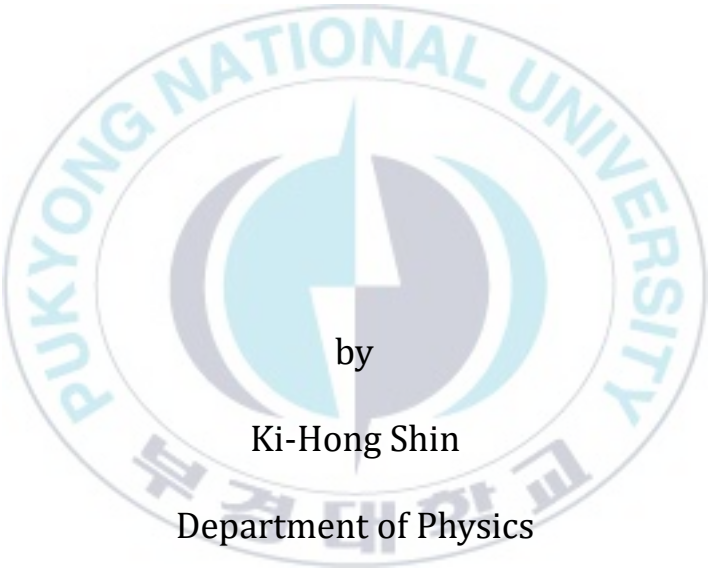
Thesis for the Degree of Doctor of Philosophy

Dynamical prediction of meteorological factors using the long short-term memory network and the deep neural network on Korean cities



by

Ki-Hong Shin

Department of Physics

The Graduate School

Pukyong National University

February 2021

Dynamical prediction of meteorological factors using the long short-term memory network and the deep neural network on Korean cities

한국도시들에서 장단기기억 네트워크와 심층신경망을 사용한 기상요소들의 동역학 예측에 관한 연구

Advisor: Prof. Sang-Wook Wu

by

Ki-Hong Shin

A thesis submitted in partial fulfillment of the requirements for the degree of doctor of Physics in Department of Physics, the Graduate School,

Pukyong National University

February 2021

신기홍의 이학박사 학위 논문을 인준함.

2021년 2월 19일

위원장 김 경 식 ㊞

위 원 우 상 욱 ㊞

위 원 이 동 인 ㊞

위 원 유 철 환 ㊞

위 원 정 재 원 ㊞

한국도시들에서 장단기기억 네트워크와 심층신경망을 사용한 기상요소들의 동역학 예측에 관한 연구

신 기 홍

부경대학교 대학원 물리학과

요약

본 논문에서는 한국의 10 개 광역시 (서울, 대전, 대구, 부산, 인천, 광주, 포항, 목포, 통영, 전주)의 신경망을 이용하여 두 가지 기상 요인 (온도, 습도)의 동적 예측을 연구하고 분석한다. 기상청에서 연도를 봄, 여름, 가을, 겨울로 구분 한 7 년 (2014 ~ 2020)의 평균 온도 및 평균 습도 시계열 데이터를 추출했다. 우리는 인공 신경망 (ANN), 심층 신경망 (DNN), 최적 학습기계 (ELM), 장단기 기억망 (LSTM), 장단기 기억-핍홀 연결망 (LSTM-PC)과 같은 다섯 가지 신경망 알고리즘을 컴퓨터 시뮬레이션으로 다룬다. 인공 신경망, 심층 신경망, 장단기 기억망 및 장단기 기억-핍홀 연결망은 2500, 5000, 7500번 학습한다. 비학습 모델인 최저 학습기계는 2500, 5000, 7500개의 예측 모델에서 생성된 온도 및 습도 값을 평균하여 예측값을 얻는데 사용한다. 5개의 학습률 (0.1, 0.3, 0.5, 0.7 및 0.9)은 인공 신경망 및 심층 신경망의 출력을 적용하는 데 사용되며, 다른 5개의 학습률 (0.001, 0.003, 0.005, 0.007 및 0.009)은 장단기 기억망 및 장단기 기억-핍홀 연결망에 적용된다. 컴퓨터 시뮬레이션은 4개의 경우에 대해 수행한다. 2일의 온도와 습도인 T_{t-1}, T_t, H_{t-1}, H_t를 입력하여 다음날의 온도 (시험1)와 습도 (시험2)를 예측하고, 3일의 온도와 습도인 T_{t-2}, T_{t-1}, T_t, H_{t-2}, H_{t-1}, H_t를 입력하여 다음날의 온도 (시험3)와 습도 (시험4)를 예측한다.

5개의 인공 신경망 모델에서 예측 정확성을 위해 시뮬레이션을 통하여 평균 제곱근 오차 (RMSE), 평균 절대비 오차 (MAPE), 평균 절대 오차 (MAE) 및 테일유 (Theil's-U) 통계를 구한 후 결과를 비교 고찰한다. 특히, 온도 예측을 위한 컴퓨터 시뮬레이션에서 봄은 통영에서 인공 신경망이 시험3 (6개의 입력 노드가 있는 입력층에서 예측된 온도)의 5000번 학습에 대해, 여름은 통영에서 장단기 기억망이 시험3의 5000번 학습에 대해, 가을은 부산에서 장단기 기억-핍홀 연결망

이 시험1 (4개의 입력 노드가 있는 입력층에서 예측된 온도)의 5000번 학습에 대해, 겨울은 대구에서 인공 신경망이 시험1의 2500번 학습에 대해 가장 작은 평균 제곱근 오차 값을 보여준다. 온도 예측에서는 장단기 기억망 모델을 적용했을 때 통영의 여름에서 0.866으로 평균 제곱근 오차 값이 가장 작았다. 습도 예측에 대해서는, 봄은 통영에서 장단기 기억-핍홀 연결망이 시험4 (6개의 입력 노드가 있는 입력층에서 예측된 습도)의 7500번 학습에 대해, 여름은 목포에서 장단기 기억망이 시험4의 2500번 학습에 대해, 가을은 목포에서 장단기 기억망이 시험2 (4개의 입력 노드가 있는 입력층에서 예측된 습도)의 7500번 학습에 대해, 겨울은 목포에서 인공 신경망이 시험4의 7500번 학습에 대해 가장 작은 평균 제곱근 오차 값을 보여준다. 습도 예측에서는 장단기 기억망 모델을 적용했을 때 목포의 여름에서 5.732로 평균 제곱근 오차 값이 가장 작았다. 온도와 습도 두 예측에서 모두 여름에서 장단기 기억망 모델이 가장 좋은 성능을 보였다.

Contents

List of Figures and Tables

Ⅰ. Introduction

Climate change is until now thought to be a cause for concern for many scientists around the world. Changes in the climate factors have resulted in considerable climate variations for complex systems [1,2]. With the numerical and analytical weather prediction of the world meteorological organization (WMO), the statistical quantities of heat transfer, solar radiation, temperature, wind, humidity, surface hydrology, and land subsidence [3] have been calculated within each grid cell of our earth, and these interactions are presently proceeding to be calculated to shed light on the atmospheric properties. Particularly, El Niño–southern oscillation (ENSO) forecast models have been categorized into three types: coupled physical models, statistical models, and hybrid models [4,5]. Among these models, the statistical models introduced for the ENSO forecasts have been the neural network (NN) model, multiple regression (MR) model, and canonical correlation analysis [6,7]. Barnston et al. [8] have found that the statistical models have reasonable accuracies in forecasting sea surface temperature (SST) anomalies.

The Artificial intelligence (AI) has been applied in various fields and research is being actively conducted. The field of deep neural network (DNN), which was once in a recession, has been applied to all fields as the era of big data has entered the era and applied to various industries and has established itself as a core technology. Machine learning (ML), originated in the 17th century, is a branch of artificial intelligence. It has used computers to simulate and analyze various models in scientific fields. The ML and the DNN are sub-fields of AI, and the DNN is the number of hidden layers in the NN. The ML improves its performance through learning, which includes supervised learning and unsupervised learning. Supervised learning uses data with targets as input values, and unsupervised learning uses input data without targets [9]. Supervised learning includes regressions such as the linear regression, logistic regression, ridge regression, and Lasso regression, and classifications such as the support vector machine (SVM) and the decision tree (DT) [10,11]. For the unsupervised learning, there are techniques such as the principle component analysis (PCA), K-means clustering, and density based spatial clustering of applications with Noise (DBSCAN) [12-14]. The reinforcement learning (RL) exists in addition to supervised and unsupervised learning. The RL is known as a learning with actions and rewards. The AlphaGo has for example become famous for its against humans [15,16].

About eight decades ago, the NN model have been proposed by McCulloch and Pitts [17], and the first learning rules were proposed by Hebb [18]. In 1958, the Perceptron model was proposed by Rosenblet [19]. However, it is proved by Minsky and Papert in 1969 that Perceptron is a linear classifier that cannot solve the XOR problem

[20]. In the field of NNs, Rumelhart proposed a multilayer perceptron that added a hidden layer between the input layer and the output layer, and solved the XOR problem, and again faced the moment of development [21]. Until now, many models have been proposed for human memory as a collective property of NNs. The NN models introduced by Little [22] and Hopfield [23] have been based on an Ising Hamiltonian extended by equilibrium statistical mechanics. A detailed discussion of the equilibrium properties of the Hopfield model was discussed in Amit et al. [24].

Furthermore, Werbos firstly proposed back-propagation (BP) for learning ANNs in his doctoral thesis [25], which was developed by Rumelhart in 1986, The BP is a method of learning a NN by calculating the error between the output value of the output layer calculated in the forward direction and the actual value propagated the error in the reverse direction. The BP is a delta rule and gradient descent method (GDM) to update weights by performing learning in the direction of minimizing errors [26,27].

In 1990, Elman proposed a simple recurrent network using the output value of the hidden layer as the input value of the next time considering time [28]. In 1990, Werbos proposed BP through time and proposed a learning method for recurrent neural network (RNN) [29]. The Long Short-Term Memory (LSTM) model, a variant of RNN that controls information flow by adding a gate to a node, was developed by Hochreiter and Schmidhuber [30]. The LSTM-peephole connections (PC) and the LSTM-GRU were developed from the LSTM [31,32]. In addition, the Convolution Neural Network (CNN) used for high-level problems, including image recognition, object detection and language processing is facing a new revival by LeCun. In 1998, the Convolution Neural Network, LeNet-5, was developed [33]. Furthermore, Huang et al. proposed an extreme learning machine (ELM) to improve the slow progression of gradient descent-based algorithms due to iterative learning. The ELM is a single-hidden layer feedforward neural network with one hidden layer, without training, and uses a matrix to obtain the output value [34].

Deep neural network (DNN) has various hyper-parameters such as the learning rate, drop out, epochs, batch size, hidden nodes, activation function and so on. The learning rate is not only a fixed value, but also a cyclic applied by changing the learning rate. Methods such as learning rate and cosine annealing are applied. In the case of weights, an initial weight value is set according to the number of nodes suggested by Xavier et al. [35-38]. In addition to the stochastic gradient descent (SGD), optimizers for the optimization such as the momentum, Nestrov, AdaGrad, RMSProp, Adam, and AdamW have also been developed [39–41].

DNN technology, which has continued to develop, is applied to the fields of stock market [42-50], transportation [51-58], weather [59-70], voice recognition [71-74], and electricity [75-81]. Tao et al. [82] have studied a state-of-the-art DNN for precipitation estimation using the satellite information, infrared (IR), and water vapor (WV) channels, and they have particularly showed a two-stage framework for precipitation estimation

from bispectral information. Although the stock market is a random and unpredictable field, DNN techniques are applied to predict the stock market [83,84]. The prediction accuracy was calculated by dividing the small, medium, large scale by applying deep learning with autoencoder and restricted Boltzmann machine (RBM), neural network with back-propagation algorithm, extreme learning machine (ELM), and radial basis function neural network (RBF) [85]. Sermpinis et al. applied traditional statistical prediction techniques and ANN, RNN, and psi-sigma neural network (PSN) for the EUR/USD exchange rate. Their results showed that the RMSE was smaller when the neural network model was used [86]. Vijh et al. predicted the closing price of US firms using a single hidden layer neural network and a random forest model [87]. Wang et al. applied the BPNN, Elman recurrent neural network (ERNN), stochastic time effective neural network (STNN), and Stochastic Time Effective Function (STNN) for SSE, TWSE, KOSPI, and Nikkei225. They have shown that artificial neural networks perform well in predicting the stock market [88].

Moustra et al. have introduced an ANN model to predict the intensity of earthquakes in Greece. They have used a multilayer perceptron (MLP) for both seismic intensity time series data and seismic electric signals as input data [89]. Gonzalez et al. [90] used the RNN and LSTM models to predict the earthquake intensity in Italy with hourly-data. Kashiwao et al. predicted rain-autumn for the local regions in Japan. They were applied the hybrid algorithm in the random optimization method [91].

Zhang and Dong have studied the CNN model to predict the temperature by using the daily temperature data of China from 1952 to 2018 as learning data [92]. Bilgile et al. has used the ANN model to predict the temperature and precipitation in Turkey, and they have analyzed 32 nodes with one hidden layer. Their results have also showed a high correlation between the predicted value and the actual value [93]. Mohammadi et al. have collected weather data from Bandar Abass and Tabass with different weather conditions. They have predicted and compared daily dew point temperature using the ELM, ANN, and SVM [94]. Maqsood et al. have predicted the temperature, windspeed, and humidity by applying the MLP, RNN, radial based function (RBF), and Hopfield Model for four seasons in Regina Airport's [95].

In this paper, we study and analyze Dynamical prediction of meteorological factors (temperature and humidity) using the NN models. We predict the temperature and the humidity for ten major cities in South Korea peninsula by applying five NN models, that is, the ANN, DNN, LSTM, LSTM-PC, and ELM. We find the RMSE, MAPE, MAE, and Theil-U from our calculated results. Data for calculations are given in Section 2. In Section 3, the basic formulas for the ANN, DNN, LSTM, LSTM-PC, and ELM are provided. Corresponding calculations and the results for these models are presented in Section 4. Concluding remarks are given in Section 5.

Ⅱ. Data

As our data, we use the temperature and the humidity of ten major cities in Korea extracted from the Korea Meteorological Administration (KMA). The ten cities we studied and analyzed are Seoul, Incheon, Daejeon, Daegu, Busan, Pohang, Tongyeong, Gwangju, Mokpo, and Jeonju. We used the data of the manned regional meteorological offices of the KMA to ensure the reliability of data, and it is daily data for seven years from 2014 to 2020. In this study, we trained and tested the NN models for two meteorological factors (temperature and humidity) during five years from 2010 to 2014.

The data used is from 2014 to 2020, and the four seasons are divided into the spring (March, April, May), the summer (June, July, August), the autumn (September, October, November), and the winter (December, January, February). In the case of the winter, data for December of the year and January and February of the following year are used as data for one year, so December 2014 to January and February 2015 are data for the winter of 2014. In this paper, training was conducted with five-years data and verified with one-year data. And, the temperature is converted into absolute temperature and used.

Ⅲ. Theoretical Background

In this section, we introduce the method and its technique for five NN models, that is, the ANN, DNN, LSTM, LSTM-PC, and ELM.

1. Artificial Neural Network (ANN) and Deep Neural Network (DNN)

The ANN is a mathematical model that presents some features of brain functions as a computer-simulation. That is, it is an artificially explored network, distinguished from a biological neural network. The basic structure of the ANN has three layers: input, hidden, and output. Each layer is determined by connection weight and its bias. In an arbitrary layer of the neural network, each node constitutes as one neuron, and one link between nodes means one connection weight of a synapse. The connection weight is corrected as feedback via a training phase, and is designed to implement self-learning. Fig. 1-1 is the ANN structure for one hidden layer with three nodes. The ANN structure with two or more hidden layers is called a DNN. We show a DNN structure with two hidden layers in Fig. 1-2.

The artificial neural network problem can be viewed as a problem of finding the optimal value of a function. If the objective function is $J(\theta)$, the gradient descent method is the method to find the minimum value of this objective function for θ. Next, the

gradient descent algorithm is defined as follows. That is,

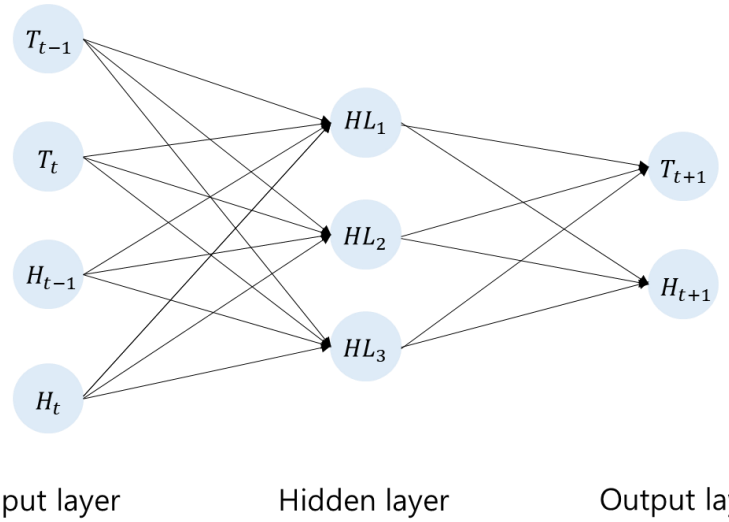


Fig. 1-1: Artificial neural network (ANN) structure with one hidden layer. Each layer is connected by weights.

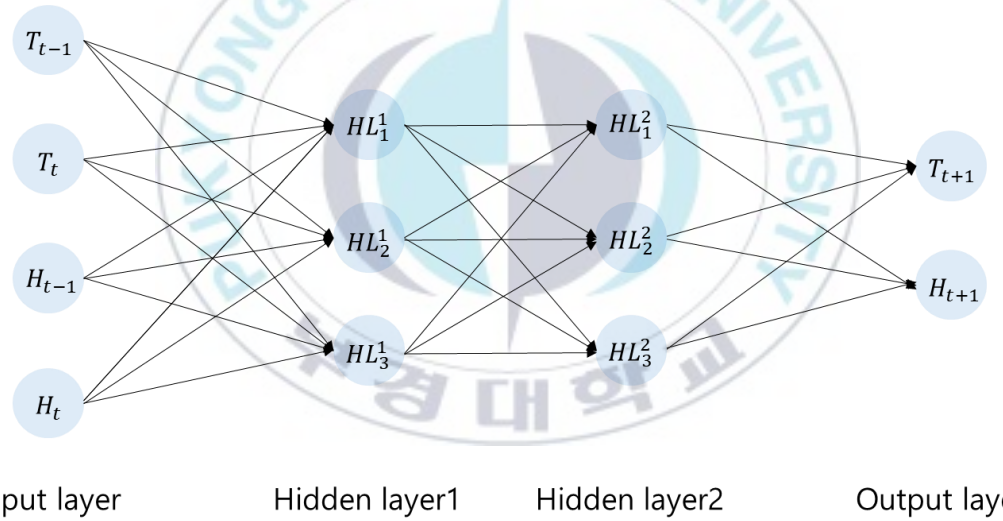


Fig. 1-2: Deep Neural Network (DNN) with 2 hidden layers. In the hidden layer, the superscript indicates the order of the layers, and the subscript indicates the order of the nodes.

$$\theta = \theta - \eta \nabla_\theta J(\theta) \ , \tag{1}$$

where $\nabla_\theta J(\theta)$ is the derivative value of the objective function with respect to θ, which is

called gradient and means $\frac{\partial J(\theta)}{\partial \theta}$. If the sign of $\nabla_\theta J(\theta)$ is negative, it means that the value of θ is adjusted in the opposite direction of the slope. The quantity η is as learning rate, and this is a factor that determines how much gradient is reflected. That is, the change of θ is determined by changing the η value. The value of θ grows larger according to increase η as the big value, but there exists a possibility that the minimum value to be found will be exceeded. On the other hand, if the η value is small, the change in θ will be small, but it may take a lot of learning time to find the minimum value. Among the gradient descent methods, stochastic gradient descent is a method of updating θ for each data set by

$$\theta = \theta - \eta \nabla_\theta J\left(\theta; x^{(i)}; y^{(i)}\right) \ , \tag{2}$$

where $x^{(i)}$ is the i-th training data, and $y^{(i)}$ is the label value of the i-th training data [41].

