Received November 5, 2018, accepted November 26, 2018, date of publication December 4, 2018, date of current version December 27, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2884827

Stable Forecasting of Environmental Time Series via Long Short Term Memory Recurrent Neural Network

KANGIL KIM[1], (Member, IEEE), DONG-KYUN KIM[2], JUNHYUG NOH[3], AND MINHYEOK KIM[4]
[1]Computer Science and Engineering Department, Konkuk University, Seoul 05029, South Korea
[2]Fisheries Science Institute, Chonnam National University, Yeosu 59626, South Korea
[3]Computer Science and Engineering Department, Seoul National University, Seoul 08826, South Korea
[4]Seocho R&D Center, LG Electronics, Seoul 06772, South Korea

Corresponding author: Dong-Kyun Kim (dkkim1004@gmail.com)

This work was supported by Konkuk University in 2016.

ABSTRACT In a recent decade, deep neural networks have been applied for many research areas after achieving dramatic improvements of accuracy in solving complex problems in vision and computational linguistics area. However, some problems, such as environmental modeling, are still limited to benefit from the deep networks because of its difficulty in collecting sufficient data of learning process. In this paper, aside from the accuracy issue, we raise another property—stability—of the deep networks useful for even such data-limited problems, especially in time-series modeling. Recurrent neural networks with memory cell structures, a deep network, can be deemed as a more robust network structure for long-term forecasting under coarse data observation and associated uncertainties, including missing values and sampling/measurement errors. The stability in forecasting is induced from balancing impact of inputs over all time steps in the networks. To analyze this property in various problem conditions, we adapt the recurrent networks with memory structure to environmental time-series problems, such as forecasting water pollution, air pollution, and ozone alarm. In the results, the recurrent networks with memory showed better performance of forecasting in non-stationary environment and long-term time lags.

INDEX TERMS Environmental forecasting, recurrent neural network, long short term memory, stability.

I. INTRODUCTION

Artificial neural networks (ANN) of machine learning algorithms have been most widely applied to environmental modeling due to its powerful performance [1]–[11]. Since the process-based and statitistical approaches had limitation in model accuracy to predict complex system dynamics, ANN has been regarded as one of the alternative modeling methods to improve the model accuracy. Although there are some skepticisms in using ANN due to lack of explanation power related to its black-box structure, the power to portray complex mechanisms of ecosystem has been improved to some extent by doing a sensitivity analysis. As a consequence, plenty of ANN research has recently shown the satisfactory level of predictions of environmental phenomena such as algal blooms, water body's toxicity, rainfall-runoff, ground-level ozone concentration, PM2.5 air pollution [1], [2], [7]–[11].

In many applications of machine learning, main focus is often to determine an appropriate model structure for improving only predictive power. Particularly for forecasting purposes, crucial is identifying the linkage (e.g., periodicity and serial correlations) between input and output at a long time span. However, the linkage among ecological entities may be even more complex, as they are nonlinearly intertwined. Due to the extremely high degree of system complexity of our interest, the current generation of machine learning seems inclined to improve model's predictability rather than model's interpretability. In this respect, conventional ANNs such as multilayer perceptron (MLP) have evolved to recursive structure such as recurrent neural network (RNN). In contrast to MLP, RNN's time-across vectors derive a competitive advantage, so that RNN may benefit from the cumulative information derived from previous observations. Although MLP can also use time-lag information, the fixed time-lagged inputs have limitation on accommodating temporal dependency between input and output. In this regard, the use of recursive structure in ANNs is essential for time-series forecasting models.

Nevertheless, RNN still suffers from the well-known gradient vanishing problem [12], [13]. This problem indicates the phenomenon to reduce the difference of partial derivatives between parameters at shallower layers when we stack deep layers. The derivatives build a gradient used for updating models toward an local optimum. In the case of recurrent neural networks, passing hidden vectors to the next layer is regarded as staking one more layer and therefore increasing the used time steps to derive the final prediction amplifies the gradient vanishing problem. The biggest disadvantage caused by the vanishing problem is that RNN cannot correctly forecast from observation in the long term. If the model Increases the interval between time steps, the model may be able to cover the long-term dependency as a macroscopic model but the model cannot capture sophisticated dependency appearning in only microscopic phenomena.

Using a memory cell with RNN is a promising solution for this problem as long short term memory (LSTM) [12]. In this type of networks using memory, it uses gated structure to indicate when to combine received information and when to apply it to the next layer. This gating can maintain high gradient values of parameters in linking inputs at early time step to a final output, which reduces the gradient vanishing effect. Particularly for long-term time-series forecasting, a deep network such as LSTM accounts more effectively for temporal dependencies between inputs than RNN does. For this reason, LSTM has shown a dramatic improvement compared to RNN in many natural language processing applications as machine translation, question answering, and usual sequential labeling problems [14]–[17] where the length of dependent words can be extremely long from the number of words in a sentence to a document.

Aside from the accuracy improvement, LSTM can offer more benefits from a perspective of forecasting stability. Since LSTM maintains high impact to the final prediction for inputs observed in all time steps, we can expect that the contribution to build the final prediction value is distributed over all time steps. This balanced contribution reversely implies that any corruption of information at a specific time step loses its impact to the final prediction. By doing so, deep structure such as LSTM can provide more stable predictive power in time-series forecast, compared to conventional neural networks.

Our study aims to analyze the predictive stability of LSTM, comparing with other popular ANNs, in environmental forecasting. We also assess the impact of LSTM's computational stability on predictive/forecasting performances using complex and highly nonlinear environmental data. Moreover, we offer the instruction to guide readers how to access and use the method used in this study. Ultimately, it is expected that the powerful deep learning neural networks can shed light on environmental forecasting systems in the context of water and air pollutions.

In Section II, we provide a background knowledge about LSTM networks and environmental problems. Section III addresses a property of LSTM enhancing stability in forecasting. Section IV proposes LSTM frameworks designed for each environmental problem, and Section V shows detailed experimental design to show the effect of the LSTM property. Section VI analyzes the results and Section VII discusses their meaning and impact for the prediction problem. We make a conclusion at Section VIII.

II. BACKGROUND

A. LONG SHORT TERM MEMORY BASED RECURRENT NEURAL NETWORK

Recurrent neural network based on long short term memory (LSTM) has been proposed in late 1990 [12], which is a type of recurrent neural network (RNN) [18] for handling time-series data. As RNN, this model has time steps and recursive edges between networks, but it derives the context vector from a memory-shaped network.

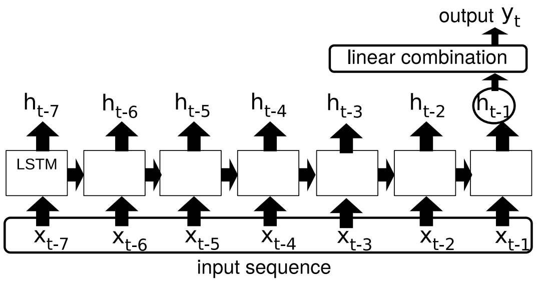


FIGURE 1. An example of forecasting process in LSTM for time-series problems.

A LSTM model is shown in Fig. 1. In detail, a LSTM network receives a context vector from previous time step and an input vecor from data. Then it determines the next context vector and gate vectors to control memory cell state vector. The gates are usually composed of input, forget, output gate, and a memory cell. The mechanism of this memory cell is deriving a context vector for the next time step from adjusting raw context vectors through the gates. Given an input data at time t and a context vector h, a raw cell vector and input vectors for each gate are created by one hidden layer. Then the cell vector is multiplied by the input vector at the input gate. The cell input is added to given previous cell vector weighted by the forget vector. The result vector is then controlled again by the output vector. There are many variants of composing the LSTM cell networks, but they commonly uses the mechanisms for learning temporal impact.

The most well-known benefit of using LSTM instead of RNN is relaxing gradient vanishing problem [13]. In predicting a class at time t, the observation in old time steps is difficult to affect the classification. In LSTM, the memory cell including the gates can store the time to emit for old observations, so that it can resolve the vanishing in RNN based structures.

Therefore, LSTM is used for learning a model in domains where the time delay between observation and class may be long such as language translation [17]. Variants as gated recurrent unit (GRU) and simple recurrent unit [19], [20] were reported to reduce training speed and to use less model parameters, but significant improvement of prediction accuracy has not been reported yet. In this paper, we use widely-known LSTM structure for more generality in applying our conclusion.

