Communications for Statistical Applications and Methods
2021, Vol. 28, No. 1, 39–57

https://doi.org/10.29220/CSAM.2021.28.1.039
Print ISSN 2287-7843 / Online ISSN 2383-4757

Two-dimensional attention-based multi-input LSTM for time series prediction

Eun Been Kim[a], Jung Hoon Park[a], Yung-Seop Lee[b], Changwon Lim[1,a]

[a]Department of Applied Statistics, Chung-Ang University, Korea;
[b]Department of Statistics, Dongguk University, Korea;

Abstract

Time series prediction is an area of great interest to many people. Algorithms for time series prediction are widely used in many fields such as stock price, temperature, energy and weather forecast; in addtion, classical models as well as recurrent neural networks (RNNs) have been actively developed. After introducing the attention mechanism to neural network models, many new models with improved performance have been developed; in addition, models using attention twice have also recently been proposed, resulting in further performance improvements. In this paper, we consider time series prediction by introducing attention twice to an RNN model. The proposed model is a method that introduces H-attention and T-attention for output value and time step information to select useful information. We conduct experiments on stock price, temperature and energy data and confirm that the proposed model outperforms existing models.

Keywords: recurrent neural network, correlation, attention, time series

1. Introduction

Time series prediction is a field that has attracted significant attention. Algorithms for time series prediction are widely used in many fields such as weather, electricity, and financial markets. They have been developed by using well-known classic models such as the autoregressive moving average (McLeod and Li, 1983) as well as recurrent neural network (RNN) (Rumelhart *et al.*, 1986; Werbos, 1990; Elman, 1991) and long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) to improve their performance. RNN is a sequential model widely used for time series data. Itx takes information from prior inputs to influence current input and output using hidden state recurrently. LSTM is a special structure of RNN. Unlike RNN, LSTM adds memory cells to calculate if the information is important or not. Three gates, input gate, forget gate and output gate, judge the input and decide to retain or forget by some special rules to capture long term dependency of time series data. In addition, as a nonlinear autoregressive exogenous (NARX) model (Chen *et al.*, 2008) using other exogenous variables that can affect the target variable, not just the past value of the target variable, was proposed, the performance of time series prediction has been improved.

The NARX model has been continuously studied, such as an improved hybrid prediction model (Pham *et al.*, 2010) and a new approach to identifying a new class of NARX models (Li *et al.*, 2011). In particular, in the case of RNN, we succeeded in increasing the prediction rate by applying it to the NARX model. However, it was difficult to capture long-term dependency. To overcome this, methods

[1] Corresponding author: Department of Applied Statistics, Chung-Ang University, 47, Heukseok-ro, Dongjak-Gu, Seoul 06974, Korea. E-mail: clim@cau.ac.kr

Published 31 January 2021 / journal homepage: http://csam.or.kr
© 2021 The Korean Statistical Society, and Korean International Statistical Society. All rights reserved.

using LSTM, gated recurrent unit algorithms (Cho *et al.*, 2014a) and encoder decoder networks (Cho *et al.*, 2014b; Sutskever *et al.*, 2014) were developed.

Research on introducing an attention mechanism has also been actively conducted, and studies using a new RNN method such as an attention-based encoder decoder network (Liu and Lane, 2016) have also been conducted. Research has also been conducted to introduce attention to the NARX model. A hierarchical attention network that selects the hidden state of the related encoder using two attention mechanisms (Yang *et al.*, 2016), a model that unifies the spatiotemporal feature extraction of exogenous variables and the temporal dynamics modeling of the target variable (Tao *et al.*, 2018), two-stage two-phase RNN (Liu *et al.*, 2019), and various algorithms have been proposed. All these algorithms show improved performance. A dual-stage attention-based RNN (DA-RNN) model (Qin *et al.*, 2017) has also been proposed, which has the advantage of selecting exogenous variables with a great influence on the prediction of target variable and being able to properly capture the long-term dependency of time series. In addition, a model using a new attention-based multi-input LSTM (MI-LSTM) (Li *et al.*, 2018) using factors with low correlation effectively, followed by a two-dimensional attention-based LSTM (2D-LSTM) (Yu and Kim, 2019) model was also proposed. 2D-LSTM is a model in which the importance of exogenous variables and the importance of time are separately calculated and combined.

In this paper, we propose a two-dimensional attention-based multi-input LSTM (2DA-MILSTM) model that includes the advantages of DA-RNN, 2D-LSTM and MI-LSTM. This model calculates the correlation between the target variable and the exogenous variables, divides the exogenous variables accordingly and then inputs them to MI-LSTM. It also applies two separate attention layers simultaneously using the output value of MI-LSTM and the hidden state of the previous layer. Two kinds of weights are created by applying the attentions to each hidden state and time step. By integrating these two weights, the predicted value can be calculated, and the importance of the hidden state and the time step can be considered to improve the long-term dependency problem and prediction performance. We compare the performance of the proposed model with existing models through prediction experiments on stock price, room temperature and energy data.

The rest of this paper is as follows. In Section 2, we review the existing models, and in Section 3 we explain the detailed parts and configurations of the proposed model in detail. We describe datasets used in the experiment and present results of the experiment in Section 4. Lastly, Section 5 draws the conclusion of this paper.

2. Existing methods

2.1. Recurrent neural network and long short-term memory

In neural network, RNNs and LSTM are the most popular model for time series including language and speech data (Huang *et al.*, 2015). Figure 1 shows that the RNN model maintains consists of input layer x, hidden layer h, and output layer y. An input layer takes the features at time t, and output layer represents a probability distribution for labels at tie t. As the recurrent layer sores history informations, RNN connects between the previous hidden state and the current hidden state.

Then the values in the hidden and output layers are computed by $h(t) = f(Ux(t) + Wh(t-1)))$, $y(t) = g(Vh(t))$, where U, W, and V are the connection weights computed in training, and $f(z)$ and $g(z)$ are sigmoid and softmax activation functions. Softmax (Jang *et al.*, 2016) is used as an activation function for the classification task. It calculates normalized output of probability distribution in the neural network. Defined as $\sigma(z_i) = \exp(z_i)/\sum_{j=1}^{K} \exp(z_j)$, which takes as input vector z of K classes, normalize output vectors dividing by the sum of all components in exponential scale.

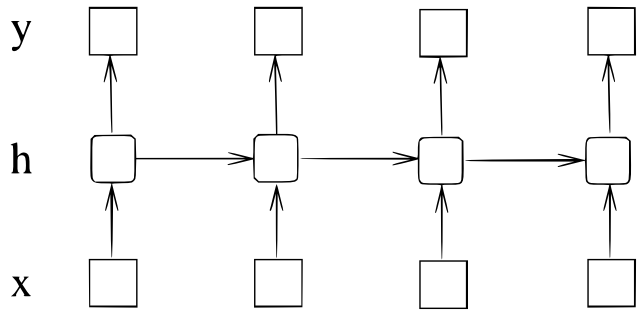

Figure 1: *Basic structure of RNN (Huang et al., 2015).*

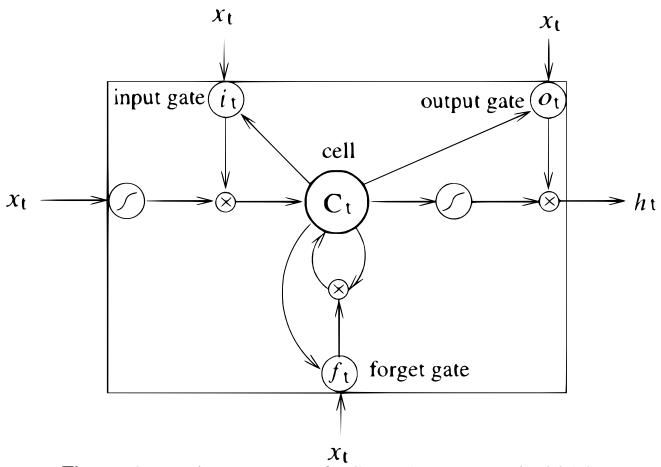

Figure 2: *Basic structure of LSTM (Huang et al., 2015).*

LSTM is also similar to RNN but LSTM captures long term dependencies of times series data better than RNN using three gates, input, forget, and output gate in the LSTM cell. In Figure 2, σ represents logistic sigmoid function, and i, f, o, and c represent the input gate, forget gate, output gate and cell vectors with same size with the hidden vector h.