Forward propagation (FP) is first performed to output a predicted value using an artificial neural network. Forward propagation refers to calculating in the order of input layer-hidden layer-output layer. When a predicted value is found through forward propagation, the error, which is the difference between the target value and the predicted value, is calculated and the error is propagated in the order of the output layer-hidden layer-input layer. This is called the error BP method. The FP of the ANN starts from the input layer and progresses as many as the number of hidden layers, and when it finally reaches the output layer, the FP is completed. Forward propagation is calculated by a simple method. After calculating the weighted sum by Eq. (3) for the layer and the layer, let us calculate the activation function with in Eq. (4) [96].

$$\text{net} = \sum \text{weight} * \text{input} + \text{bias} \tag{3}$$

and

$$\text{output} = \sigma(\text{net}) \ , \tag{4}$$

where σ is a sigmoid function, one of the nonlinear functions, and is called an activation function, and has the following Eq. (5).

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{5}$$

In addition, the differential form of the sigmoid function is shown as

$$\frac{d\sigma(x)}{dx} = \sigma(x)(1-\sigma(x)) \ . \tag{6}$$

Next, we show the functional form of the sigmoid function and its derivative form in Fig. 2 (a) and (b), respectively.

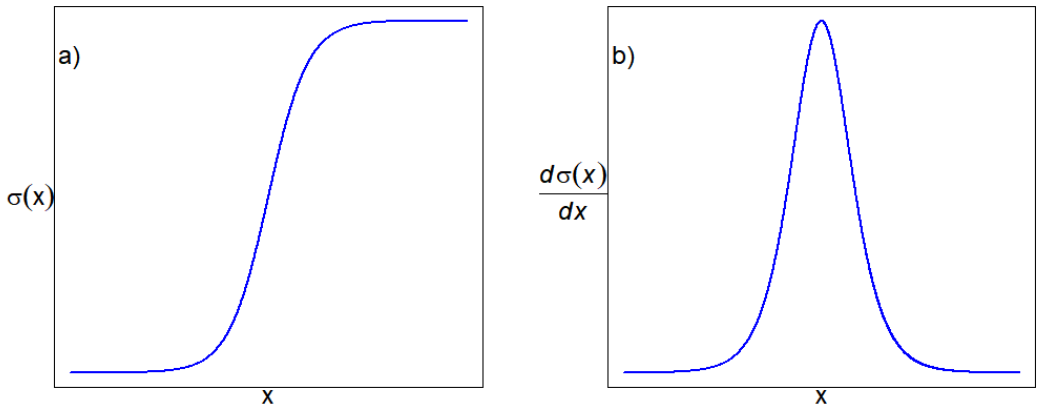


Fig. 2: Sigmoid function $\sigma(x)$ and derivative of sigmoid function $\frac{d\sigma(x)}{dx}$.

First of all, the FP between the input layer and the hidden layer is obtained as follows:

$$y_j = w_{ji}x_i + b_j \tag{7}$$

$$h_j = \sigma(y_j) \tag{8}$$

Here, x_i is the i-th neuron of the input layer, w_{ji} is the weight of the j-th neuron of the first hidden layer and the i- th neuron of the input layer, and b_j is the bias of the j-th neuron of the first hidden layer. h_j is an output value from the output layer. If the hidden layer consists of more than one layer, the above process can be repeated with h_j as the input value of the next layer. The FP of the hidden layer and the output layer is performed by

$$y_k = w_{kj}h_j + b_k \tag{9}$$

$$o_k = \sigma(y_k)\,, \tag{10}$$

where w_{kj} is the weight between the hidden layer and the output layer, and b_k is the bias of the output layer. When the FP calculation is completed for all training data, the training data is learned using a learning method called error back-propagation method. The purpose of the ANN is to find weights that minimize errors. The BP method is updating the weights between layers by sending the error between the target value and the predicted value to the hidden layer and the input layer.

Firstly, the error between the target value and the predicted value of the output layer is calculated. As for the error, the MSE, which is specifically called the cost function, is calculated as

$$E = \frac{1}{2}\sum_{k=1}^{N}(t_k - o_k)^2 \tag{11}$$

In error back-propagation [21,96], the weight updated by the gradient descent method can be expressed as

$$w' = w - \eta \frac{\partial E}{\partial \mathrm{w}} , \tag{12}$$

where w' represents the updated new weight, and w is the weight connecting the layer before the update.

Secondly, the weight between the output layer and the hidden layer is adjusted. When calculating the gradient of the cost function in Eq. (12), it is calculated by applying the chain rule as

$$\begin{aligned}\frac{\partial E}{\partial w_{kj}} &= \frac{\partial E}{\partial o_k}\frac{\partial o_k}{\partial y_k}\frac{\partial y_k}{\partial w_{kj}} \\ &= -(t_k - o_k)(y_k)\big(1 - \sigma(y_k)\big)h_j \\ &= \delta_k h_j\end{aligned} \tag{13}$$

and δ_k is given by

$$\begin{aligned}\delta_{\mathrm{k}} &= \frac{\partial \mathrm{E}}{\partial \mathrm{y_k}} \\ &= -(t_k - o_k)\sigma(y_k)(1\text{-}\,\sigma(y_k)) .\end{aligned} \tag{14}$$

Here, Eq. (14) is called the delta rule. The weight updated by output layer and hidden layer is finally shown as

$$w'_{kj} = w_{kj} + \eta\delta_k h_j . \tag{15}$$

Lastly, the process updated the weights of the hidden layer and the input layer is as follows. In the output layer, the error between the target value and the predicted value can be calculated, but in the hidden layer, the error cannot be calculated because there is no target value of the hidden layer. After δ_k of the output layer is backpropagated and used as an error value in the hidden layer, the gradient for the weight of the hidden layer-input layer is calculated as

$$\begin{aligned}\frac{\partial E}{\partial w_{ji}} &= \frac{\partial E}{\partial h_j}\frac{\partial h_j}{\partial y_j}\frac{\partial y_j}{\partial w_{ji}} \\ &= \frac{\partial E}{\partial y_k}\frac{\partial y_k}{\partial h_j}\frac{\partial h_j}{\partial y_j}\frac{\partial y_j}{\partial w_{ji}} \\ &= (\sum_{k=1}^{N}\delta_k w_{kj})\sigma(y_k)(1-\sigma(y_k))x_i \\ &= \delta_j x_i \end{aligned} \tag{16}$$

The final update for the weights of hidden layer and input layer is obtained by

$$w'_{ji} = w_{ji} + \eta\delta_j x_i \,. \tag{17}$$

2. Long Short Term Memory (LSTM)

The LSTM is an artificial neural network model modified from RNN. Like the ANN model, a basic RNN consists of an input layer, a hidden layer, and an output layer. However, the difference from the existing ANN is that the output value from the hidden layer is used as the input value of the input layer. That is, the output value of the hidden layer at the current time t is used as an input value together with the input value of the input layer at the next time step $t+1$. Fig. 3 shows the structure of the RNN with respect to time. In Fig. 3(a), the structure in which the output value of the hidden layer is again used as the input value of the input layer can be confirmed. Fig. 3(b) shows the unfolded structure Fig. 3(a). In this way, the RNN makes it possible to find more accurate predicted values by continuously remembering past information. However, the characteristic of remembering the past acts as a disadvantage as the time step deepens. Information from the past too far is difficult to reflect in the present time step, which is called gradient vanishing. The LSTM is a model that overcomes the gradient vanishing problem.

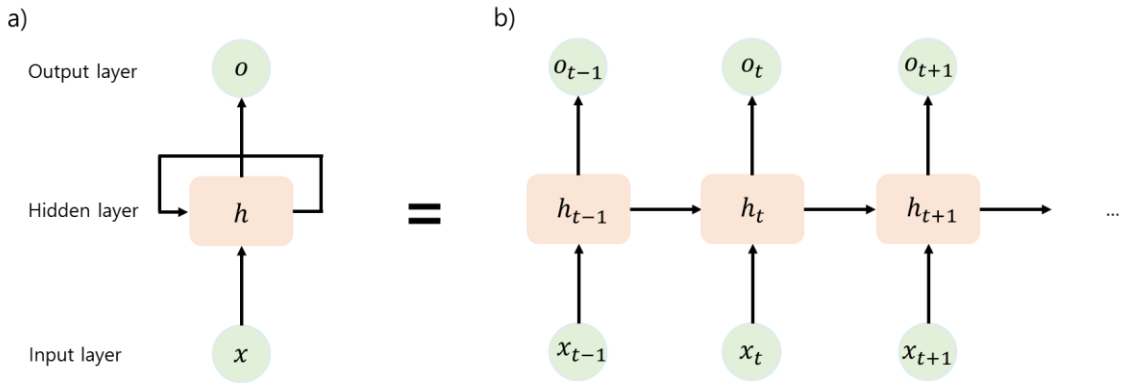


Fig. 3: RNN structure. In (a), it can be seen that the output value of the hidden layer goes back to the input value of the input layer, and (b) is a structure in which RNNs are expanded in chronological order.

Fig. 4 is structure of the LSTM. The LSTM is a model that transforms the node of the hidden layer into a memory cell in the RNN. The memory cell has a forget gate, an input gate, an output gate, and a state that is the current state of the cell, so that past information can be controlled more efficiently.

Forward and back-propagation of the LSTM [97] is achieved by the control of each gate in the LSTM memory cell. There are three types of the LSTM. Input weight connecting the input value and the gate. input weights : $w_f, w_i, w_o, w_z \in R^{N \times M}$, connect the gate to the output of the previous layer, recurrent weights : $u_f, u_i, u_o, u_z \in R^{N \times M}$, and there are bias weights : $b_f, b_i, b_o, b_z \in R^N$. Here, M is the length of the input value x_t at time t, and N is the number of gate nodes. The input value input to the gate passes through the activation function layer of each gate and updates the current cell state.

First of all, the forget gate is a gate that determines how much information from the past will be forgotten. In the forget gate, the output value h_{t-1} of $t-1$ and the input value x_t at the current time t are input and calculated as

$$\overline{f_t} = w_f x_t + u_f h_{t-1} + b_f \ , \tag{18}$$

$$f_t = \sigma(\overline{f_t}) \ . \tag{19}$$

where f_t passes through the sigmoid function and has a value in the range of 0 to 1, and determines how many memories of the past will be remembered. The closer to 0, the less information is remembered, and the closer to 1, the more information from the past is remembered.

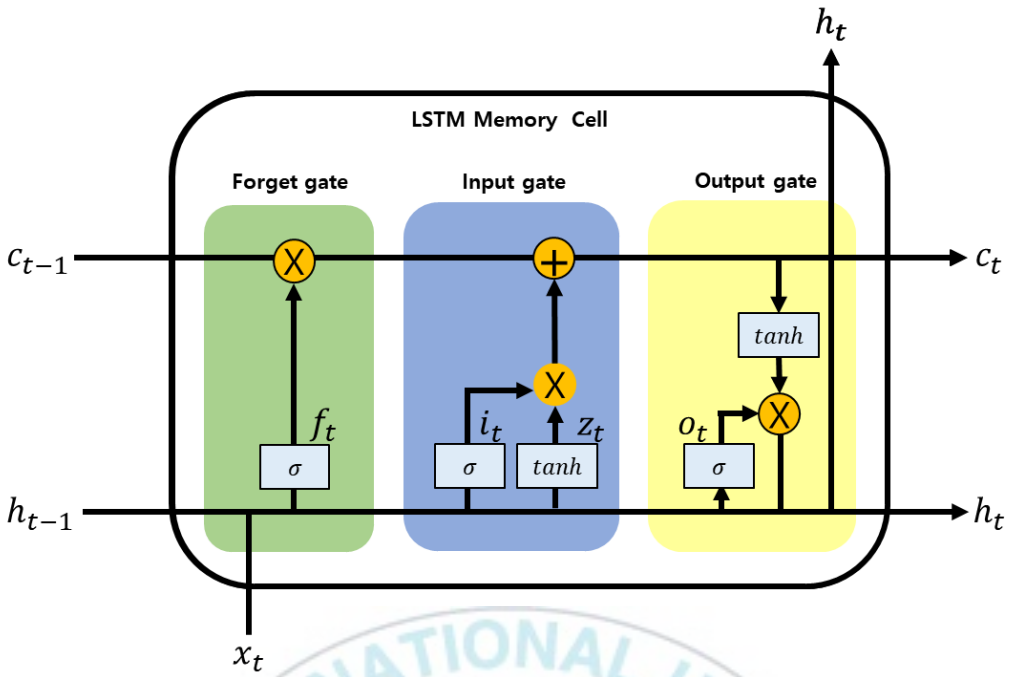


Fig. 4: LSTM memory cell structure.

The input gate determines the new information of the current time step and how much the information is reflected in the current state. The input gate updates the current state of t by passing through two activation functions, the *sigmoid layer* and the *tanh layer* as in Eqs. (21) and (23).

$$\overline{i_t} = w_i x_t + u_i h_{t-1} + b_i \quad (20)$$

$$i_t = \sigma(\overline{i_t}) \quad (21)$$

$$\overline{z_t} = w_z x_t + u_z h_{t-1} + b_z \quad (22)$$

$$z_t = tanh(\overline{z_t}) . \quad (23)$$

i_t passes the sigmoid function and outputs a value between 0 and 1, which is a layer that determines how much information about the input value is reflected.
z_t Has a value between -1 and 1 through tanh, and is a candidate vector of a new state to be reflected in state update.

Let us update the past state c_{t-1} using f_t, i_t, and z_t in the forget gate and input gate. The new state c_t can be obtained by the following Eq. (24).

$$c_t = i_t \odot z_t + c_{t-1} \odot f_t \tag{24}$$

Here, $\odot$ represents element-wise multiplication. In the first term of Eq. (24), the state update candidate z_t(range -1 to 1) is multiplied by the value of i_t(0 to 1) to determine how many state update candidates are reflected. The second term $c_{t-1} \odot f_t$ is an element that determines how much c_{t-1} is to be remembered. The updated c_t is obtained by adding the past state c_{t-1} and the present state candidates.

The output gate is a gate that determines how much state of the current cell is to be exported, and gets a value between 0 and 1 through the sigmoid function, which acts as a factor that determines whether or not the current cell state is completely exported.

$$\overline{o_t} = w_o x_t + u_o h_{t-1} + b_o\,, \tag{25}$$
$$o_t = \sigma(\overline{o_t})\ . \tag{26}$$

The updated state c_t is passed through $tanh$ to create a candidate for a new output value, and the final output value of the cell is calculated by Eq. (27).

$$h_t = tanh(c_t) \odot o_t \tag{27}$$

The output value by Eq. (27) is inputted as the input value of the next time step at the same time as the output value of the cell.

In the LSTM similar to ANN, the error BP method is used as a method for learning. In the LSTM, the Back-Propagation Through Time (BPTT) method, which is an error method dependent on time, is the same as the conventional error BP used in the ANN. We can simply calculate the error between the output value and the target value, if backpropagating the value after the gradient of the error is calculated. However, as it is backpropagated in consideration of time, we can calculate the gradient of the output value h_t as

$$\delta h_t = \Delta_t + u_z^T \delta z_{t+1} + u_i^T \delta i_{t+1} + u_f^T \delta f_{t+1} + u_o^T \delta o_{t+1} \tag{28}$$

Next, let us calculate the gradient values of the forget gate, input gate, and output gate as shown in the following equations

$$\delta f_t = \delta c_t \odot c_{t-1} \odot \sigma(\overline{f_t})(1 - \sigma(\overline{f_t})) \tag{29}$$
$$\delta i_t = \delta c_t \odot z_t \odot \sigma(\overline{i_t})(1 - \sigma(\overline{i_t})) \tag{30}$$
$$\delta o_t = \delta h_t \odot tanh\,(c_t) \odot \sigma(\overline{o_t})(1 - \sigma(\overline{o_t})) \tag{31}$$
$$\delta z_t = \delta c_t \odot i_t \odot (1 - tanh^2(\overline{z_t}))\,. \tag{32}$$

Finally, we calculate the gradient of the cell state as

$$\delta c_t = \delta h_t \odot o_t \odot (1 - tanh^2(c_t)) + \delta c_{t+1} \odot \delta f_{t+1} \tag{33}$$

We can calculate the updated quantities for the weights w, u, b using the gradient obtained through Eqs. (29)- (32). The gradients of the weights for w are calculated as

$$\delta w_f = \sum_{t=0}^{T} \delta f_t \times x_t \tag{34}$$

$$\delta w_i = \sum_{t=0}^{T} \delta i_t \times x_t \tag{35}$$

$$\delta w_o = \sum_{t=0}^{T} \delta o_t \times x_t \tag{36}$$

$$\delta w_z = \sum_{t=0}^{T} \delta z_t \times x_t \ . \tag{37}$$

Next, the gradients of the weights for u are calculated as in Eqs. (38) – (41).

$$\delta u_f = \sum_{t=0}^{T-1} \delta f_{t+1} \times h_t \tag{38}$$

$$\delta u_i = \sum_{t=0}^{T-1} \delta i_{t+1} \times h_t \tag{39}$$

$$\delta u_o = \sum_{t=0}^{T-1} \delta o_{t+1} \times h_t \tag{40}$$

$$\delta u_z = \sum_{t=0}^{T-1} \delta z_{t+1} \times h_t \ . \tag{41}$$

Finally, the gradients of the weights for b are obtained as

$$\delta b_f = \sum_{t=0}^{T} \delta f_t \tag{42}$$

$$\delta b_i = \sum_{t=0}^{T} \delta i_t \tag{43}$$

$$\delta b_o = \sum_{t=0}^{T} \delta o_t \tag{44}$$

$$\delta b_z = \sum_{t=0}^{T} \delta z_t \ . \tag{45}$$

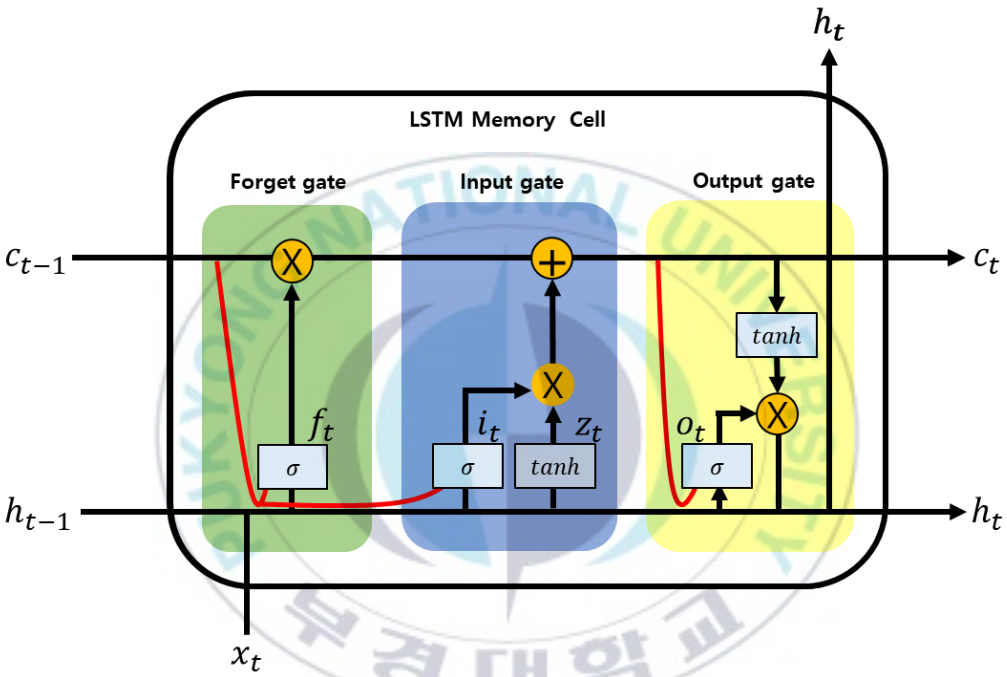


Fig. 5: LSTM-PC structure.

