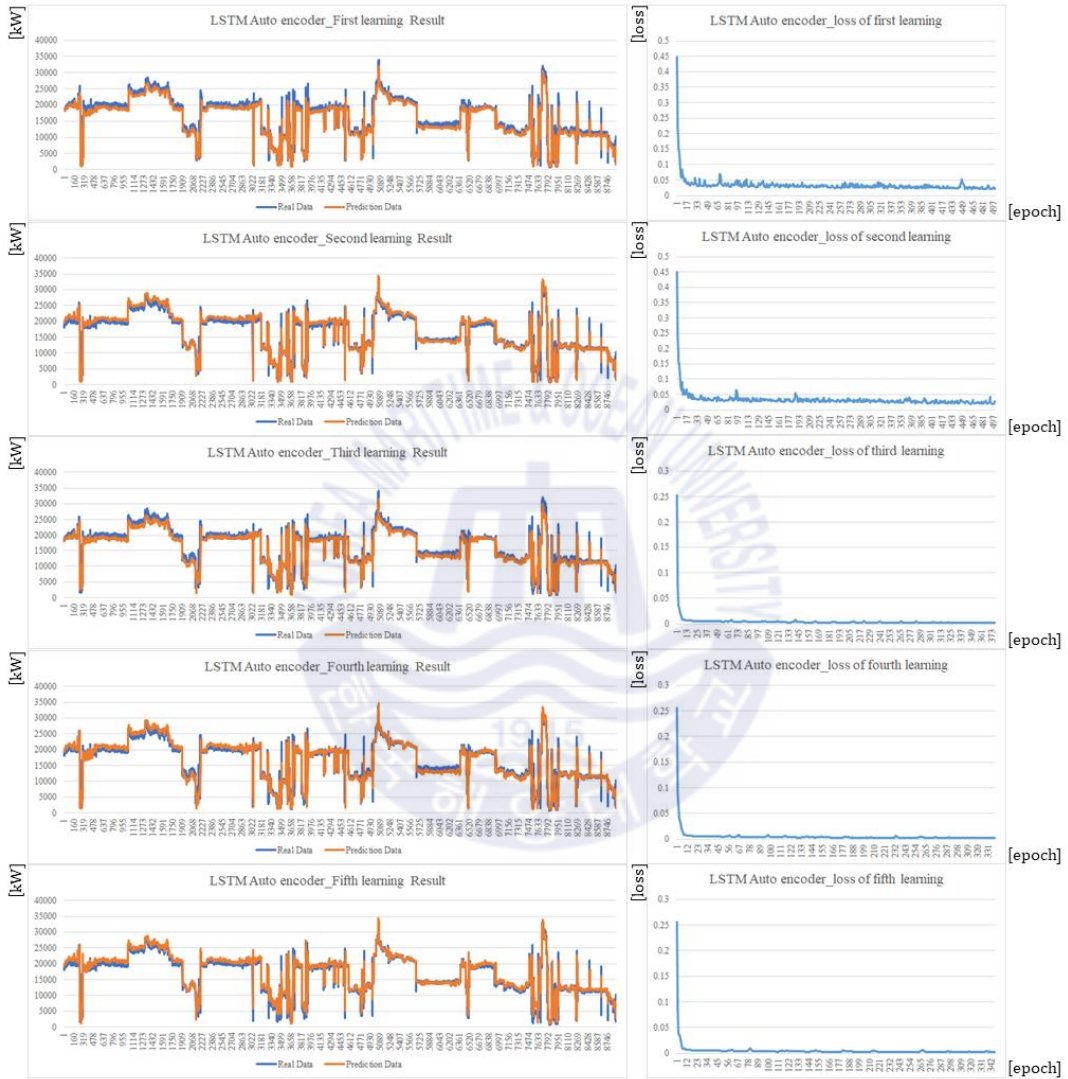


Fig. 4.7 Result of LSTM auto encoder model experiment

4.3.8 ANN

ANN 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 다음 Table 4.16과 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 3,975.6이며, 가장 적은 RMSE 값은 1,618.7으로 모델의 성능 평균 분포가 낮은 것을 확인할 수 있다.

Table 4.16 RMSE result of ANN model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
2,077.3	2,089.4	3,975.6	1,618.7	2,519.3

Table 4.17은 ANN 모델의 1차 실험부터 5차 실험까지의 RMSE값 통계를 나타내었다. 평균값은 2,456이 나왔으며, 표준편차는 907.2으로 모델의 재현율이 매우 낮은 것을 확인할 수 있었다. 최솟값은 1,618.7 제1 사분 위수(Q1)은 2,077.3, 제2 사분 위수(Q2)는 2,089.4, 제3 사분 위수 (Q3)는 2,519.3로 사분위간 범위는 2,077.3~2,519.3로 확인되었다. 마지막으로 최대값은 3,975.6로 확인되었으며 모델의 성능이 낮은 것을 확인할 수 있다.

Table 4.17 RMSE' s statistical information of ANN model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
2,456	907.2	1,618.7	2,077.3	2,089.4	2,519.3	3,975.6

다음 Fig. 4.8은 ANN 모델의 실험 결과를 나타낸다.