Then the procedure of LSTM cell using matrix notation is:

$$
\begin{aligned}
i_t &= \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right), \\
f_t &= \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right), \\
c_t &= f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right), \\
o_t &= \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right), \\
h_t &= o_t \tanh(c_t).
\end{aligned}
$$

For explanation, W_{xo} represents the input-output gate matrix, W_{hi} represents the hidden-input gate matrix. These calculations concentrate on which information should be focused to capture short and long term dependencies in time series data as well as determine how much information should be forgotten using the forget gate.

2.2. Multi-input long short-term memory model

The target variable is defined as $\mathbf{Y} = (y_1, y_2, \ldots, y_T)^\top \in \mathbb{R}^T$, and the exogenous variables related to the prediction of the target variable are defined as $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T)^\top \in \mathbb{R}^{T\times D}$ where T is the time window size and D is the number of exogenous variables. To express the exogenous variables in detail, two notations are used. First, $\mathbf{X}_t = (x_t^1, x_t^2, \ldots, x_t^D)^\top \in \mathbb{R}^D$ denotes the values of all exogenous variables at time t, and $\mathbf{X}^k = (x_1^k, x_2^k, \ldots, x_T^k)^\top \in \mathbb{R}^T$ denotes the values of the kth exogenous variable. The matrix $\mathbf{X}$ contains three kinds of exogenous variables: $\mathbf{X_p} \in \mathbb{R}^{T\times P}$ having a positive correlation with the target variable, $\mathbf{X_n} \in \mathbb{R}^{T\times N}$ having a negative correlation with the target variable, index variables $\mathbf{X_i} \in \mathbb{R}^T$ for stock price data if available, where $D = P + N + 1$.

The Pearson correlation coefficient is used to measure the correlation between the exogenous variables and the target variable, and $\mathbf{X_p}$ and $\mathbf{X_n}$ are generated using P-largest and N-smallest variables with positive and negative correlation coefficients, respectively. In our experiment, $\{4, 6, 10, 15\}$ is a set of possible values for P and N, and 15, the largest value, is finally selected.

We define "Self", "Index", "Positive", and "Negative" based on $\mathbf{Y}$, $\mathbf{X_i}$, $\mathbf{X_p}$, and $\mathbf{X_n}$ values, respectively, by using LSTM with the hidden state size of r. The definitions are:

- Self:

$$\tilde{\mathbf{Y}} = \text{LSTM}(\mathbf{Y}).$$

- Index:

$$\tilde{\mathbf{I}} = \text{LSTM}(\mathbf{X_i}),$$

  where $\tilde{\mathbf{Y}}, \tilde{\mathbf{I}} \in \mathbb{R}^{T\times r}$.

- Positive:

$$\tilde{\mathbf{P}} = \frac{1}{P}\sum_{p=1}^{P} \text{LSTM}\left(\mathbf{X_p^p}\right).$$

- Negative:

$$\tilde{\mathbf{N}} = \frac{1}{N}\sum_{n=1}^{N} \text{LSTM}\left(\mathbf{X_n^n}\right),$$

  where $\tilde{\mathbf{P}}, \tilde{\mathbf{N}} \in \mathbb{R}^{T\times r}$.

The MI-LSTM is a model that can extract valuable information from small correlation factors and discard negative noises using extra input gates controlled by convincing factors. This model deviates from the existing LSTM methods and shows that a model having multi-inputs to the LSTM can improve prediction performance. As defined above, MI-LSTM uses four input values: the past value of the target variable, exogenous variables with positive and negative correlation, and an index exogenous variable. The input values are expressed as $\tilde{\mathbf{Y}}$, $\tilde{\mathbf{P}}$, $\tilde{\mathbf{N}}$, $\tilde{\mathbf{I}}$, and the values after passing through MI-LSTM can be expressed as

$$\tilde{\mathbf{Y}}' = \text{MILSTM}\left(\tilde{\mathbf{Y}}, \tilde{\mathbf{P}}, \tilde{\mathbf{N}}, \tilde{\mathbf{I}}\right)$$

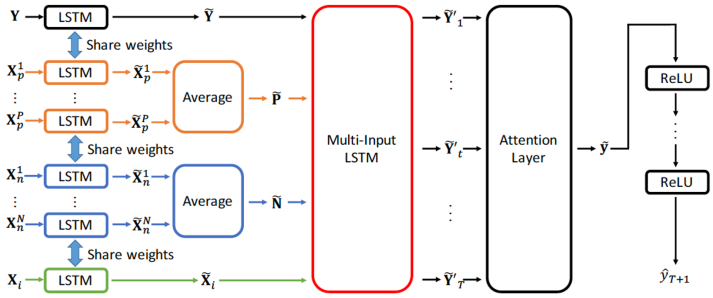

Figure 3: *Structure of MI-LSTM (Li et al., 2018).*

and an output value of $\tilde{\mathbf{Y}}' = (\tilde{\mathbf{Y}}'_1, \tilde{\mathbf{Y}}'_2, \ldots, \tilde{\mathbf{Y}}'_T)^\top \in \mathbb{R}^{T\times p}$ is obtained.

Figure 3 is the structure of MI-LSTM. The following attention is applied to $\tilde{\mathbf{Y}}'$ to give weights:

$$
\begin{aligned}
j_t &= \mathbf{v}_b^\top \tanh\left(\mathbf{W}_b \left(\tilde{\mathbf{Y}}'_t\right)^\top + \mathbf{b}_b\right), \\
\boldsymbol{\beta} &= \mathrm{Softmax}\left(\left[j_1, j_2, \ldots, j_T\right]^\top\right), \\
\tilde{\mathbf{y}} &= \boldsymbol{\beta}^\top \tilde{\mathbf{Y}}', \\
\hat{y}_{T+1} &= \mathrm{ReLU}\left(\mathbf{W}\tilde{\mathbf{y}} + b\right),
\end{aligned}
$$

where the matrix $\mathbf{W}_b \in \mathbb{R}^{p\times p}$, bias $\mathbf{b}_b \in \mathbb{R}^p$, and vector $\mathbf{v}_b \in \mathbb{R}^p$ are the parameters to be learned, $\tilde{\mathbf{y}} \in \mathbb{R}^p$ is the attention output, $\mathbf{W} \in \mathbb{R}^{1\times p}$ and $b \in \mathbb{R}$ are also parameters to be learned, and ReLU (rectified linear unit) is an activation function. ReLU was introduced by Nair and Hinton (2010) to make the deep learning model train better. Marked as $\mathrm{ReLU}(x) = \max(0, x)$, unlike the activation functions before such as sigmoidal activation, it gets fewer vanishing gradient problems which prohibit neurons to learn weights in the propagation steps.

2.3. Dual-stage attention-based recurrent neural network

DA-RNN consists of two steps. In the first step it extracts the hidden state of the previous encoder and the relevant exogenous variables in each step by using the input attention. Then, in the second step the model selects the output value of the related decoder using the temporal attention mechanism. This model can select the most relevant exogenous variables as well as adequately capture the long-term dependency of the time series. Figure 4 and Figure 5 show the structure of DA-RNN.