3. LSTM-peephole Connection (LSTM-PC)

The LSTM-PC is a transformed model of the LSTM, and Fig. 5 shows the structure of the LSTM-PC. It is a model that uses more information for learning by putting c_t, the long-term state of the LSTM cell, into the forget gate, input gate, and output gate as an input value. Forward and back-propagation of the LSTM-PC proceeds [97] in the same way as the LSTM. However, the difference from the existing the LSTM is that c_{t-1}, which is the state value of the previous time step, is entered as an input value. First, the Forget

gate is calculated as

$$\overline{f_t} = w_f x_t + u_f h_{t-1} + p_f \odot c_{t-1} + b_f \ , \tag{46}$$

$$f_t = \sigma(\overline{f_t}) \, . \tag{47}$$

Next, the input gate is described by

$$\overline{i_t} = w_i x_t + u_i h_{t-1} + p_i \odot c_{t-1} + b_i \ , \tag{48}$$

$$i_t = \sigma(\overline{i_t}) \, . \tag{49}$$

The z_t value is showed in the same way as Eq. (23). The output gate is calculated as

$$\overline{o_t} = w_o x_t + u_o h_{t-1} + p_o \odot c_t + b_o \ , \tag{50}$$

$$o_t = \sigma(\overline{o_t}) \, . \tag{51}$$

Here, $p_f, p_i, p_o \in R^N$ is the peephole weights. the state value c_t and the output value h_t of the cell are simulated in the same manner as the Eqs. (24) and (27) in the LSTM.

The LSTM-PC learning is also applied to the BPTT method used in the LSTM. It is the same as the BPTT method of the LSTM, On the other hand, the PC is added at δc_t, and the gradients of the weights value of the peephole weight is additionally calculated. Eq. (33) for δc_t is transformed into an equation with peephole weight added as

$$\begin{aligned} \delta c_t = \delta h_t \odot o_t \odot (1 - tanh^2(c_t)) + \delta c_{t+1} \odot \delta f_{t+1} \\ + p_o \odot \delta o_t + p_i \odot \delta i_{t+1} + p_f \odot \delta f_{t+1} \end{aligned} \tag{52}$$

Then, the gradients of the weights of the peephole weight are calculated as.

$$\delta p_i = \sum_{t=0}^{T-1} c_t \odot \delta i_{t+1} \tag{53}$$

$$\delta p_f = \sum_{t=0}^{T-1} c_t \odot \delta f_{t+1} \tag{54}$$

$$\delta p_o = \sum_{t=0}^{T} c_t \odot \delta o_t \ . \tag{55}$$

In the case of w, u, and b, it is the same variables as Eqs. (34) - (45).

4. Extreme Learning Machine (ELM)

The ELM is an artificial neural network model with one hidden layer, as shown in Fig. 1. In other words, it is the same as the ANN model with one hidden layer, but the ELM is characterized in that it does not learn differently from the existing ANN model.

If N training datasets are (x_i, t_i), then $x_i = [x_{i1}, x_{i2} \dots, x_{in}]^T \in R^n$ and $t_i = [t_{i1}, t_{i2} \dots, t_{im}]^T \in R^m$. $w_i = [w_{i1}, w_{i2} \dots, w_{in}]^T$ the weight vector of which the i-th hidden layer node and the input nodes are connected. The quantity $\beta_i = [\beta_{i1}, \beta_{i2} \dots, \beta_{in}]^T$ is a weight vector in which the i-th hidden layer node and output nodes are connected. The quantity b_i is the bias value of the i-th hidden layer node. Then, the output value o_j can be obtained as follows [34] :

$$o_j = \sum_{i=1}^{\tilde{N}} \beta_i \, \sigma_i(x_j) = \sum_{i=1}^{\tilde{N}} \beta_i \, \sigma_i(w_i \cdot x_j + b_i), \ j = 1,2, \dots, N. \tag{56}$$

If Eq. (56) is expressed in the matrix form, it is shown as Eq. (57) below. That is,

$$H\beta = T, \tag{57}$$

where T is the target value, and H is the output matrix of the hidden layer. These forms are shown as

$$H(\boldsymbol{w}_1, \dots, \boldsymbol{w}_{\tilde{N}}, b_1, \dots, b_{\tilde{N}}, \boldsymbol{x}_1, \dots, \boldsymbol{x}_{\tilde{N}}) = \begin{bmatrix} \sigma(\boldsymbol{w}_1 \cdot \boldsymbol{x}_1 + b_1) & \cdots & \sigma(\boldsymbol{w}_{\tilde{N}} \cdot \boldsymbol{x}_1 + b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ \sigma(\boldsymbol{w}_1 \cdot \boldsymbol{x}_N + b_1) & \cdots & \sigma(\boldsymbol{w}_{\tilde{N}} \cdot \boldsymbol{x}_N + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}} \tag{58}$$

$$\beta = \left[\beta_1^T \cdots \beta_{\tilde{N}}^T\right]_{\tilde{N} \times m}^{\mathrm{T}} \tag{59}$$

$$T = \left[t_1^T \cdots t_N^T\right]_{N \times m}^{\mathrm{T}} \tag{60}$$

where T is transpose of matrix. The solution of the linear system of Eq. (57) is to find $\hat{\beta}$ that satisfies

$$\left\| H(\boldsymbol{w}_1, \dots, \boldsymbol{w}_{\tilde{N}}, b_1, \dots, b_{\tilde{N}}, \boldsymbol{x}_1, \dots, \boldsymbol{x}_{\tilde{N}}) \hat{\beta} - T \right\|$$

$$= \min_{\beta} \|H(\boldsymbol{w}_1, \dots, \boldsymbol{w}_{\tilde{N}}, b_1, \dots, b_{\tilde{N}}, \boldsymbol{x}_1, \dots, \boldsymbol{x}_{\tilde{N}})\beta - T\| . \quad (61)$$

The solution of Eq. (57) can be obtained as

$$\hat{\beta} = H^{\dagger}T, \quad (62)$$

where $H^{\dagger}$ is moore-penrose generalized inverse of H.

The weight is an element connecting each layer in the artificial neural network model. When training a neural network model, it is a common method to randomly set the weight in the range of -1 to 1 or 0 to 1. However, the setting of these weights can act as a factor that hinders proper learning of the model. Therefore, Xavier proposed [38] a method for setting initial weights for effective model learning. In the uniform distribution between [-1, 1], if the node n_{in} of the previous layer and the node of the next layer are n_{out}, the initial weight can be set as shown in the following Eq. (63).

$$W \sim U(-\sqrt{\frac{6}{n_{in}+n_{out}}}, +\sqrt{\frac{6}{n_{in}+n_{out}}}) \quad (63)$$

The Xavier weight initialization is also a suitable method when the activation function is a sigmoid function.

In this paper, four evaluation scales RMSE, MAPE, MAE and Theil's U were calculated to evaluate the performance of the artificial neural network model as the following Eqs. (64) – (67). Here, *A* and *P* are represented as an actual value and a predicted value, respectively:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(A_i - P_i)^2} \quad (64)$$

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{A_i - P_i}{A_i}\right| \times 100\% \quad (65)$$

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|A_i - P_i| \quad (66)$$

and

$$\text{Theil's U} = \frac{\sqrt{\frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2}}{\sqrt{\frac{1}{N}\sum_{t=1}^{N} y_i^2} + \sqrt{\frac{1}{N}\sum_{t=1}^{N} \hat{y}_i^2}} \ . \tag{67}$$

where we normalize and convert the input data as $y = \frac{x - x_{min}}{x_{max} - x_{min}}$.

Ⅳ. Numerical results

4-1. Testings 1 and 2

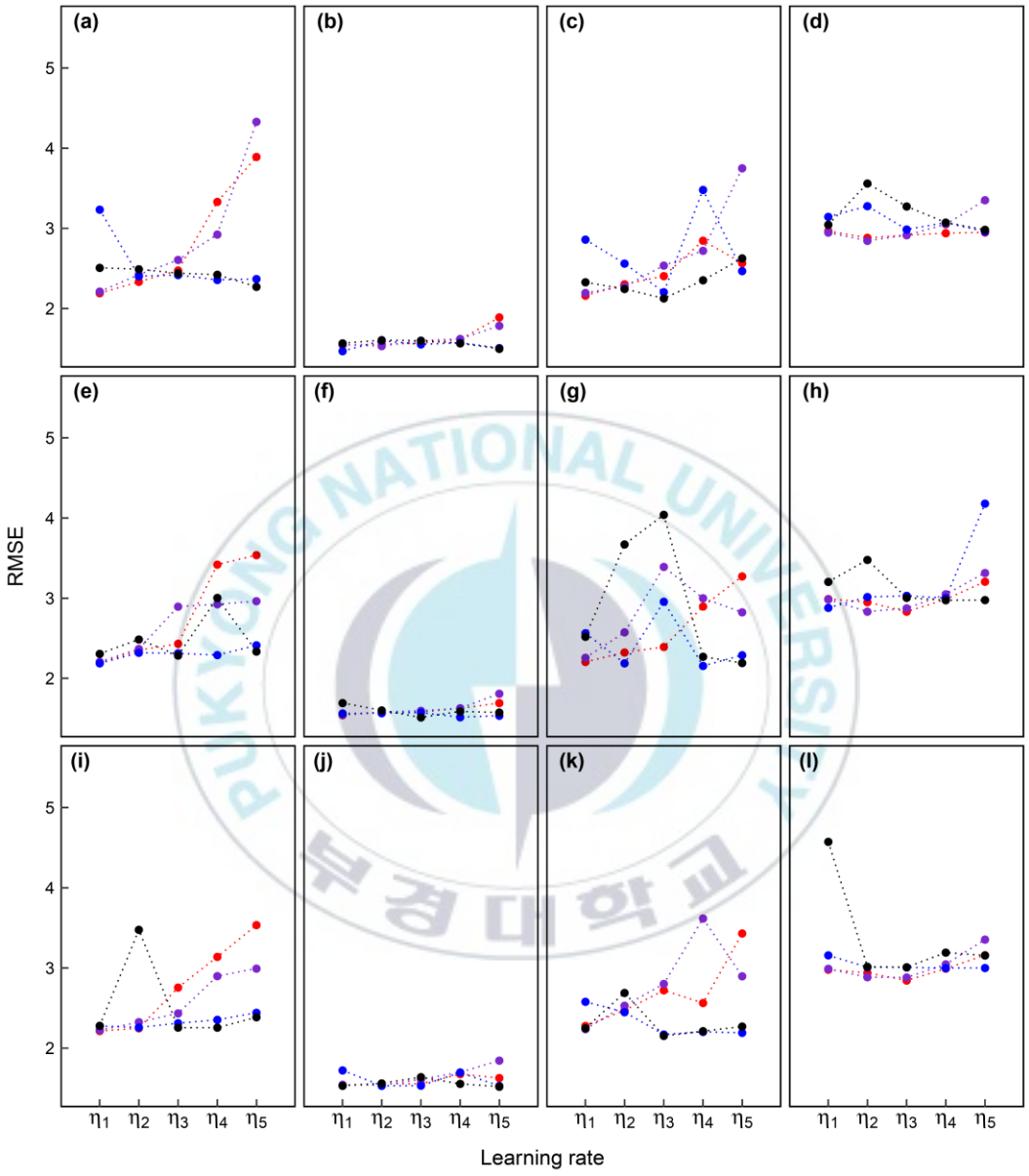


Fig. 6: Values of the RMSE as a function of the learning rate in the ANN (red circle), DNN (purple circle), LSTM (blue circle), and LSTM-PC (black circle) in all seasons of Seoul in testing 1. Here, (a)-(d) , (e)-(h), and (i)-(l) are, respectively, the results for training 2500, 5000, and 7500 epochs in the spring, summer, autumn, winter.

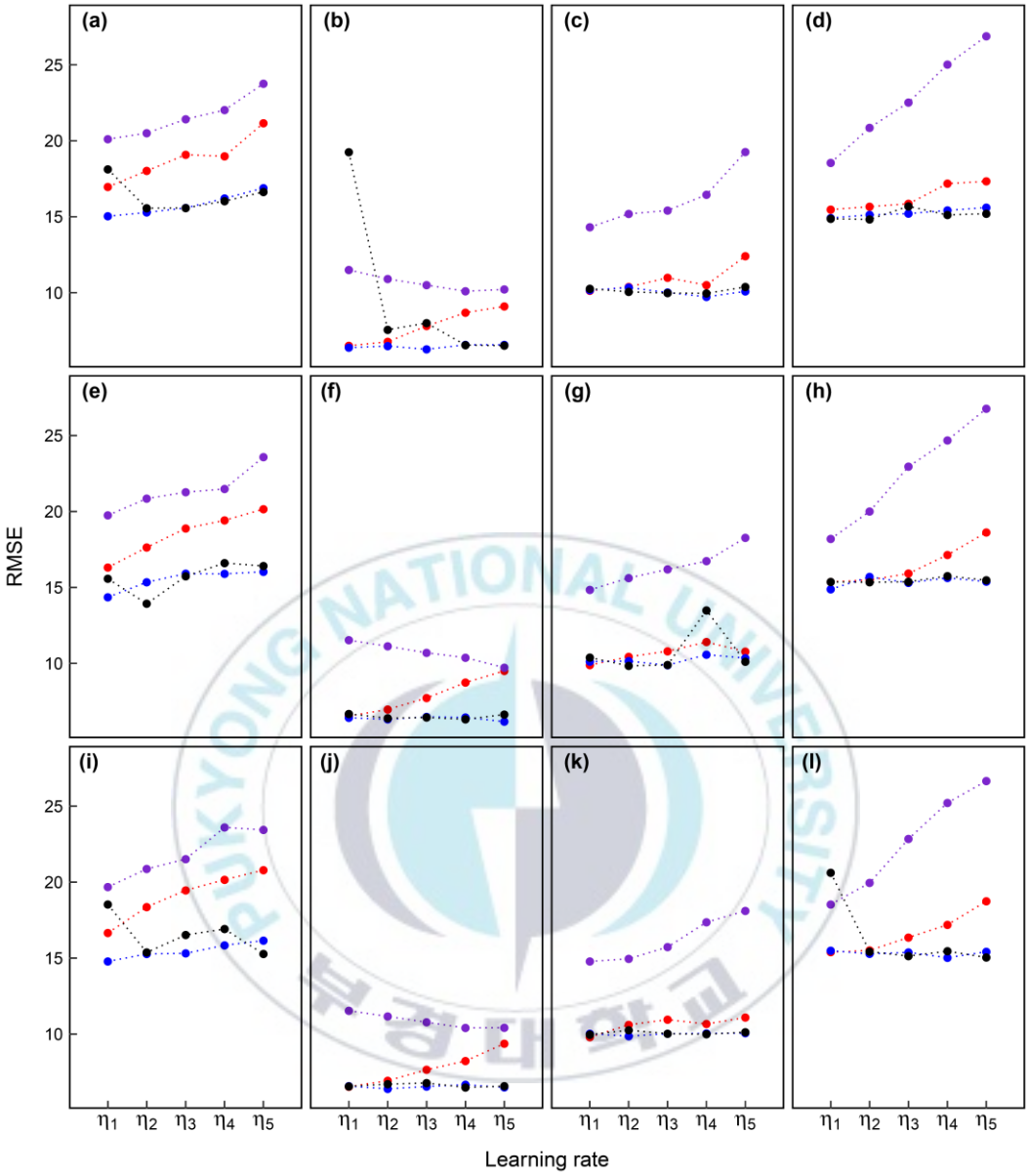


Fig. 7: Values of the MAPE as a function of the learning rate in the ANN (red circle), DNN (purple circle), LSTM (blue circle), and LSTM-PC (black circle) in all seasons of Tongyeong in testing 2. Here, (a)-(d), (e)-(h), and (i)-(l) are, respectively, the results for training 2500, 5000, and 7500 epochs in the spring, summer, autumn, winter.

Table. 1: Learning rate

Leraning rate	ANN, DNN	LSTM, LSTM-PC
η_1	0.1	0.001
η_2	0.3	0.003
η_3	0.5	0.005
η_4	0.7	0.007
η_5	0.9	0.009

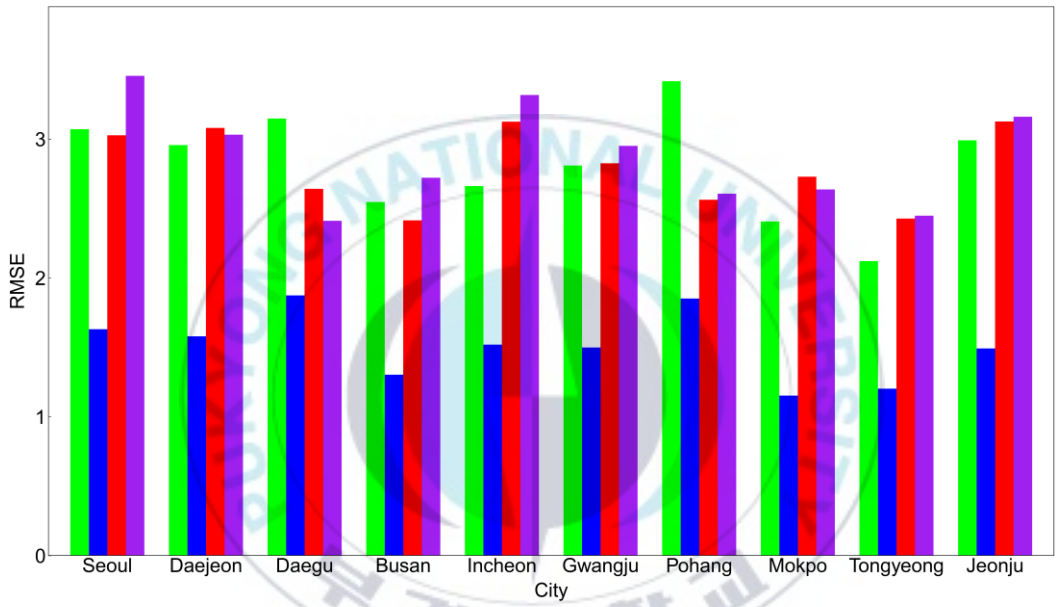


Fig. 8: The RMSE of ELM for 7500 epochs in spring (light green bar), summer (blue bar), autumn (red bar), winter (purple bar) of testing 1.

Table. 2: The RMSE of ELM for 7500 epochs for spring, summer, autumn, and winter in 10 cities in testing 1, where the numbers in parentheses indicate the learning rate.

