CONVOLUTIONAL NEURAL NETWORK (CNN)-BASED TRANSFER LEARNING IMPLEMENTED TO TIME-SERIES FLOOD FORECAST

NOBUAKI KIMURA
Institute for rural engineering, National Agriculture & food Research Organization, Tsukuba, Japan, kimuran590@affrc.go.jp

DAICHI BABA
ARK Information Systems, INC., Tokyo, Japan, baba.daichi@ark-info-sys.co.jp

ABSTRACT

A deep neural network (DNN) model has recently been employed in a riverine flood forecast system. The accurate DNN model requires lots of data for machine learning. However, large data for flood events are hardly obtained in field measurements because abundant manpower and expensive sensor installation and maintenance are required in current research. Additionally, only a few flood events are normally recorded per year even if long-term measurement is successfully conducted. To improve data limitation, we introduced transfer learning, which pre-trains a model in a source domain and then reuses it in a different domain. This method can be beneficial when implementing the DNN model to the flood forecast at the location where insufficient data is available. The transfer learning with time-series DNN model potentially has not been successful. Instead, we focused on a convolutional neural network (CNN) model, a deep learning architecture and an image analysis tool, because it has numerous examples of image analyses with a transfer learning approach. Our previous study implemented the CNN model-based transfer learning to time-series flood prediction. The model pre-trained in large-size flood events in a source watershed was retrained with a small size data in a target watershed. This study performed the extension and improvement of the model with transfer learning in the previous study. An accuracy improvement of the model was achieved with a change of training datasets in a validation method. The model prediction was able to be extended till the 3h-lead time with reduced or equivalent errors when compared with those of no-pre-trained model.

Keywords: CNN, transfer learning, flood forecast, time-series data, insufficient dataset

1. INTRODUCTION

One of the machine learning methods, an artificial neural network, has been implemented to riverine flood modeling (e.g., Hsu et al., 2010). As deep learning (Bengio et al., 2013) has been dramatically developed, deep neural network (DNN) models are currently utilized for riverine flood forecasts as accurate and quick-response models. For example, Hitokoto et al. (2016) developed a conventional-based model, such as a multilayer perceptron (MLP) model (McCulloch and Pitts, 1943), coupled with deep layers. As another example of the DNN models, the long, short-term memory (LSTM, Hochreiter and Schmidhuber, 1997) models that can accurately predict time-series data were implemented to local flood events (Yamada et al., 2018; Hu et al., 2018). The DNN models are more efficient to set up than being a physical-based model such as a rain-runoff model. No preparation of geological and hydrological information is required for the DNN models. However, an accurate prediction strongly requires high quality and large-size datasets. In particular, large datasets are hard to obtain for riverine flood modeling in local, mid- to small-size watersheds because of poor monitoring tools such as videos and sensors for water level and discharge measurements. To solve this problem, Kimura et al. (2020) proposed a DNN model coupled with a transfer learning approach, which is that a pre-trained model in a source watershed was applied to a different (target) watershed to predict the water levels in the 1h-lead time for larger flood events (Figure 1). Their study reported that the pre-trained model in the source watershed successfully reduced a computational cost and slightly improved accuracy in the target watershed after retraining parts of the model. Note that a convolutional neural network (CNN) model proposed by LeCun et al. (1998) was employed as the DNN model because the CNN model has provided numerous good examples with transfer learning in image analyses.

The CNN model created by Kimura et al. (2020) still has crucial two tasks that should be improved to make it more robust. One task is related to the improvement of prediction accuracy. The pre-trained model poorly captured the maximum peaks of water level with 10–20% errors for the prediction for the top-three largest flood events in past datasets. The errors were calculated by the typical evaluation method that consists of three processes: training that tunes inner parameters, validation that tunes hyperparameters, and test that provides the

prediction compared with the observation. This method is often used in machine learning, but how to assign datasets to each process may affect the magnitude of errors. That is, choosing datasets may be a key to create a robust trained model, especially in the training process. The poor training provides insufficient accuracy if some feature quantities among large flood events are learned. Therefore, the pre-trained model with poor training of large flood events in the source watershed must have resulted only in a little improvement in the target watershed when compared with the output from no pre-trained model. Another task is that lead times of the model prediction should be extended to understand a temporal trend of the prediction because the previous study dealt only with the 1h-lead time. Extending lead times is not easy in this model because it is necessary to separately pre-train the model in the 1h-, 3h-, … lead times one by one corresponding to the prediction in each time step.