The n exogenous variables are represented by matrix $\mathbf{X} = (\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n)^\top = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T) \in \mathbb{R}^{n\times T}$, where T is the time window size, $\mathbf{x}^k = (x_1^k, x_2^k, \ldots, x_T^k)^\top \in \mathbb{R}^T$ represents the kth exogenous variable, and $\mathbf{x}_t = (x_t^1, x_t^2, \ldots, x_t^n)^\top \in \mathbb{R}^n$ represents the vectors of n exogenous variables at time t. DA-RNN creates a new input vector by multiplying this by the weight obtained through the following input attention mechanism. It inputs the new input vectors back to the LSTM and makes predictions by obtaining weights through a temporal attention mechanism.

$$
\begin{aligned}
e_t^k &= \mathbf{v}_e^\top \tanh\left(\mathbf{W}_e\left[\mathbf{h}_{t-1}; \mathbf{s}_{t-1}\right] + \mathbf{U}_e \mathbf{x}^k\right), \\
\alpha_t^k &= \frac{\exp\left(e_t^k\right)}{\sum_{i=1}^n \exp\left(e_t^i\right)}, \\
\tilde{\mathbf{x}}_t &= \left(\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, \ldots, \alpha_t^n x_t^n\right)^\top,
\end{aligned}
$$

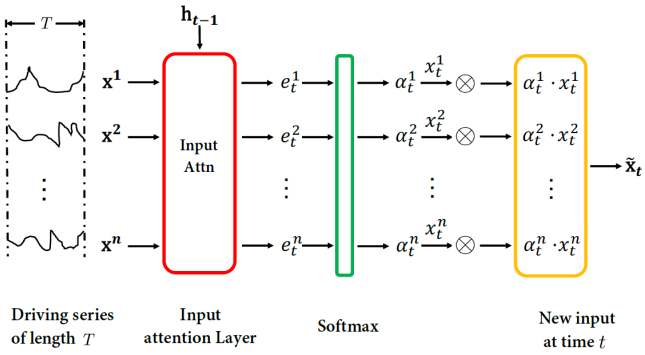

Figure 4: *Input attention mechanism of DA-RNN (Qin et al., 2017).*

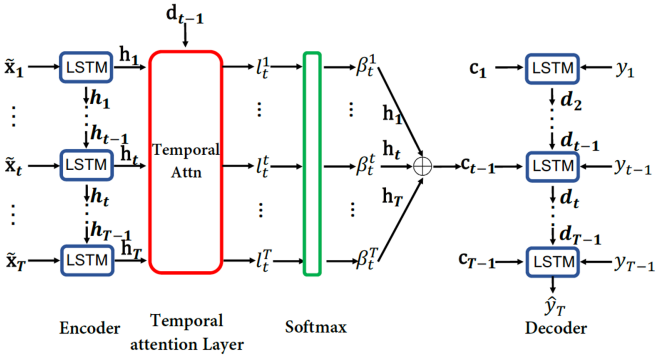

Figure 5: *Temporal attention mechanism of DA-RNN (Qin et al., 2017).*

where $\mathbf{h}_{t-1} \in \mathbb{R}^m$ and $\mathbf{s}_{t-1} \in \mathbb{R}^m$ are the previous hidden state and the cell state in the encoder LSTM unit, respectively, and $\mathbf{v}_e \in \mathbb{R}^T$, $\mathbf{W}_e \in \mathbb{R}^{T\times 2m}$, and $\mathbf{U}_e \in \mathbb{R}^{T\times T}$ are parameters to be learned, and m is the size of the hidden state of the encoder. As in the above equations, a new input value is created by multiplying the weight, and after inputting it into LSTM as shown in the following equations, it is finally input into the temporal attention mechanism of the decoder.

$$
\begin{aligned}
\mathbf{h}_t &= f_1(\mathbf{h}_{t-1}, \tilde{\mathbf{x}}_t), \\
l_t^i &= \mathbf{v}_d^\top \tanh\left(\mathbf{W}_d \left[\mathbf{d}_{t-1}; \mathbf{s}'_{t-1}\right] + \mathbf{U}_d \mathbf{h}_i\right), \quad 1 \le i \le T, \\
\beta_t^i &= \frac{\exp\left(l_t^i\right)}{\sum_{j=1}^{T} \exp\left(l_t^j\right)}, \\
\mathbf{c}_t &= \sum_{i=1}^{T} \beta_t^i \mathbf{h}_i, \\
\tilde{y}_{t-1} &= \tilde{\mathbf{w}}^\top \left[y_{t-1}; \mathbf{c}_{t-1}\right] + \tilde{b}, \\
\mathbf{d}_t &= f_2(\mathbf{d}_{t-1}, \tilde{y}_{t-1}), \\
\hat{y}_T &= F(y_1, \ldots, y_{T-1}, \mathbf{x}_1, \ldots, \mathbf{x}_T) = \mathbf{v}_y^\top \left(\mathbf{W}_y [\mathbf{d}_T; \mathbf{c}_T] + \mathbf{b}_w\right) + b_v,
\end{aligned}
$$

where $\mathbf{v}_d \in \mathbb{R}^m$, $\mathbf{W}_d \in \mathbb{R}^{m\times 2p}$, and $\mathbf{U}_d \in \mathbb{R}^{m\times m}$ are parameters to be learned, p is the size of the

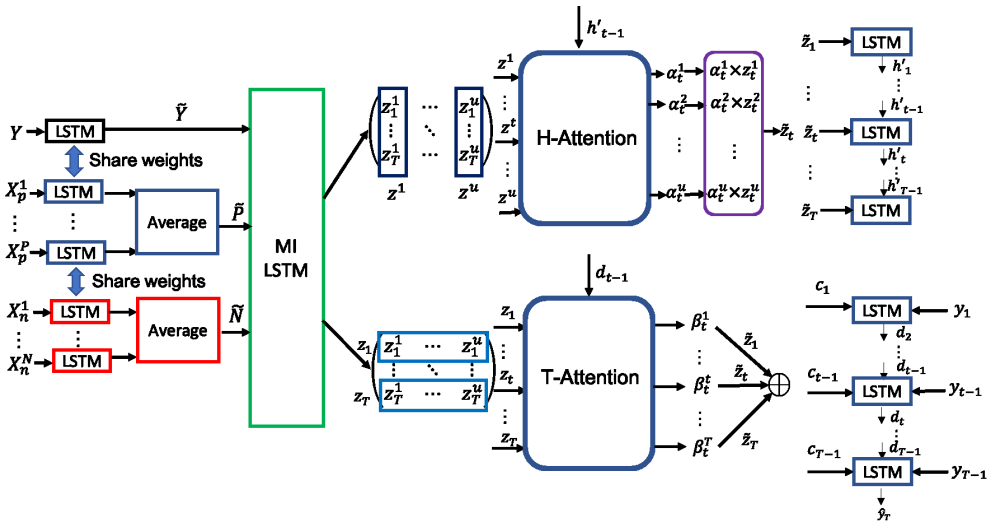

Figure 6: *Structure of 2DA-MILSTM.*

decoder's hidden state, f_1 and f_2 are LSTMs, and $\mathbf{v}_y \in \mathbb{R}^p$, $\mathbf{W}_y \in \mathbb{R}^{p\times(p+m)}$, $\mathbf{b}_w \in \mathbb{R}^p$ and $b_v \in \mathbb{R}$ are also parameters to be learned.

3. Proposed method

3.1. Two-dimensional attention-based multi-input LSTM

In this section we present the details of the two-dimensional attention-based multi-input LSTM (2DA-MILSTM) model, a new model that includes the advantages of the MI-LSTM and DA-RNN models described in Section 2 and 2D-LSTM. Figure 6 is the structure of the 2DA-MILSTM model.