This vanilla LSTM is composed of a hidden layer and an output layer. The inputs of the hidden layer are observations $\boldsymbol{x}_t$ at time t and context $\boldsymbol{h}_{t-1}$ at $t-1$. From the inputs, we derive a hidden context vector $\boldsymbol{h}_t$ by the definition of a LSTM cell as the following equations.

$$\begin{aligned}
\boldsymbol{i} &= \mathrm{sigm}(W_i\boldsymbol{x}_t + U_i\boldsymbol{h}_{t-1} + \boldsymbol{b}_i)\\
\boldsymbol{o} &= \mathrm{sigm}(W_o\boldsymbol{x}_t + U_o\boldsymbol{h}_{t-1} + \boldsymbol{b}_o)\\
\boldsymbol{f} &= \mathrm{sigm}(W_f\boldsymbol{x}_t + U_f\boldsymbol{h}_{t-1} + \boldsymbol{b}_f)\\
\boldsymbol{c}_t &= \mathrm{sigm}(W_cl\boldsymbol{x}_t + U_c\boldsymbol{h}_{t-1} + \boldsymbol{b}_c)\\
\boldsymbol{h}_t &= \tanh(\boldsymbol{i}\cdot\boldsymbol{c}_t + \boldsymbol{f}\cdot\boldsymbol{c}_{t-1})\cdot\boldsymbol{o}\\
\boldsymbol{y}_t &= W_1\boldsymbol{h}_t + \boldsymbol{b}_1
\end{aligned}$$

where $\boldsymbol{y}_t$ is a raw output vector used for calculating cost values. The gate and cell vectors $\boldsymbol{i}, \boldsymbol{o}, \boldsymbol{f}, \boldsymbol{c}_t$ are generated by passing the concatenation of input $\boldsymbol{x}_t$ and $\boldsymbol{h}_{t-1}$ through a sigmoid layer composed of weight matrices W, U, and a corresponding bias vector. After obtaining $\boldsymbol{h}_t$ from the vectors, the LSTM derives the final vector $\boldsymbol{y}_t$ by applying linear combination of the output layer composed of W_1 and $\boldsymbol{b}_1$. The final vector $\boldsymbol{y}_t$ is used for calculating various costs depending on applications.

B. TEMPORAL DEPENDENCY IN VARIOUS NEURAL NETWORKS FOR TIME SERIES

MLP is the basic, but good performing approach to learn complex nonlinear relationships between input and output variables. Yet, it is known that the network is not suitable for time-series problems, because the time length between input and output is constant. Given that meaningful time-lags between time-series variables are unknown, the model should cover the maximum time steps to reduce the risk of missing important old observations. The risk may be relaxed by increasing the length of time-lags. However, the change of length induces a dimensional increase of model, often reducing predictivity due to data sparsity. Another problem of MLP in learning the temporal dependency is assuming uniform distribution over time steps. In an ideal condition, MLP can represent temporal dependencies such as decayed impact to the output variable by time. However, usual MLP is trained with random initialization of weights and training mechanisms have no bias on which variables to amplify. Therefore, the approximated nonlinear function of MLP easily uses only a few variables selected without consideration of temporal relations.

RNN is another popularly used neural network, which solves the restriction of the fixing input sites in MLP. However, in the perspective of learning complex temporal dependency, RNN is still limited by the gradient vanishing problem [13] which critically decreases the impact of observations with large time lag. This phenomenon appears when the network layers are deep, usually more than two, caused by the information for training network weights disappears at the shallow layers (close to the input layer) in gradient-based optimization algorithms. The gradient is a vector composed of partial derivatives of model parameters with respect to a function, which indicates the update amount of the parameters to maximally increase the function value at the current model. Repeating updates by calculating the gradient at each model, the algorithms converge to a local optimal. This gradient is almost neutralized over parameters in the shallow layers, because input variables are used to generate values of all hidden nodes. Any reward from the final cost function to hidden nodes is shared for all input variables, so the gradient in the shallow layers is not any more informative to indicate how to update for obtaining better cost. In RNN, time steps are similar to depth because the same network is applied to the results of the previous networks at each time step. Therefore this vanishing problem equally appears in terms of time steps and observation before two time steps is almost completely ignored in training.

A representative structure of using memory structure with RNN is LSTM. As mentioned in Section II, it solves the gradient vanishing problem of RNN via using the memory structure. Thus it can maintain high gradient of each input regardless of its time step and generate sufficiently high impact of inputs at specific time steps to outputs if it is required. It is clear that this property dramatically ease the restriction of learning the temporal dependency compared to RNN. In the perspective of bias on time steps, MLP and LSTM both have no restriction on learning any temporal dependency.

C. WATER QUALITY FORECASTING

The eutrophication of freshwater systems is an internationally important issue, seen worldwide, resulting from the interaction of a wide variety of factors [21]. In particular, cyanobacterial blooms are one of the world's commonest - and problematic - water pollution and eutrophication phenomena, occurring in both lotic and lentic ecosystems [22]. Thus eutrophication has been an adequate experimental field related to prediction of undesirable conditions in ecological modeling. Since river ecosystems with large populations generally revolve around multiple anthropogenic impacts such as dam construction and water resource supply limitations, there are more difficult challenges in prediction of such phenomena in relation to water quality deterioration [23]. Due to inaccurate (or dissatisfied) prediction associated with great uncertainty in process knowledge, many recent modeling projects have attempted to reveal the possible relationship among plankton dynamics, physicochemical parameters and climate conditions using machine learning algorithms [24]–[26].

In water quality prediction of river, the long time dependency may not be very important, because strong flow or flooding may completely reset the states of ecosystems.

Even in previous literature, the networks showed a successful improvement of the water quality prediction. However it is not yet deeply discussed how complex temporal dependecies should be considered in the problem which may be a key issue to obtain more accurate models. In previous works using MLP [3]–[7], most of the models manually determines the length of time steps of inputs, called as memory (different to the memory cell in LSTM). This approach implicitly suffers from the uncertain constraint by fixing the time steps. In the other approaches using RNN [1], [2], models can learn from inputs in various time steps, however the gradient vanishing problem has never been raised as an issue which implies their implicitely downgraded performance by ignoring long temporal dependencies.

D. AIR POLLUTION FORECASTING

Air pollution is another envionmental domain which we confront in relation to public health. Various chemical compounds, such as carbon monoxide (CO), nitrogen dioxide (NO_2), ozone (O_3), sulfur dioxide (SO_2) and particles, have become the main causes of many respiratory diseases. For instance, lung cancer, cardiopulmonary illnesses are most typical disease that caused by air pollutant [32], [33].

In previous works, MLP is used to predict maximum concentrations of air pollutants into the Palermo [34]. RNN with LSTM and GRU is also studied in analyzing China National Environmental Monitoring Center(CNEMC) data [35]. In air pollution forecasting, it is also shown that 10 day measurement can reduce errors less than 2% in neural networks [29].

E. OZONE ALARM

Ground-level ozone concentration has recently drawn attention to the public, since its abrupt fluctuation and high concentration were considered harmful and toxic to human in relation to respiratory disease, decrease of crop productivity, and climate changes. Previous studies have shown that rising ambient ozone concentrations caused visible injury on plants and threats yields of wheat and rice [36], [37]. It is also reported that ozone can increase possibility of human disease like Pneumonia and Pulmonary disease [38]. Moreover ozone is greenhouse gas causing global warming and it makes hurt ecosystem as well as human beings [39].

Neural networks and Bayesian networks have been applied for ground-level ozone forecasting [40], [41] which selected predictors based on linear and partial auto-correlation with nonlinear sensitivity analysis. In those works, effective time lags is approximately 14 days and fully connected feed forward neural network shows that it can successfully estimate daily maximum ozone level.

III. TEMPORAL IMPACT OF NEURAL NETWORKS

The impact of each input in various time steps is differently trained in neural networks by their distinguished optimization characteristics. In this paper, we focus on three major networks containing MLP, RNN, and LSTM rather than covering all variants introduced with different topologies.