	Seoul	Daejeon	Daegu	Busan	Incheon	Gwangju	Pohang	Mokpo	Tongyeong	Jeonju
Spring	3.072	2.958	3.149	2.548	2.662	2.81	3.417	2.407	2.122	2.992
Summer	1.631	1.58	1.875	1.304	1.52	1.5	1.852	1.153	1.203	1.493
Autumn	3.028	3.081	2.642	2.415	3.126	2.827	2.564	2.73	2.428	3.127
Winter	3.456	3.032	2.412	2.722	3.318	2.952	2.607	2.638	2.449	3.162

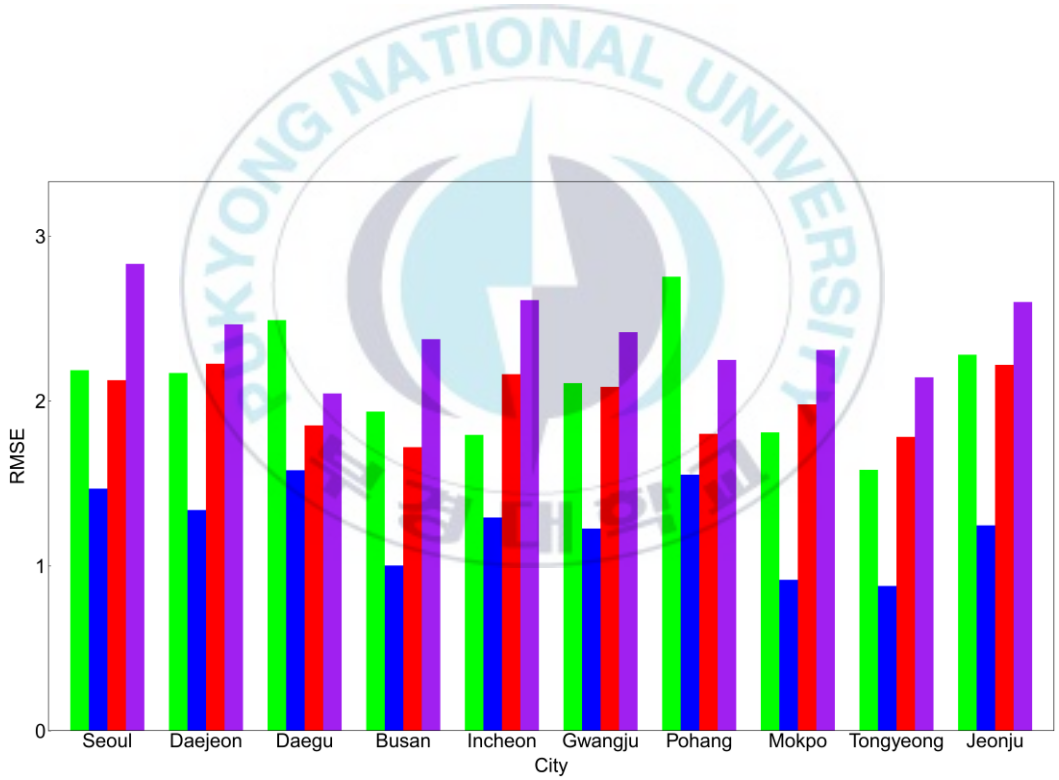


Fig. 9: The Lowest RMSE of five NN models for all three kinds of epochs in spring (light green bar), summer (blue bar), autumn (red bar). winter (purple bar) in testing 1.

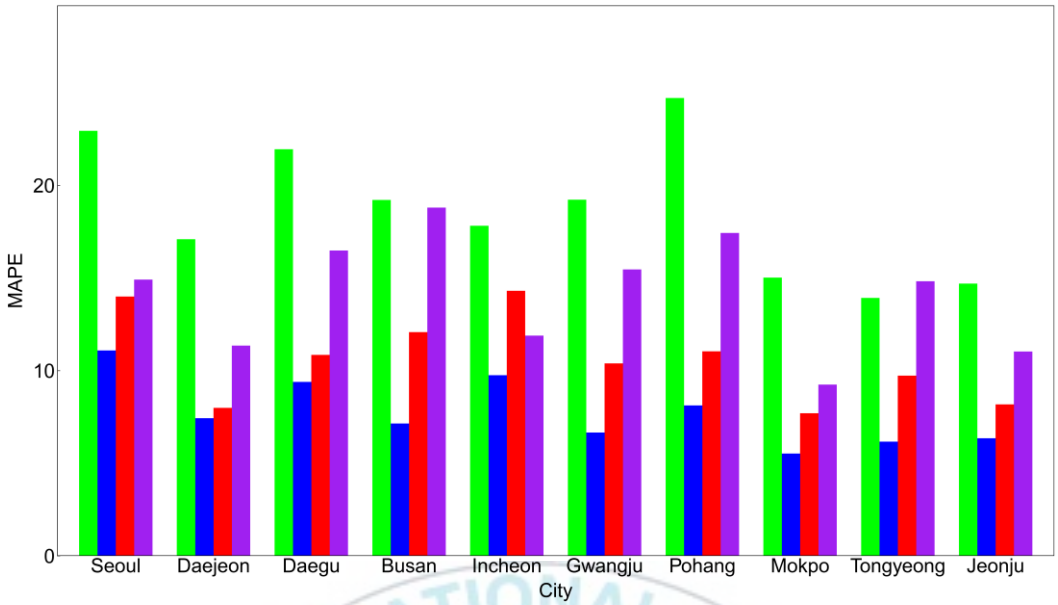


Fig. 10: The Lowest MAPE of five NN models for all three kinds of epochs in spring (light green bar), summer (blue), autumn (red). winter (purple) in testing 2.

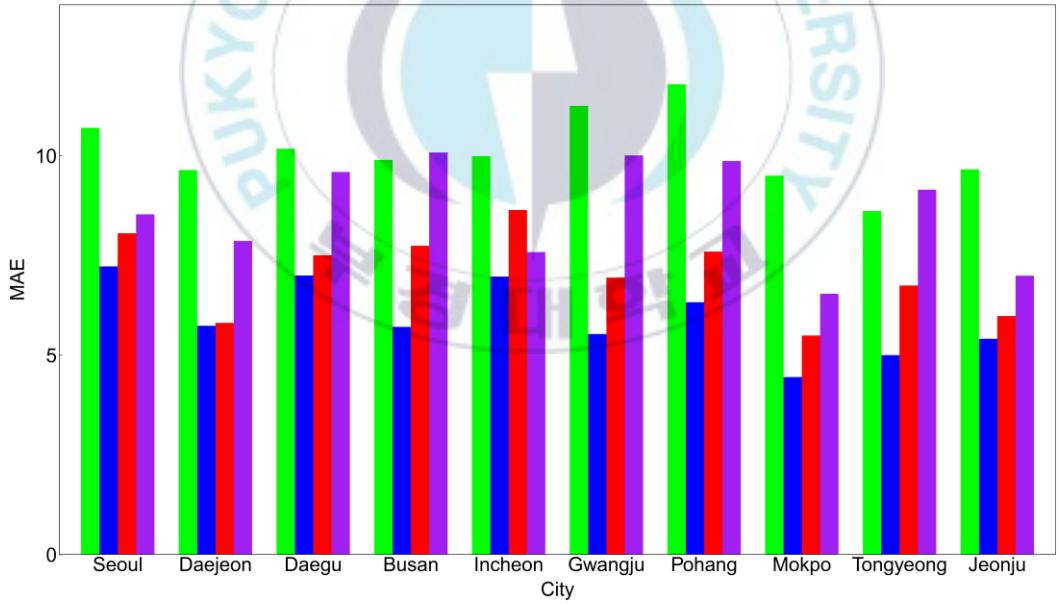


Fig. 11: The Lowest MAE of five NN models for all three kinds of epochs in spring (light green bar), summer (blue bar), autumn (red bar). winter (purple bar) in testing 2.

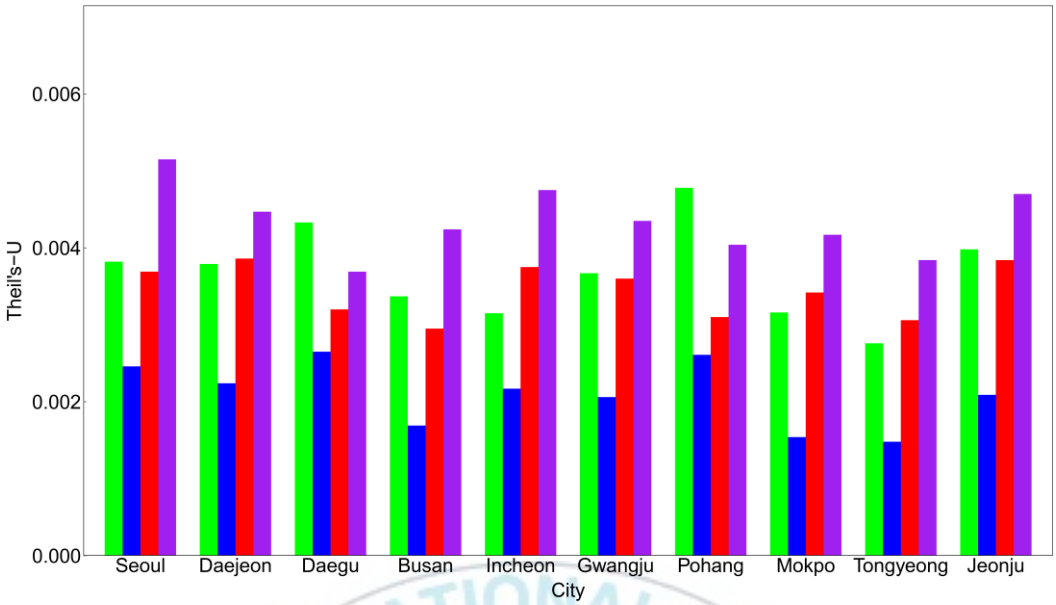


Fig. 12: The Lowest Theil's-U of five NN models for all three kinds of epochs in spring (light green bar), summer (blue bar), autumn (red bar). winter (purple bar) in testing 1.

Table. 3: RMSE, MAPE, MAE and Theil's-U of testing 1.

City	Season	RMSE	MAPE	MAE	Theil's-U ($\times 10^{-3}$)
Seoul	Spring	2.187 (LSTM $\eta = 0.001$)	0.61 (DNN $\eta = 0.1$)	1.747 (DNN $\eta = 0.1$)	3.823 (LSTM $\eta = 0.001$)
	Summer	1.469 (LSTM $\eta = 0.001$)	0.392 (LSTM $\eta = 0.001$)	1.169 (LSTM $\eta = 0.001$),	2.462 (LSTM $\eta = 0.001$)
	Autumn	2.126 (LSTM-PC $\eta = 0.005$)	0.392 (LSTM-PC $\eta = 0.009$)	1.575 (LSTM-PC $\eta = 0.005$)	3.686 (LSTM-PC $\eta = 0.005$)
	Winter	2.832 (ANN $\eta = 0.5$)	0.552 (LSTM-PC $\eta = 0.005$)	2.046 (DNN $\eta = 0.3$)	5.153 (ANN $\eta = 0.5$)
Daejeon	Spring	2.17 (ANN $\eta = 0.1$)	0.591 (DNN $\eta = 0.1$)	1.694 (DNN $\eta = 0.1$)	3.785 (ANN $\eta = 0.1$)
	Summer	1.339 (LSTM $\eta = 0.001$)	0.333 (LSTM $\eta = 0.007$)	0.992 (LSTM $\eta = 0.007$)	2.244 (LSTM $\eta = 0.001$)
	Autumn	2.227 (LSTM-PC $\eta = 0.007$)	0.58 (LSTM-PC $\eta = 0.007$)	1.66 (LSTM-PC $\eta = 0.007$)	3.855 (LSTM-PC $\eta = 0.007$)
	Winter	2.465 (ANN	0.692 (ANN	1.909 (ANN	4.469 (ANN

		$\eta = 0.5$)	$\eta = 0.5$)	$\eta = 0.5$)	$\eta = 0.5$)
Daegu	Spring	2.491 (LSTM-PC $\eta = 0.007$)	0.677 (LSTM-PC $\eta = 0.009$)	1.947 (LSTM-PC $\eta = 0.009$)	4.328 (LSTM-PC $\eta = 0.007$)
	Summer	1.58 (LSTM $\eta = 0.001$)	0.407 (LSTM $\eta = 0.001$)	1.209 (LSTM $\eta = 0.001$)	2.646 (LSTM $\eta = 0.001$)
	Autumn	1.852 (LSTM $\eta = 0.009$)	0.491 (LSTM-PC $\eta = 0.007$)	1.412 (LSTM-PC $\eta = 0.007$)	3.202 (LSTM $\eta = 0.009$)
	Winter	2.046 (ANN $\eta = 0.3$)	0.545 (ANN $\eta = 0.3$)	1.509 (ANN $\eta = 0.3$)	3.694 (ANN $\eta = 0.3$)
Busan	Spring	1.936 (DNN $\eta = 0.1$)	0.522 (ANN $\eta = 0.3$)	1.499 (ANN $\eta = 0.3$)	3.366 (DNN $\eta = 0.1$)
	Summer	1.003 (LSTM $\eta = 0.001$)	0.266 (LSTM $\eta = 0.001$)	0.789 (LSTM $\eta = 0.001$)	1.686 (LSTM $\eta = 0.001$)
	Autumn	1.72 (LSTM-PC $\eta = 0.005$)	0.448 (LSTM-PC $\eta = 0.005$)	1.298 (LSTM $\eta = 0.005$)	2.955 (LSTM-PC $\eta = 0.005$)
	Winter	2.375 (DNN $\eta = 0.3$)	0.64 (DNN $\eta = 0.1$)	1.793 (DNN $\eta = 0.1$)	4.241 (DNN $\eta = 0.3$)
Incheon	Spring	1.795 (ANN $\eta = 0.1$)	0.493 (DNN $\eta = 0.001$))	1.405 (NN $\eta = 0.1$))	3.149 (ANN $\eta = 0.1$)
	Summer	1.294 (LSTM $\eta = 0.009$)	0.334 (LSTM $\eta = 0.009$)	0.997 (LSTM $\eta = 0.009$)	2.173 (LSTM $\eta = 0.009$)
	Autumn	2.163 (LSTM $\eta = 0.00$	0.569 (ANN $\eta = 0.3$)	1.628 (ANN $\eta = 0.3$)	3.746 (ANN $\eta = 0.3$)
	Winter	2.612 (LSTM-PC $\eta = 0.005$)	0.726 (DNN $\eta = 0.005$)	1.991 (DNN $\eta = 0.5$)	4.751 (LSTM-PC $\eta = 0.005$)
Gwangju	Spring	2.108 (LSTM-PC $\eta = 0.007$)	0.565 (ANN $\eta = 0.1$)	1.617 (ANN $\eta = 0.1$)	3.674 (LSTM-PC $\eta = 0.007$)
	Summer	1.227 (LSTM-PC $\eta = 0.001$)	0.314 (LSTM-PC $\eta = 0.001$)	0.934 (LSTM-PC $\eta = 0.1$)	2.058 (LSTM-PC $\eta = 0.001$)
	Autumn	2.086 (LSTM $\eta = 0.005$)	0.518 (LSTM-PC $\eta = 0.007$)	1.487 (LSTM-PC $\eta = 0.007$)	3.599 (LSTM $\eta = 0.005$)
	Winter	2.418 (LSTM-PC $\eta = 0.001$)	0.633 (ANN $\eta = 0.3$)	1.762 (ANN $\eta = 0.3$)	4.354 (LSTM-PC $\eta = 0.001$)
Pohang	Spring	2.754 (LSTM $\eta = 0.001$)	0.745 (LSTM $\eta = 0.009$)	2.147 (LSTM $\eta = 0.009$)	4.783 (LSTM $\eta = 0.001$)
	Summer	1.554 (LSTM	0.436 (LSTM	1.299 (LSTM	2.606 (LSTM

		$\eta = 0.005$)	$\eta = 0.001$	$\eta = 0.001$)	$\eta = 0.005$)
	Autumn	1.801 (ANN $\eta = 0.1$)	0.482 (LSTM-PC $\eta = 0.007$)	1.393 (LSTM-PC $\eta = 0.007$)	3.101 (ANN $\eta = 0.1$)
	Winter	2.249 (DNN $\eta = 0.1$)	0.598 (DNN $\eta = 0.1$)	1.666 (DNN $\eta = 0.1$)	4.041 (DNN $\eta = 0.1$)
Mokpo	Spring	1.81 (LSTM $\eta = 0.001$)	0.511 (LSTM $\eta = 0.001$)	1.458 (LSTM $\eta = 0.001$)	3.163 (LSTM $\eta = 0.001$)
	Summer	0.916 (LSTM-PC $\eta = 0.007$)	0.243 (LSTM-PC $\eta = 0.007$)	0.722 (LSTM-PC $\eta = 0.007$)	1.539 (LSTM-PC $\eta = 0.007$)
	Autumn	1.98 (ANN $\eta = 0.1$)	0.492 (LSTM-PC $\eta = 0.005$)	1.413 (LSTM-PC $\eta = 0.005$)	3.416 (ANN $\eta = 0.1$)
	Winter	2.31 (ANN $\eta = 0.005$)	0.611 (ANN $\eta = 0.5$)	1.695 (ANN $\eta = 0.5$)	4.166 (ANN $\eta = 0.5$)
Tongyeong	Spring	1.583 (ANN $\eta = 0.1$)	0.439 (LSTM $\eta = 0.005$)	1.26 (LSTM $\eta = 0.005$)	2.758 (ANN $\eta = 0.1$)
	Summer	0.878 (LSTM-PC $\eta = 0.003$)	0.226 (LSTM-PC $\eta = 0.003$)	0.671 (LSTM-PC $\eta = 0.003$)	1.478 (LSTM-PC $\eta = 0.003$)
	Autumn	1.783 (LSTM-PC $\eta = 0.005$)	0.463 (LSTM-PC $\eta = 0.005$)	1.336 (LSTM-PC $\eta = 0.005$)	3.063 (LSTM-PC $\eta = 0.005$)
	Winter	2.144 (ANN $\eta = 0.3$)	0.585 (ANN $\eta = 0.3$)	1.635 (ANN $\eta = 0.3$)	3.840 (ANN $\eta = 0.3$)
Jeonju	Spring	2.281 (LSTM-PC $\eta = 0.005$)	0.622 (DNN $\eta = 0.1$)	1.779 (DNN $\eta = 0.1$)	3.983 (LSTM-PC $\eta = 0.005$)
	Summer	1.246 (LSTM-PC $\eta = 0.001$)	0.324 (LSTM-PC $\eta = 0.003$)	0.961 (LSTM-PC $\eta = 0.003$)	2.090 (LSTM-PC $\eta = 0.001$)
	Autumn	2.22 (LSTM $\eta = 0.009$)	0.584 (LSTM-PC $\eta = 0.007$)	1.675 (LSTM-PC $\eta = 0.007$)	3.837 (LSTM $\eta = 0.009$)
	Winter	2.601 (ANN $\eta = 0.3$)	0.716 (ANN $\eta = 0.3$)	1.984 (ANN $\eta = 0.3$)	4.699 (ANN $\eta = 0.3$)

Table. 4: RMSE, MAPE, MAE and Theil's-U of testing 2.