The exogenous variables are divided using the value of the correlation coefficient, and the output value after inputting it to MI-LSTM is expressed as $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_T)^\top \in \mathbb{R}^{T\times u}$, where u is the number of MI-LSTM hidden units, and $\mathbf{z}_t = (z_t^1, z_t^2, \ldots, z_t^u)^\top \in \mathbb{R}^u$ represents the u hidden unit input vector at time t. The proposed model uses H-attention and T-attention for $\mathbf{Z}$, which can be calculated as:

$$\hat{y}_{T+1} = F(\mathbf{Y}, \mathbf{X}, \mathbf{Z}),$$

where F is the function we want to learn.

In this model, as in the existing MI-LSTM model, "Self", "Positive", and "Negative" are used as input values of MI-LSTM. However, unlike in the existing model, three factors excluding index variable are input. This is because there is no index exogenous variable in the data we consider in this paper. Therefore, we use the $\tilde{\mathbf{Y}}, \tilde{\mathbf{P}}, \tilde{\mathbf{N}} \in \mathbb{R}^{T\times r}$ as the input of the three factors, and $\tilde{\mathbf{Y}}_t, \tilde{\mathbf{P}}_t, \tilde{\mathbf{N}}_t \in \mathbb{R}^r$ as the corresponding input vector for $t = 1, 2, \ldots, T$. Figure 7 shows the structure of the modified MI-LSTM.

The forget gate and the output gate of the MI-LSTM remain the same when compared to the

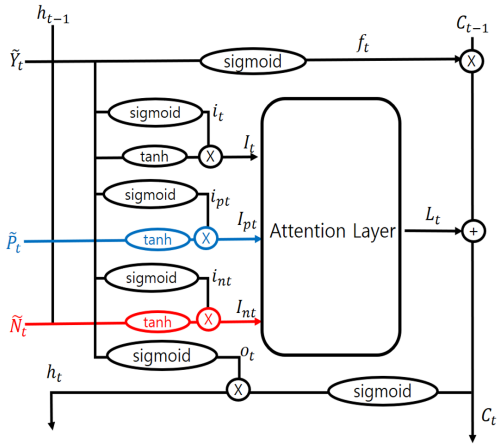

Figure 7: *Structure of a modified MI-LSTM.*

original LSTM as shown in the following equations:

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_f\left[\mathbf{h}_{t-1}; \tilde{\mathbf{Y}}_t\right] + \mathbf{b}_f\right), \tag{3.1}$$

$$\mathbf{o}_t = \sigma\left(\mathbf{W}_o\left[\mathbf{h}_{t-1}; \tilde{\mathbf{Y}}_t\right] + \mathbf{b}_o\right), \tag{3.2}$$

where $\mathbf{f}_t, \mathbf{o}_t \in \mathbb{R}^u$ are the forget gate and output gate, respectively, $\mathbf{h}_{t-1} \in \mathbb{R}^u$ is the hidden state of the previous time step and u is the number of the MI-LSTM hidden units. $\mathbf{W}_f, \mathbf{W}_o \in \mathbb{R}^{u\times(u+r)}$ are the weights of the forget gate and the output gate, respectively, and $\mathbf{b}_f, \mathbf{b}_o \in \mathbb{R}^u$ are the biases.

All input gates are determined by the "Self" variable and the previous hidden state to control the auxiliary factors as in the following equations:

$$\mathbf{i}_t = \sigma\left(\mathbf{W}_i\left[\mathbf{h}_{t-1}; \tilde{\mathbf{Y}}_t\right] + \mathbf{b}_i\right), \tag{3.3}$$

$$\mathbf{i}_{pt} = \sigma\left(\mathbf{W}_{ip}\left[\mathbf{h}_{t-1}; \tilde{\mathbf{Y}}_t\right] + \mathbf{b}_{ip}\right), \tag{3.4}$$

$$\mathbf{i}_{nt} = \sigma\left(\mathbf{W}_{in}\left[\mathbf{h}_{t-1}; \tilde{\mathbf{Y}}_t\right] + \mathbf{b}_{in}\right), \tag{3.5}$$

where $\mathbf{i}_t, \mathbf{i}_{pt}, \mathbf{i}_{nt} \in \mathbb{R}^u$ are the input gates of three factors, $\mathbf{W}_i, \mathbf{W}_{ip}, \mathbf{W}_{in} \in \mathbb{R}^{u\times(u+r)}$ are the weights, and $\mathbf{b}_i, \mathbf{b}_{ip}, \mathbf{b}_{in} \in \mathbb{R}^u$ are the biases. The cell states of three factors $\tilde{\mathbf{C}}_t, \tilde{\mathbf{C}}_{pt}, \tilde{\mathbf{C}}_{nt} \in \mathbb{R}^u$ are obtained as:

$$\tilde{\mathbf{C}}_t = \tanh\left(\mathbf{W}_c\left[\mathbf{h}_{t-1}; \tilde{\mathbf{Y}}_t\right] + \mathbf{b}_c\right), \tag{3.6}$$

$$\tilde{\mathbf{C}}_{pt} = \tanh\left(\mathbf{W}_{cp}\left[\mathbf{h}_{t-1}; \tilde{\mathbf{P}}_t\right] + \mathbf{b}_{cp}\right), \tag{3.7}$$

$$\tilde{\mathbf{C}}_{nt} = \tanh\left(\mathbf{W}_{cn}\left[\mathbf{h}_{t-1}; \tilde{\mathbf{N}}_t\right] + \mathbf{b}_{cn}\right), \tag{3.8}$$

where $\mathbf{W}_c, \mathbf{W}_{cp}, \mathbf{W}_{cn} \in \mathbb{R}^{u\times(u+r)}$ are the weights, and $\mathbf{b}_c, \mathbf{b}_{cp}, \mathbf{b}_{cn} \in \mathbb{R}^u$ are the biases.

In LSTM, the element-wise multiplication of $\tilde{\mathbf{C}}_t$ and $\mathbf{i}_t$ is the final cell state input at time t:

$$\mathbf{l}_t = \tilde{\mathbf{C}}_t \odot \mathbf{i}_t, \tag{3.9}$$

$$\mathbf{l}_{pt} = \tilde{\mathbf{C}}_{pt} \odot \mathbf{i}_{pt}, \tag{3.10}$$

$$\mathbf{l}_{nt} = \tilde{\mathbf{C}}_{nt} \odot \mathbf{i}_{nt}, \tag{3.11}$$

where $\mathbf{l}_t, \mathbf{l}_{pt}, \mathbf{l}_{nt} \in \mathbb{R}^u$ are the cell inputs of the factors, and $\odot$ is the element-wise multiplication. Then, the sum of the weights is performed for the cell input as:

$$\mathbf{L}_t = \alpha_t \mathbf{l}_t + \alpha_{pt} \mathbf{l}_{pt} + \alpha_{nt} \mathbf{l}_{nt}, \tag{3.12}$$

where $\alpha_t, \alpha_{pt}, \alpha_{nt} \in \mathbb{R}$ are the attention weights, and $\mathbf{L}_t$ is the input of the final cell state at time t. The attention weights are determined by the cell state input itself and the cell state in the previous step as shown in the following equations:

$$u_t = \tanh\left(\mathbf{l}_t^\top \mathbf{W}_a \mathbf{C}_{t-1} + b_a\right), \tag{3.13}$$

$$u_{pt} = \tanh\left(\mathbf{l}_{pt}^\top \mathbf{W}_a \mathbf{C}_{t-1} + b_{ap}\right), \tag{3.14}$$

$$u_{nt} = \tanh\left(\mathbf{l}_{nt}^\top \mathbf{W}_a \mathbf{C}_{t-1} + b_{an}\right), \tag{3.15}$$

$$\left[\alpha_t, \alpha_{pt}, \alpha_{nt}\right]^\top = \mathrm{Softmax}\left(\left[u_t, u_{pt}, u_{nt}\right]^\top\right), \tag{3.16}$$

where $u_t, u_{pt}, u_{nt} \in \mathbb{R}$ are values that appear in the middle of the calculation of the attention weights, and $\mathbf{W}_a \in \mathbb{R}^{u \times u}$ and $b_a, b_{ap}, b_{an} \in \mathbb{R}$ are parameters to be learned. Also, $\mathbf{C}_{t-1} \in \mathbb{R}^u$ is the state of the cell at time $t-1$. $[u_t, u_{pt}, u_{nt}]^\top$ is the attention score and is input to the softmax layer. When the update vector of the cell state $\mathbf{L}_t$ is stabilized, the remaining MI-LSTM units proceed in the same manner as the original LSTM.

$$\mathbf{C}_t = \mathbf{C}_{t-1} \odot \mathbf{f}_t + \mathbf{L}_t, \tag{3.17}$$

$$\mathbf{h}_t = \tanh\left(\mathbf{C}_t\right) \odot \mathbf{o}_t, \tag{3.18}$$

where $\mathbf{C}_t, \mathbf{h}_t \in \mathbb{R}^u$ are the cell state and hidden state at time t, respectively.