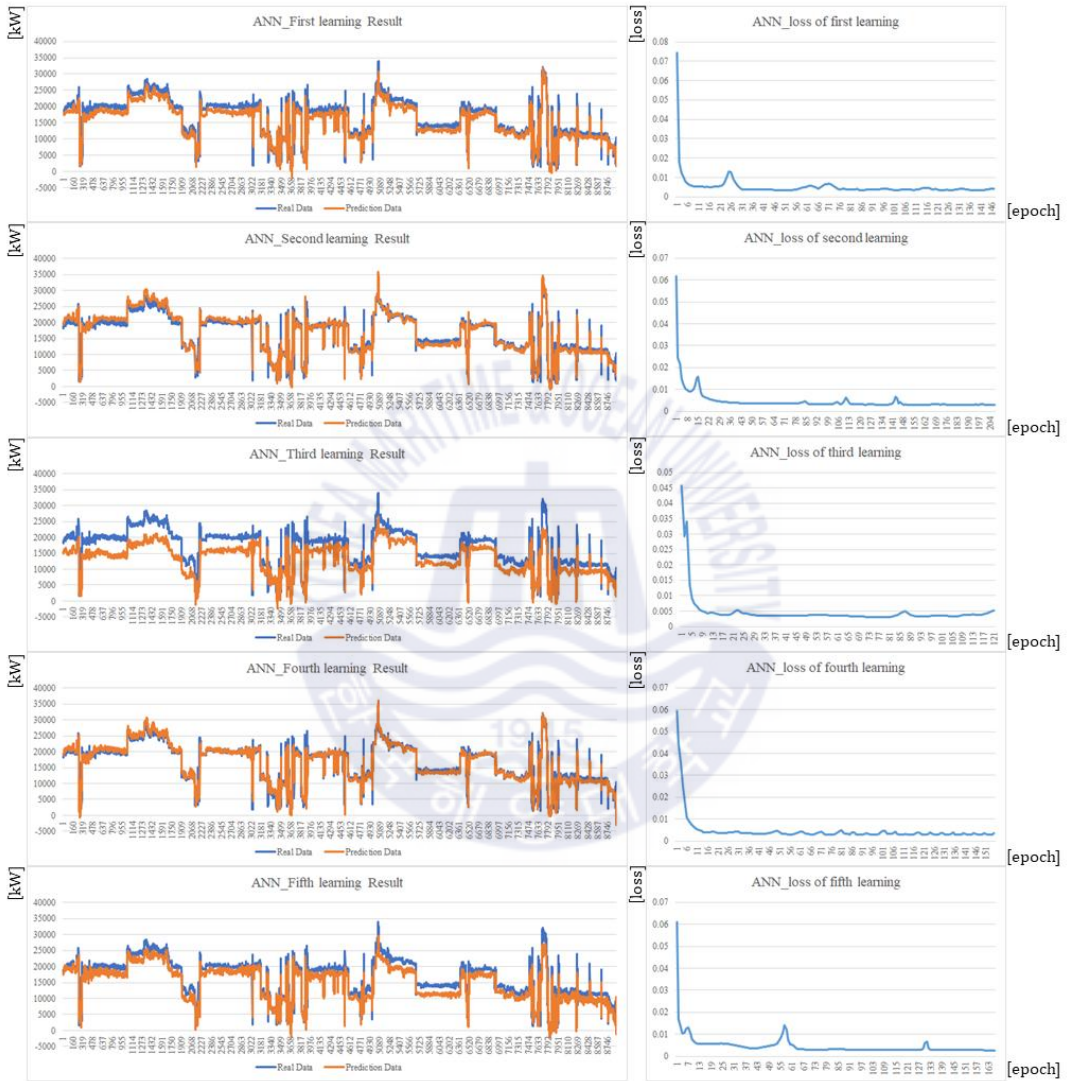


Fig. 4.8 Result of ANN model experiment

4.3.9 DNN

ANN 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 다음 Table 4.18과 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 2,427.5이며, 가장 적은 RMSE 값은 1,858.3으로 모델의 성능 평균 분포가 낮은 것을 확인할 수 있다.

Table 4.18 RMSE result of DNN model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
2,427.5	1,858.3	1,944.3	1,879.9	1,863

Table 4.19은 ANN 모델의 1차 실험부터 5차 실험까지의 RMSE값 통계를 나타내었다. 평균값은 1,994.6이 나왔으며, 표준편차는 244.4으로 모델의 재현율이 낮은 것을 확인할 수 있었다. 최솟값은 1,858.3 제1 사분 위수(Q1)은 1,863, 제2 사분 위수(Q2)는 1,879.9, 제3 사분 위수 (Q3)는 1,944.3로 사분위간 범위는 1,863~1,944.3로 확인되었다. 마지막으로 최대값은 2,427.5로 확인되었으며 모델의 성능이 낮은 것을 확인할 수 있다.

Table 4.19 RMSE' s statistical information of DNN model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
1,994.6	244.4	1,858.3	1,863	1,879.9	1,944.3	2,427.5

다음 Fig. 4.9는 DNN 모델의 실험 결과를 나타낸다.