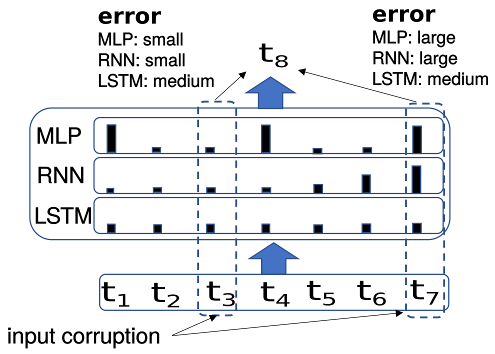


FIGURE 2. Distinguished distributions of impacts of input noise values to a final output between neural networks.

A. IMPACT DISTRIBUTION OF NEURAL NETWORKS

As different properties in learning mechanisms of RNN, LSTM, and MLP shown in Section II, the impact of inputs variables on the outputs has been distributed in different ways. Using the explained bias, we illustrate an example distribution in Fig. 2. In the case of MLP, there is no bias on the impact of inputs, the selected variables completely relies on the process of optimization. This optimization and balance distribution is, however, somewhat unstable because of huge variance of local optima by slight initial change of weights. Typically in RNN, timely closer input variables exert larger influences on prediction of desired outputs, while LSTM maintains the influences of inputs throughout all time steps minimizing bias. If an input at a time step has noise or is omitted, prediction of output relies on its estimated value and the other correct inputs. In the case of MLP, the output will maintain correct value even if the wrong input has small impact, but the output may critically change if the impact of the wrong input is large. Unbalanced distribution of RNN also has the same problem. LSTM, compared to two models, balances the impact between all time steps and therefore it can overcome the sudden drop of output accuracy. MLP and LSTM both have no bias in determining lengths of sensitive inputs, but the impact of inputs to an output may be all focused on a few time steps in MLP while it is distributed over time steps by combinatorial process of input information over time steps in LSTM.

B. EMPIRICAL EVIDENCE OF IMPACT BALANCING OF LSTM

Our preliminary result of sensitivity evaluation supports the impact distribution example as shown in Fig. 4. To obtain the result, we create a new test set by changing the value of a randomly selected input variable at a time step in training data The changed value is uniformly distributed values in the range of mean$\pm$standard deviation of the input variable in the training data. After training the neural networks, we evaluated the standard deviation of an output material in the newly generated test set and averaged it over runs with changing different variable and time step.

In this result, we can confirm that LSTM generate almost no error on output even if wide range of error for a single input occurs. Compared to LSTM, RNN shows higher sensitivity in the latest time steps and MLP shows significantly high errors. In the case of MLP, the high average sensitivity is observed because sensitivity is extremely high or low in different runs. More detailed environment of this sensitivity test is introduced in Section V.

The explanation of impact distribution and this experimental evidence implies that LSTM balances the impact on inputs between time steps and the balancing reduces the error in losing some information of inputs.

C. BENEFIT OF IMPACT BALANCING

Expected benefit of balancing impact of input time steps is robustness to unexpectedly generated input errors. Because RNN and MLP both have unbalanced impacts, randomly selected time steps may generate large error, which leads to sudden drop of forecasting performance. However, LSTM distributes all impacts to over time steps and therefore such sudden drop does not occur. In another interpretation, we can expect that remaining correct time steps supports LSTM to generate sufficiently accurate outputs even if there is a few errors are introduced.

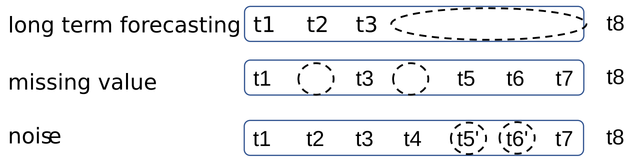


FIGURE 3. Unstable observation-environment in forecasting (long-term forecasting, noise, missing values).

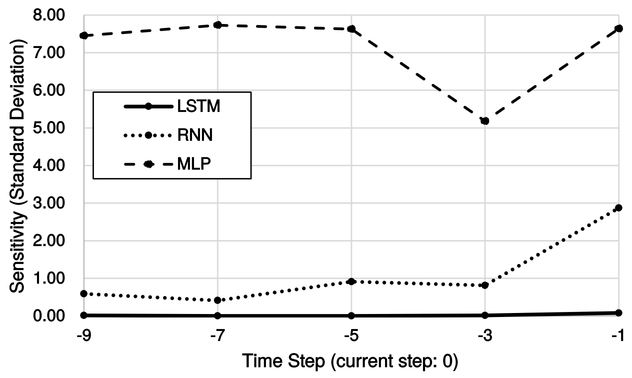


FIGURE 4. Standard deviation (sensitivity) of output values generated by random sampling of input variables in various time steps.

The scenario of using this robustness is shown in Fig. 3. First of all, we can use it for stable forecasting when missing values are introduced. Missing value is a common phenomenon where data is observed by human. For this reason, human-collecting data are easy to include noise in the use of trained models. Another example is long-term forecasting. This example is understood as missing values from a few latest time steps. In the example, models should forecast the output from given input data with long time lags. Because LSTM maintains impact of early time steps, it can again preserve somewhat accurate outputs compared to other frameworks.

IV. NETWORK CONFIGURATION FOR ENVIRONMENTAL PROBLEMS

To analyze effects of balancing temporal impact of LSTM, we evaluate MLP, RNN, and LSTM in three environmental problems to forecast following materials. This section is composed of the following four categories of configuration.

- common setting between problems
- specific setting for air pollution
- specific setting for water quality
- specific setting for ozone alarm

A. COMMON SETTING

We selected the simplest topologies for each network to evaluate distinct effects of impact balancing. As the figure, MLP is set to have a hidden and an output layer and activation of hidden node is sigmoid function. RNN is a simple Elman network [18], which concatenates hidden vector generated at previous time step with current input vector. LSTM is designed as the network represented in Section II, which is a primitive LSTM topology. To apply them to the same time-series modeling, we set MLP to receive inputs over time steps as one single input vectors. Thus, the dimension of input vector of MLP is the multiplication of maximum number of time steps and the dimension of the input vector at a step in RNN and LSTM.

$$\boldsymbol{y}_{t+l} = \mathrm{ANN}(I_k) \quad \text{(raw output of ANN)} \tag{1}$$
$$I_k = (\boldsymbol{x}_i)_{t-k}^{t-1} \tag{2}$$
$$\hat{v} = \mathrm{output}(\boldsymbol{y}_{t+l}) \quad \text{(for forecasting)} \tag{3}$$

The goal of the networks is to predict correct output v by generating its estimate $\hat{v}$. To calcaulte the estimate with a problem-specific output function, MLP, RNN, and LSTM generate a raw output vector $\boldsymbol{y}_{t+l}$ at time step $t + l$ from the given vector sequence I_k composed of input parameter values $\boldsymbol{x}_i$ at a time step i in $t - k$ to $t - 1$. For each problem, the ANNs use specifically defined output functions. In implementation of the ANNs, we modified open source[1] frequently used in many other areas. We randomly initialize weights and biases of networks from [-0.08, 0.08] for all networks. This small range is selected to locate hidden vectors near the origin of vector space, which helps training by locating all vectors near effective regions of hyperplanes represented by a neuron.

To optimize the networks, we used gradient-based optimization method with adaptive strategy to control learning rate via momentum [42]. Mini-batch approach is used for exploration of optimization process, which separately updates parameters for each mini-batch. The size of mini-batch is 64 for all problems and mini-batches of full training data are randomly shuffled in training. The mini-batch size is one of usual small setting in training neural networks to give sufficient randomness in gradient-based optimization for more exploration ability.

To suppress overfitting, we equally applied sufficient regularization via batch normalization to all networks [43] instead

[1]http://deeplearning.net/tutorial/

of other regularizations as L1, L2, and dropout [44]. In MLP and RNN, batch normalization is only applied to the hidden layer. In LSTM, the normalization is applied to the outputs of LSTM cells which are used as inputs to the final output layer. To reduce errors according to difference of statistics between training and test set, we linearly interpolated batch-normalized and original activations with a rate parameter and decayed the parameter from 0.99 to near 10^{-5} in training.

In fact, our focus is not on finding the best hyper parameter settings and topologies for environmental time series, but on evaluating the difference of impact balancing of neural networks. For this reason, fine-tuning of settings is out of interest of this paper and we selected reasonably good-working and ordinary settings. Therefore, many recently proposed techniques to train deep networks can consistently improve overall performance of the used networks as using many stacks of layers, rectified linear units, pre-training of layers, group normalization, better initialization, drop-out, and so on.