The structure of MI-LSTM is the same as the above equations. In this paper, the equations (3.1)-(3.18) are expressed as F_2 to summarize as:

$$\mathbf{h}_t = F_2\left(\mathbf{h}_{t-1}, \tilde{\mathbf{Y}}_t, \tilde{\mathbf{P}}_t, \tilde{\mathbf{N}}_t\right).$$

It is also intended to be expressed simply as:

$$\mathbf{Z} = \mathrm{MILSTM}\left(\tilde{\mathbf{Y}}, \tilde{\mathbf{P}}, \tilde{\mathbf{N}}\right),$$

and we get an output value $\mathbf{Z} = (\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_T) \in \mathbb{R}^{T \times u}$ for each time step.

3.2. H-attention

Considering the kth output value $\mathbf{z}^k = (z_1^k, z_2^k, \ldots, z_T^k) \in \mathbb{R}^T$, we can construct the attention model by referring to the previous hidden state $\mathbf{h}'_{t-1}$ and the cell state $\mathbf{C}'_{t-1}$ in the LSTM unit. By constructing an attention model, an attention weight α_t^k can be obtained. We focus on the output value in the MI-LSTM structure and consider whether the output value contains accurate information and sufficiently explains the information by looking at the importance at time step when using it. Therefore, we introduce an H-attention mechanism that can select which value is of high importance among the output values from MI-LSTM as:

$$u_t^k = \mathbf{v}_e^\top \tanh\left(\mathbf{W}_e\left[\mathbf{h}'_{t-1}; \mathbf{C}'_{t-1}\right] + \mathbf{U}_e \mathbf{z}^k\right)$$

$$\alpha_t^k = \frac{\exp\left(u_t^k\right)}{\sum_{i=1}^u \exp\left(u_t^i\right)}$$

where $\mathbf{v}_e \in \mathbb{R}^T$, $\mathbf{W}_e \in \mathbb{R}^{T\times 2m}$, and $\mathbf{U}_e \in \mathbb{R}^{T\times T}$ are parameters to be learned, and α_t^k is the attention weight that measures the importance of the kth hidden unit at time t. In addition, the softmax function is applied to u_t^k, so that the weight values sum to 1.

A new input vector can be extracted by multiplying the weight derived through the H-attention mechanism by the input value.

$$\tilde{\mathbf{z}}_t = \left(\alpha_t^1 z_t^1, \alpha_t^2 z_t^2, \ldots, \alpha_t^u z_t^u\right)^\top$$

Using this new input vector, the hidden state at time t can be updated as:

$$\mathbf{h}'_t = f_{\text{lstm}}\left(\mathbf{h}'_{t-1}, \tilde{\mathbf{z}}_t\right),$$

where f_{lstm} is the LSTM function.

3.3. T-attention

In order to predict the output $\hat{y}_{T+1}$, another LSTM-based RNN is used to introduce an attention that can select the importance of time step. Considering the input series $\mathbf{z}_t = (z_t^1, z_t^2, \ldots, z_t^u)^\top \in \mathbb{R}^u$ at time t, the attention weight of each hidden state is calculated based on the previous hidden state $\mathbf{d}_{t-1} \in \mathbb{R}^u$ and the cell state $\mathbf{C}''_{t-1} \in \mathbb{R}^u$ of the LSTM unit, and the attention model can be constructed as:

$$r_t^i = \mathbf{v}_d^\top \tanh\left(\mathbf{W}_d\left[\mathbf{d}_{t-1}; \mathbf{C}''_{t-1}\right] + \mathbf{U}_d \mathbf{z}_i\right), \quad 1 \le i \le T,$$

$$\beta_t^i = \frac{\exp\left(r_t^i\right)}{\sum_{j=1}^T \exp\left(r_t^j\right)},$$

where $[\mathbf{d}_{t-1}; \mathbf{C}''_{t-1}] \in \mathbb{R}^u$ is a concatenation of the previous hidden state and the cell state of the LSTM unit, and $\mathbf{v}_d \in \mathbb{R}^m$, $\mathbf{W}_d \in \mathbb{R}^{m\times 2u}$, and $\mathbf{U}_d \in \mathbb{R}^{m\times m}$ are parameters to be learned. The attention weight β_t^i denotes the importance of the ith hidden state for prediction. The attention mechanism computes the context vector $\mathbf{c}_t$ as the weighted average of all new input vectors as:

$$\mathbf{c}_t = \sum_{i=1}^T \beta_t^i \tilde{\mathbf{z}}_i.$$

Note that the context vector $\mathbf{c}_t$ is distinct at each time step.

Once we get the context vector, we can obtain $\tilde{y}_{t-1}$ as:

$$\tilde{y}_{t-1} = \tilde{\mathbf{w}}^\top [y_{t-1}; \mathbf{c}_{t-1}] + \tilde{b},$$

where $[y_{t-1}; \mathbf{c}_{t-1}] \in \mathbb{R}^{m+1}$ is a concatenation of the decoder input y_{t-1} and the calculated context vector $\mathbf{c}_{t-1}$. The parameters $\tilde{\mathbf{w}} \in \mathbb{R}^{m+1}$ and $\tilde{b} \in \mathbb{R}$ map the connected part to the input size of the T-attention.

The newly calculated $\tilde{y}_{t-1}$ can be used to update the decoder hidden state at time t as:

$$\mathbf{d}_t = f_2(\mathbf{d}_{t-1}, \tilde{y}_{t-1})$$

where f_2 is the LSTM function. Then $\mathbf{d}_t$ can be updated as:

$$\begin{aligned}
\mathbf{f}'_t &= \sigma\left(\mathbf{W}'_f\left[\mathbf{d}_{t-1}; \tilde{y}_{t-1}\right] + \mathbf{b}'_f\right), \\
\mathbf{i}'_t &= \sigma\left(\mathbf{W}'_i\left[\mathbf{d}_{t-1}; \tilde{y}_{t-1}\right] + \mathbf{b}'_i\right), \\
\mathbf{o}'_t &= \sigma\left(\mathbf{W}'_o\left[\mathbf{d}_{t-1}; \tilde{y}_{t-1}\right] + \mathbf{b}'_o\right), \\
\mathbf{C}''_t &= \mathbf{f}'_t \odot \mathbf{C}''_{t-1} + \mathbf{i}'_t \odot \tanh\left(\mathbf{W}'_c\left[\mathbf{d}_{t-1}; \tilde{y}_{t-1}\right] + \mathbf{b}'_c\right), \\
\mathbf{d}_t &= \mathbf{o}'_t \odot \tanh\left(\mathbf{C}''_t\right),
\end{aligned}$$

Table 1: Variables of KOSPI 200 data (https://kr.investing.com/)

Target variable	KOSPI200 index
Exogenous variable	165 among 200 companies such as Samsung Electronics, SK Hynix, Samsung Biologics, Naver, Hyundai Motors, and LG Chem.