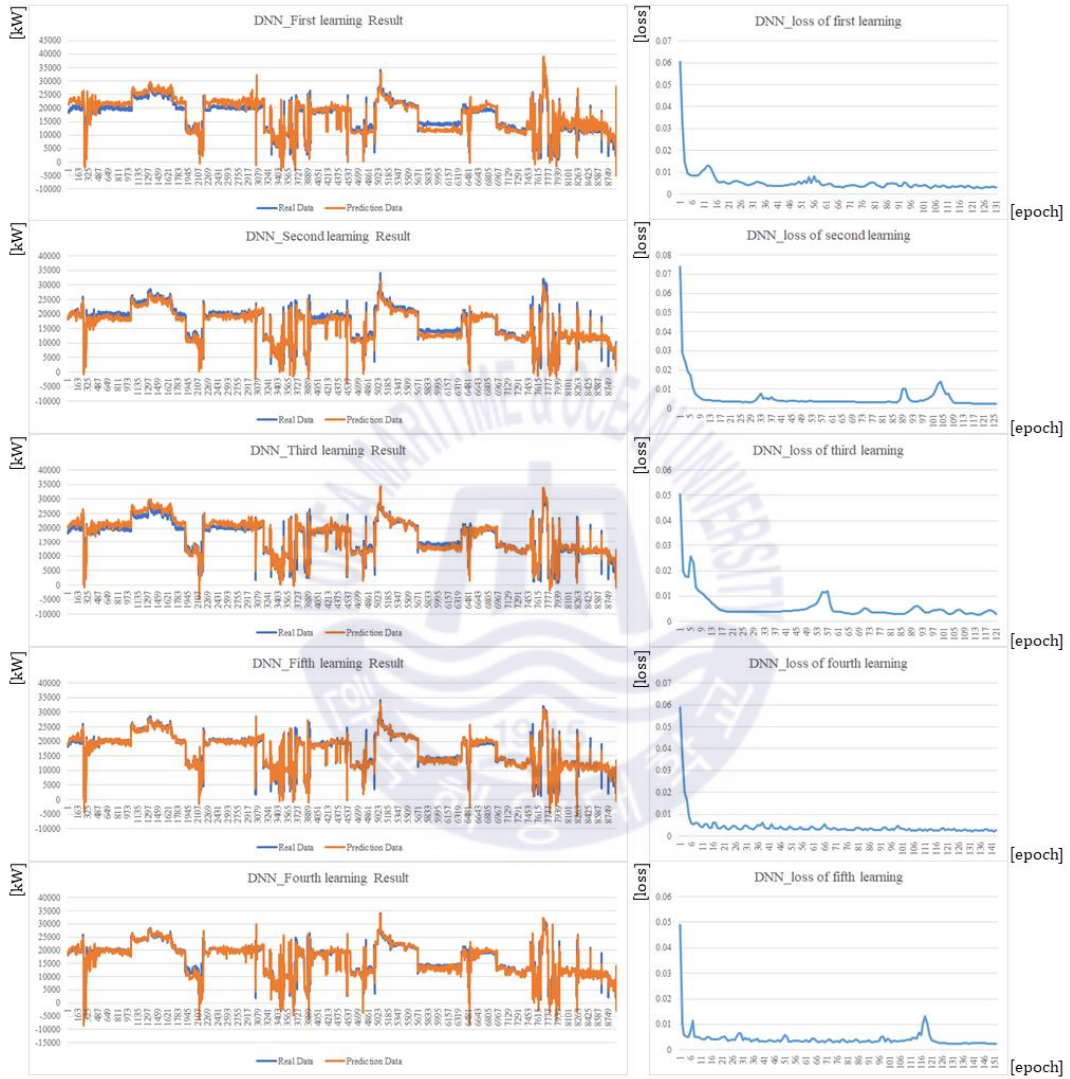


Fig. 4.9 Result of DNN model experiment

4.3.10 CNN

CNN 모델을 활용한 전기추진 선박 부하 예측 실험 결과는 다음 Table 4.18과 같다. 실험 횟수별 RMSE 값을 활용하여, 모델의 성능을 비교하였다. 가장 큰 RMSE 값은 2,191.9이며, 가장 적은 RMSE 값은 1,691.6으로 모델의 성능 평균 분포가 낮은 것을 확인할 수 있다.

Table 4.20 RMSE result of CNN model experiment

1차 실험	2차 실험	3차 실험	4차 실험	5차 실험
1,726.1	2,007.7	1,991.5	2,191.9	1,691.6

Table 4.19은 CNN 모델의 1차 실험부터 5차 실험까지의 RMSE값 통계를 나타내었다. 평균값은 1,921.8이 나왔으며, 표준편차는 210.1으로 모델의 재현율이 낮은 것을 확인할 수 있었다. 최솟값은 1,691.6 제1 사분 위수(Q1)은 1,726.1, 제2 사분 위수(Q2)는 1,991.5 제3 사분 위수 (Q3)는 2,007.7로 사분위간 범위는 1,726.1~2,007.7로 확인되었다. 마지막으로 최대값은 2,191.9로 확인되었으며 모델의 성능이 낮은 것을 확인할 수 있다.

Table 4.21 RMSE' s statistical information of CNN model experiment results

평균값	표준편차	최솟값	4분위수			최대값
			25%	50%	75%	
1,921.8	210.1	1,691.6	1,726.1	1,991.5	2,007.7	2,191.9

다음 Fig. 4.10은 CNN 모델의 실험 결과를 나타낸다.