TABLE 1. Selected input and output variables in this paper (δ: change, $*$: three values at K-indea sea level pressures of 500, 700, 850 hpa).

problem	$\|\boldsymbol{x}_t\|$	input variables $\boldsymbol{x}_t$	output v
water quality (WQ) [7], [27]	17	conductivity alkalinity turbidity Secchi depth silica DO saturation phosphate dam discharge (four sites) flow rate rainfall water temperature pH nitrate	chlorophyll-a
air pollution (AP) [28], [29]	7	Temperature Benzene Total Nitrogen Oxides Nitrogen Dioxide Relative Humidity Absolute Humidity Non-Metanic Hydrocarbons	CO
ozone alarm (OA) [30], [31]	72	temperature (T) per hour max, average, total T wind speed (WS) per hour max, average, total WS T* δT* East to west WS* δ East to west WS* Relative Humidity* δRelative Humidity* North to south WS* δNorth to south WS* Geopotential Height* δGeopotential Height*	danger or not

B. AIR POLLUTION

To evaluate the performance of networks in forecasting air pollution, we used the data released by Center for Machine Learning and Intelligent Systems, University of California, Irvine.[2] We selected input variables and output variables as shown in Table 1. The variables are base setting and are adjusted for various empirical analyses in in Section V.

[2]https://archive.ics.uci.edu/ml/datasets/Air+quality

The output function for this problem is a scaled sigmoid function defined as

$$\hat{v} = \alpha \times \mathrm{sigm}(y_{t+l}). \tag{4}$$

where y_{t+l} is the raw output value generated by linear combination of the latest hidden vector. This function is used for restricting the range of prediction values to $(0, \alpha)$. This objective function is designed for the purpose of forecasting whether the risk of pollution is sufficiently high or not, rather than the exact metric values. For this reason, We empirically tuned $\frac{\alpha}{2}$ to be located at sigm(0) where neural networks most sensitively react. The constant α for air pollution is set by 24.

The cost function is defined as follows.

$$\mathrm{cost}(D) = \sqrt{\sum_{(I_k, v) \in D} \frac{(v - \hat{v})^2}{v}} \tag{5}$$

This cost is a root mean squared error normalized by correct output value v. As the reason of deciding the constant for output scaling, we selected this normalization to train neural networks more sensitive to the value of extremely high peaks, but an intermediate value working as a criterion to decide dangerous pollution.

This data is collected from sensors to observe the amount of pollution materials and time interval between samples is regularly one hour. To evaluate performance in varying data set, we created cross validation sets for time-series. Each set is composed of a training and a test set which covers sequential time steps by their size N_{tr} and N_{te}. All cross-validation sets are disjoint and sequetially located in total time-series data. Given the fixed amount of time-series data, we adjusted N_{tr} and N_{te} to the sufficiently large numbers for representing most phenomena of the problem. The validation set to select the best trained model is generated by random selection from the training set. Detailed data statistics of this cross-validation set is shown in Table 2.

TABLE 2. Statistics of cross validation data set (WQ: water quality, AP: air pollution, OA: ozone alarm, a random seed is used for each run in cross-validation).

problem	sets	cases per set			input	a step	total
		train	valid	test	steps	unit	steps
WQ	12	188	20	52	8	week	832
AP	10	2790	310	620	10	hour	9300
OA	4	365	365	365	30	day	2190

C. WATER QUALITY

For forecasting water quality, we used observed ecological time-series data for 16 years from Nakdong river in South Korea [7], [27]. The interval of observation is inconsistent because of difficulty of human observation, but roughly one week. We linearly interpolated this data to generate weekly time-series. For brevity, we skip the description of all detailed measuring methods and conditions for each material, and the map of the river with its study sites, which are all shown in the paper [7], [27] Selected input and output variables are shown in Table 1. As air pollution data, we built cross-validation sets as shown in Table 2. Output and loss function have the equal forms to air pollution problem, but the scale constant α is set to 400.

D. OZONE ALARM

For ozone alarm problem, we used the daily data released by the equal center of releasing the data for air pollution problem.[3] In this problem, we set input and output variables as Table 1 and generated cross-validation data as Table 2. To generate missed values, we used linear interpolation.

TABLE 3. Summary of model setting (SD: standard deviation, nRMSE: normalized root mean squared error, CE: cross-entropy).

parameter	LSTM	RNN	MLP
common setting between problems			
input			
normalization	mean, SD	mean, SD	mean, SD
hidden layer			
number	1	1	1
dimension	128	128	128
activation	sigmoid	sigmoid	sigmoid
output			
dimension	1	1	1
function	scaled sigmoid	scaled sigmoid	softmax
scale	400	24	–
cost	nRMSE	nRMSE	CE
optimizer	AdaDelta	AdaDelta	AdaDelta
regularization	batch-norm	batch-norm	batch-norm
WQ specific setting			
input dimension	17	17	127
input time steps	8	8	1
AP specific setting			
input dimension	8	8	80
input time steps	10	10	1
OA specific setting			
input dimension	72	72	2160
input time steps	30	30	1

Because this problem is a binary classification, cost function is set to softmax differently to the scaled sigmoid of the other regression problems.

$$p(v) = \frac{e^{(y_{t+l})_v}}{\sum_w e^{(y_{t+l})_w}} \tag{6}$$

which is a vector composed of probabilities to select a correct class v. Then, its cost function is defined as cross entropy between the probability distribution of correct classes and estimated classes.

$$\text{cost}(D) = \sum_{(I_t,v)\in D} -\log p(v) \tag{7}$$

Parameters of models for each problem are summarized in Table 3. In the topology setting, 1 hidden layer is used for all networks in all problems. This setting is the most basic but potential to represent all complex relations between input and outputs unless the hidden layer is too small. Then, we empirically selected 128 nodes to give sufficient expression power to the networks and used a recently developed effective regularization method as batch normalization to avoid overfitting.

[3]http://archive.ics.uci.edu/ml/datasets/Ozone+Level+Detection

V. EXPERIMENT SETTING

We performed following tests to evaluate robustness of the impact balancing of LSTM in the configuration of networks.

- long-term forecasting
- autoregressive long-term forecasting
- input noise
- missing value imputation

TABLE 4. Experiment conditions for test in unstable environment (WQ: water quality, AP: air pollution, OA: ozone alarm, SD: standard deviation).

condition	WQ	AP	OA
time lag	{1,2,3,4,5,6,7,8}	{1,6,12,18,24}	{1,3,5,7,9}
missing rate	{0.0, 0.1,· · · ,0.9}	{0.0, 0.1,· · · ,0.9}	–
noise rate			
SD=0.1	{0.0, 0.1,· · · ,0.9}	{0.0, 0.1,· · · ,0.9}	–
SD=1.0	{0.0, 0.1,· · · ,0.9}	{0.0, 0.1,· · · ,0.9}	–

A. LONG-TERM FORECASTING

We evaluated performance in longterm forecasting varying time lag between output and the last input. Table 4 shows intervals of the lag change in each application. To obtain statistically meaningful results, we applied the cross validation set. Increasing time lag reduces the total time-series length, but the composition of data is retained by using slightly smaller time-series.

B. AUTOREGRESSIVE LONG-TERM FORECASTING

We performed the test in autoregressive modeling problems which can be more accustumed to long-term forecasting. In autoregressive modeling, the output variable to forecast uses its observed value at previous time step, and therefore the differential relation between outputs are trained rather than the relation between other inputs. In this test, we only added the output variable as an input variable to the setting for long-term forecasting test.

C. FORECASTING WITH NOISY INPUT

Noise of observation is common phenomenon in environmental problems. To evaluate the effect of the impact balancing with noisy inputs, we generated gaussian noise to original input data whose input and output delay is 1 time step. In detail, the Gaussian noise is generated for stochastically selected variables with a given standard deviation, denoted as noise-level in this paper. The exact equation for noise generation is as follows.

$$x' = x \cdot (1 + N(0, \sigma)) \tag{8}$$

where x is original value and x' is the noise-imputed value and N is the gaussian distribution. The rate of selected time steps varies from 10% to 90% for each noise level which is set to 0.1 or 1.