Table 2: Variables of SML 2010 data (https://archive.ics.uci.edu/ml/datasets/SML2010)

Target variable	Indoor temperature (dining room)
Exogenous variable	Indoor temperature (room)
	Weather forecast temperature
	Carbon dioxide in ppm (dining room)
	Carbon dioxide in ppm (room)
	Relative humidity (dining room)
	Relative humidity (room)
	Lighting (dining room)
	Lighting (room)
	Rain (Percentage of the last 15 minutes in which rain was detected)
	Sun dusk
	Wind (m/s)
	Sun light in west facade
	Sun light in east facade
	Enthalpic motor 2, 0, or 1
	Enthalpic motor turbo, 0 or 1
	Outdoor temperature

where $[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] \in \mathbb{R}^{u+1}$ is a concatenation of the previous hidden state $\mathbf{d}_{t-1}$ and the input value $\tilde{y}_{t-1}$ of the LSTM of the T-attention. $\mathbf{W}'_f$, $\mathbf{W}'_i$, $\mathbf{W}'_o$, $\mathbf{W}'_c \in \mathbb{R}^{u\times(u+1)}$ and $\mathbf{b}'_f$, $\mathbf{b}'_i$, $\mathbf{b}'_o$, $\mathbf{b}'_c \in \mathbb{R}^u$ are parameters to be learned. σ and $\odot$ are logistic sigmoid functions and element multiplications, respectively.

4. Experiment

We conduct an experiment using two types of data sets to check whether the proposed model has good performance in stock price data as well as in time series data in other fields. A stock price prediction experiment is conducted using KOSPI 200 data. A general prediction experiment on time series data in various fields other than stock prices is conducted using temperature data and energy data.

4.1. Data description

- KOSPI 200: This is a data set where, among the listed companies on KOSPI, 200 stocks considered to represent KOSPI due to their large market capitalizations and large trading volume are selected, and used to predict the stock index. This is daily data measured for a total of 2158 days from August 23, 2011 to May 29, 2020. In our experiment, the KOSPI 200 index is used as the target variable, and the stock prices for 165 companies are used as exogenous variables (Table 1). These companies are selected so that there are no missing values in the data for companies such as Samsung Electronics, SK Hynix, Samsung Biologics, Naver, Hyundai Motor Company, and LG Chem.

- SML 2010 (Zamora-Martinez *et al.*, 2014): This is a public dataset used for indoor temperature prediction as data collected from a monitor system installed in a general house. Data was sampled every minute and calculated every 15 minutes on average. It consists of approximately 40 days (before and after 43 days) of monitoring data. We use 4137 data. In the experiment, 16 exogenous variables are used as shown in Table 2, and data having missing values and variables such as date

Table 3: Variables of Appliances energy data (https://archive.ics.uci.edu/ml/datasets/Appliances+energy +prediction)

Target variable	Appliances, energy use in Wh
Exogenous variable	lights, energy use of light fixtures in the house in Wh
	T1, Temperature in kitchen area, in Celsius
	RH_1, Humidity in kitchen area, in %
	T2, Temperature in living room area, in Celsius
	RH_2, Humidity in living room area, in %
	T3, Temperature in laundry room area
	RH_3, Humidity in laundry room area, in %
	T4, Temperature in office room, in Celsius
	RH_4, Humidity in office room, in %
	T5, Temperature in bathroom, in Celsius
	RH_5, Humidity in bathroom, in %
	T6, Temperature outside the building (north side), in Celsius
	RH_6, Humidity outside the building (north side), in %
	T7, Temperature in ironing room , in Celsius
	RH_7, Humidity in ironing room, in %
	T8, Temperature in teenager room 2, in Celsius
	RH_8, Humidity in teenager room 2, in %
	T9, Temperature in parents room, in Celsius
	RH_9, Humidity in parents room, in %
	To, Temperature outside (from Chievres weather station), in Celsius
	Pressure (from Chievres weather station), in mm Hg
	RH_out, Humidity outside (from Chievres weather station), in %
	Wind speed (from Chievres weather station), in m/s
	Visibility (from Chievres weather station), in km
	Tdewpoint (from Chievres weather station), in Celsius
	rv1, Random variable 1, nondimensional
	rv2, Random variable 2, nondimensional

and time are removed. Among the indoor temperatures, the temperature of dinning room is used as the target variable.

- Appliances energy (Candanedo *et al.*, 2017): This data has been used to predict appliance energy use in low energy buildings. Data were measured at intervals of 10 minutes over a period of 4.5 months. The total size of data is 19734, and as shown in Table 3; 27 exogenous variables are used (excluding date) and appliances, energy consumption, are used as the target variable. For time series data, there should be stationarity before make any forecast. So we preprocessed the three time series data with min max scaling using Scikit-learn packages (Pedregosa et al., 2011).

4.2. Design of experiment

The parameters of our model include the LSTM dimension r, the MI-LSTM dimension u, time window size T, the number of exogenous variables with positive correlation P and the number of exogenous variables with negative correlation N, the size of hidden states of the LSTM for H-attention, q, and the size of hidden states of the LSTM for T-Attention, m.

In our experiment, in the case of r and m, we selected 128 as a result of conducting experiments with 64 and 128. In addition, P and N are set to be the same and selected among {4,6,10,15}; in addition, the time window size T is selected among {10,16,32}. Lastly, 0.8, 0.1, and 0.1 are used as the ratio of training, validation, and test datasets.

We consider three different metrics to evaluate the performance of the models in time series pre-

Table 4: Time series prediction results over the KOSPI 200 dataset, SML 2010 dataset and Appliances energy dataset (best performance displayed in **boldface**)

Model	KOSPI 200 dataset			SML 2010 dataset			Appliances energy dataset		
	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE
ARIMA	**6.358**	29.696	23.791	19.440	3.680	3.098	49.855	97.498	53.859
DA-RNN	10.982	2.647	2.664	16.872	11.272	1.148	32.883	32.488	6.049
MI-LSTM	10.182	0.420	0.556	5.380	0.240	0.321	7.120	0.562	1.243
2DA-MILSTM	9.100	**0.356**	**0.472**	**2.300**	**0.198**	**0.289**	**5.110**	**0.510**	**1.166**

diction experiments. Assuming y_t is the target at time t and $\hat{y}_t$ is the predicted value at time t,

- Mean absolute percentage error (MAPE): a value obtained by taking the average of the absolute percentage of the residual

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_t^i - \hat{y}_t^i}{y_t^i}\right|.$$

- Root mean square error (RMSE): a value obtained by taking the root of the average of the squares of the residual

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_t^i - y_t^i\right)^2}.$$

- Mean absolute error (MAE): average of all absolute errors

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{y}_t^i - y_t^i\right|.$$

Note that the MSE, RMSE, and MAE are calculated using normalized data due to huge difference in amount among different time series data.

To compare the performance of 2DA-MILSTM, we conduct a comparative experiment with two models, DA-RNN and MI-LSTM. Also, not based on neural network model, we conduct the classical time series model in statistics field, ARIMA. Designed by Hyndman and Khandakar, auto.arima in forecast package for R (Hyndman and Benítez, 2016), automatically determines the orer of ARIMA model using step-wise algorithm for forecasting. The experiment was conducted using KOSPI 200 data (stock price data), SML 2010 data (room temperature data), and appliances energy data (energy data).