City	Season	RMSE	MAPE	MAE	Theil's-U
Seoul	Spring	13.302 (LSTM-PC $\eta = 0.001$)	22.944 (LSTM-PC $\eta = 0.009$)	10.7 (LSTM-PC $\eta = 0.009$)	0.126 (LSTM-PC $\eta = 0.001$)
	Summer	9.399	11.087	7.226	0.071

		(LSTM-PC $\eta = 0.003$)	(LSTM-PC $\eta = 0.007$)	(LSTM $\eta = 0.009$)	(LSTM-PC $\eta = 0.003$)
	Autumn	11.313 (LSTM-PC $\eta = 0.001$)	14 (LSTM $\eta = 0.007$)	8.055 (LSTM $\eta = 0.007$)	0.091 (LSTM-PC $\eta = 0.001$)
	Winter	11.054 (LSTM-PC $\eta = 0.001$)	14.915 (LSTM-PC $\eta = 0.001$)	8.531 (LSTM-PC $\eta = 0.001$)	0.095 (LSTM-PC $\eta = 0.009$)
Daejeon	Spring	11.468 (LSTM-PC $\eta = 0.001$)	17.095 (LSTM-PC $\eta = 0.001$)	9.64 (LSTM-PC $\eta = 0.001$)	0.094 (LSTM-PC $\eta = 0.001$)
	Summer	7.46 (ANN $\eta = 0.1$)	7.433 (LSTM $\eta = 0.009$)	5.737 (ann $\eta = 0.1$)	0.048 (ANN $\eta = 0.1$)
	Autumn	7.827 (LSTM-PC $\eta = 0.001$)	7.99 (LSTM-PC $\eta = 0.005$)	5.809 (LSTM-PC $\eta = 0.001$)	0.051 (LSTM-PC $\eta = 0.001$)
	Winter	10.403 (LSTM $\eta = 0.001$)	11.352 (LSTM-PC $\eta = 0.001$)	7.862 (LSTM-PC $\eta = 0.001$)	0.074 (LSTM-PC $\eta = 0.007$)
Daegu	Spring	12.798 (LSTM $\eta = 0.001$)	21.951 (LSTM-PC $\eta = 0.003$)	10.176 (LSTM-PC $\eta = 0.003$)	0.121 (LSTM $\eta = 0.001$)
	Summer	9.307 (ANN $\eta = 0.1$)	9.392 (LSTM-PC $\eta = 0.007$)	6.996 (LSTM-PC $\eta = 0.007$)	0.064 (ANN $\eta = 0.1$)
	Autumn	9.614 (LSTM-PC $\eta = 0.003$)	10.853 (LSTM-PC $\eta = 0.003$)	7.504 (LSTM $\eta = 0.001$)	0.067 (LSTM $\eta = 0.001$)
	Winter	13.404 (LSTM $\eta = 0.001$)	16.484 (LSTM-PC $\eta = 0.001$)	9.594 (LSTM-PC $\eta = 0.001$)	0.114 (LSTM-PC $\eta = 0.005$)
Busan	Spring	12.085 (LSTM $\eta = 0.001$)	19.212 (LSTM $\eta = 0.001$)	9.893 (LSTM $\eta = 0.001$)	0.101 (LSTM-PC $\eta = 0.003$)
	Summer	7.166 (ANN $\eta = 0.1$)	7.147 (ANN $\eta = 0.1$)	5.708 (ann $\eta = 0.1$)	0.045 (ANN $\eta = 0.1$)
	Autumn	10.032 (LSTM-PC $\eta = 0.009$)	12.081 (LSTM $\eta = 0.007$)	7.744 (LSTM $\eta = 0.007$)	0.072 (LSTM $\eta = 0.005$)
	Winter	14.173 (LSTM-PC $\eta = 0.001$)	18.804 (LSTM-PC $\eta = 0.003$)	10.081 (LSTM-PC $\eta = 0.001$)	0.131 (ANN $\eta = 0.007$)
Incheon	Spring	12.629 (ANN $\eta = 0.5$)	17.822 (ANN $\eta = 0.3$)	9.989 (ANN $\eta = 0.3$)	0.097 (ANN $\eta = 0.5$)
	Summer	8.864 (LSTM-PC $\eta = 0.005$)	9.755 (LSTM-PC $\eta = 0.005$)	6.969 (LSTM-PC $\eta = 0.005$)	0.057 (LSTM-PC $\eta = 0.005$)
	Autumn	10.941 (LSTM-PC $\eta = 0.005$)	14.308 (LSTM-PC $\eta = 0.005$)	8.64 (LSTM-PC $\eta = 0.005$)	0.083 (LSTM-PC $\eta = 0.005$)
	Winter	9.832	11.892	7.585	0.078

		(LSTM-PC $\eta = 0.009$)	(LSTM-PC $\eta = 0.003$)	(LSTM-PC $\eta = 0.003$)	(LSTM-PC $\eta = 0.009$)
Gwangju	Spring	13.862 (LSTM $\eta = 0.001$)	19.23 (LSTM $\eta = 0.001$)	11.248 (LSTM $\eta = 0.001$)	0.109 (LSTM $\eta = 0.001$)
	Summer	7.093 (ANN $\eta = 0.7$)	6.659 (LSTM-PC $\eta = 0.007$)	5.53 (LSTM-PC $\eta = 0.007$)	0.044 (ANN $\eta = 0.7$)
	Autumn	8.895 (LSTM $\eta = 0.003$)	10.391 (LSTM $\eta = 0.003$)	6.945 (LSTM $\eta = 0.003$)	0.059 (LSTM $\eta = 0.003$)
	Winter	12.703 (LSTM-PC $\eta = 0.005$)	15.463 (ANN $\eta = 0.5$)	10.008 (LSTM-PC $\eta = 0.005$)	0.096 (LSTM-PC $\eta = 0.009$)
Pohang	Spring	14.322 (LSTM $\eta = 0.001$)	24.717 (LSTM-PC $\eta = 0.003$)	11.793 (LSTM $\eta = 0.001$)	0.123 (LSTM $\eta = 0.001$)
	Summer	7.99 (ANN $\eta = 0.7$)	8.122 (LSTM $\eta = 0.001$)	6.326 (LSTM $\eta = 0.001$)	0.051 (ANN $\eta = 0.9$)
	Autumn	9.408 (ANN $\eta = 0.3$)	11.043 (ANN $\eta = 0.3$)	7.598 (ANN $\eta = 0.3$)	0.065 (ANN $\eta = 0.3$)
	Winter	12.832 (ANN $\eta = 0.5$)	17.431 (LSTM $\eta = 0.007$)	9.866 (ANN $\eta = 0.5$)	0.109 (ANN $\eta = 0.5$)
Mokpo	Spring	11.435 (ANN $\eta = 0.7$)	15.024 (ANN $\eta = 0.7$)	9.502 (LSTM-PC $\eta = 0.007$)	0.082 (ANN $\eta = 0.7$)
	Summer	5.839 (ANN $\eta = 0.1$)	5.525 (LSTM $\eta = 0.007$)	4.451 (LSTM $\eta = 0.007$)	0.036 (ANN $\eta = 0.1$)
	Autumn	6.891 (LSTM $\eta = 0.005$)	7.702 (ANN $\eta = 0.3$)	5.494 (ANN $\eta = 0.1$)	0.046 (LSTM $\eta = 0.003$)
	Winter	8.16 (ANN $\eta = 0.1$)	9.248 (ANN $\eta = 0.1$)	6.54 (ANN $\eta = 0.1$)	0.058 (ANN $\eta = 0.1$)
Tongyeong	Spring	10.479 (LSTM $\eta = 0.001$)	13.926 (LSTM-PC $\eta = 0.003$)	8.613 (LSTM-PC $\eta = 0.003$)	0.076 (LSTM $\eta = 0.001$)
	Summer	6.549 (LSTM $\eta = 0.009$)	6.166 (LSTM $\eta = 0.009$)	5.002 (LSTM $\eta = 0.009$)	0.039 (LSTM $\eta = 0.005$)
	Autumn	8.63 (ANN $\eta = 0.1$)	9.729 (LSTM $\eta = 0.007$)	6.747 (LSTM-PC $\eta = 0.003$)	0.059 (ANN $\eta = 0.1$)
	Winter	12.371 (LSTM $\eta = 0.001$)	14.826 (LSTM-PC $\eta = 0.003$)	9.15 (LSTM-PC $\eta = 0.001$)	0.101 (ANN $\eta = 0.7$)
Jeonju	Spring	12.011 (LSTM-PC $\eta = 0.003$)	14.698 (LSTM $\eta = 0.001$)	9.658 (LSTM $\eta = 0.001$)	0.088 (LSTM-PC $\eta = 0.003$)
	Summer	7.027	6.351	5.412	0.041

	(ANN $\eta = 0.9$)	(LSTM $\eta = 0.007$)	(LSTM-PC $\eta = 0.005$)	(ANN $\eta = 0.9$)
Autumn	7.386 (LSTM-PC $\eta = 0.001$)	8.178 (LSTM-PC $\eta = 0.001$)	5.98 (LSTM-PC $\eta = 0.001$)	0.047 (LSTM-PC $\eta = 0.1$)
Winter	8.899 (ANN $\eta = 0.1$)	11.033 (ANN $\eta = 0.1$)	6.992 (ANN $\eta = 0.1$)	0.066 (ANN $\eta = 0.1$)

In this section, the computer-simulation is performed for two testings as follows: testing 1 has the four nodes T_{t-1}, T_t, H_{t-1}, H_t, in the input layer and the one output node T_{t+1}in output layer, and testing 2 has also the four input nodes T_{t-1}, T_t, H_{t-1}, H_t, and the one output node H_{t+1}. We set the five learning rates 0.1, 0.2, 0.3, 0.4, 0.5 for the ANN and the DNN, while the learning rates for LSTM and LSTM-PC are set as 0.001, 0.003, 0.005, 0.007, 0.009, for different train set sizes over three runs, 2500, 5000, 7500 epochs. The predicted values of the ELM are obtained by averaging the results over 2500, 5000, and 7500 epochs. A prediction model is created and the average of the prediction values is obtained through the prediction model was used as the final prediction value. The ANN and the DNN can be trained in the range of 0.1 - 0.9, while we can train the LSTM and the LSTM-PC in the range of 0.001 - 0.009, as summarize in Table 1.

Fig. 6 shows the predicted values of the RMSE as a function of the learning rate η in the ANN, DNN, LSTM, and LSTM-PC in all seasons of Seoul in testing 1. Here, Figs. 6(a)-6(d), 6(e)-6(h), and 6(i)-6(l) are, respectively, the results for training 2500, 5000, and 7500 epochs in the spring, summer, autumn, winter.

From Fig. 6(a) for 2500 epochs in spring in Seoul, the ANN and the DNN show a tendency to increase the RMSE as each learning rate increases from η_1 to η_5. The RMSE value of ANN is 2.192 at η_1, and has a high value of 3.892 at η_5. The RMSE value of DNN is 2.212 (4.33) at η_1 (η_5). Then, the RMSE value gradually increases as each learning rate increases from η_1 to η_5. In Fig. 6(e) for 5000 epochs in summer in Seoul, the ANN and the DNN show a tendency to increase the RMSE as the learning rate increases from η_1 to η_5. The ANN RMSE value is 2.193 at η_1, and has a high value of 3.538 at η_5. The DNN RMSE value is 2.207 (2.963) at η_1 (η_5). The RMSE of LSTM and LSTM-PC RMSEs do not show a clear trend. However, the ANN exhibits a higher RMSE value at η_5 compared to other learning rates. The RMSE of LSTM has significantly a value of 3.234 (2.187) for training 2500 (5000) epochs at η_1. In Fig. 6(i) for 7500 epochs, the ANN and DNN shows a tendency to increase RMSE as the learning rate increases from η_1 to η_5. The RMSE value of ANN (DNN) is 2.214 (2.226) at η_1, and has a highest value of 3.535 (2.993) at η_5. The RMSE value (3.476) of LSTM-PC at η_2 is larger than other learning rates. The RMSE of LSTM has a lower (higher) value of 2.259 (2.442) at η_2 (η_5).

In Fig. 6(b), the RMSE of ANN and DNN has larger values than those of LSTM and LSTM-PC for 2500 epochs in summer in Seoul. The RMSE of ANN (DNN) value is 1.89

(1.785) at η_5, and the RMSE of LSTM (LSTM-PC) has a lower value of 1.507 (1.498) at η_5. In addition, as the learning rate increases, the RMSE value of ANN tends to increase. The RMSE of ANN had a lowest (highest) value of 1.545 (1.89) at η_1 (η_5). In Fig. 6(f) for training 5000 epochs, the RMSE of DNN has a lowest value of 1.548 at η_1, but the RMSEs of ANN, LSTM, and LSTM-PC do not show a distinct trend. In Fig. 6(j) for 7500epochs, the RMSE of ANN shows very similar result with 1.554 (1.555) at η_2 (η_3). the LSTM-PC showed a higher RMSE value at η_3, compared to other learning rates.

Fig. 6(c) for 2500 epochs in autumn in Seoul, the RMSE of DNN tends to increase as the learning rate increases. The RMSE value of DNN has a lowest (highest) value of 2.195 (3.75) at η_1 (η_5). In the LSTM-PC, the RMSE decreased and then increased again based on η_3. In Fig. 6(g) for 5000 epochs, the RMSEs of DNN, LSTM, and LSTM-PC had highest values of 3.39, 2.956, and 4.04 at η_3, respectively. The RMSE of ANN had a lowest (highest) value of 2.205 (3.273) at η_1 (η_5). In Fig. 6(k) for 7500 epochs, the RMSE values of LSTM and LSTM-PC are 2.171 and 2.153 at η_3, respectively. and it was confirmed that learning was better than the result of Fig. 6(g). The RMSE of ANN has a lowest (highest) value of 2.278 (3.431) at η_1 (η_5). Hence, the learning rate gradually increases, the RMSE shows a tendency to increase.

Figs. 6(d), 6(h), and 6(l) are the results of learning 2500, 5000, and 7500 epochs for winter temperature. In Fig. 6(d), the RMSE of ANN and the DNN shows a tendency to increase from η_1 to η_5., while that of LSTM-PC shows a tendency to decrease from η_2 to η_5. The RMSE of LSTM do not show clearly a trend. In Fig. 6(h), the RMSE value of LSTM shows higher than that of other NN models at η_5, and the RMSE value of LSTM-PC shows higher at η_2 compared to other learning rates for each model. In Fig. 6(l), the RMSE of LSTM-PC shows a very higher value of 2.88, compared with those at η_1 in Fig. 6(d)-6(h). On the other hand, at η_2, the RMSE of LSTM-PC shows better performance than those of Figs. 6(d) and 6(h).

In Fig. 7, we obtain the predicted values of the MAPE as a function of the learning rate in the ANN, DNN, LSTM, and LSTM-PC in all seasons of Tongyeong in testing 2. Here, Figs. 7(a)-7(d), 7(e)-7(h), and 7(i)-7(l) are, respectively, the results for training 2500, 5000, and 7500 epochs in the spring, summer, autumn, winter. Fig. 8 shows the RMSE of ELM for 7500 epochs in spring, summer, autumn, winter in testing 1, and Table 2 also illustrates the comparison of the RMSE of ELM for 7500 epochs for spring, summer, autumn, and winter in ten cities in testing.

Figs. 9-12 plot the lowest RMSE, MAPE, MAP, and Theil's-U of ten cities for all three training (2500, 5000, 7500) epochs in spring, summer, autumn, winter in testings 1 and 2, respectively. Tables 3 and 4 is illustrated the comparison of the RMSE, MAPE, MAE and Theil's-U statistics in four seasons of ten cities in testings 1 and 2, respectively.

From testing 1 of Table 3, the RMSE (MAPE) of ANN (LSTM) has a lowest value

of 1.583 (0.439) at $\eta = 0.1$ ($\eta = 0.005$) in spring in Tongyeong. The RMSE (MAE) of LSTM-PC has a lowest value of 0.878 (0.671) at $\eta = 0.003$ ($\eta = 0.003$) in summer in Tongyeong. In autumn of Busan, the RMSE (MAE) value of LSTM-PC is a lowest value 1.72 (1.298) at $\eta = 0.005$. In winter of Daegue, the MAPE (Theil's-U) of ANN has a value of 0.545 ($3.694 \text{ x } 10^{-3}$) at $\eta = 0.3$ lowest than that of other cities.

From testing 2 of Table 4, in spring of Tongyeong, the RMSE (MAPE) of LSTM (LSTM-PC) has a lowest value of 10.479 (13.926) at $\eta = 0.001$ ($\eta = 0.003$). The RMSE (MAE) of ANN (LSTM) has a lowest value of 5.839 (4.451) at learning rate $\eta = 0.1$ ($\eta = 0.007$) in Summer in Mokpo. In autumn of Mokpo, the RMSE (MAE) value of LSTM (ANN) is a lowest value 6.891 (5.494) at $\eta = 0.005$ ($\eta = 0.1$). It is not good accuracy characteristically in autumn of testing 2, but in autumn, the RMSE value of DNN is 8.076 at $\eta = 0.1$ in Jeonju, while that of ELM is 8.196 in Mokpo for 5000 epochs. In winter of Mokpo, the MAPE (Theil's-U) of ANN has a value of 9.248 (0.058) at $\eta = 0.1$ lowest than that of other cities.

4-2. Testings 3 and 4

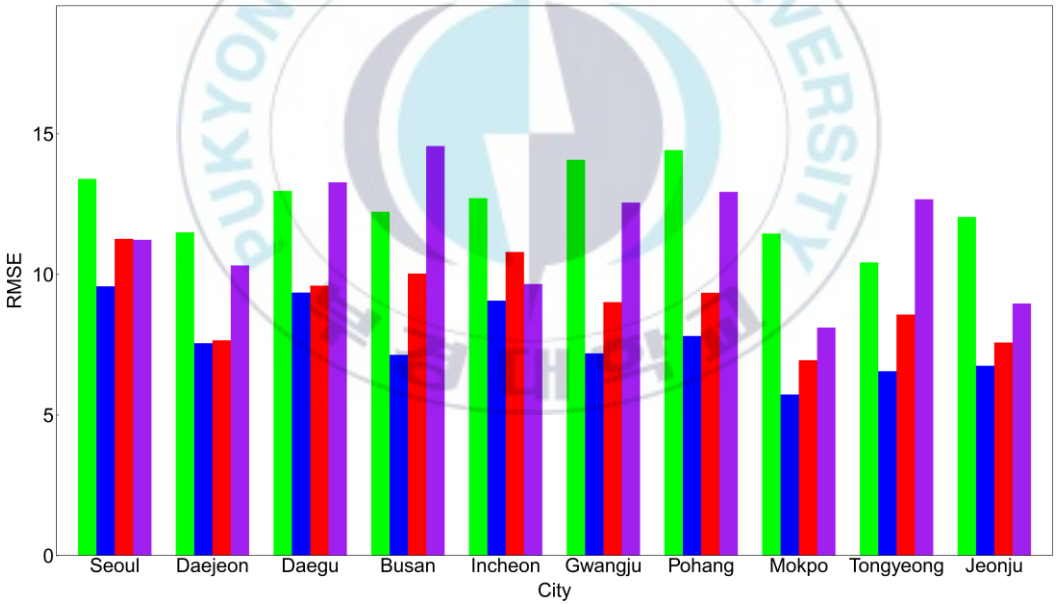


Fig. 13: The lowest RMSE of five NN models for all three kinds of epochs in spring (light green bar), summer (blue bar), autumn (red bar), winter (purple bar) of testing 4.

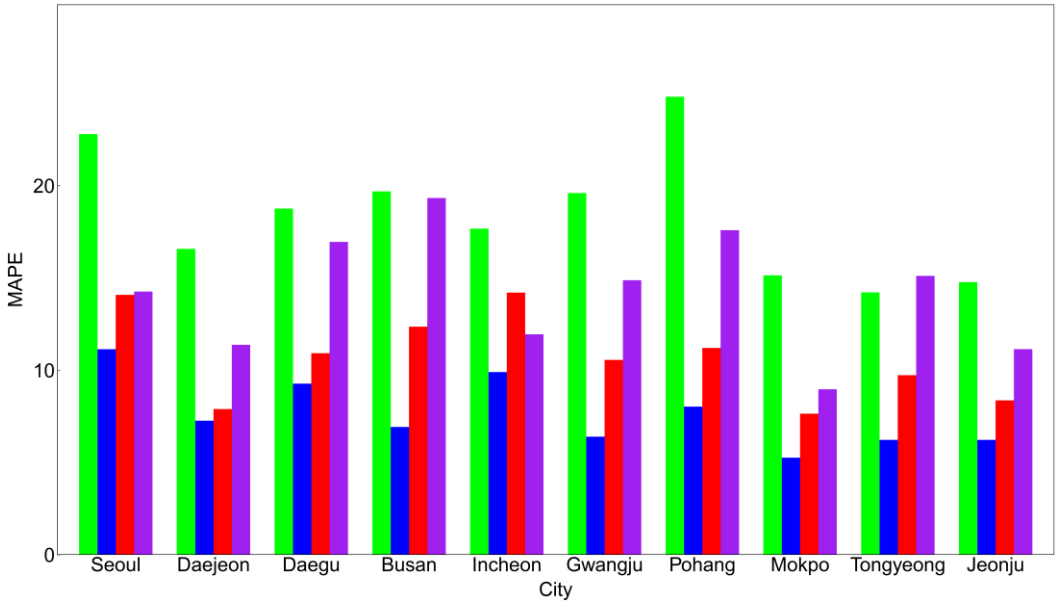


Fig. 14: The lowest MAPE of five NN models for all three kinds of epochs in spring (light green bar), summer (blue bar), autumn (red bar), winter (purple bar) of testing 4.