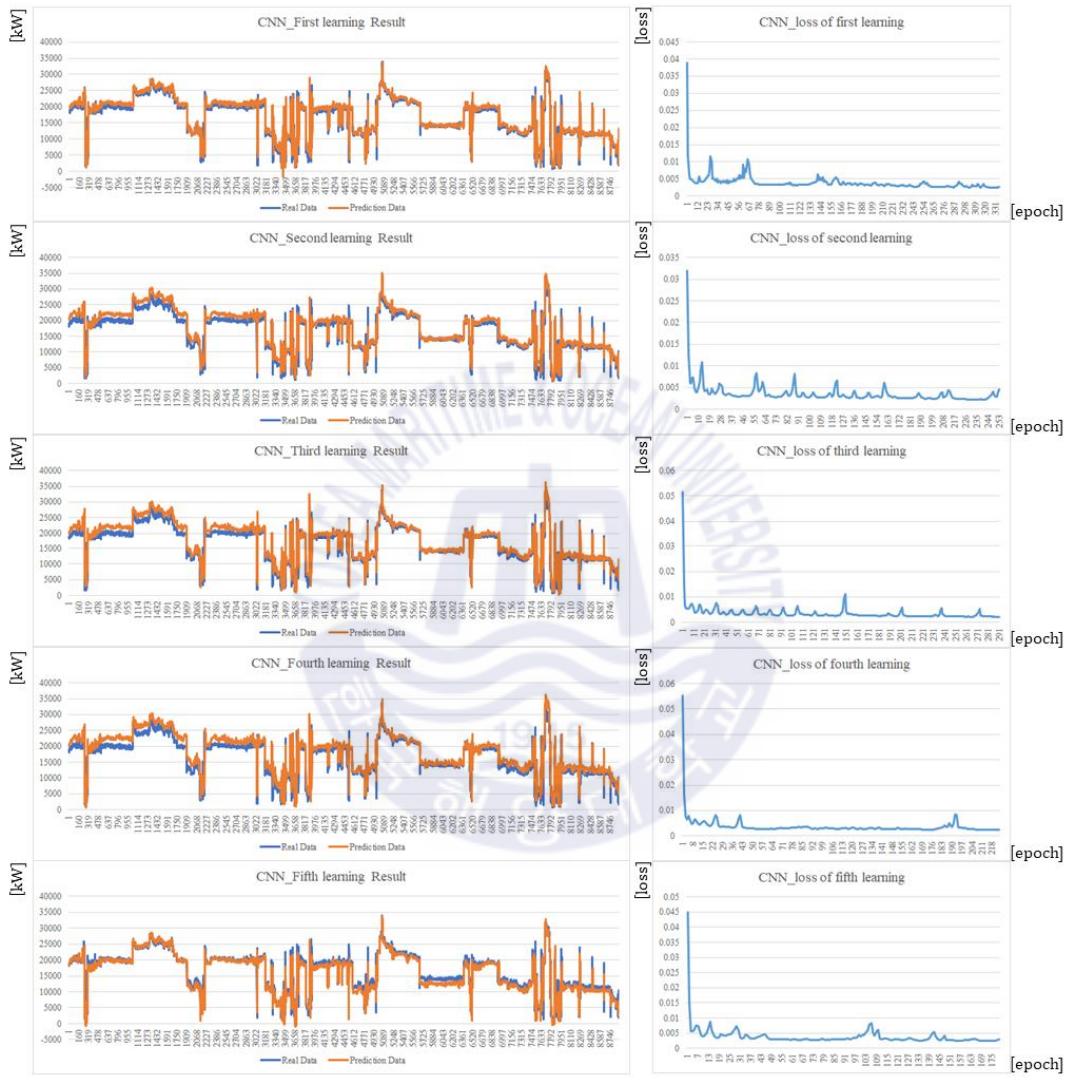


Fig. 4.10 Result of CNN model experiment

제 5 장 모델 비교 분석 및 고찰

본 장에서는 4장에 기재한 전기추진 선박의 부하예측 모델별 학습 평가 결과를 활용하여, 전체 모델의 통계적 결과를 비교하고, 5번의 반복 학습에 따른 모델별 RMSE 값의 변화를 라인차트를 활용하여 분석한다. 또한 box plot을 활용하여 각 모델별 출력된 outlier와 사분위수의 범위를 시각적으로 분석하였다.

5.1 모델 비교 분석

모델별 실험 결과를 기준으로 모델별 비교 분석을 진행한다. 모델별 비교 분석은 반복 학습 결과에 따른 RMSE 값의 통계와 box plot의 분석을 활용한다. 다음 Table 5.1은 전체 모델의 통계를 나타낸다.

Table 5.1 RMSE' s statistical information of experiment results

모델	평균값	표준편차	최솟값	4분위수			최대값
				25%	50%	75%	
LSTM	1747.7	35.5	1709.8	1713	1758.6	1765.1	1792.1
Bidrectional LSTM	1,704	58.8	1,602.8	1,704	1,731.7	1,732.7	1,748.8
CNN-LSTM (direct)	1,726.1	133.0	1,523.6	1,679.3	1,752	1,806.6	1,869.2
CNN-Bidirectional LSTM (direct)	1,552.7	67.4	1,501.7	1,520.2	1,534.4	1,536.3	1,670.7
C N N - L S T M (parallel)	<u>1,502</u>	35.3	<u>1,461</u>	1,476.6	1,507.6	1,512.9	<u>1,552</u>
CNN-Bidirectional LSTM (parallel)	1,517.7	36.2	1,484	1,501.9	1,509.1	1,514.4	1,579.1
LSTM Auto encoder	1,686.7	<u>31.6</u>	1,656.2	1,660.6	1,677.5	1,710.7	1,728.3

ANN	2,456	907.2	1,618.7	2,077.3	2,089.4	2,519.3	3,975.6
DNN	1,994.6	244.4	1,858.3	1,863	1,879.9	1,944.3	2,427.5
CNN	1,921.8	210.1	1,691.6	1,726.1	1,991.5	2,007.7	2,191.9

위 Table 5.1에서 확인할 수 있듯이 5번의 반복 실험 결과를 확인하였을 때 평균값을 확인할 경우, 범용적인 ANN, DNN, CNN 구조는 모델의 예측 정확도가 떨어짐을 알 수 있으며, CNN-LSTM (parallel)의 성능이 1,502으로 좋음을 확인할 수 있다. 그리고 표준편차는 LSTM Auto encoder이 가장 낮음을 확인할 수 있었다.

또한 전체적으로, CNN과 LSTM의 결합 모델의 성능이 가장 높음을 알 수 있다. 특히 CNN-LSTM (direct) 결합보다, CNN-LSTM 결합 모델의 경우 병렬 형태로 구성을 하는 것이 전기추진 선박의 부하를 가장 잘 예측 하는 것으로 확인할 수 있다. Fig. 5.1은 모델간 오차율 비교를 나타내었다.

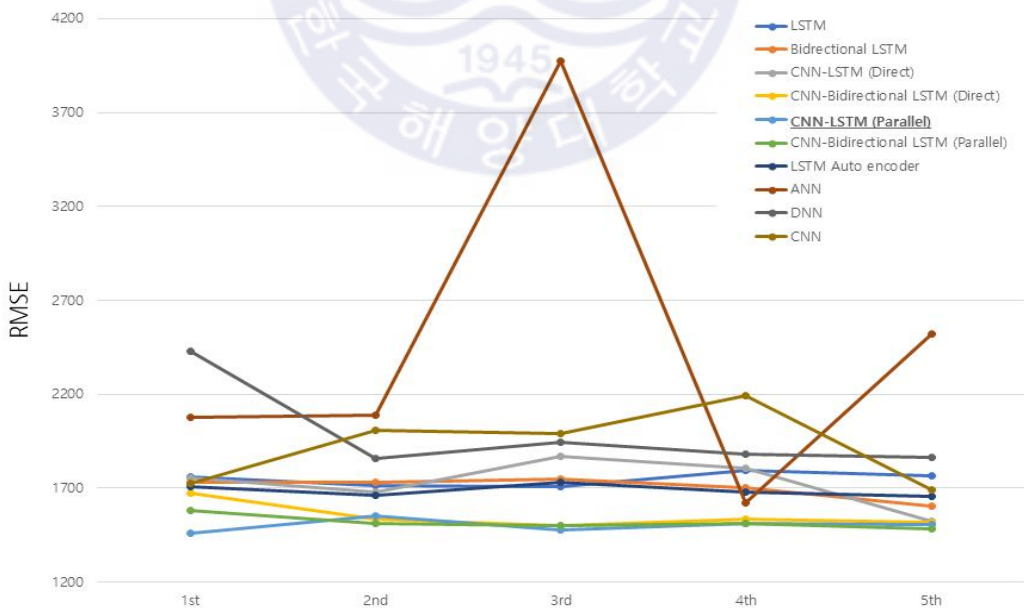


Fig. 5.1 Comparison of model's RMSE value

Fig. 5.1의 분석결과 다음과 같은 결론을 얻을 수 있었다.