D. FORECASTING WITH MISSING VALUE

Missing observation is also easy to occur in problems relying on human observation. As the noise test, we imputed missing

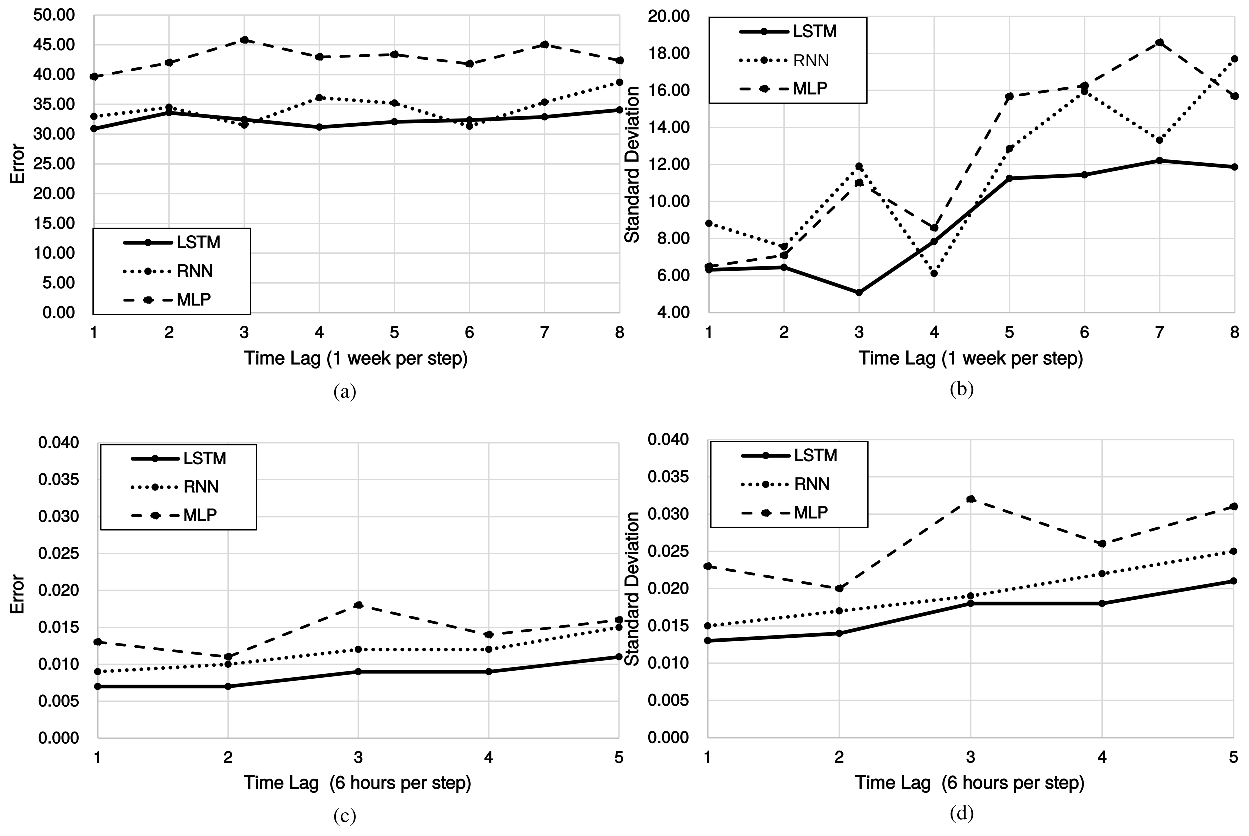


FIGURE 5. **Test errors in various time lags between final input step and output step for water quality and air pollution forecasting. (a) Water quality mean. (b) Water quality SD. (c) Air pollution mean. (d) Air pollution SD.**

values to stochastically selected time steps with varying probabilities from 10% to 90%. When observed values at a time step are missed, we imputed mean vector of all parameters over the full training set. Therefore, the missing value test is supposed to show effects in more generalized sampling environment than the noise test and the amount error may be stronger than the noise test. Detailed configuration values are shown in Table 4.

VI. RESULTS

A. LONG-TERM FORECASTING PERFORMANCE

The results of performance test for three environmental problems are shown in Fig. 5 and Fig. 9. We evaluated mean and standard deviation of root mean squared error widely used in general regression evaluation. This measure is different to the cost function used in training.

In the water quality problem, average errors slowly increased in all networks by the increase of time lag between input and output with small fluctuation. Standard deviation also increased by the time with higher rate than the mean values. Compared to MLP and RNN, the errors of LSTM increases slowly. LSTM is the most accurate in forecasting the closest step. These phenomena also appeared in air pollution forecasting. The phenomena are similar in air pollution problem.

In ozone alarm test, we evaluated F1-measure ($\frac{2\times\text{precision}\times\text{recall}}{\text{precision}+\text{recall}}$), which is a widely used evaluation metric for binary classification. In this problem, F1 values of networks decreased by longer time lags, which implies the increase of accuracy. In average, LSTM shows the highest values compared to RNN and MLP. At the longest lag, LSTM maintained higher values compared to RNN and MLP critically dropping the errors. In case of MLP, the accuracy decrease is relatively small, but overall accuracy is lower than the other recurrent models. The accuracy of RNN was higher than LSTM at the shortest time lag, but it rapidly decreases to a worse point than LSTM at the longest lag.

B. LONG-TERM FORECASTING WITH AUTOREGRESSIVE MODEL

We only performed autoregressive modeling only for air pollution problem because of restriction on computational costs. in the results shown in Fig. 10, the errors increased only in the MLP result. Compared to nonautoregressive modeling, MLP was more accurate. Standard deviation showed the similar pattern of increasing errors by time lags as the slow increase of errors in LSTM and rapid increase of MLP.

C. MISSING VALUE IMPUTATION

The results of missing value imputation is shown in Fig. 8. We performed this test for regression problems. In the graph (a), increasing missing rate induced more errors to all models. Increasing rate of RNN is the highest and ends up to

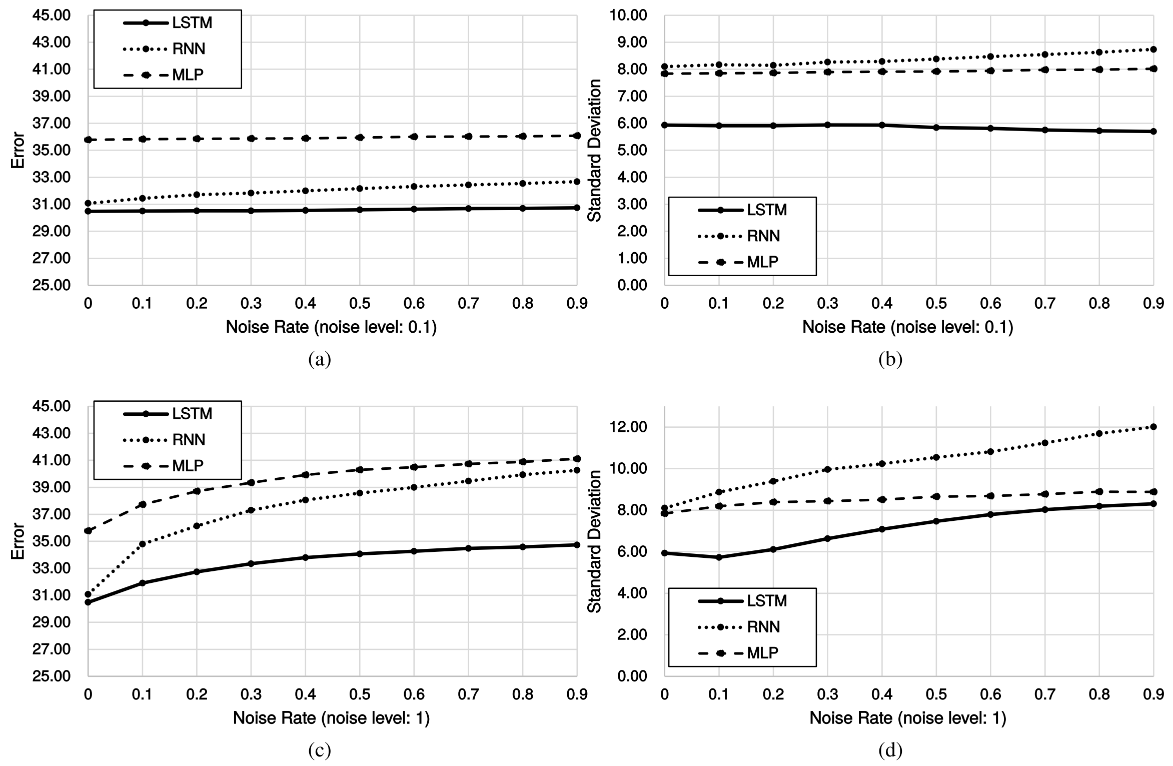


FIGURE 6. Test errors in various rates and strengths of generating noise for water quality forecasting. (a) Water quality mean. (b) Water quality SD. (c) Water quality mean. (d) Water quality SD.