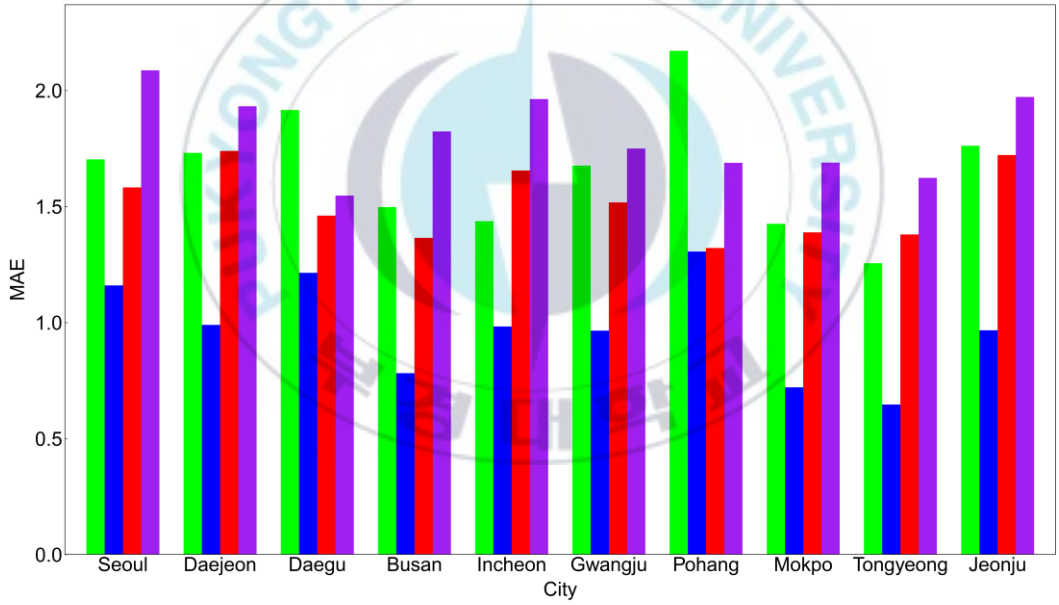


Fig. 15: The lowest MAE of five NN models for all three kinds of epochs in spring (light green bar), summer (blue bar), autumn (red bar), winter (purple bar) of testing 3.

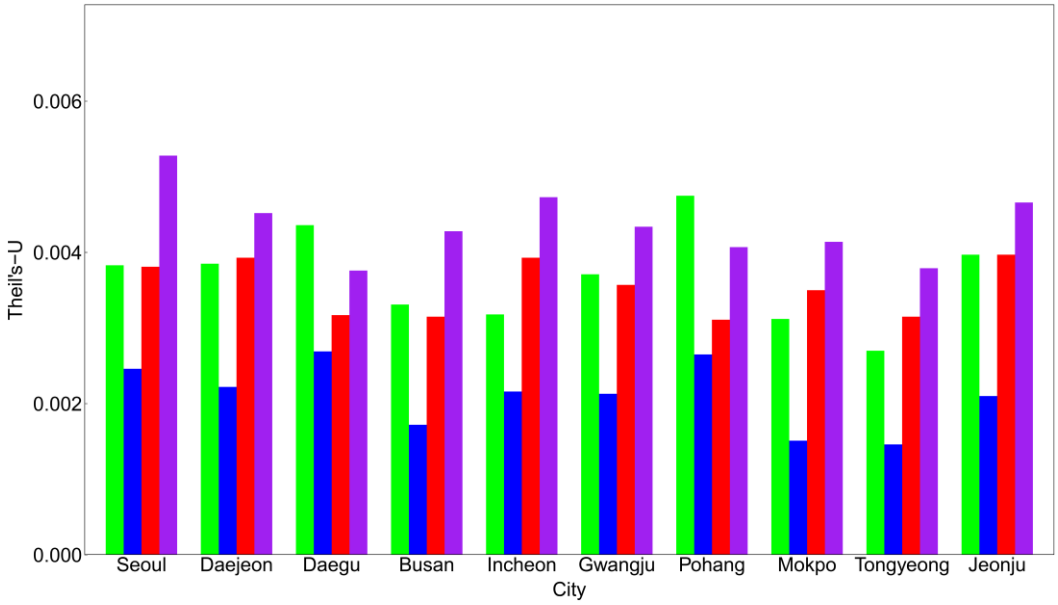


Fig. 16: The lowest Theil's-U of five NN models for all three kinds of epochs in spring (light green bar), summer (blue bar), autumn (red bar), winter (purple bar) of testing 3.

Table. 5: RMSE, MAPE, MAE and Theil's-U values of testing 3.

City	Season	RMSE	MAPE	MAE	Theils'-U ($\times 10^{-3}$)
Seoul	Spring	2.19 (DNN $\eta = 0.1$)	0.595 (DNN $\eta = 0.1$)	1.703 (DNN $\eta = 0.1$)	3.826 (DNN $\eta = 0.1$)
	Summer	1.468 (ANN $\eta = 0.3$)	0.389 (ANN $\eta = 0.3$)	1.16 (ANN $\eta = 0.3$)	2.46 (ANN $\eta = 0.3$)
	Autumn	2.195 (LSTM $\eta = 0.005$)	0.555 (LSTM $\eta = 0.003$)	1.583 (LSTM $\eta = 0.003$)	3.807 (LSTM $\eta = 0.005$)
	Winter	2.901 (DNN $\eta = 0.3$)	0.761 (DNN $\eta = 0.3$)	2.087 (DNN $\eta = 0.3$)	5.278 (DNN $\eta = 0.3$)
Daejeon	Spring	2.207 (ANN $\eta = 0.1$)	0.604 (ANN $\eta = 0.1$)	1.732 (ANN $\eta = 0.1$)	3.85 (ANN $\eta = 0.1$)
	Summer	1.322 (LSTM-PC $\eta = 0.001$)	0.333 (LSTM $\eta = 0.003$)	0.99 (LSTM $\eta = 0.003$)	2.216 (LSTM-PC $\eta = 0.001$)
	Autumn	2.268 (LSTM $\eta = 0.005$)	0.607 (LSTM $\eta = 0.005$)	1.74 (LSTM $\eta = 0.001, \eta = 0.005$)	3.93 (LSTM $\eta = 0.005$)
	Winter	2.492	0.699	1.932	4.519

		(ANN $\eta = 0.3$)	(DNN $\eta = 0.3$)	(DNN $\eta = 0.3$)	(ANN $\eta = 0.3$)
Daegu	Spring	2.509 (ANN $\eta = 0.1$)	0.666 (ANN $\eta = 0.1$)	1.916 (ANN $\eta = 0.1$)	4.361 (ANN $\eta = 0.1$)
	Summer	1.605 (LSTM $\eta = 0.003$)	0.408 (ANN $\eta = 0.3$)	1.215 (ANN $\eta = 0.3$)	2.688 (LSTM $\eta = 0.003$)
	Autumn	1.832 (LSTM $\eta = 0.005$)	0.508 (LSTM $\eta = 0.001$)	1.461 (LSTM $\eta = 0.001$)	3.168 (LSTM $\eta = 0.005$)
	Winter	2.081 (LSTM-PC $\eta = 0.009$)	0.559 (DNN $\eta = 0.1$)	1.547 (DNN $\eta = 0.1$)	3.756 (LSTM-PC $\eta = 0.009$)
Busan	Spring	1.904 (DNN $\eta = 0.1$)	0.521 (DNN $\eta = 0.1$)	1.498 (DNN $\eta = 0.1$)	3.31 (DNN $\eta = 0.1$)
	Summer	1.023 (LSTM-PC $\eta = 0.003$)	0.264 (ANN $\eta = 0.1$)	0.782 (ANN $\eta = 0.1$)	1.72 (LSTM-PC $\eta = 0.003$)
	Autumn	1.832 (LSTM-PC $\eta = 0.001$)	0.472 (LSTM-PC $\eta = 0.001$)	1.365 (LSTM-PC $\eta = 0.001$)	3.145 (LSTM-PC $\eta = 0.1$)
	Winter	2.396 (ANN $\eta = 0.3$)	0.652 (ANN $\eta = 0.3$)	1.824 (ANN $\eta = 0.3$)	4.279 (ANN $\eta = 0.3$)
Incheon	Spring	1.814 (ANN $\eta = 0.1$)	0.504 (ANN $\eta = 0.1$)	1.437 (ANN $\eta = 0.1$)	3.182 (DNN $\eta = 0.1$)
	Summer	1.288 (LSTM-PC $\eta = 0.009$)	0.33 (ANN $\eta = 0.1$, DNN $\eta = 0.1$))	0.983 (ANN $\eta = 0.1$)	2.163 (LSTM-PC $\eta = 0.009$)
	Autumn	2.268 (ANN $\eta = 0.3$)	0.581 (LSTM-PC $\eta = 0.001$)	1.655 (LSTM-PC $\eta = 0.001$)	3.926 (ANN $\eta = 0.3$)
	Winter	2.604 (DNN $\eta = 0.3$)	0.715 (DNN $\eta = 0.3$)	1.963 (DNN $\eta = 0.3$)	4.734 (DNN $\eta = 0.3$)
Gwangju	Spring	2.127 (LSTM $\eta = 0.003$)	0.586 (LSTM $\eta = 0.005$)	1.676 (LSTM $\eta = 0.005$)	3.705 (LSTM $\eta = 0.003$)
	Summer	1.267 (LSTM-PC $\eta = 0.001$)	0.325 (LSTM $\eta = 0.001$)	0.965 (LSTM $\eta = 0.001$)	2.126 (LSTM-PC $\eta = 0.001$)
	Autumn	2.069 (LSTM $\eta = 0.001$)	0.529 (LSTM-PC $\eta = 0.5$)	1.518 (LSTM-PC $\eta = 0.005$)	3.569 (LSTM $\eta = 0.001$)
	Winter	2.412 (LSTM $\eta = 0.003$)	0.629 (DNN $\eta = 0.3$)	1.75 (DNN $\eta = 0.3$)	4.343 (ANN $\eta = 0.3$)
Pohang	Spring	2.737 (LSTM $\eta = 0.001$)	0.753 (LSTM $\eta = 0.001$)	2.171 (LSTM $\eta = 0.001$)	4.752 (LSTM $\eta = 0.001$)

	Summer	1.582 (DNN $\eta = 0.1$)	0.439 (LSTM $\eta = 0.001$)	1.306 (LSTM $\eta = 0.001$)	2.654 (DNN $\eta = 0.1$)
	Autumn	1.806 (LSTM-PC $\eta = 0.001$)	0.459 (LSTM-PC $\eta = 0.001$)	1.321 (LSTM-PC $\eta = 0.001$)	3.107 (LSTM-PC $\eta = 0.001$)
	Winter	2.27 (ANN $\eta = 0.3$)	0.607 (LSTM $\eta = 0.009$)	1.688 (LSTM $\eta = 0.009$)	4.073 (ANN $\eta = 0.3$)
Mokpo	Spring	1.785 (LSTM $\eta = 0.003$)	0.243 (LSTM $\eta = 0.003$)	1.426 (LSTM $\eta = 0.003$)	3.12 (LSTM $\eta = 0.003$)
	Summer	0.896 (DNN $\eta = 0.1$)	0.243 (ANN $\eta = 0.1$, DNN $\eta = 0.1$))	0.721 (ANN $\eta = 0.1$)	1.506 (DNN $\eta = 0.1$)
	Autumn	2.025 (LSTM-PC $\eta = 0.005$)	0.485 (LSTM-PC $\eta = 0.001$)	1.389 (LSTM-PC $\eta = 0.001$)	3.496 (LSTM-PC $\eta = 0.005$)
	Winter	2.294 (DNN $\eta = 0.1$)	0.609 (ANN $\eta = 0.3$)	1.689 (ANN $\eta = 0.3$)	4.14 (DNN $\eta = 0.1$)
Tongyeong	Spring	1.547 (ANN $\eta = 0.1$)	0.438 (LSTM $\eta = 0.007$)	1.256 (LSTM $\eta = 0.007$)	2.695 (ANN $\eta = 0.1$)
	Summer	0.866 (LSTM $\eta = 0.005$)	0.218 (LSTM $\eta = 0.005$)	0.647 (LSTM $\eta = 0.005$)	1.457 (LSTM $\eta = 0.005$)
	Autumn	1.832 (LSTM-PC $\eta = 0.001$)	0.477 (LSTM-PC $\eta = 0.001$)	1.38 (LSTM-PC $\eta = 0.001$)	3.15 (LSTM-PC $\eta = 0.001$)
	Winter	2.118 (ANN $\eta = 0.3$)	0.581 (DNN $\eta = 0.1$)	1.624 (DNN $\eta = 0.1$)	3.791 (ANN $\eta = 0.3$)
Jeonju	Spring	2.272 (ANN $\eta = 0.1$)	0.616 (ANN $\eta = 0.1$)	1.762 (ANN $\eta = 0.1$)	3.968 (ANN $\eta = 0.1$)
	Summer	1.25 (ANN $\eta = 0.1$)	0.325 (ANN $\eta = 0.1$)	0.967 (ANN $\eta = 0.1$)	2.097 (ANN $\eta = 0.1$)
	Autumn	2.296 (ANN $\eta = 0.1$)	0.6 (LSTM $\eta = 0.005$)	1.722 (LSTM $\eta = 0.005$)	3.971 (ANN $\eta = 0.1$)
	Winter	2.579 (ANN $\eta = 0.3$)	0.712 (ANN $\eta = 0.3$)	1.972 (ANN $\eta = 0.3$)	4.658 (ANN $\eta = 0.3$)

Table. 6: RMSE, MAPE, MAE and Theil's-U values of testing 4.

City	Season	RMSE	MAPE	MAE	Theils'-U
Seoul	Spring	13.398 (LSTM	22.798 (LSTM-PC	10.6 (LSTM-PC	0.127833 (LSTM

		$\eta = 0.001$)	$\eta = 0.003$)	$\eta = 0.003$)	$\eta = 0.001$)
	Summer	9.575 (LSTM-PC $\eta = 0.009$)	11.136 (LSTM $\eta = 0.007$)	7.17 (LSTM $\eta = 0.007$)	0.071154 (LSTM $\eta = 0.001$)
	Autumn	11.266 (LSTM $\eta = 0.001$)	14.082 (LSTM $\eta = 0.005$)	8.01 (LSTM $\eta = 0.001$)	0.09058 (LSTM $\eta = 0.001$)
	Winter	11.232 (ANN $\eta = 0.1$)	14.262 (ANN $\eta = 0.1$)	8.377 (ANN $\eta = 0.1$)	0.100045 (LSTM-PC $\eta = 0.003$)
Daejeon	Spring	11.497 (LSTM-PC $\eta = 0.003$)	16.577 (LSTM-PC $\eta = 0.003$)	9.536 (LSTM-PC $\eta = 0.003$)	0.095455 (LSTM-PC $\eta = 0.003$)
	Summer	7.554 (LSTM $\eta = 0.009$)	7.267 (LSTM $\eta = 0.007$)	5.696 (LSTM $\eta = 0.007$)	0.049259 (LSTM $\eta = 0.009$)
	Autumn	7.657 (ANN $\eta = 0.001$)	7.898 (ANN $\eta = 0.1$)	5.742 (ANN $\eta = 0.1$)	0.050439 (ANN $\eta = 0.1$)
	Winter	10.32 (ELM)	11.379 (LSTM $\eta = 0.001$)	7.914 (LSTM $\eta = 0.001$)	0.075578 (ELM)
Daegu	Spring	12.969 (LSTM-PC $\eta = 0.001$)	18.761 (LSTM-PC $\eta = 0.007$)	9.748 (LSTM-PC $\eta = 0.007$)	0.123016 (LSTM-PC $\eta = 0.001$)
	Summer	9.354 (LSTM $\eta = 0.009$)	9.275 (LSTM-PC $\eta = 0.003$)	6.977 (LSTM-PC $\eta = 0.003$)	0.064614 (LSTM $\eta = 0.009$)
	Autumn	9.602 (LSTM-PC $\eta = 0.003$)	10.922 (LSTM-PC $\eta = 0.003$)	7.506 (LSTM $\eta = 0.001$)	0.067986 (LSTM-PC $\eta = 0.003$)
	Winter	13.274 (LSTM-PC $\eta = 0.003$)	16.955 (ANN $\eta = 0.1$)	9.534 (LSTM-PC $\eta = 0.003$)	0.111212 (LSTM-PC $\eta = 0.003$)
Busan	Spring	12.227 (LSTM $\eta = 0.001$)	19.691 (LSTM $\eta = 0.001$)	10.047 (LSTM $\eta = 0.001$)	0.101468 (LSTM $\eta = 0.001$)
	Summer	7.139 (ANN $\eta = 0.1$)	6.928 (ANN $\eta = 0.1$)	5.556 (ANN $\eta = 0.1$)	0.044902 (ANN $\eta = 0.1$)
	Autumn	10.029 (LSTM-PC $\eta = 0.001$)	12.362 (LSTM-PC $\eta = 0.1$)	7.705 (LSTM-PC $\eta = 0.001$)	0.07212 (LSTM-PC $\eta = 0.001$)
	Winter	14.565 (ANN $\eta = 0.3$)	19.332 (LSTM $\eta = 0.005$)	10.424 (LSTM $\eta = 0.001$)	0.132957 (ANN $\eta = 0.5$)
Incheon	Spring	12.714 (ANN $\eta = 0.3$)	17.671 (ANN $\eta = 0.1$)	9.974 (ANN $\eta = 0.1$)	0.097865 (ANN $\eta = 0.5$)
	Summer	9.067 (LSTM-PC $\eta = 0.003$)	9.902 (LSTM-PC $\eta = 0.007$)	7.132 (LSTM-PC $\eta = 0.003$)	0.059158 (LSTM-PC $\eta = 0.003$)
	Autumn	10.798 (LSTM	14.206 (LSTM-PC	8.463 (LSTM	0.080832 (LSTM