- ANN, DNN, CNN 등의 모델은 평균적으로 오차율이 높으며, 다른 모델과 비교하였을 때 분산이 상대적으로 큰 것을 알 수 있다.
- LSTM과 bidirectional LSTM의 경우 ANN, DNN, CNN 보다는 성능이 높으나, 나머지 다른 모델들과 비교하였을 때 성능이 낮은 것을 확인할 수 있다.
- LSTM auto encoder의 경우 일정한 수준의 성능을 보장하는 것을 알 수 있으나 모델의 성능 비교시 LSTM, bidirectional LSTM 보다는 우수하나, CNN-LSTM 결합 모델보다는 성능이 낮음을 알 수 있다.
- CNN과 LSTM 결합 모델이 성능이 좋은 것으로 확인된다. 그러나 ANN의 오차 최대 값이 크기 때문에 정확한 모델의 비교가 어려운 특징이 있다.

다음 Fig. 5.2는 CNN과 LSTM 결합 모델의 오차율 비교를 나타내었다.



Fig. 5.2 Comparison of CNN-LSTM combine model's RMSE value

Fig. 5.2 분석결과 다음과 같은 결과를 얻을 수 있다.

- CNN-LSTM(direct) 모델은 학습 별 모델의 RMSE의 변동이 크다.
- CNN-Bidirectional LSTM (direct) 모델은 1번 테스트 와 3번 테스트의 RMSE 오차가 크게 발생한다.
- CNN-LSTM (parallel) 모델은 학습 별 모델의 오차가 가장 적다.
- CNN-Bidirectional LSTM (Parallel) 모델의 경우 CNN-bidirectional LSTM (direct) 모델 보다는 오차율이 적으나 평균 RMSE를 비교 하였을 때 CNN-LSTM (parallel) 모델 보다 성능이 상대적으로 낮음을 알 수 있다.

다음 Fig. 5.3은 모델별 RMSE 값을 box plot을 활용하여 나타낸 것이다.

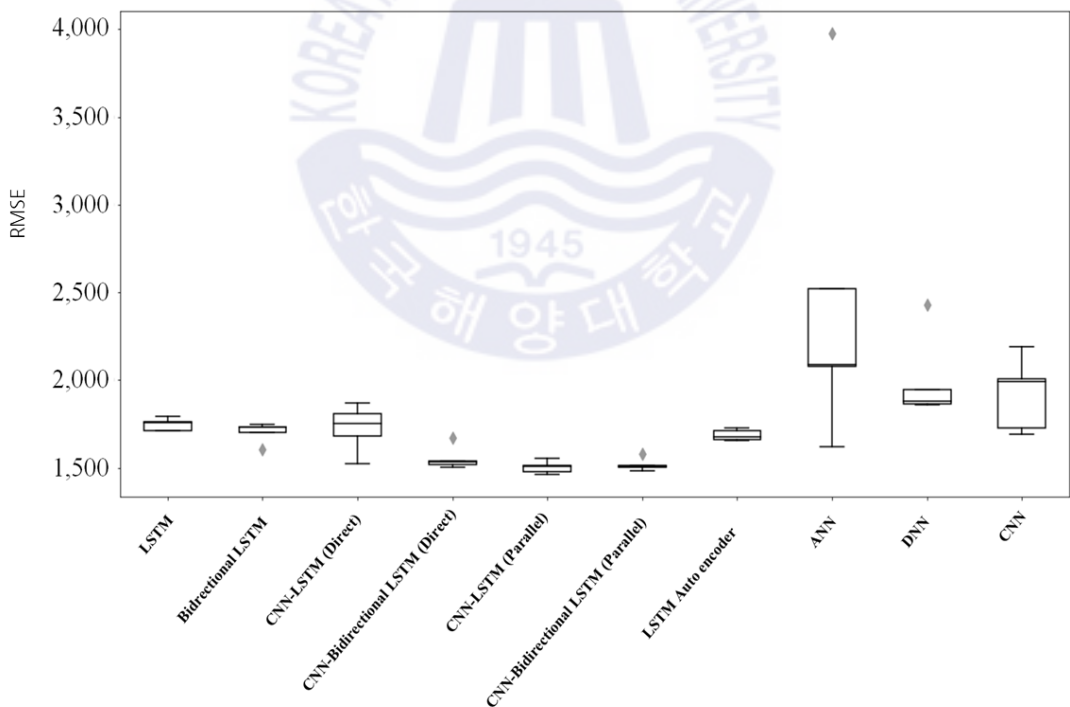


Fig. 5.3 Comparison of model's RMSE using box plot

- ANN 모델의 경우 값의 편차가 매우 크며 outlier 값이 발생함을 확인할 수 있다.
- DNN 모델의 경우 모델의 분차는 ANN 모델, CNN 모델 그리고 CNN-LSTM(direct) 모델보다 적음을 확인할 수 있으나, 모델의 정확도가 상대적으로 낮으며, outlier가 발생함을 알 수 있다.
- CNN 모델의 경우 모델의 정확도가 낮으며 box plot의 최대 값과 uppder quatile 값의 차이가 큰 것을 확인할 수 있다.
- LSTM 모델의 경우 분포가 적으나 정확도가 다른 모델에 비하여 낮은 특징을 보여준다.
- bidirectional LSTM 모델의 경우 LSTM 모델과 비교하여, 정확도와 분포가 낮은 것을 확인할 수 있으나 outlier 값이 발생하는 것을 확인할 수 있으며, CNN과 LSTM 결합 모델과 비교하였을 때 상대적으로 성능이 낮은 것을 확인할 수 있다.
- CNN-LSTM(direct) 모델은 LSTM 모델, bidirectional LSTM 모델과 비교하였을 때 최대 성능은 높으나 모델 자체의 분포가 매우 큰 특징을 보여준다.
- LSTM auto encoder 모델은 낮은 분포와 안정적인 성능을 보여주고 있으나 CNN과 LSTM 결합 모델과 비교하였을 때 상대적으로 성능이 낮은 것을 확인할 수 있다.
- CNN-Bidirectional LSTM(direct), CNN-LSTM(parallel), CNN-bidirectional LSTM(parallel) 모델의 경우 다른 모델과 비교하였을 때 분포와 성능 모두 우수한 것으로 확인된다. 그러나 Fig. 4.13의 데이터 중 ANN 모델의 큰 분포를 나타내며, 세 개 모델 분석의 어려움이 존재한다.

Fig. 5.4는 CNN-LSTM(direct) 모델을 제외한 CNN과 LSTM 결합 모델을 boxplot을 활용한 오차율 비교이다. 세 개 모델의 비교 분석 결과 다음과 같은 결론을 얻을 수 있다.

