Supplementary Materials

Development of daily PM_{10} and $PM_{2.5}$ prediction system using a deep long short-term memory neural network model

Hyun Soo Kim[1,†], Inyoung Park[2,†], Chul Han Song[1], Kyunghwa Lee[1], Jae Woong Yun[2], Hong Kook Kim[2], Moongu Jeon[2], Jiwon Lee[2]

Shortened title: Deep LSTM model for daily PM_{10} and $PM_{2.5}$ predictions

1. School of Earth Sciences and Environmental Engineering, Gwangju Institute of Science and Technology (GIST), Gwangju 61005, Korea
2. School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology (GIST), Gwangju 61005, Korea

***Corresponding author:** C. H. Song
Email: chsong@gist.ac.kr, Phone: +82-62-715-3276

[†]Both authors are equally contributed to this work

(**Submitted to Atmos. Chem. & Phys**)

LSTM cell architecture

LSTM neural network is a kind of recurrent neural networks (RNNs) developed for being able to learn long-term dependencies with gradient descent. In contrast with RNNs, the vanishing gradient problem is resolved by controlling the hidden state from the gates in LSTM. The architecture of LSTM can differ depending on the structure and logistic function for its gate activations. For a given time step *t*, the gates in the LSTM calculate the gate output vectors with the input vector, x_t, the previous hidden state vector, h_{t-1}, and the previous cell vector, c_{t-1}, by the following equations:

$$i_t = \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right) \tag{S1}$$

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{S2}$$

$$o_t = \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \tag{S3}$$

where i_t, f_t, and o_t are the output vectors of input, forget, and output gate at the time step *t*; σ is an activation function that is usually realized as the logistic sigmoid function ranging from 0 to 1, which is denoted as

$$\sigma(x) = \begin{cases} 0 & when\ x < -2.5 \\ 0.2x + 0.5 & when -2.5 \le x \le 2.5 \\ 1 & when\ x < -2.5 \end{cases} \tag{S4}$$

In addition, Ws and Us in Eqs. (S1)-(S3) denote weight matrices for computing gate output vectors. For example, W_i in Eq. (S1) is a weight matrix connecting the input vector, x_t, to the input gate output vector, i_t. Finally, b_i, b_f, and b_o are the biases that are added to each gate to adjust the center of a data space according to the training data.

The LSTM cell takes a block input to decide whether it forgets the entire previous memory or ignores new input data by applying an activation function that is implemented by a hyperbolic tangent function. Consequently, the cell vector at time *t* is computed as

$$z_t = \tanh\left(W_c x_t + U_c h_{t-1} + b_c\right) \tag{S5}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ z_t \tag{S6}$$

where c_{t-1} is the cell state vector at the previous time step; h_{t-1} is the hidden state vector at time step t-1; $\circ$ represents the element-wise multiplication; and b_c is a bias for the cell. In addition, the hyperbolic tangent function is represented as

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{S7}$$

As shown in Eqs. (S5) and (S6), the memory of LSTM keeps the amount of the multiplication of the previous cell vector, c_{t-1}, and the forget gate output vector, f_t, as well as the partial amount of the input gate output vector, i_t, controlled by z_t.

Next, the cell vector, c_t, is used to obtain the hidden state vector, h_t, by applying the hyperbolic tangent activation function to c_t, such as

$$h_t = o_t \circ \tanh(c_t) \tag{S8}$$

Unlike to the RNN, the LSTM has three gates that control properly truncating the lengths of data sequences, which can solve the vanishing gradient problem at small computational extra-costs.

Fig. S3 presents the architecture of an LSTM block applied in this study. As shown in Fig. S3, there are three gates and one cell state (i.e., f_t, i_t, o_t, and c_t) in the LSTM cell. The key element of LSTM is c_t (check the upper horizontal line in Fig. S3), which