		$\eta = 0.001$)	$\eta = 0.009$)	$\eta = 0.001$)	$\eta = 0.001$)
	Winter	9.66 (LSTM-PC $\eta = 0.001$)	11.946 (LSTM-PC $\eta = 0.001$)	7.572 (LSTM-PC $\eta = 0.001$)	0.076879 (LSTM-PC $\eta = 0.001$)
Gwangju	Spring	14.076 (LSTM $\eta = 0.001$)	19.601 (LSTM $\eta = 0.001$)	11.404 (LSTM $\eta = 0.001$)	0.110535 (LSTM $\eta = 0.003$)
	Summer	7.193 (ANN $\eta = 0.1$)	6.401 (ANN $\eta = 0.1$)	5.333 (ANN $\eta = 0.1$)	0.044155 (ANN $\eta = 0.1$)
	Autumn	9.013 (LSTM $\eta = 0.003$)	10.56 (LSTM $\eta = 0.003$)	7.045 (LSTM $\eta = 0.003$)	0.060579 (LSTM $\eta = 0.003$)
	Winter	12.555 (LSTM-PC $\eta = 0.001$)	14.88 (ANN $\eta = 0.3$)	9.785 (ANN $\eta = 0.3$)	0.095765 (LSTM-PC $\eta = 0.001$)
Pohang	Spring	14.421 (LSTM $\eta = 0.001$)	24.822 (LSTM-PC $\eta = 0.003$)	12.089 (LSTM $\eta = 0.001$)	0.123603 (LSTM $\eta = 0.001$)
	Summer	7.81 (ANN $\eta = 0.7$)	8.028 (ANN $\eta = 0.1$)	6.365 (LSTM $\eta = 0.007$)	0.049416 (ANN $\eta = 0.7$)
	Autumn	9.349 (ANN $\eta = 0.3$)	11.208 (ANN $\eta = 0.3$)	7.621 (ANN $\eta = 0.3$)	0.065148 (ANN $\eta = 0.3$)
	Winter	12.935 (ANN $\eta = 0.3$)	17.59 (ANN $\eta = 0.3$)	9.808 (ANN $\eta = 0.3$)	0.111622 (ANN $\eta = 0.3$)
Mokpo	Spring	11.454 (ANN $\eta = 0.5$)	15.142 (ANN $\eta = 0.5$)	9.526 (ANN $\eta = 0.5$)	0.082099 (ANN $\eta = 0.5$)
	Summer	5.732 (LSTM $\eta = 0.005$)	5.263 (LSTM $\eta = 0.005$)	4.296 (LSTM $\eta = 0.005$)	0.035826 (LSTM $\eta = 0.005$)
	Autumn	6.951 (ANN $\eta = 0.1$)	7.647 (ANN $\eta = 0.1$)	5.393 (ANN $\eta = 0.1$)	0.046935 (ANN $\eta = 0.1$)
	Winter	8.109 (ANN $\eta = 0.1$)	8.971 (ANN $\eta = 0.1$)	6.391 (ANN $\eta = 0.1$)	0.057606 (ANN $\eta = 0.1$)
Tongyeong	Spring	10.427 (LSTM-PC $\eta = 0.003$)	14.22 (LSTM-PC $\eta = 0.003$)	8.55 (LSTM-PC $\eta = 0.003$)	0.076959 (LSTM-PC $\eta = 0.003$)
	Summer	6.557 (ANN $\eta = 0.1$)	6.227 (LSTM-PC $\eta = 0.003$)	5.085 (LSTM-PC $\eta = 0.003$)	0.039583 (ANN $\eta = 0.1$)
	Autumn	8.572 (ANN $\eta = 0.1$)	9.732 (LSTM $\eta = 0.005$)	6.729 (LSTM $\eta = 0.005$)	0.058932 (ANN $\eta = 0.1$)
	Winter	12.672 (ANN $\eta = 0.1$)	15.113 (ANN $\eta = 0.1$)	9.415 (ANN $\eta = 0.1$)	0.10209 (ANN $\eta = 0.5$)
Jeonju	Spring	12.046 (LSTM	14.774 (LSTM-PC	9.768 (LSTM	0.088428 (LSTM

	$\eta = 0.003$)	$\eta = 0.009$)	$\eta = 0.003$)	$\eta = 0.003$)
Summer	6.751 (ANN $\eta = 0.9$)	6.225 (LSTM $\eta = 0.009$)	5.308 (ANN $\eta = 0.9$)	0.038613 (ANN $\eta = 0.9$)
Autumn	7.58 (LSTM $\eta = 0.001$)	8.365 (LSTM $\eta = 0.001$)	6.101 (LSTM $\eta = 0.001$)	0.048543 (LSTM $\eta = 0.001$)
Winter	8.965 (ANN $\eta = 0.1$)	11.145 (ANN $\eta = 0.1$)	7.015 (ANN $\eta = 0.1$)	0.066511 (ANN $\eta = 0.1$)

In this subsection, we perform the computer-simulation for two testings as follows: testing 3 has the six input nodes T_{t-2}, T_{t-1}, T_t, H_{t-2}, H_{t-1}, H_t days in the input layer and the one output node T_{t+1} in the output layer, and testing 4 has also the six input nodes T_{t-2}, T_{t-1}, T_t, H_{t-2}, H_{t-1}, H_t days in the input layer and the one output node H_{t+1} in the output layer. We here test the predicted accuracies for temperature T_{t+1} and humidity H_{t+1} at time lag t+1. For testings 3 and 4, we set the five learning rates 0.1, 0.2, 0.3, 0.4, 0.5 for the ANN and the DNN, while the learning rates for LSTM and LSTM-PC are set as 0.001, 0.003, 0.005, 0.007, 0.009, for different train set sizes over three runs (2500, 5000, and 7500 epochs). The predicted values of the ELM are obtained by averaging the results over 2500, 5000, and 7500 epochs. A prediction model is created and the average of the prediction values is obtained through the prediction model was used as the final prediction value.

Figs. 13-16 shows the lowest predicted values of RMSE, MAPE, MAE, and Theil's-U statics of ten cities including all five NN models (the ANN, DNN, LSTM, and LSTM-PC) for all three kinds of epochs in four seasons of ten cities in testings 3 and 4, respectively. Tables 5 and 6 is illustrated the comparison of the RMSE, MAPE, MAE and Theil's-U statistics in four seasons of ten cities in testings 3 and 4, respectively.

From Fig. 13 and Table 6 of testing 4, the RMSE of LSTM has a lowest value of 5.732 in summer in Mokpo (rank1), rather than 6.557 in summer in Tongyeong (rank 2) and 6.751 in summer in Jeonju (rank 3). The lowest MAPE value of LSTM is 5.263 in summer in Mokpo (rank1), lower than 6.225 in summer in Jeonju (rank 2) and 6.227 in summer in Tongyeong (rank 3), as shown in Fig. 14 of testing 4.

From Fig. 15 and Table 5 of testing 3, the MAE of LSTM has a lowest value of 0.647 in summer in Tongyeong (rank1), rather than 0.721 in summer in Mokpo (rank 2) and 0.967 in summer in Jeonju (rank 3). The lowest Theil's-U value of LSTM is 1.457 in summer in Tongyeong (rank1), lower than 1.506 in summer in Mokpo (rank 2) and 2.097 in summer in Jeonju (rank 3), as shown in Fig. 16 of testing 3.

Hence, we find that the RMSE of LSTM in humidity prediction has a lowest value of 5.732 at η=0.005 in summer in Mokpo, while the lowest MAE value of LSTM is 0.647 at η=0.01 in summer in Tongyeong in temperature prediction.

4-3 Lowest values of temperature and humidity prediction

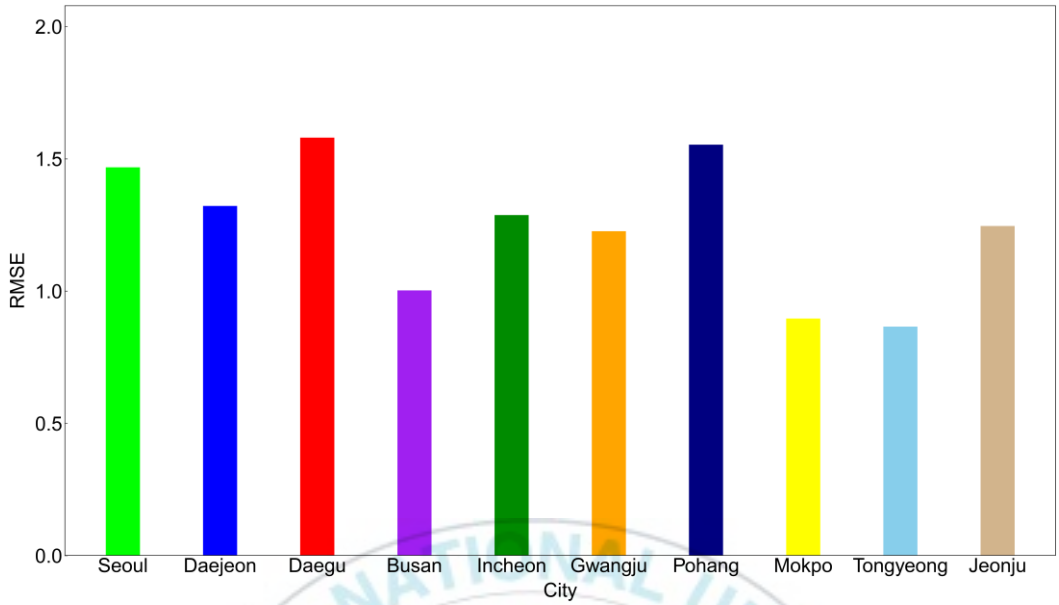


Fig. 17: The lowest RMSE of temperature prediction of ten cities for five NN models in four seasons in testings 1 and 3.

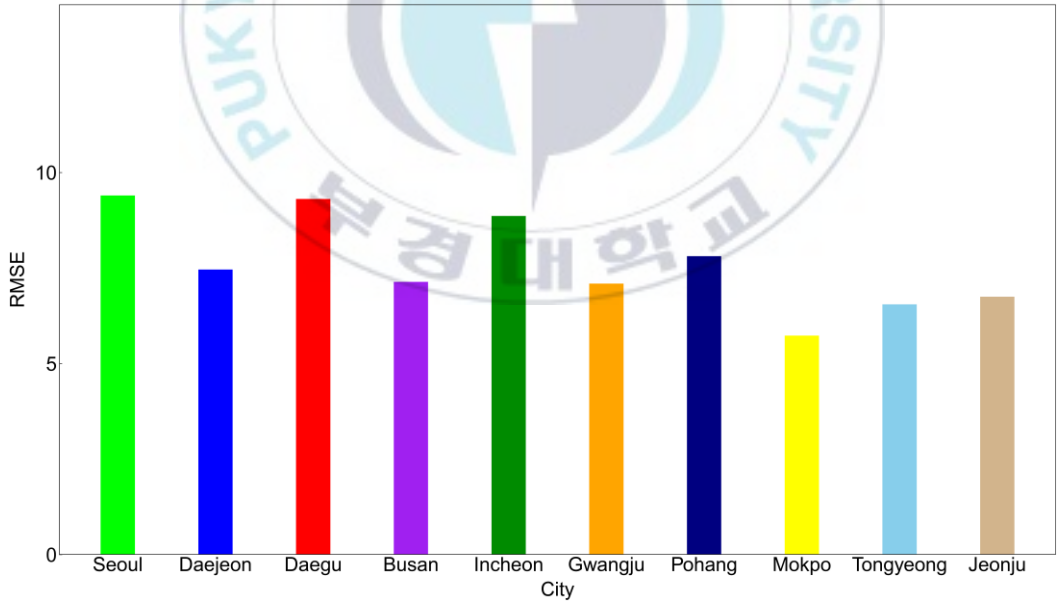


Fig. 18: The lowest RMSE of humidity prediction of ten cities for five NN models in four seasons in testings 2 and 4.

Table. 7: The lowest RMSE values of temperature prediction

City	Season	Model	RMSE	testing
Seoul	Summer	ANN ($\eta = 0.3$)	1.468	3
Incheon	Summer	LSTM-PC ($\eta = 0.009$)	1.288	3
Daejeon	Summer	LSTM-PC ($\eta = 0.001$)	1.322	3
Daegu	Summer	LSTM ($\eta = 0.001$)	1.58	1
Busan	Summer	LSTM ($\eta = 0.001$)	1.003	1
Pohang	Summer	LSTM ($\eta = 0.005$)	1.554	1
Tongyeong	Summer	LSTM ($\eta = 0.005$)	0.866	3
Gwangju	Summer	LSTM-PC ($\eta = 0.001$)	1.227	1
Jeonju	Summer	LSTM-PC ($\eta = 0.001$)	1.246	1
Mokpo	Summer	DNN ($\eta = 0.1$)	0.896	3

Table. 8: Lowest RMSE values of humidity prediction

City	Season	Model	RMSE	testing
Seoul	Summer	LSTM-PC ($\eta = 0.003$)	9.399	2
Incheon	Summer	LSTM-PC ($\eta = 0.005$)	8.864	2
Daejeon	Summer	ANN ($\eta = 0.1$)	7.46	2
Daegu	Summer	ANN ($\eta = 0.1$)	9.307	2
Busan	Summer	ANN ($\eta = 0.1$)	7.139	4
Pohang	Summer	ANN ($\eta = 0.7$)	7.81	4
Tongyeong	Summer	LSTM ($\eta = 0.009$)	6.549	2
Gwangju	Summer	ANN ($\eta = 0.7$)	7.093	2
Jeonju	Summer	ANN ($\eta = 0.9$)	6.751	4
Mokpo	Summer	LSTM ($\eta = 0.005$)	5.732	4

Fig. 17 and Fig. 18 shows the lowest RMSE of temperature prediction and humidity prediction of ten cities for the ANN, DNN, LSTM, LSTM-PC, and ELM in four seasons in testings 1-4, respectively.

From Fig. 17 and Table 7, in temperature prediction, the RMSE of LSTM has a lowest value of 0.866 in summer in Tongyeong (rank1), rather than 0.896 in summer in Mokpo (rank 2) and 1.003 in summer in Busan (rank 3).

From Fig. 18 and Table 8, in humidity prediction, the RMSE of LSTM has a lowest value of 5.732 in summer in Mokpo (rank1), rather than 6.549 in summer in Tongyeong (rank 2) and 6.751 in summer in Jeonju (rank 3).

Hence, we find that the RMSE of LSTM in temperature prediction has a lowest value of 0.866 at learning rate η=0.005 in summer in Tongyeong, while the lowest RMSE value of LSTM in humidity prediction is 5.732 at learning rate η=0.005 in summer in Tongyeong.

Ⅴ. Conclusion

In this paper, the daily average temperature and relative humidity of 10 major cities (Seoul, Daejeon, Daegu, Busan, Incheon, Gwangju, Pohang, Mokpo, Tongyeong, and Jeonju) in Korea are predicted using the NN models. We have simulated using the ANN with one hidden layer, DNN with two hidden layers, LSTM, LSTM-PC, and ELM. We simulate four testings: that is, input four nodes T_{t-1}, T_t, H_{t-1}, H_t days of temperature (testing 1) and humidity (testing 2), and input six nodes T_{t-2}, T_{t-1}, T_t, H_{t-2}, H_{t-1}, H_t. T_{t+1} and H_{t+1} days of temperature (testing 3) and humidity (testing 4). The five learning rates for the ANN and the DNN are set to 0.1, 0.3, 0.5, 0.7, and 0.9, while those for LSTM and LSTM-PC are set to 0.001, 0.003, 0.005, 0.007, 0.009, for 2500, 5000, 7500 epochs. The predicted values of the ELM are obtained by averaging the results trained 2500, 5000, and 7500 epochs. From the result of outputs, the root mean squared error (RMSE), mean absolute percentage error (MAPE), mean absolute error (MAE), Theil-U statistics are simulated for performance evaluation, and we compare each other after the manipulation of five NN models.

We simulate and analyze the testings 1-4 as follows: (ⅰ) In testing 1, the RMSE value is 1.583 for the 7500 training epochs of the ANN (learning $\eta = 0.1$) in spring in Tongyeong. The RMSE value of the LSTM-PC ($\eta = 0.003$) in summer in Tongyeong is 0.878 for 2500 training epochs, while that for the LSTM-PC ($\eta = 0.005$) is 1.72 for the LSTM-PC ($\eta = 0.005$) for 5000 training epochs in autumn in Busan. The RMSE value of the ANN

($\eta = 0.3$) is 2.046 for 2500 training epochs in winter in Daegu. Among the four seasons, the LSTM-PC shows good performance in two seasons (summer, autumn). Particularly, when the LSTM ($\eta = 0.003$) is trained 2500 training epochs in summer in Tongyeong, the RMSE has the smallest value with 0.878. (ii) In testing 2, In spring, the RMSE value of the LSTM ($\eta = 0.001$) is 10.479 for the 5000 training epochs in spring in Tongyeong. The RMSE value of the ANN ($\eta = 0.1$) is 5.839 for 2500 training epochs in summer in Mokpo. The RMSE of the LSTM ($\eta = 0.005$) was 6.891 for 7500 training epochs in autumn in Mokpo, The RMSE value is 8.16 for 2500 training epochs of the ANN ($\eta = 0.1$) in winter in Mokpo. When the ANN ($\eta = 0.1$) is trained 2500 training epochs in the summer in Mokpo, the RMSE has the smallest value with 5.839. (iii) In testing 3, the RMSE value is 1.547 for 5000 training epochs of the ANN ($\eta = 0.1$) in spring in Tongyeong, and the LSTM ($\eta = 0.005$) has an RMSE of 0.866 for 5000 training epochs in summer in Tongyeong. The RMSE value is 1.806 for 5000 training epochs of the LSTM-PC ($\eta = 0.001$) in autumn in Pohang, while that is 2.081 for the 7500 training epoch of The LSTM-PC ($\eta = 0.009$) in winter in Daegu. The LSTM in summer and the LSTM-PC in autumn and winter show good performances, and as in testing 1, the LSTM series also show good performances. (iv) In testing 4, the RMSE value is 10.427 for the 7500 training epochs of the LSTM-PC ($\eta = 0.003$) in spring in Tongyeong. The RMSE value of LSTM ($\eta = 0.005$) is 5.732 for 2500 training epochs in summer in Mokpo. The RMSE value is 6.951 for 2500 training epochs of the ANN ($\eta = 0.1$) in autumn in Mokpo, while that ($\eta = 0.1$) is 8.109 for 7500 training epochs in winter. In this case, The LSTM value outperforms any values of other models in summer in Mokpo.

Particularly, from the computer-simulation in order to predict the temperature in spring, the RMSE of the ANN in Tongyeong shows the smallest value for 5000 training epochs in testing 3 (the temperature predicted in the input layer with six input nodes). In summer, The RMSE of the LSTM in testing 3 has the smallest value in Tongyeong for 5000 training epochs. In the autumn, The LSTM-PC in testing 1 (the temperature predicted in the input layer with four input nodes) has the smallest value in Busan for 5000 training epoch. In winter, the ANN in testing 1 shows the smallest error in 2500 training epoch of Daegu. In the temperature prediction, when using the LSTM model in testing 3 in Tongyeong in the summer among the four seasons, we find that the smallest value of RMSE is 0.866. In the simulation to predict the humidity in spring, the LSTM-PC in Tongyeong for 7500 training epochs has the smallest RMSE in testing 4 (the humidity predicted in the input layer with six input nodes). The LSTM in testing 4 has the smallest value in 2500 training epochs of Mokpo in the summer, while the RMSE of the LSTM in testing 2 (the humidity predicted in the input layer with four input nodes) in the autumn has the smallest value for 7500 training epochs in Mokpo. In winter, the RMSE of the ANN has smallest value in testing 4 trained 7500 epochs in Mokpo. In the humidity prediction,

when using in the summer in Mokpo, the RMSE of LSTM model is shown the smallest value with 5.732. In both the temperature and humidity predictions, the RMSEs are the smallest in summer.

From calculated results, the difference between the actual value and the predicted value of humidity is greater than the temperature, and the reason for this is that the actual value of humidity is more chaotic than that of temperature as shown in the previous paper [98]. The reason why land cities are given less error than coastal cities is that the data of land cities are inherently more chaotic, non-linear, and non-stationary time series.

Our result provides the evidence that the LSTM is an effective method of predicting one meteorological factor (temperature) rather than the DNN. Prediction value of temperature among our result is consistent to the result of Chen et al. [85], and they found this value via deep learning network from Chinese stock data. We will conduct a study to further improve the accuracy of the meteorological element prediction model by applying a learning method that applies optimization algorithms such as genetic algorithm and particle swarm optimization to other types of neural network models such as and back-propagation algorithms [99,100]. There exists complicatedly and rebelliously correlated relation between several meteorological factors (temperature, wind velocity, humidity, surface hydrology, heat transfer, solar radiation, surface hydrology, land subsidence, and so on), the research in future can treat and apply complex network theory to input variables of the meteorological data, and the LSTM and LSTM-PC models can promote and develop the predictive performance.