- CNN-bidirectional LSTM(direct) 모델은 정확도 측면에서 성능이 좋으나, outlier가 발생하는 문제를 내포하고 있다.
- CNN-LSTM(parallel) 모델은 CNN-Bidirectional LSTM(parallel) 모델에 비하여 오차율 분포가 상대적으로 낮으나, outlier가 존재하지 않으며, 정확도 역시 우수함을 알 수 있다.
- CNN-bidirectional LSTM(parallel)은 정확도가 우수하나 outlier가 존재하는 문제점을 가지고 있다.

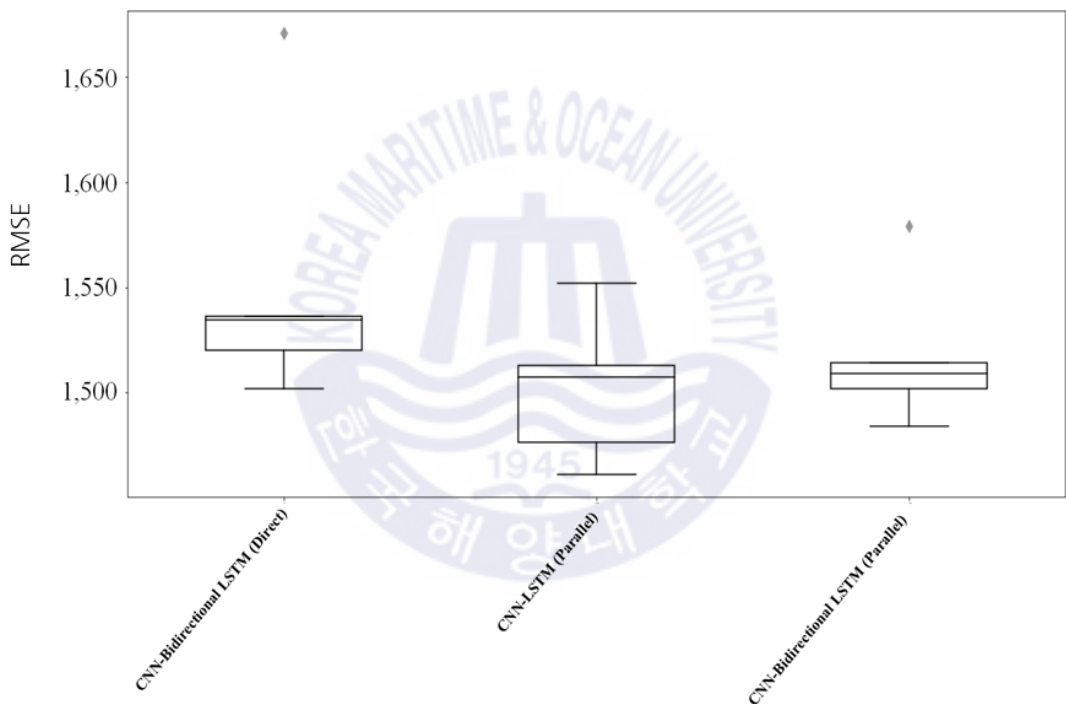


Fig. 5.4 Comparison of model's RMSE using box plot

5.2 연구의 고찰

기존 선행연구 결과와 AI 모델에 대한 분석을 활용하여, 부하의 변동성이 심한 전기추진 선박의 예측 모델을 분석하였다. 그러나 선행 연구의 경우 부하의 변동성이 상대적으로 낮은 태양광 발전 전력, 주가 예측을 수행하는 특징이 존재하였고, 변동성이 심한 데이터의 경우 전력 부하의 범위가 상대적으로 낮은 특징이 존재하였다. 그리고 AI 모델의 경우 개별 모델의 융합 결과에 따른 관련 연구가 공개데이터를 중심으로 제한적으로 이루어지는 특징을 가지고 있었다.

본 논문은 데이터의 특징 추출과 시계열 데이터 예측을 중점으로 가설을 수립하였다. 우선 CNN 모델과 LSTM-auto encoder 모델을 활용하여 전기추진 선박의 부하 데이터 특징 추출 성능을 비교하였으며, 실험 결과 CNN 모델의 성능이 전기추진 선박의 변동성 높은 전력 부하의 특징을 잘 반영할 수 있다는 것을 확인하였다.

CNN 모델과 시계열 데이터 모델의 결합을 활용한 전기추진 선박의 부하 예측모델의 초기 가설은 CNN 모델이 데이터의 특징을 추출하고, 추출된 특징값을 LSTM 등의 시계열 데이터 모델에서 시계열 데이터 예측을 수행하는 CNN-LSTM(direct) 모델의 성능이 우수할 것이라 예측하였다. 그러나 실험결과 CNN-LSTM(parallel) 모델의 성능이 우수했으며, 전기추진 선박 데이터의 특성상 데이터의 특징 추출과 시계열 데이터의 예측을 별도로 진행한 결과 값을 합성하는 모델이 전기추진 선박 데이터를 우수하게 예측할 수 있음을 확인할 수 있었다.

제 6 장 결론

전기추진 선박의 부하는 수심, 유속, 풍향, 풍속, 선속 등 다양한 요인에 영향을 받는다. 또한 최근 시계열 데이터 예측과 관련된 다양한 연구가 진행되고 있으며, 보유한 데이터의 특성에 따라 다양한 형태의 모델을 적용하여, 시계열 데이터 예측의 정확도를 높이는 중이다. 그러나 기존의 연구는 화이트박스 기반의 모델링이나, 주거지 전력 데이터, 정수장 데이터, 태양전지 발전량 등으로 예측하고자 하는 데이터의 높낮이가 낮으며, 데이터의 변화가 상대적으로 적은 특성을 지니고 있으며, 전기추진 선박의 부하를 효과적으로 예측할 수 있는 모델에 대한 선행 연구가 존재하지 않는 문제점을 확인할 수 있었다.

본 연구에서는 데이터의 변화가 심하고, 범위가 넓은 전기추진 선박 부하를 예측하기 위하여, 선박의 부하에 영향을 미칠 수 있는 변수를 선정하고, 데이터 산출 및 보완을 활용하여 누락된 데이터를 복구하였다. 또한 잘못된 입력된 데이터를 경험적 방법론에 근거하여 정상적인 데이터로 변환하였다. 이후 복구가 불가능한 일부 데이터를 삭제하여, 총 29,635개의 데이터를 사용하여 예측 모델용 데이터로 활용하였다.