determines whether it forgets the previous state or ignores the current state. As shown in Eq. (S6), c_t can be estimated by merging the two terms. The first term represents the amount of forget, represented by the dot product between f_t and c_{t-1}. The first vertical line on the left-hand side of Fig. S3 denotes forget gate, which determines the loss of the past memory. Because the forget gate has a sigmoid logistic function, the outcome of f_t is between 0 and 1. When f_t equals 0, the memory of the previous time step (c_{t-1}) is completely erased from c_t. In contrast, when f_t is 1, the previous memory is completely stored. The second gate (i_t) decides how much of the current information to memorize (check the two vertical lines merged into one line in Fig. S3). The values of its vector matrix are the outcomes of element-wise multiplication between i_t and $\tanh\left(W_c x_t + U_c h_{t-1} + b_c\right)$. The operation principle of i_t is the same for f_t. The value of $\tanh\left(W_c x_t + U_c h_{t-1} + b_c\right)$ is from -1 to 1 because of the hyperbolic tangent function. The storing ratio and directionality of current state are determined by i_t and $\tanh\left(W_c x_t + U_c h_{t-1} + b_c\right)$. The third gate of LSTM is output gate (o_t). As shown in Eq. (S8), the value of the hidden layer output (h_t) is the Hadamard product between o_t and $\tanh(c_t)$. In the calculations of h_t, o_t and $\tanh(c_t)$ control the intensity and directionality of output vectors as in the previous gates.

ADAM optimizer

Based on the interacting ways with the input variables, there are two categories in optimization algorithms: (i) deterministic and (ii) stochastic. Deterministic algorithms update the weight and bias only once for one epoch. In contrast, stochastic optimization algorithms estimate learnable parameters several times for one iteration because they use training data set divided into mini-batches in the model training. Since the loss is calculated only for the

collection of small data (mini-batch) instead of whole data set, stochastic algorithms are more efficient than deterministic ones. In addition, such stochastic approaches have less risk of local minima.

In this study, we utilized adaptive moment estimation (ADAM) as an optimizer. In this algorithm, the individual adaptive learning rates for different learning parameters are computed by estimating the first and second moments of their gradient. The detailed expressions of the ADAM algorithm are shown below:

$$g_t = \nabla_\theta f(\theta_{t-1}) \tag{S9}$$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{S10}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^{\,2} \tag{S11}$$

$$\hat{m}_t = m_1 / (1-\beta_1^t) \tag{S12}$$

$$\hat{v}_t = m_2 / (1-\beta_2^t) \tag{S13}$$

$$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \varepsilon}} \tag{S14}$$

where g_t is the gradient with respect to stochastic objective at time step t; m_t and v_t are the first and second moment of gradient; $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first and second moment; β_1, β_1, α, and ε are set to 0.9, 0.999, 0.001, and 10^{-8}, respectively.[28] m_t provides inertia in the gradient descent (backpropagation). Since inertial force can accelerate the speed of gradient descent, ADAM performs fast computation and increases the possibility of global minima. Moreover, because v_t is the exponential moving average of the squared gradient, the relative difference between the variables of recent variation can be maintained without increasing v_t infinitely. Here, the decay rates of m_t and v_t are governed by β_1

and β_2 (see Eqs. (S10)-(S11)). In the calculation of weight and bias, α is the learning rate that determines the adjusting degree. In addition, ε prevents the divergence of learnable parameters (see Eq. (S14)).

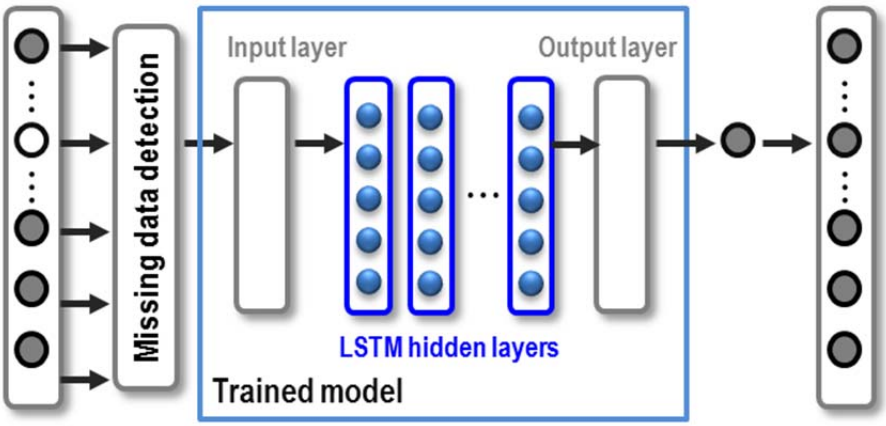

Figure S1. Schematic diagram for the generation of missing values.