VI. References

[1] P.D. Jones, T.J. Osborn, K.R. Briffa, The Evolution of Climate Over the Last Millennium, Science 292 (2001) 662-667.

[2] P. Lynch, The origins of computer weather prediction and climate modeling, J. Comput. Phys. 227 (2008) 3431-3444.

[3] Chen B, Gong H, Chen Y, Li. X, Zhou. C, Lei. K, Zhu. L, Duan. L, Zhao. X, Land subsidence and its relation with groundwater aquifers in Beijing Plain of China, Science of Total Environment 735 (2020) 139111.

[4] J.M. Wallace, E.M. Rasmusson, T.P. Mitchell, V.E. Kousky, E.S. Sarachik, H. von Storch, On the structure and evolution of enso-related climate variability in the tropical pacific: lessons from toga, J. Geophys. Res. 103 (C7) (1998) 14241-14259.

[5] M. Latif, D. Anderson, T.P. Barnett, M.A. Cane, R. Kleeman, A. Leetmaa, A review of the predictability and prediction of ENSO, J Geophys. Res. 103 (1998) 14375-14393.

[6] A.G. Barnston, C.F. Ropelewski, Prediction of enso episodes using canonical correlation analysis, J. Clim 5 (1992) 1316-1345 .
[7] A. Wu, W.W. Hsieh, B. Tang, Neural network forecasts of the tropical pacific sea surface temperatures, Neural Netw. 19 (2006) 145-154.
[8] A.G. Barnston, H.M. van den Dool, S.E. Zebiak, T.P. Barnett, M. Ji, D.R. Rodenhuis, Longlead seasonal forecasts - where do we stand, Bull. Am. Meteor. Soc. 75 (1994) 2097-2114.
[9] N.K. Chauhan, K. Singh, A Review on Conventional Machine Learning vs Deep Learning, 2018 International Conference on Computing, Power and Communication Technologies (GUCON) (2018) 347–352.
[10] S.B. Kotsiantis, Supervised machine learning: A review of classification techniques, Informatica 31 (2007) 249-268.
[11] R. Muthukrishnan, R. Rohini, LASSO: A Feature Selection Technique In Predictive Modeling For Machine Learning, IEEE International Conference on Advances in Computer Applications (ICACA) (2016) 18-20.
[12] W. Lai. M. Zhou. M, F. Hu, K. Bian, Q. Song, A new DBSCAN parameters determination method based on improved MVO, IEEE Access 7 (2019) 104085-104095.
[13] C. Ding, X. He, K-means Clustering via Principal Component Analysis, Proceedings of the 21st International Conference on Machine Learning (2004) 29-36.
[14] Khan. K., Rehman. S. U, Aziz. K, Fong. S, Sarasvady. S, DBSCAN: Past, Present and Future, In Applications of Digital Information and Web Technologies (ICADIWT), IEEE (2014) 232-238.
[15] Glasera. J. I, Benjamin. A. S, Farhoodi. R, Kording. K. P, The roles of supervised machine learning in systems neuroscience, Progress in Neurobiology 175 (2019) 126-137.
[16] Silver. D, Huang. A, Maddison. C. J, Guez. A, Sifre. L, Driessche. G, Schrittwieser. J, Antonoglou. I, Panneershelvam. V, Lanctot. M, Dieleman. S, Grewe. D, Nham. J, Kalchbrenner. N, Sutskever. I, Lillicrap. T, Leach. M, Kavukcuoglu. K, Graepel. T, Hassabis. D, Mastering the game of Go with deep neural networks and tree search, Nature 529 (2016) 484-489.
[17] W.S. Mcculloch, Pitts. W, A logical calculus of the ideas immanent in nervous activity, Bulletin of Mothemnticnl Biology 52 (1990) 99-115.
[18] Hebb. D. O, *The Organization of Behavior*, McGill Univentgli (1949).
[19] Rosenblatt. F, The perceptron: a probabilistic model for information storage and organization in the brain, Psych. Rev. 65 (1958) 386-408.
[20] Minsky. M, Papert. S. A, *Perceptrons: An Introduction to Computational Geometry,* MIT Press (1969).
[21] Rumelhart. D. E, Hinton. G. E, Williams. R. J, Learning Internal Representations by Error Propagation, MIT Press, Cambridge, (1986) 318-362.

[22] W.A. Little, The existence of persistent states in the brain, Math. Biosci. 19 (1974) 101-120; W.A.Little, Q.L. Shaw, A statistical theory of short and long term memory, Behav. Biol. 14 (1975) 115-133.
[23] J.J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, Proc. Natl. Acad. Sci. USA 79 (1982) 2554-2558; Neurons with graded response have collective computational properties like those of two-state neurons, Proc. Natl. Acad. Sci. USA 81 (1984) 3088-3092.
[24] D.J. Amit, H. Gutfreund, H. Sompolinsky, Spin-glass models of neural networks, Phys. Rev. A, 32 (1985) 1007-1018; Storing Infinite Numbers of Patterns in a Spin-Glass Model of Neural Networks, Phys. Rev. Lett. 55 (1985) 1530-1533.
[25] Werbos. P. J, BEYOND REGRESSION: NEW TOOLS FOR PREDICTION AND ANALYSIS IN THE BEHAVIORAL SCIENCES, Ph. D thesis of Harvard University (1974).
[26] Rumelhart. D. E, Hinton. G. E, Williams. R. J, Learning representations by back-propagating errors, Nature 323 (1986) 533-536.
[27] Bishop. C.M, Neural networks and their applications, Review of Scientific Instruments 65 (1994) 1803-1832.
[28] Elman. J. L, Finding structure in time, Cong. Sci. 14 (1990) 179-211.
[29] Werbos. P. J, Backpropagation through time: what it does and how to do it, Proceedings of the IEEE 78 (1990) 1550-1560.
[30] Hochreiter. S, Schmidhuber. J, Long short-term memory, Neural Comput 9 (1997) 1735-1780.
[31] Gers. F. A, Schmidbuber. J, 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 3 (2000) 189-194.
[32] Cho. K, Van Merriënboer. B, Gulcehre. C, Bahdanau. D, Bougares. F, Schwenk. H, Bengio. Y, Learning phrase representations using RNN encoder-decoder for statistical machine translation, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (2014) 1724-1734.
[33] LeCun. Y, Bottou. L, Bengio. Y, Haffner. P, Gradient-based learning applied to document recognition, PROC OF THE IEEE 86 (1998) 2278-2324.
[34] Huang. G. -B, Zhu. Q. -Y, Siew. C. -K, Extreme learning machine: theory and applications, Neurocomputing 70 (2006) 489-501.
[35] Koutsoukas. A, Monaghan. K. J, Li. X, Huan. J, Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data, J. Cheminform 9 (2017) 42.
[36] Smith. L. N, Cyclical learning rates for training neural networks, In 2017 IEEE Winter Conference on Applications of Computer Vision (WACV) (2017) 464-472.
[37] Loshchilov. I, Hutter. F, SGDR: Stochastic gradient descent with warm restarts, arXiv

preprint arXiv:1608.03983 (2016).
[38] Glorot. X, Bengio. Y, Understanding the difficulty of training deep feedforward neural networks, In Proceedings of the thirteenth international conference on artificial intelligence and statistics (2010) 249-256.
[39] Duchi. J, Hazan. E, Singer. Y, Adaptive subgradient methods for online learning and stochastic optimization, Journal of machine learning research 12 (2011) 2121-2159.
[40] Kingma. D. P, Ba. J. L, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).
[41] Ruder. S, An overview of gradient descent optimization algorithms, arXiv preprint arXiv:1609.04747 (2016).
[42] Jarrah. M, Salim. N, A recurrent neural network and a discrete wavelet transform to predict the Saudi stock price trends, Int. J. Adv. Comput. Sci. Appl 10 (2019) 155-162.
[43] Vargas. M. R, dos Anjos. C. E. M, Bichara. G. L. G, Evsukoff. A. G, Deep leaming for stock market prediction using technical indicators and financial news articles, In 2018 International Joint Conference on Neural Networks (IJCNN), IEEE (2018) pp. 1-8.
[44] Wang. J, Wang. J, Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks, Neurocomputing 156 (2015) 68-78.
[45] Khare. K, Darekar. O, Gupta. P, Attar, V. Z, Short term stock price prediction using deep learning, In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), IEEE (2017) pp. 482-486.
[46] Zhang. K, Zhong. G, Dong. J, Wang. S, Wang. Y, Stock market prediction based on generative adversarial network, Procedia computer science 147 (2019) 400-406.
[47] Pawar. K, Jalem. R. S, Tiwari. V, Stock market price prediction using LSTM RNN, *Emerging Trends in Expert Applications and Security,* Springer, Singapore, (2019) 493-503.
[48] Mehtab. S, Sen. J, Dutta. A, Stock price prediction using machine learning and LSTM-based deep learning models, arXiv preprint arXiv:2009.10819 (2020).
[49] Hiransha. M, Gopalakrishnan. E. A, Menon. V. K, Soman. K. P, NSE stock market prediction using deep-learning models, Procedia computer science 132 (1018) 1351-1362.
[50] Selvin. S, Vinayakumar. R, Gopalakrishnan. E. A, Menon. V. K, Soman. K. P, Stock price prediction using LSTM, RNN and CNN-sliding window model, In 2017 international conference on advances in computing, communications and informatics (icacci), IEEE (2017) 1643-1647.
[51] Wu. Y, Tan. H, Qin. L, Ran. B, Jiang. Z, A hybrid deep learning based traffic flow prediction method and its understanding, Transportation Research Part C: Emerging Technologies 90 (2018) 166-180.
[52] Zhang. D, Kabuka. M. R, Combining weather condition data to predict traffic flow: a

GRU-based deep learning approach, IET Intelligent Transport Systems 12 (2018) 578-585.
[53] Arif. M, Wang. G, Chen. S, Deep learning with non-parametric regression model for traffic flow prediction, In 2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), IEEE (2018) 681-688.
[54] Dai. X, Fu. R, Lin. Y, Li. L, Wang. F. -Y, DeepTrend: A deep hierarchical neural network for traffic flow prediction, arXiv preprint arXiv:1707.03213, (2017).
[55] Xiao. Y, Yin. Y, Hybrid LSTM neural network for short-term traffic flow prediction, Information 10 (2019) 105.
[56] Yang. H. -F, Dillon. T. S, Chen. Y. P. -P, Optimized structure of the traffic flow forecasting model with a deep learning approach, IEEE transactions on neural networks and learning systems 28 (2016) 2371-2381.
[57] Yu. L, Zhao. J, Gao. Y, Lin. W, Short-Term Traffic Flow Prediction Based On Deep Learning Network, In 2019 International Conference on Robots & Intelligent System (ICRIS), IEEE (2019) 466-469.
[58] Yu. H, Wu. Z, Wang. S, Wang. Y, Ma. X, Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks, Sensors 17 (2017) 1501.
[59] Cifuentes. J, Marulanda. G, Bello. A, Reneses. J, Air temperature forecasting using machine learning techniques: a review, Energies 13 (2020) 4215.
[60] Zhu. S, Heddam. S, Wu. S, Dai. J, Jia. B, Extreme learning machine-based prediction of daily water temperature for rivers, Environmental Earth Sciences 78 (2019) 202.
[61] Mendes. D, Marengo. J. A, Temporal downscaling: a comparison between artificial neural network and autocorrelation techniques over the Amazon Basin in present and future climate change scenarios, Theoretical and Applied Climatology 100 (2010) 413-421.
[62] Yen. M. -H, Liu. D. -W, Hsin. Y. -C, Lin. C. -E, Chen. C. -C, Application of the deep learning for the prediction of rainfall in Southern Taiwan, Sci, Rep. 9 (2019) 1-9.
[63] Ise. T, Oba. Y, Forecasting climatic trends using neural networks: An experimental study using global historical data, Frontiers in Robotics and AI 6 (2019) 32.
[64] Guo. Y, Cao. X, Liu. B, Peng. K, El Niño Index Prediction Using Deep Learning with Ensemble Empirical Mode Decomposition, Symmetry 12 (2020) 893.
[65] Zhang. Y, Zhang. C, Zhao. Y, Gao. S, Wind speed prediction with RBF neural network based on PCA and ICA, Journal of Electrical Engineering 69 (2018) 148-155.
[66] Salman. A. G, Heryadi. Y, Abdurahman. E, Suparta. W, Single layer & multi-layer long short-term memory (LSTM) model with intermediate variables for weather forecasting, Procedia Computer Science 135 (2018) 89-98.
[67] Elhoseiny. M, Huang. S, Elgammal. A, Weather classification with deep convolutional

neural networks, In 2015 IEEE International Conference on Image Processing (ICIP), IEEE (2015) 3349-3353.
[68] Poornima. S, Pushpalatha. M, Prediction of rainfall using intensified LSTM based recurrent neural network with weighted linear Units, Atmosphere 10 (2019) 668.
[69] Ramírez. M. C. V, Ferreira. N. J, Velho. H. F. C, Linear and nonlinear statistical downscaling for rainfall forecasting over southeastern Brazil, Weather and forecasting 21 (2006) 969-989.
[70] Zhang. Z, Pan. X, Jiang. T, Sui. B, Liu. C, Sun. W, Monthly and Quarterly Sea Surface Temperature Prediction Based on Gated Recurrent Unit Neural Network, J. Mar. Sci. Eng. 8 (2006) 249.
[71] Deng. L, Deep learning: from speech recognition to language and multimodal processing, APSIPA Transactions on Signal and Information Processing 5 (2016) 1-15.
[72] Lim. C. P, Woo. S. C, Loh. A. S, Osman. R, Speech recognition using artificial neural networks, In Proceedings of the First International Conference on Web Information Systems Engineering 1, IEEE (2000) 419-423.
[73] Venayagamoorthy. G. K, Moonasar. V, Sandrasegaran. K, Voice recognition using neural networks, In Proceedings of the 1998 South African Symposium on Communications and Signal Processing-COMSIG'98 (Cat. No. 98EX214), IEEE (1998) 29-32.
[74] Nichie. A, Mills. G. A, Voice Recognition Using Artificial Neural Networks And Gaussian Mixture Models, International Journal of Engineering Science and Technology 5 (2013) 1120.
[75] Tian. C, Ma. J, Zhang. C, Zhan. P, A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network, Energies 11 (2018) 3493.
[76] Hamedmoghadam. H, Joorabloo. N, Jalili, M, Australia's long-term electricity demand forecasting using deep neural networks. arXiv preprint arXiv:1801.02148, (2018).
[77] Lalis. J. T, Maravillas. E, Dynamic forecasting of electric load consumption using adaptive multilayer perceptron (AMLP), In 2014 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), IEEE (2014) 1-7.
[78] Zheng. J, Xu. C, Zhang. Z, Li. X, Electric Load Forecasting in Smart Grids Using Long-Short-Term-Memory based Recurrent Neural Network, In 2017 51st Annual Conference on Information Sciences and Systems (CISS). IEEE (2017) 1-6.
[79] Park. D. C, El-Sharkawi. M. A, Marks II. R. J, Atlas. L. E, Damborg. M. J, Electric load forecasting using an artificial neural network, IEEE transactions on Power Systems 6 (1991) 442-449.
[80] Azadeh. A, Ghaderi. S. F, Tarverdian. S, Saberi. M, Integration of artificial neural

networks and genetic algorithm to predict electrical energy consumption, Applied Mathematics and Computation 186 (2007) 1731-1741.
[81] Yokoyama. R, Wakui. T, Satake, R, Prediction of energy demands using neural network with model identification by global optimization, Energy Conversion and Management 50 (2009) 319-327.
[82] Y. Tao, K. Hsu, A. Ihler, X. Gao, S. Sorooshian, A Two-Stage Deep Neural Network Framework for Precipitation Estimation from Bispectral Satellite Information, J. Hydrometeor. 19 (2018) 393-408.
[83] Nabipour. M, Nayyeri. P, Jabani. H, Mosavi. A, Salwana. E, S. Shahab, Deep Learning for Stock Market Prediction, Entropy 22 (2020) 840.
[84] Yudong. Z, Learn. W, Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network, Expert systems with applications 36 (2009) 8849-8854.
[85] Chen. L, Qiao. Z, Wang. M, Wang. C, Du. R, Stanley. H. E, Which artificial intelligence algorithm better predicts the Chinese stock market?, IEEE Access 6 (2008) 48625-48633.
[86] Sermpinis. G, Dunis. C, Laws. J, Stasinakis. C, Forecasting and trading the EUR/USD exchange rate with stochastic Neural Network combination and time-varying leverage, Decision Support Systems 54 (2012) 316-329.
[87] Vijh. M, Chandola. D, Tikkiwal. V. A, Kumar. A, Stock Closing Price Prediction using Machine Learning Techniques, Procedia Computer Science 167 (2020) 599-606.
[88] Wang. J, Wang. J, Fang. W, Niu. H, Financial time series prediction using elman recurrent random neural networks, Computational intelligence and neuroscience, 2016.
[89] Moustra. M, Avraamides. M, Christodoulou. C, Artificial neural networks for earthquake prediction using time series magnitude data or seismic electric signals, Expert systems with applications 38 (2011) 15032-15039.
[90] González. J, Yu. W, Telesca. L, Earthquake Magnitude Prediction Using Recurrent Neural Networks, Multidisciplinary Digital Publishing Institute Proceedings 24 (2019) 22.
[91] Kashiwao. T, Nakayama. K, Ando. S, Ikeda. K, Lee. M, Bahadori. A, A neural network-based local rainfall prediction system using meteorological data on the Internet: A case study using data from the Japan Meteorological Agency, Applied Soft Computing 56, (2017) 317-330.
[92] Zhang. Z, Dong. Y, Temperature Forecasting via Convolutional Recurrent Neural Networks Based on Time-Series Data, Complexity (2020).
[93] Bilgili. M, Sahin. B, Prediction of long-term monthly temperature and rainfall in Turkey, Energy Sources, Part A: Recovery, Utilization, and Environmental Effects 32 (2009) 60-71.
[94] Mohammadi. K, Shamshirband. S, Motamedi. S, Petković. D, Hashim. R, Gocic. M, Extreme learning machine based prediction of daily dew point temperature, Computers

and Electronics in Agriculture 117 (2015) 214-225.
[95] Maqsood, I, Khan. M. R, Abraham. A, Weather forecasting models using ensembles of neural networks, *Intelligent Systems Design and Applications*, Springer, Berlin, Heidelberg, (2003) 33-42.
[96] Buscema. M, Back propagation neural networks, Substance use & misuse 33 (1998) 233-270.
[97] Greff. K, Srivastava. R. K, Koutník. J, Steunebrink. B. R, Schmidhuber. J, LSTM: A search space odyssey, IEEE transactions on neural networks and learning systems 28 (2016) 2222-2232.
[98] K.-H. Shin, W. Baek, K. Kim, C.-H. You, K.-H. Chang, D.-I. Lee, S. S. Yum, Physica A 523 (2019) 778–796.
[99] Chen. N, Xiong. C, Du. W, Wang. C, Lin. X, Chen. Z, An Improved Genetic Algorithm Coupling a Back-Propagation Neural Network Model (IGA-BPNN) for Water-Level Predictions, Water 11 (2019) 1795.
[100] Du. J, Liu. Y, Yu. Y, Yan. W, A prediction of precipitation data based on support vector machine and particle swarm optimization (PSO-SVM) algorithms, Algorithms 10 (2017) 57.