예측 모델은 LSTM, CNN, bidirectional LSTM, LSTM autoencoder 등을 활용하여, 모델간의 데이터 예측 성능을 비교하였다. 우선 다양한 변수를 활용한 데이터의 특징을 잘 추출할 수 있는 CNN을 활용하여 CNN-LSTM 그리고 CNN-bidirectional LSTM 모델을 결합 방식에 따라 구분하였고, CNN과의 성능 비교를 위하여 LSTM auto encoder를 활용하였다. 마지막으로 위의 모델들과의 비교를 위하여, LSTM, bidirectional LSTM, ANN, DNN, CNN 등의 모델을 활용하였다.

모델의 실험은 각 모델별 5번의 반복 학습을 진행하였으며, RMSE를 활용하여 모델별 평균 RMSE와 boxplot의 정규분포 비교를 통해 모델간의 성능을 분석하였다. LSTM 모델은 1,747.7의 RMSE를 가졌으며, bidirectional LSTM은 1,704의 RMSE를 지니고 있었다. CNN-LSTM (Direct)은 1,726.1의 RMSE를 나타내었고, CNN-bidirectional LSTM (Direct)은 1,522.7을 나타내었다.

그리고 CNN-LSTM (parallel) 1,502의 RMSE로 가장 좋은 성능을 나타냈고, CNN-Bidirectional LSTM (parallel)은 1,517.7의 RMSE를 나타내었다. LSTM auto encoder는 1,686.7로 일반 LSTM과 bidirectional LSTM 보다 좋은 성능을 나타내었다. 마지막으로 ANN은 2,456, DNN은 1,994.6, CNN은 1,921.8의 RMSE를 나타내었다. 또한 boxplot 분석결과 CNN-LSTM (parallel) 구조가 outlier를 나타내지 않아 가장 안정적인 모델임을 알 수 있었다.

본 연구를 통하여 컨테이너 전기추진 선박의 부하를 가장 효과적으로 예측할 수 있는 모델을 선정할 수 있었다. 그러나 다양한 종류의 전기추진 선박의 부하예측을 위하여 모델의 신뢰성 문제를 해결할 수 있어야 한다. 특히 선박의 부하는 운항 목적에 따라 다양한 형태의 부하 특성을 나타낸다. 그러므로 다양한 형태의 선박의 데이터를 수집할 수 있어야 하며, 모든 형태의 전기추진 선박의 부하예측을 할 수 있는 모델의 연구가 필요하다.



감사의 글

2008년 9월 이후 약 12년간 지속적인 지도편달을 해주신 교수님께 감사드립니다. 교수님의 조언과 관심으로, 대학원에 처음 입학하였을 때 가졌던 호기심을 조금이나마 해결할 수 있었던 시간이었습니다. 또한 지난 시간 E2E 실험실 식구들의 도움에 대한 많은 감사의 마음을 표현하고 싶습니다.

학부 시절부터 저에게 지식적 가르침과 삶에 대한 깊은 고민을 할 수 있게 아낌없는 조언을 해주신 유희한 교수님, 소명옥 교수님, 오세준 교수님, 곽준호 박사님께 논문의 지도 편달에 깊은 감사의 말씀을 올립니다. 새로운 영역을 공부함에 부족함을 많이 느꼈으나 앞으로도 더욱 열심히 노력하고 최선을 다하겠습니다.

대학원 시절 저에게 많은 깨달음을 얻게 해주신 박재현 선배님, 이지영 선배님, 곽준호 선배님, 김연형 선배님, 이종호 선배님, 조관준 선배님, 배수영 선배님, 한성훈 선배님, 박도영 선배님, 정성영 선배님 그리고 이현석, 강훈, 강영민, 장재희, 손나영, 강민승에게 항상 좋은 일만 가득하시길 바랍니다. 그리고 대학원에 남아 공부하는 김민욱, 이종학, 오지현은 원하시는 목표를 꼭 이루어 내시길 바랍니다.

“鞠躬盡瘁 死而後已”

E2E 실험실에서 배운 것처럼, 제가 맡아서 하고 있는 일 그리고 앞으로 제가 맡게 될 모든 일에 최선을 다해 힘을 쓰겠습니다.

마지막으로 오늘의 제가 있기까지 아낌없이 지원해주신 어머님께 감사의 마음을 전하며 가족들의 얼굴을 떠올리며 항상 씩씩하게 나아가도록 하겠습니다.

참고문헌

- [1] 박유상, “해상환경규제 강화에 따른 조선산업 영향”, 이슈브리프, 2018.
- [2] Chae-og Lim, Jeong-hoon Bae, Byeong-cheol Park, Sung-chul Shin, “international Conference on Systems and Informatics”, 2017. 11
- [3] 이성근, 배재류, 김진강, “선박 추진시스템 기술동향과 기술개발전략”, 대한조선학회지, 52(2), pp.13-18
- [4] 장은혜, “유럽의 친환경선박 전환 관련 정책 동향”, 한국법제연구원, 2019.12
- [5] 한국에너지공단, “노르웨이 100% 전기추진선박 개발”, [Internet] http://www.energy.or.kr/web/kem_home_new/energy_issue/mail_vol64/pdf/issue_167_01_02.pdf
- [6] 공길영, “[해양수산칼럼] 미래 선박용 연료와 전기추진선”, 국제신문, 2020.01
- [7] Tae-Young Kim, Sung-Bae Cho, “Predicting residential energy consumption using CNN-LSTM neural networks”, Energy, 2019
- [8] Kerang Cao, Hangyung Kim, Chulhyun Hwang, Hoekyung Jung, “CNN-LSTM Coupled Model for Prediction of Waterworks Operation Data”, Journal of Information Processing Systems, Vol. 14, No. 6, pp. 1508-1520, 2018.12
- [9] 정호철, 선영규, 이동구, 김수현, 황유민, 심이삭, 오상근, 송승호, 김진영 “에너지인터넷에서 1D-CNN과 양방향 LSTM을 이용한 에너지 수요예측”, 전기전자 학회 논문지, Vol. 23 Issue 1, Pages.134-142, 2019
- [10] Donghun Lee, Kwanho Kim, “Recurrent Neural Network-Based Hourly Prediction of Photovoltaic Power Output Using Meteorological Information”, Energies, Vol. 12, No. 2, 2019. 01
- [11] Ji-Yoon Kim, Hun-Seok Lee, Jin-Seok Oh, “Study on prediction of ship’s power using light GBM and XGBoost”, 한국마린엔지니어링학회지, Vol.44, No.2, pp. 174-180, 2020