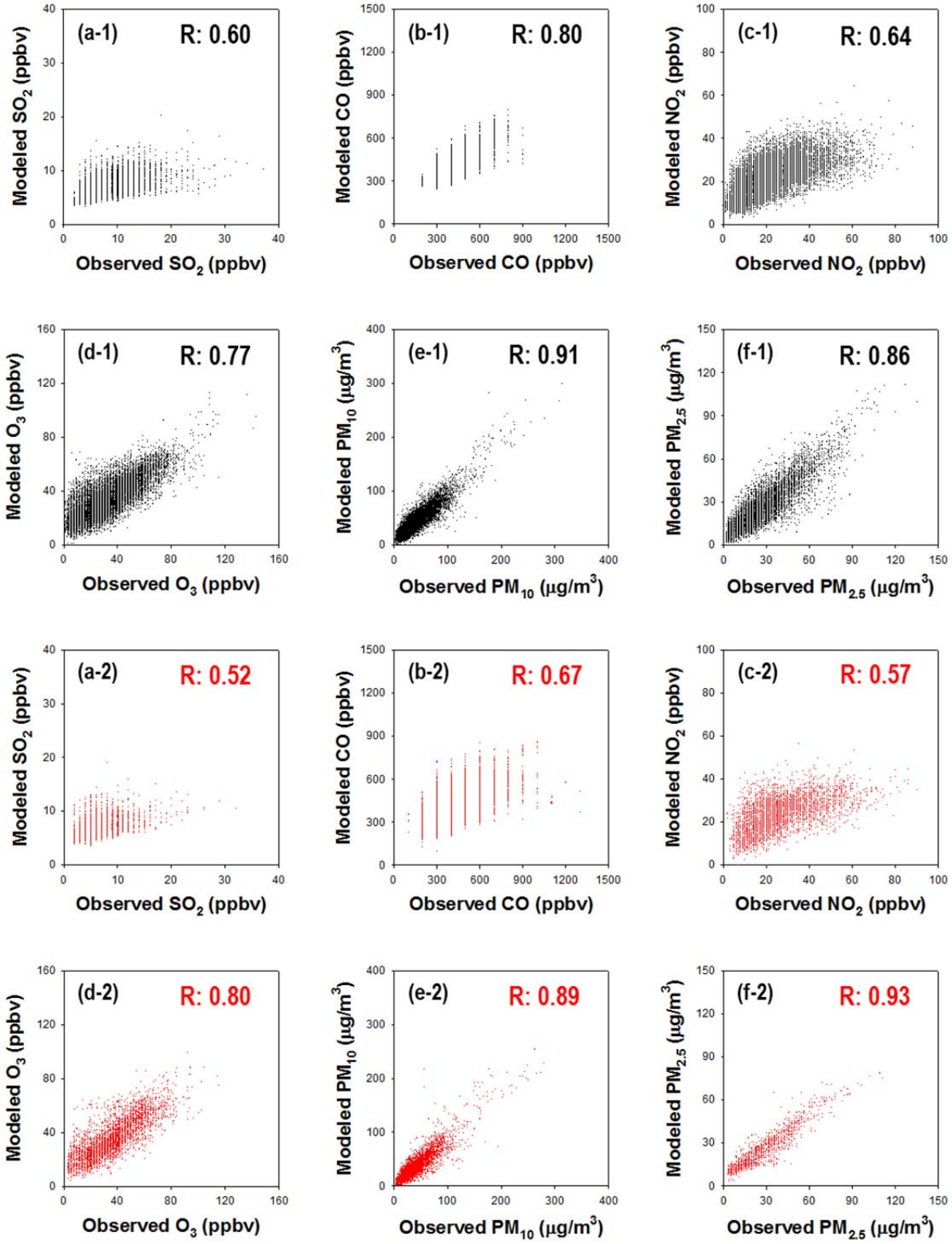

Figure S2. Evaluation of missing value generations for Seoul-1 site: (a) SO_2; (b) CO, (c) NO_2; (d) O_3; (e) PM_{10}; (f) $PM_{2.5}$. Black and red dots represent the training and validation results, respectively.