4.3. Results

The time series prediction results of 2DA-MILSTM and three existing methods over the three datasets are shown in Table 4. In Table 4, we observe that the MAPE, RMSE and MAE of ARIMA are larger than those of the other neural net-based methods. There are two interesting exceptions for such observation, one of which is MAPE for KOSPI 200 dataset. ARIMA outperforms the other methods for the data in terms of MAPE. The possible reasons are (1) The stock price by nature reveals higher volatility as its level goes up, (2) The MAPE puts a heavier penalty on negative errors than on positive errors. These reasons are because of the high volatility at high level of stock price it is difficult to

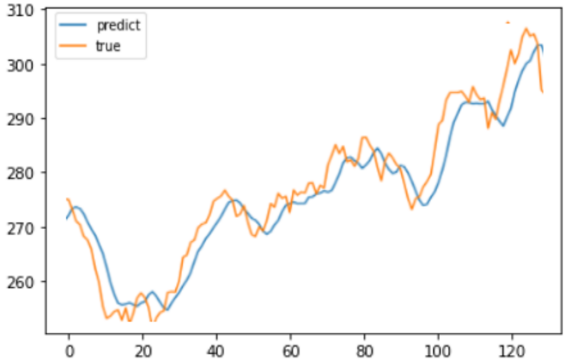

Figure 8: *Prediction of KOSPI 200 using DA-RNN.*

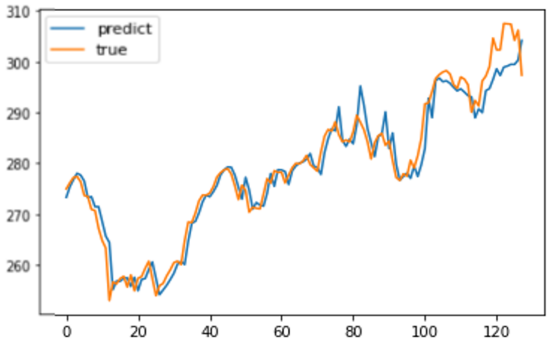

Figure 9: *Prediction of KOSPI 200 using MI-LSTM.*

predict it accurately and there are possibly more positive errors at high level. Subsequently, it can be small even though a model does not predict well at high level since the MAPE puts a lighter penalty on positive errors.

We can also see that the proposed model outperforms the other methods for all three datasets in terms of all three criteria, except the case decsribed above. This suggests that it might be beneficial to consider the correlation between exogenous variables and target variable by introducing MI-LSTM. We can consider the importance of exogenous variables using the input attention; however, DA-RNN just downgrades the importance of variables with negative correlation with the target variable. With integration of the MI-LSTM as well as two attention mechanisms, the proposed 2DA-MILSTM achieves the best MSE, RMSE and MAE across three datasets since it uses the MI-LSTM to consider the correlation of exogenous variables effectively as well as employs H- and T-attention mechanisms to capture relevant hidden features across all time steps as well as potential output information.

For visual comparison, we show the prediction results of the three neural net-based methods over certain ranges of the three datasets in Figures 8–16.

Figures 8–10 show that our model makes better predictions for KOSPI 200 data than the other two models. In particular, in the case of DA-RNN, it follows the overall trend, but we can some time shift across all time period. In the case of MI-LSTM, the prediction is accurate until time point of 70, but after that time, the result deteriorates. However, we can see that 2DA-MILSTM predicts almost

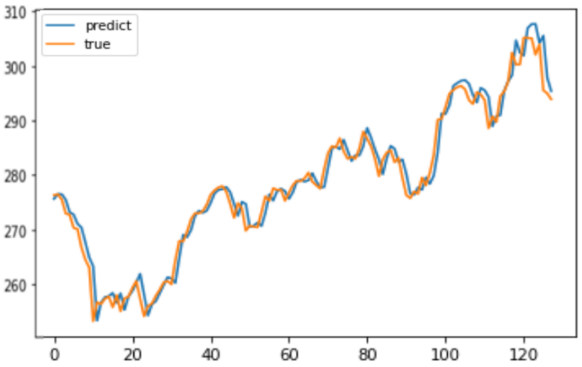

Figure 10: *Prediction of KOSPI 200 using 2DA-MILSTM.*

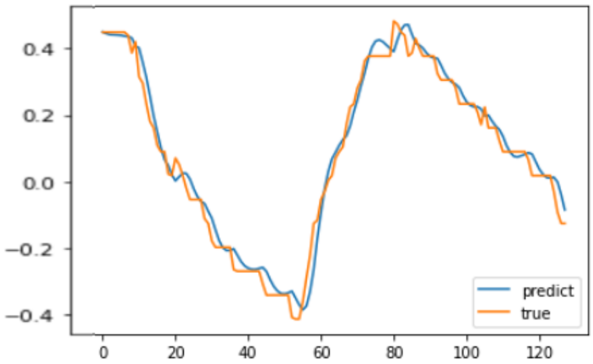

Figure 11: *Prediction of SML 2010 using DA-RNN.*

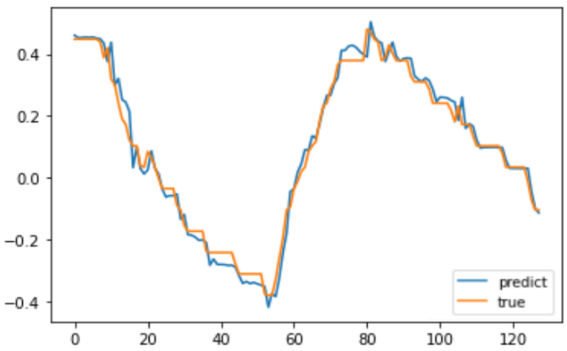

Figure 12: *Prediction of SML 2010 using MI-LSTM.*

exactly the actual value except slight overestimating and shifting after time point of 115.

Figures 11–13 show that our model predicts SML 2010 data better than the other two models. In the case of both DA-RNN and MI-LSTM, we can see that the prediction before time point of 60 is not accurate (the prediction is accurate only between 60 and 75 for DA-RNN). Using MI-LSTM, in

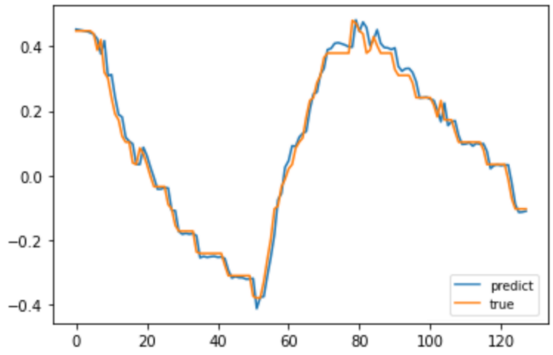

Figure 13: *Prediction of SML 2010 using 2DA-MILSTM.*

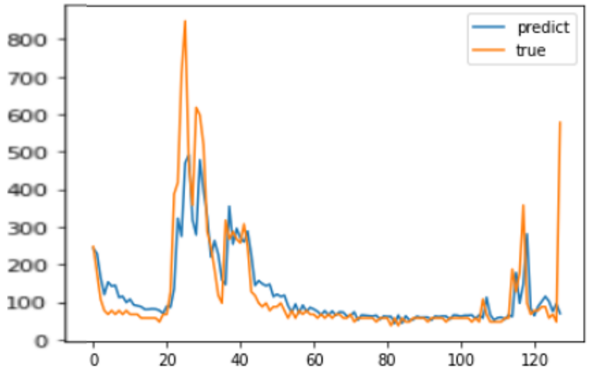

Figure 14: *Prediction of Appliances energy using DA-RNN.*

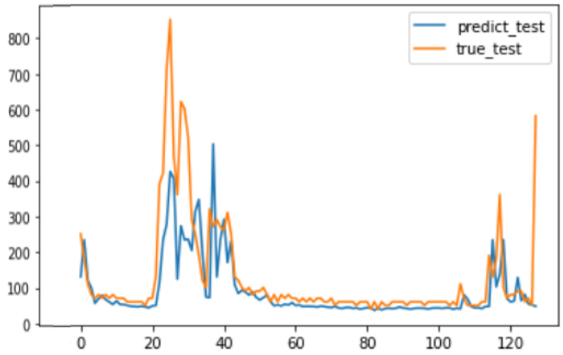

Figure 15: *Prediction of Appliances energy using MI-LSTM.*

addition to the range of 60–75, the prediction is pretty accurate after time point of 105. However, we can see that 2DA-MILSTM predicts very accurately over all the time period except between 70 and 95.