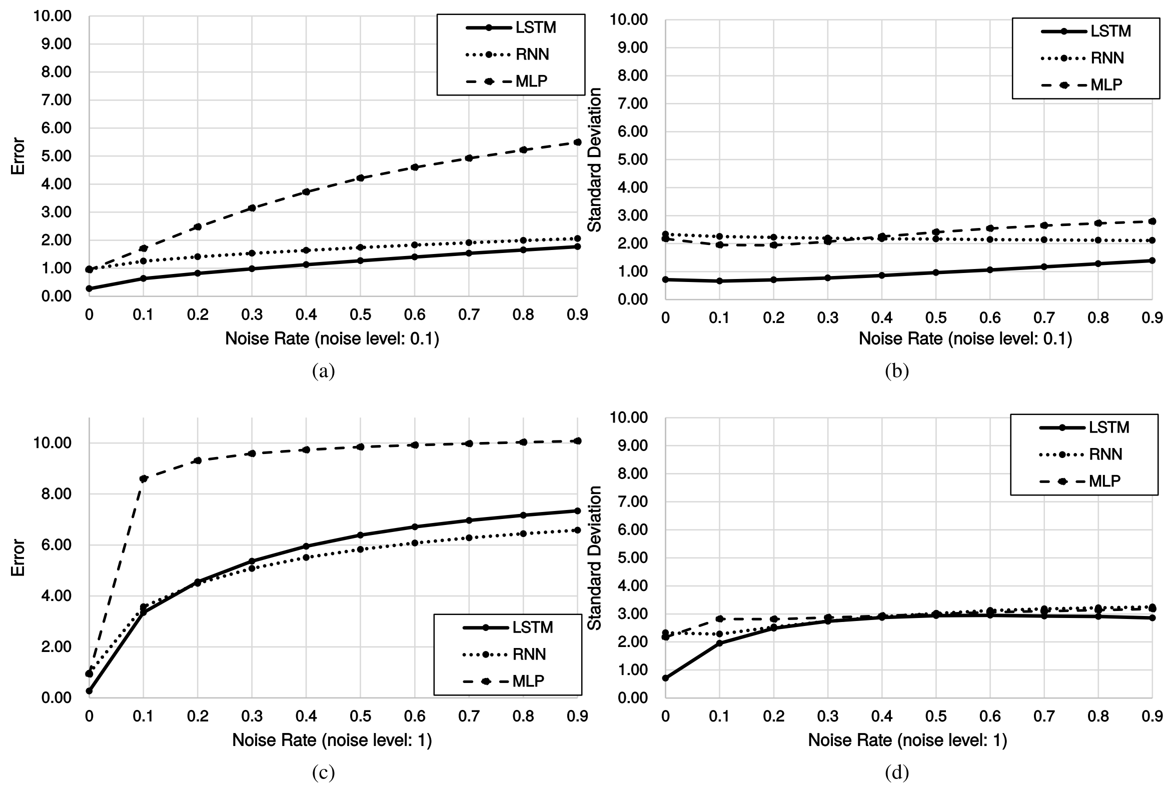


FIGURE 7. Test errors in various rates and strengths of generating noise for water quality forecasting. (a) Air pollution mean. (b) Air pollution SD. (c) Air pollution mean. (d) Air pollution SD.

generating the largest error when most observation is missing. LSTM and MLP shows similar level of errors, but base errors of LSTM is less than MLP by about 4.86 for overall time lags.

In air pollution forecasting, MLP largely increased errors by time lags compared to RNN and LSTM. In comparison of RNN and LSTM, errors were smaller in RNN than LSTM

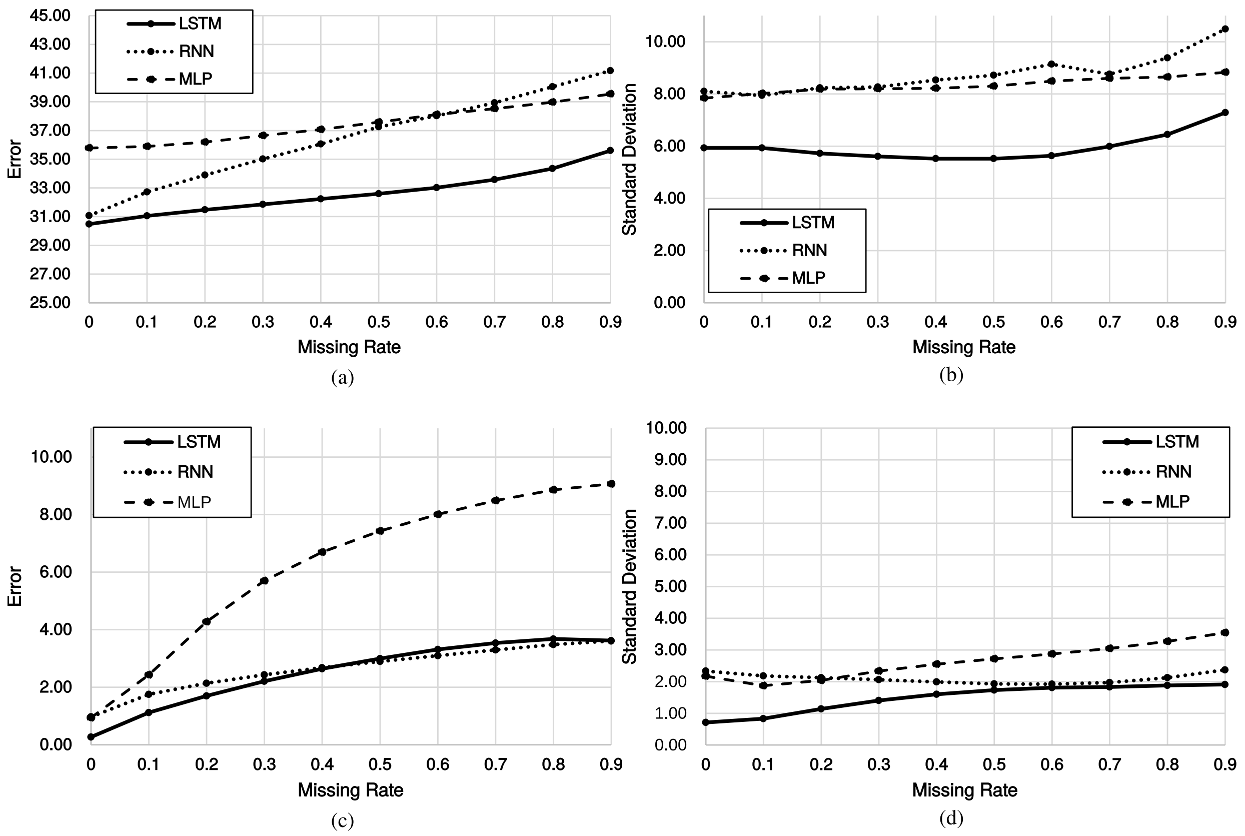


FIGURE 8. Test errors in various rates of missing input values for water quality forecasting. (a) Water quality mean. (b) Water quality SD. (c) AP mean. (d) Air pollution SD.

at the starting point, but LSTM increased with larger rate and then generates higher error than RNN at the longest lags. Standard deviation had similar pattern to water quality forecasting, but RNN decreased the standard deviation by time lags.

D. NOISE IMPUTATION

In noise imputation test of water quality shown in Fig. 6, observed results were similar to the missing value test. RNN had the highest increase rate and LSTM showed the lowest mean and standard deviation of errors. MLP showed significantly higher errors. Increasing noise level from 0.1 to 1.0, increase rate were amplified in all models. RNN is especially affected by the noise level, showing rapid increase of mean and standard deviation of errors by time.