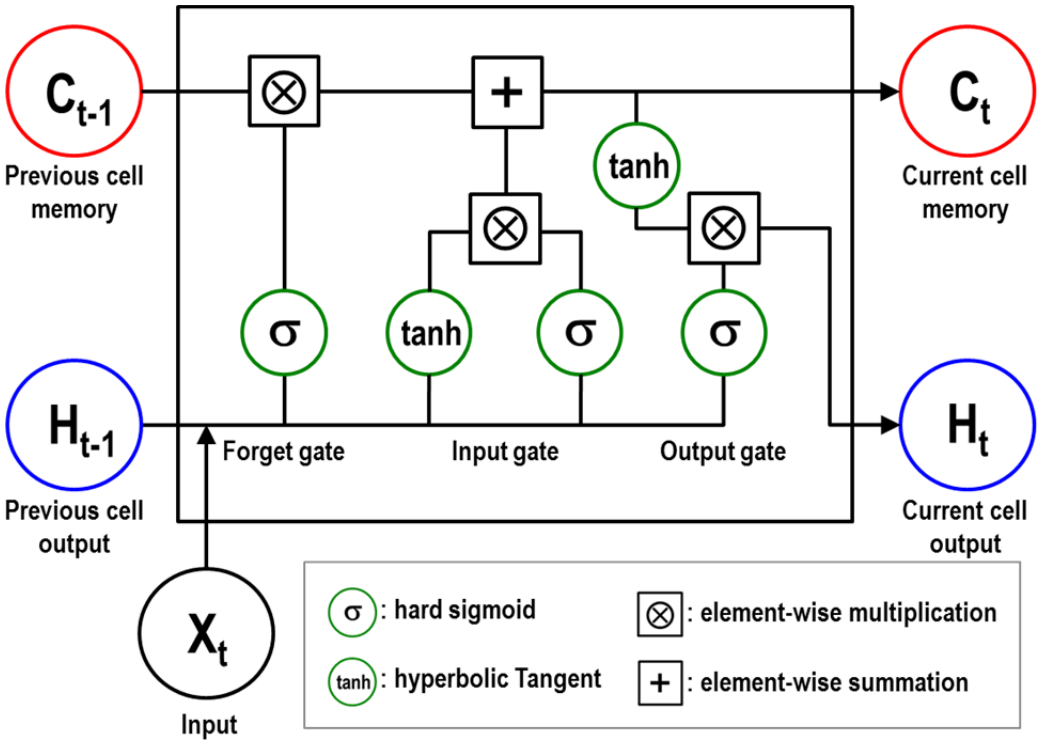

Figure S3. Structure of LSTM memory cell embedded as hidden layers of the newly-developed PM_{10} and $PM_{2.5}$ prediction system.

Table S1. Summary of statistical analysis in the generation of missing observations