Figures 14–16 show that our model makes better predictions for appliances energy data than the

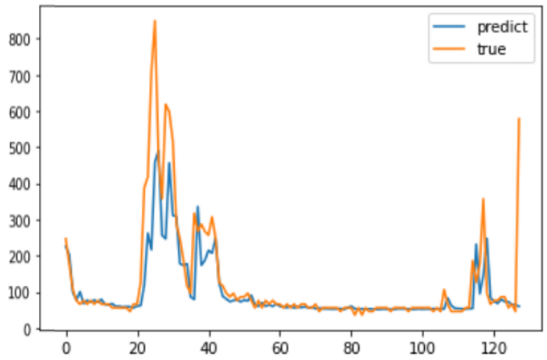

Figure 16: *Prediction of Appliances energy using 2DA-MILSTM.*

other two models. In the case of this data, all three methods do not predict accurately between time point of 20 and 40. DA-RNN predicts slightly better than the other two models for the time period. However, it overestimates before time point of 20. We can also see that MI-LSTM underestimates between 40 and 110. However, 2DA-MILSTM predicts very accurately before 20 and between 40 and 110.

5. Conclusion

In this paper, we proposed a two-dimensional attention-based MI-LSTM (2DA-MILSTM) motivated by the advantages of MI-LSTM, which effectively uses exogenous variables with low correlation, and DA-RNN and 2D-ALSTM, in which an attention mechanism is introduced twice. The proposed method was applied to KOSPI 200, SML 2010 and appliance energy datasets. The results show that our proposed 2DA-MILSTM can outperform existing methods for time series prediction.

The proposed model, 2DA-MILSTM, can selectively use output value information and temporal information by introducing H-attention and T-attention. H-attention can better capture potential information of the output value, and T-attention can capture temporal information. 2DA-MILSTM, which uses the two attention mechanisms selects the output value with accurate information as well as captures temporal information appropriately to improve predictive performance. Better forecasting depends on capturing long- and short-term dependencies. A longer forecasted time series increases variability. While MI-LSTM capturing the dependencies, T-attention and H-attention make the output values of MI-LSTM stable considering temporal and potential values. This means that this task can control the increasing variability of time series data because the two attention mechanism weights control both temporal and potential information.

The stock price has high volatility at high level. Therefore, if one uses the log-return of the stock price rather than the original scale, then the prediction at high level might be more accurate; consequently, the performance of the model can be improved. However, in the literature of NARX field, there are several papers where the stock prices are used as the original scale, and two of which are the motivation for our research, DA-RNN and MI-LSTM. Therefore, we believe that using the same (original) scale for the stock price makes it easier for readers to compare our proposed model with the two models.

As a follow-up study, visualization method for H-attention and T-attention weights could be considered to check where the two attention mechanisms concentrate among exogenous variables, time

stamp and logits from MI-LSTM each. In addition, non-linear relation between exogenous variables and target variable should be considered. When applying non-linear relations, it can be expected that the proposed 2DA-MILSTM model receives additional information between exogenous variables and the target variables. Also, various ranges of the hyperparameters should be experimented with while considering the relationship between hyperparameters.

Acknowledgement

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation (NRF) of Korea funded by the Ministry of Science, ICT (NRF-2017M3C4A7083281). We thank the referees for many important comments which helped to improve the presentation of the manuscript.

References

Candanedo LM, Feldheim V, and Deramaix D (2017). Data driven prediction models of energy use of appliances in a low-energy house, *Energy and Buildings*, **140**, 81–97.

Chen S, Wang XX, and Harris CJ (2008). NARX-based nonlinear system identification using orthogonal least squares basis hunting, *IEEE Transactions on Control Systems Technology*, **16**, 78–84.

Cho K, van Merriënboer B, Bahdanau D, and Bengio Y (2014a). On the properties of neural machine translation: Encoder-decoder approaches, arXiv:1409.1259.

Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, and Bengio Y (2014b). Learning phrase representations using RNN encoder-decoder for statistical machine translation, arXiv:1406.1078.

Elman JL (1991). Distributed representations, simple recurrent networks, and grammatical structure, *Machine Learning*, **7**, 195–225.

Hochreiter S and Schmidhuber J (1997). Long short-term memory, *Neural Computation*, **9**, 1735–1780.

Huang Z, Xu W, and Yu K (2015). Bidirectional LSTM-CRF models for sequence tagging. arXiv: 1508.01991.

Hyndman RJ and Benítez JM (2016). Bagging exponential smoothing methods using STL decomposition and Box-Cox transformation, *International Journal of Forecasting*, **32**, 303–312.

Jang E, Gu S, and Poole B (2016). Categorical reparameterization with gumbel-softmax, arXiv:1611.01144.

Li H, Shen Y, and Zhu Y (2018). Stock price prediction using attention-based multi-Input LSTM. In *Proceedings of the 10th Asian Conference on Machine Learning*, 454–469.

Li G, Wen C, Zheng W, and Chen Y (2011). Identification of a class of nonlinear autoregressive models with exogenous inputs based on kernel machines, *IEEE Transactions on Signal Processing*, **59**, 2146–2159.

Liu B and Lane I (2016). Attention-based recurrent neural network models for joint intent detection and slot filling, arXiv:1609.01454.

Liu Y, Gong C, Yang L, and Chen Y (2019). DSTP-RNN: a dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction, arXiv:1904.07464.

McLeod AI and Li WK (1983). Diagnostic checking ARMA time series models using squared-residual autocorrelations, *Journal of Time Series Analysis*, **4**, 269–273.

Nair V and Hinton GE (2010). Rectified linear units improve restricted boltzmann machines. In *ICML*.

Pedregosa F, Varoquaux G, Gramfort A, *et al.* (2011). Scikit-learn: machine learning in Python, *The Journal of Machine Learning Research*, **12**, 2825–2830.

Pham H, Tran V, and Yang BS (2010). A hybrid of nonlinear autoregressive model with exogenous input and autoregressive moving average model for long-term machine state forecasting, *Expert Systems with Applications*, **37**, 3310–3317.

Qin Y, Song D, Chen H, Cheng W, Jiang G, and Cottrell G (2017). A dual-stage attention-based recurrent neural network for time series prediction, arXiv:1704.02971.

Rumelhart DE, Hinton GE, and Williams RJ (1986). Learning internal representations by backpropagating errors, *Nature*, **323**, 533–536.

Sutskever I, Vinyals O, and Le QV (2014). Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, 3104–3112.

Tao Y, Ma L, Zhang W, Liu J, Liu W, and Du Q (2018). Hierarchical attention based recurrent highway networks for time series prediction, arXiv:1806.00685.

Werbos P (1990). Backpropagation through time: What it does and how to do it. In *Proceedings of the IEEE*, **78**, 1550–1560.

Yang Z, Yang D, Dyer C, He X, Smola A, and Hovy E (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1480–1489.

Yu Y and Kim YJ (2019). Two-dimensional attention-based LSTM model for stock index prediction, *Journal of Information Processing Systems*, **15**, 1231–1242.

Zamora-Martinez F, Romeu-Guallart P, and Pardo J (2014). UCI Machine Learning Repository: SML2010 Data Set, *UCI Machine Learning Repository*. Available from: https://archive.ics.uci.edu/ml/ datasets/SML2010

Received August 19, 2020; Revised December 19, 2020; Accepted December 23, 2020