In air pollution forecasting shown in Fig. 7, MLP shows the highest errors and standard deviation at overall noise rates. Compared to LSTM, RNN slowly increased errors by noise rates, but LSTM showed the lowest errors and variances. In the large noise-level setting, MLP still showed the highest errors. LSTM in this case showed a higher increase rate than RNN, but standard deviation was similar.

VII. DISCUSSION

In the long-term forecasting test, we could confirm that LSTM shows the most accurate results compared to MLP

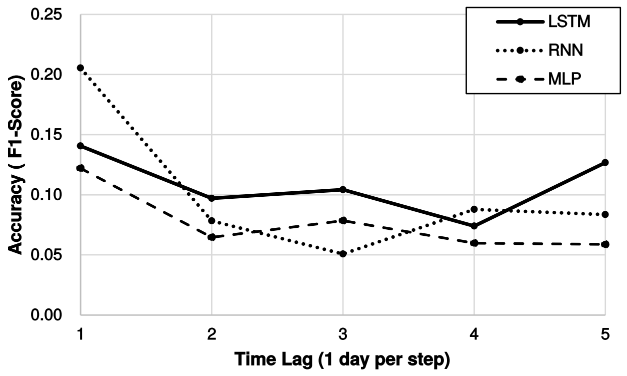
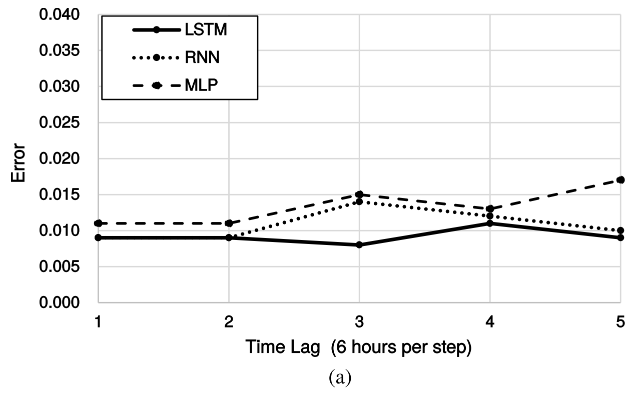


FIGURE 9. Test errors in various time lags between final input step and output step for ozone alarm forecasting.

and RNN. In the autoregressive modeling, this superiority is maintained. It is notable that the lowest standard deviation across different time lags in all long-term test results. This finding is strong empirical evidence to support our hypothesis that the impact of inputs is balanced over time steps in LSTM. We reason it that the low standard deviation is deemed as low probability to cause sudden drop of accuracy given the same level of information loss.

In the missing value and noise injection test, the lowest standard deviation of LSTM implies high stability of the performance under the unstable observation environments.



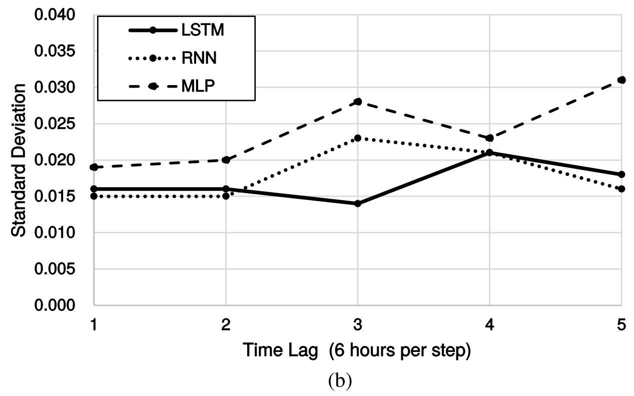


FIGURE 10. Test errors of autoregressive models in various time lags between final input step and output step for air pollution forecasting. (a) Air pollution mean. (b) Air pollution SD.

Mean errors of LSTM is the lowest in water quality problems, but RNN slightly exceeds the LSTM performance in air pollution. This lower error reversely implies that temporal dependency to represent in a model is so short that RNN can have high accuracy. If the dependency length is small, then, most noise values generated at the longer inputs are ignored and therefore errors by noise values can be less than LSTM.

In overall results, the LSTM shows the best accuracy in almost all experiments. This observation is not surprising, because the accuracy gain of using LSTM is well-known in many other forecasting problems from time series and sequential data. The key feature of the results is that LSTM consistently shows significantly low variances compared to the other types of neural networks. The low variance implies that errors at some input time steps do not cause sufficient change in the output. The cause of this small change is not prematured convergence of training, because LSTM shows the best accuracy in almost all experiments. Therefore, we can derive a conclusion from the implications that the remaining correct input data strongly maintain the correct output value even if some errors are introduced. Why can LSTM stick to the correct output values in unstable environments? The main reason is because all input time steps of LSTM sufficiently contribute to generate the output values not relying on a few dominating time steps.

VIII. CONCLUSION

In this paper, we evaluated stability of deep neural network in environmental time-series regression and classification problems. In many scientific applications, deep networks are difficult to improve performance of existing machine learning techniques, because of its high standard for the amount of training data. However, they may need to be reconsidered to use, because they are more robust to longterm forecasting, noisy, and missing value imputation by balanced distribution of impact to forecasting over times.

IX. IMPLEMENTATION ENVIRONMENT

This section describes the system and development environments for implementing the neural networks and setting the experiments.

- OS: Unbuntu 16.04
- virtual environment manager: Anaconda 2.0
- Python: 2.7
- Python GPU Library: Theano 0.9
- Cuda Library: cuda 8.0
- computing power: 4 cluster nodes
- cluster node: single Xeon quad-core CPU, 128GB RAM, two GTX1080Ti cards

REFERENCES

[1] K.-S. Jeong, G.-J. Joo, H.-W. Kim, K. Ha, and F. Recknagel, "Prediction and elucidation of phytoplankton dynamics in the Nakdong River (Korea) by means of a recurrent artificial neural network," *Ecol. Model.*, vol. 146, nos. 1–3, pp. 115–129, 2001.

[2] K.-S. Jeong, D.-K. Kim, J.-M. Jung, M.-C. Kim, and G.-J. Joo, "Non-linear autoregressive modelling by temporal recurrent neural networks for the prediction of freshwater phytoplankton dynamics," *Ecol. Model.*, vol. 211, nos. 3–4, pp. 292–300, 2008.

[3] A. Dedecker, P. Goethals, W. Gabriels, and N. De Pauw, "Optimisation of Artificial Neural Network (ANN) model design for prediction of macroinvertebrate communities in the Zwalm river basin (Flanders, Belgium)," in *Proc. 1st Biennial Meeting Int. Environ. Modelling Softw. Soc. Integr. Assessment Decis. Support*, vol. 2, 2002, pp. 142–147.

[4] W. Huang and S. Foo, "Neural network modeling of salinity variation in Apalachicola River," *Water Res.*, vol. 36, no. 1, pp. 356–362, 2002.

[5] Q. Zhang and S. J. Stanley, "Forecasting raw-water quality parameters for the North Saskatchewan River by neural network modeling," *Water Res.*, vol. 31, no. 9, pp. 2340–2350, 1997.

[6] H. M. Nagy, K. Watanabe, and M. Hirano, "Prediction of sediment load concentration in rivers using artificial neural network model," *J. Hydraulic Eng.*, vol. 128, no. 6, pp. 588–595, 2002.

[7] D. K. Kim, K. S. Jeong, R. I. B. McKay, T. S. Chon, and G. J. Joo, "Machine learning for predictive management: Short and long term prediction of phytoplankton biomass using genetic algorithm based recurrent neural networks," *Int. J. Environ. Res.*, vol. 6, no. 1, pp. 95–108, 2012.

[8] W. G. Cobourn, L. Dolcine, M. French, and M. C. Hubbard, "A comparison of nonlinear regression and neural network models for ground-level ozone forecasting," *J. Air Waste Manage. Assoc.*, vol. 50, no. 11, pp. 1999–2009, 2000.

[9] A. Russo, P. G. Lind, F. Raischel, R. Trigo, and M. Mendes, "Neural network forecast of daily pollution concentration using optimal meteorological data at synoptic and local scales," *Atmos. Pollut. Res.*, vol. 6, no. 3, pp. 540–549, 2015.

[10] M. D. Adams and P. S. Kanaroglou, "Mapping real-time air pollution health risk for environmental management: Combining mobile and stationary air pollution monitoring with neural network models," *J. Environ. Manage.*, vol. 168, pp. 133–141, Mar. 2016.