Site	Training						Validation						
	Statistics	SO_2	CO	NO_2	O_3	PM_{10} [3]	$PM_{2.5}$ [3]	Statistics	SO_2	CO	NO_2	O_3	PM_{10} [3]	$PM_{2.5}$ [3]
Seoul-1	IOA	0.74	0.90	0.92	0.96	0.98	0.94	IOA	0.46	0.82	0.88	0.95	0.94	0.94
	RMSE[1]	1.83	145.55	9.48	7.00	9.72	6.55	RMSE[1]	1.93	153.39	9.87	7.06	9.82	6.69
	MB[1]	0.13	-7.15	0.54	-1.11	0.14	-0.59	MB[1]	-0.75	-79.76	-0.39	-0.95	-1.32	-1.44
	MNGE[2]	43.75	31.99	26.55	55.02	18.64	26.62	MNGE[2]	24.62	24.50	26.77	40.33	19.84	25.08
	MNB[2]	2.72	-1.58	1.49	-5.79	0.28	-2.41	MNB[2]	-17.00	-20.25	-1.26	-5.69	-3.10	-5.76
Seoul-2	IOA	0.84	0.94	0.91	0.96	0.97	0.95	IOA	0.56	0.88	0.88	0.96	0.96	0.94
	RMSE[1]	1.13	119.20	8.15	6.65	11.59	6.68	RMSE[1]	1.32	134.23	9.69	7.89	13.02	6.70
	MB[1]	-0.08	19.80	1.29	-1.50	-1.03	0.68	MB[1]	0.21	-52.39	0.86	-0.56	-2.28	1.11
	MNGE[2]	14.52	18.52	21.05	30.49	5.81	31.32	MNGE[2]	18.31	16.06	27.30	48.38	17.51	27.96
	MNB[2]	-1.36	3.87	3.19	-4.67	-12.88	2.79	MNB[2]	3.50	-9.70	2.41	-2.16	-3.96	3.89
Daejeon	IOA	0.93	0.89	0.84	0.95	0.95	-	IOA	0.84	0.82	0.77	0.96	0.95	-
	RMSE[1]	0.79	112.55	6.03	9.04	12.59	-	RMSE[1]	0.86	132.47	6.09	9.18	12.75	-
	MB[1]	-0.01	-6.90	0.06	-0.61	1.56	-	MB[1]	-0.27	-60.69	-0.79	4.90	-1.49	-
	MNGE[2]	27.08	5.23	54.69	55.14	33.62	-	MNGE[2]	21.93	25.31	45.76	52.63	22.69	-
	MNB[2]	-0.19	0.95	0.45	-2.20	3.76	-	MNB[2]	-9.65	-14.70	-7.03	15.34	-3.02	-
Gwangju	IOA	0.87	0.83	0.79	0.94	0.96	0.89	IOA	0.67	0.75	0.79	0.93	0.95	0.93
	RMSE[1]	1.06	174.00	9.07	8.86	13.22	8.35	RMSE[1]	1.11	181.52	9.38	9.40	14.18	8.37
	MB[1]	-0.06	35.21	-0.28	1.07	4.40	-0.80	MB[1]	-0.42	-35.66	1.36	0.24	0.61	-0.69
	MNGE[2]	24.21	30.61	37.55	64.58	47.31	43.22	MNGE[2]	23.74	21.61	49.96	74.92	29.81	31.70
	MNB[2]	-1.64	6.93	-7.46	3.92	11.55	-3.81	MNB[2]	-13.83	-6.57	6.41	0.77	1.31	-2.88
Daegu	IOA	0.81	0.89	0.88	0.97	0.91	0.91	IOA	0.68	0.87	0.89	0.94	0.87	0.88
	RMSE[1]	2.38	123.99	8.83	6.81	15.48	8.57	RMSE[1]	2.46	127.17	11.30	6.97	16.30	8.72
	MB[1]	0.08	-2.86	0.66	-0.95	1.65	-0.11	MB[1]	0.01	-49.65	-5.08	1.98	7.42	-0.87
	MNGE[2]	65.92	70.80	38.95	50.66	34.72	26.07	MNGE[2]	54.14	25.02	28.68	94.54	34.69	26.85
	MNB[2]	2.15	-0.42	2.82	-4.08	1.10	-0.39	MNB[2]	0.23	-11.59	-15.71	4.12	14.77	-3.12
Ulsan	IOA	0.85	0.83	0.89	0.88	0.96	0.94	IOA	0.70	0.76	0.88	0.88	0.97	0.94
	RMSE[1]	5.72	145.54	7.64	9.50	11.58	7.28	RMSE[1]	5.92	151.36	7.99	9.81	11.62	7.46
	MB[1]	-0.22	0.25	0.77	0.35	-0.80	0.34	MB[1]	1.65	41.02	2.51	-1.99	0.52	1.49
	MNGE[2]	44.70	21.49	41.18	38.76	24.18	32.12	MNGE[2]	52.01	26.06	34.88	35.87	28.09	38.49
	MNB[2]	-2.82	0.04	3.98	1.16	-1.71	1.33	MNB[2]	26.50	7.80	12.53	-5.63	1.06	5.75
Busan	IOA	0.65	0.87	0.75	0.86	0.94	0.92	IOA	0.63	0.78	0.67	0.88	0.93	0.94
	RMSE[1]	2.48	73.69	9.34	10.85	11.36	7.88	RMSE[1]	2.56	97.16	11.85	10.87	12.94	7.90
	MB[1]	-0.23	-6.42	0.13	1.00	2.76	-0.60	MB[1]	0.58	1.65	-2.78	-0.09	-3.71	-2.09
	MNGE[2]	26.36	13.06	46.96	47.13	31.98	30.64	MNGE[2]	36.95	19.77	40.16	39.10	25.86	32.06
	MNB[2]	-3.36	-1.52	0.61	2.96	7.08	-2.37	MNB[2]	9.26	0.40	-10.56	-0.25	-9.31	-7.54

[1] Units are in ppbv, except for PM_{10} and $PM_{2.5}$; [2] units are in %; [3] for PM_{10} and $PM_{2.5}$, the units of RMSE and MB are in μg/