- [12] Zhiyong Cui, Ruimin Ke, Ziyuan Pu, Yin Hai Wang, “Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction” [Internet], <https://arxiv.org/ftp/arxiv/papers/1801/1801.02143.pdf>
- [13] Huang Tang, Yong Yin, Helong Shen, “A model for vessel trajectory prediction based on long short-term memory neural network”, *Journal of Marine Engineering & Technology*, 2019.09.10.
- [14] 장재희, “LCS에 의한 전기추진장치용 하이브리드 전력원의 에너지관리 연구”, 한국해양대학교 박사 학위 논문, 2019
- [15] CG Hodge, DJ Mattick, “The electric warship II, Institute of Marine Engineers”, 1998
- [16] Man-B&W Instruction, “Diesel-electric Drives”, [Internet] <https://marine.mandieselturbo.com/docs/librariesprovider6/marine-broschures/diesel-electric-drives-guideline.pdf>
- [17] QiangMeng, YuquanDu, YadongWanga, “Shipping log data based container ship fuel efficiency modeling”, *Transportation Research Part B*, pp.83. 2016, 207-229
- [18] Yu-Hsien Lin, Ming-Chung Fang, Ronald W. Yeung, “The optimization of ship weather-routing algorithm based on the composite influence of multi-dynamic elements”, *Applied Ocean Research*, Vol. 43, 2013, pp. 184-194
- [19] Melissa J. Azur, Elizabeth A. Stuart, Constantine Frangakis, Philip J. Leaf, “Multiple imputation by chained equations: What is it and how does it work?”, *International Journal of Methods in Psychiatric Research*, Vol. 40-49, pp. 40-49, 2011.03
- [20] Y.H. Kim, Y.K. Hwang, T.G. Kang, K.M. Jung., “LSTM Language Model Based Korean Sentence Genera,” Vol. 41, No. 5, pp.592-601, 2016
- [21] H.G. Youn, D.S.J eong, G.M. Jeong, “Structure And Handwriting Recognition of Recurrent Neural Networks”, *CKIISE*, Vol. 09 No. 33, pp. 42-48, 2015.

- [22] H.S Kim, H.C. Song, S.K. Ko, B.T. Lee, J.W. Shin, “RNN-LSTM based Short-Term Electricity Demand Forecasting using Holiday Information” , IEIE, No. 11, pp. 552-555, 2016.
- [23] S.H. Dae, J.H. Seok, “A Study on the Prediction interest rate using LSTM systems” KICS, pp. 308-0309, 2018.
- [24] C.Y. Kim, J.W. Kang, “LSTM based Anomaly Detection on semiconductor manufacturing data” KIISE, Vol. 2017, No. 12, 760-762. 2017.
- [25] L.S. Kim, “Time Series Prediction Using a Recurrent Neural Network Model” JKPC, Vol. 4, No. 3, pp. 33-46, 2000.
- [26] M. Schuster, K.K. Paliwal, “Bidirectional recurrent neural networks” , IEEE Transactions on Signal Processing, Vol. 45, Issue. 11, 1997
- [27] Saad Albawi, Tareq Abed Mohammed, Saad Al-Zawi, “Understanding of a convolutional neural network” , IEEE, 2017.08
- [28] Andrew Ng, “NIPS 2016“, [Internet] <https://www.youtube.com/watch?v=F1ka6a13S9I>
- [29] F. A. Gers and J. Schmidhuber, “Recurrent nets that time and count,“ Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium, Como, Italy, vol.3, pp. 189-194 , 2000
- [30] N Kalchbrenner, I Danihelka, A Graves, “Grid Long Short-Term Memory” , Cornell University. 2016
- [31] Khaled A. Althelaya, El-Sayed M. El-Alfy, Salahadin Mohammed, “Evaluation of Bidirectional LSTM for Short- and Long-Term Stock Market Prediction” , 2018 9th International Conference on Information and Communication Systems (ICICS), 2018,
- [32] Tae-Young Kim, Sung-Bae Cho, “Predicting residential energy consumption using CNN-LSTM neural networks” , Energy 182, 72e81, 2019

- [33] Ercan Izgi, Ahmet Oztopal b, Bihter Yerli, Mustafa Kemal Kaymak, Ahmet Duran Sahin, “Short-mid-term solar power prediction by using artificial neural networks” , Solar Energy 86 (2012) 725-733, 2012
- [34] S.I Sulaiman, T.K Abdul Rahman, and I. Musirin, “Partial Evolutionary ANN for Output Prediction of a Grid-Connected Photovoltaic System” , International Journal of Computer and Electrical Engineering, Vol. 1, No. 1, 2009.04
- [35] S. W. Bae, J.S. Yu, “Predicting the Real Estate Price Index Using Deep Learning,” RSS. Vol. 27, No. 3, pp.16-17, 2017
- [36] S. Wang, N. Zhang, Y. Zhao and J. Zhan, “Photovoltaic system power forecasting based on combined grey model and BP neural network,” 2011 International Conference on Electrical and Control Engineering, Yichang, pp. 4623-4626, doi: 10.1109/ICECENG.2011.6057634, 2011
- [37] Junseo Son, Yongtae Park, Junu Lee and Hyogon Kim, “Sensorless PV Power Forecasting in Grid-Connected Buildings through Deep Learning” , Sensors 2018, 18, 2529; doi:10.3390/s18082529, 2018
- [38] S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon and K. P. Soman, “Stock price prediction using LSTM, RNN and CNN-sliding window model” 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, pp. 1643-1647, 2017
- [39] O. H. Jafari, O. Groth, A. Kirillov, M. Y. Yang and C. Rother, “Analyzing modular CNN architectures for joint depth prediction and semantic segmentation” 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, pp. 4620-4627, 2017