[11] M. S. Baawain and A. S. Al-Serihi, "Systematic approach for the prediction of ground-level air pollution (around an industrial port) using an artificial neural network," *Aerosol Air Qual. Res.*, vol. 14, no. 1, pp. 124–134, 2014.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[14] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. (Nov. 2015). "Sequence level training with recurrent neural networks." [Online]. Available: https://arxiv.org/abs/1511.06732

[15] Y. Wu *et al.* (Oct. 2016). "Google's neural machine translation system: Bridging the gap between human and machine translation." [Online]. Available: https://arxiv.org/abs/1609.08144

[16] M.-T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba. (Oct. 2014). "Addressing the rare word problem in neural machine translation." [Online]. Available: https://arxiv.org/abs/1410.8206

[17] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[18] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[19] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. (Oct. 2014). "On the properties of neural machine translation: Encoder-decoder approaches." [Online]. Available: https://arxiv.org/abs/1409.1259

[20] T. Lei, Y. Zhang, S. I. Wang, H. Dai, and Y. Artzi, "Simple recurrent units for highly parallelizable recurrence," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4470–4481.

[21] R. G. Wetzel, *Limnology*. San Diego, CA, USA: Academic, 2001.

[22] B. R. Moss, *Ecology of Fresh Waters: Man and Medium, Past to Future*. Oxford, U.K.: Blackwell Science, 1998.

[23] A. R. G. Large and G. E. Petts, "Rehabilitation of river margins," in *The Rivers Handbook: Hydrological and Ecological Principles*, vol. 2. Oxford, U.K.: Blackwell Scientific, 1992.

[24] H. R. Maier, M. D. Burch, and M. Bormans, "Flow management strategies to control Blooms of the cyanobacterium, *Anabaena circinalis*, in the River Murray at Morgan, South Australia," *Regulated Rivers, Res. Manage.*, vol. 17, no. 6, pp. 637–650, 2001.

[25] G. J. Joo and K. S. Jeong, "Modelling community changes of cyanobacteria in a flow regulated river (the lower Nakdong River, S. Korea) by means of a self-organizing map (SOM)," in *Modelling Community Structure in Freshwater Ecosystems*. Berlin, Germany: Springer, 2005, pp. 273–287.

[26] S. M. Mitrovic, B. C. Chessman, L. C. Bowling, and R. H. Cooke, "Modelling suppression of cyanobacterial Blooms by flow management in a lowland river," *River Res. Appl.*, vol. 22, no. 1, pp. 109–114, 2006.

[27] D.-K. Kim, B. Mckay, H. Shin, Y.-G. Lee, and X. H. Nguyen, "Ecological application of evolutionary computation: Improving water quality forecasts for the Nakdong River, Korea," in *Proc. IEEE Cong. Evol. Comput.*, Jul. 2010, pp. 1–8.

[28] S. De Vito, E. Massera, M. Piga, L. Martinotto, and G. Di Francia, "On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario," *Sens. Actuators B, Chem.*, vol. 129, no. 2, pp. 750–757, 2008.

[29] S. De Vito, M. Piga, L. Martinotto, and G. Di Francia, "CO, NO_2 and NO_x urban pollution monitoring with on-field calibrated electronic nose by automatic Bayesian regularization," *Sens. Actuators B, Chem.*, vol. 143, no. 1, pp. 182–191, 2009.

[30] K. Zhang, W. Fan, X. Yuan, I. Davidson, and X. Li, "Forecasting skewed biased stochastic ozone days: Analyses and solutions," in *Proc. IEEE 6th Int. Conf. Data Mining (ICDM)*, Dec. 2006, pp. 753–764.

[31] K. Zhang and W. Fan, "Forecasting skewed biased stochastic ozone days: Analyses, solutions and beyond," *Knowl. Inf. Syst.*, vol. 14, no. 3, pp. 299–326, 2008.

[32] C. A. Pope, III, *et al.*, "Lung cancer, cardiopulmonary mortality, and long-term exposure to fine particulate air pollution," *J. Amer. Med. Assoc.*, vol. 287, no. 9, pp. 1132–1141, 2002.

[33] D. W. Dockery *et al.*, "An association between air pollution and mortality in six U.S. cities," *New England J. Med.*, vol. 329, no. 24, pp. 1753–1759, 1993.

[34] U. Brunelli, V. Piazza, L. Pignato, F. Sorbello, and S. Vitabile, "Two-days ahead prediction of daily maximum concentrations of SO_2, O_3, PM10, NO_2, CO in the urban area of Palermo, Italy," *Atmos. Environ.*, vol. 41, no. 14, pp. 2967–2995, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1352231006012696

[35] V. Athira, P. Geetha, R. Vinayakumar, and K. P. Soman, "DeepAirNet: Applying recurrent networks for air quality prediction," *Procedia Comput. Sci.*, vol. 132, pp. 1394–1403, Jun. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050918308007

[36] Z. Feng, J. Sun, W. Wan, E. Hu, and V. Calatayud, "Evidence of widespread ozone-induced visible injury on plants in Beijing, China," *Environ. Pollut.*, vol. 193, pp. 296–301, Oct. 2014.

[37] X. Wang *et al.*, "Effects of elevated O_3 concentration on winter wheat and rice yields in the Yangtze River Delta, China," *Environ. Pollut.*, vol. 171, pp. 118–125, Dec. 2012.

[38] M. Medina-Ramón, A. Zanobetti, and J. Schwartz, "The effect of ozone and PM10 on hospital admissions for pneumonia and chronic obstructive pulmonary disease: A national multicity study," *Amer. J. Epidemiol.*, vol. 163, no. 6, pp. 579–588, 2006.

[39] A. M. De Schryver, K. W. Brakkee, M. J. Goedkoop, and M. A. J. Huijbregts, "Characterization factors for global warming in life cycle assessment based on damages to humans and ecosystems," *Environ. Sci. Technol.*, vol. 43, no. 6, pp. 1689–1695, 2009.

[40] T. A. Solaiman, P. Coulibaly, and P. Kanaroglou, "Ground-level ozone forecasting using data-driven methods," *Air Qual., Atmos. Health*, vol. 1, no. 4, pp. 179–193, 2008.

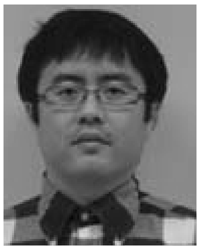
[41] J. Yi and V. R. Prybutok, "A neural network model forecasting for prediction of daily maximum ozone concentration in an industrialized urban area," *Environ. Pollut.*, vol. 92, no. 3, pp. 349–357, 1996.

[42] M. D. Zeiler. (Dec. 2012). "ADADELTA: An adaptive learning rate method." [Online]. Available: https://arxiv.org/abs/1212.5701

[43] S. Ioffe and C. Szegedy. (Mar. 2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift." [Online]. Available: https://arxiv.org/abs/1502.03167

[44] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

KANGIL KIM received the B.Sc. degree in computer science from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2006, and the Ph.D. degree from Seoul National University, Seoul, South Korea, in 2012. He has been a Senior Researcher with the Natural Language Processing Group, Electronics and Telecommunications Research Institute, Seoul, since 2013. He is currently an Assistant Professor with the Computer Science and Engineering Department, Konkuk University. His research interests include artificial intelligence, evolutionary computation, machine learning, and natural language processing.



DONG-KYUN KIM received the Ph.D. degree in ecological informatics/modeling from Pusan National University, Busan, South Korea, in 2008. His area of specialization involves both process-based (mechanistic) and statistical (empirical) modeling approaches. He also expands his research to application of innovative computational techniques to elucidate the functioning of ecosystems associated with climate change. His recent research promotes a critical approach to environmental modeling that focuses on the rigorous assessment of ecosystem responses while explicitly quantifying the underlying uncertainties.

JUNHYUG NOH received the B.S. degree in computer science and engineering & statistics and the M.S. degree in computer science and engineering from Seoul National University, Seoul, South Korea, in 2013 and 2015, respectively, where he is currently pursuing the Ph.D. degree in computer science and engineering. His research interests include artificial intelligence, machine learning, and computer vision.

MINHYEOK KIM received the B.Sc. degree in computer science and the Ph.D. degree from Seoul National University, Seoul, South Korea, in 2005 and 2013, respectively. Since 2015, he has been a Senior Researcher with LG electronics. His research interests include machine learning and its applications.