The purpose of this study is to solve the two tasks related to model accuracy and functional extension for the previous model created by Kimura et al. (2020). The first task is to improve a prediction accuracy of the model with a transfer learning approach in the target watershed by means of the error reduction of the pre-trained model for the past three largest flood events in the source watershed. To solve this task, the leave-one-out cross validation will be introduced for the error evaluation of the pre-trained model. Another task will be solved by adding the output function of the model that is able to predict till the 6h-lead time in the target watershed as well as the source watershed.

2. METHOD

2.1 Study site

This study chose two watersheds that do not have large dams and ponds for flood control (Figure 1). These watersheds were named source watershed and target watershed for an introduction of a transfer learning approach. The source watershed is located in southwest Japan, which is a part of the entire river basin to avoid a large reservoir in the downstream area. The source watershed has an area of 861 km^2, a main river with a length of 52 km up to a key gauge station where a model predicted, and an elevation that ranges from 118–1,350 m. The basin has experienced numerous severe flood events due to typhoon attacks and rainy season. The target watershed is located in north Japan. It has an area of 1,380 km^2, a main river with a length of 115 km that flows to the North Pacific, and an elevation that ranges from 0–978 m. Heavy rainfall events have seldom occurred due to weak impacts of typhoons and rainy season.

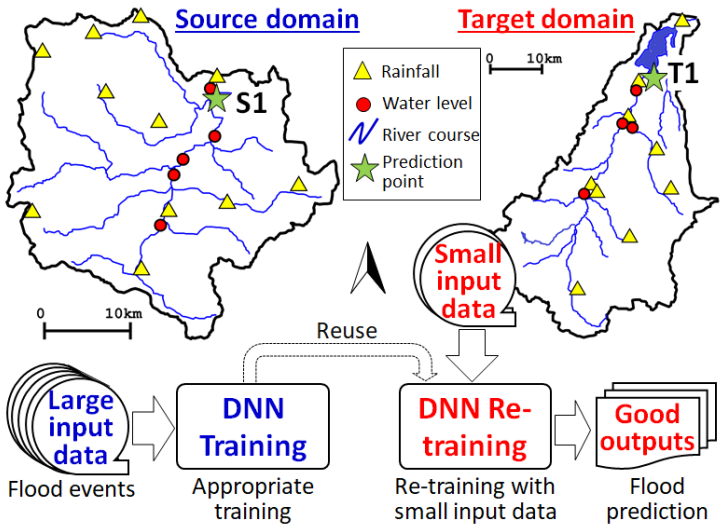


Figure 1. The maps for source and target watersheds with a transfer learning approach. DNN is a deep neural network.

2.2 Field data acquisition

The hourly datasets for rainfalls and water levels during 1992–2017 in the source watershed and during 2000–2019 in Target were obtained from the hydrology and water quality database (www1.river.go.jp) and the meteorological database (www.data.jma.go.jp/gmd/risk/obsdl/index.php) managed by the Ministry of Land, Infrastructure, Transport, and Tourism in Japan (MLIT Japan) and the Japan Meteorological Agency (JMA), respectively. The observed datasets in the source watershed were obtained from five-gauge stations for water level and 11-gauge stations for rainfall. Target had four water-level stations and nine rainfall stations. The downstream stations S1 and T1 were selected as the locations where the model predicts water levels (Figure 1). This study collected past flood events at each watershed. Each flood event, including the flood maximum peak, has a 123h duration, which consists of three days before the peak and two days after the peak. One flood event

was identified when the water level exceeded a certain criterion related to risk levels (approximately 45–60% of the highest peak of the observed datasets among all flood events). If multiple peaks exist locally in one flood event, only the highest peak is regarded as the maximum value of the event. The numbers of datasets of the source and target watersheds are 43 and 17, respectively (Tables 1 and 2). Detailed information of these datasets is shown in the previous study (Kimura et al., 2020).

Table1. Flood events in the source watershed

Events in the source watershed[1]	**Maximum water level (m) during each event[2]**	**Remarks**
8/5/1992	4.56	
7/29/1993	9.50	3rd highest peak
8/7/1993	8.04	
8/13/1993	6.99	
6/10/1994	4.80	
6/1/1995	4.66	
6/22/1995	6.90	
7/15/1996	6.84	
8/11/1996	5.18	
8/4/1997	5.05	
7/24/1999	6.72	
8/14/1999	6.03	Moderate peak
9/11/1999	8.26	
9/21/1999	5.42	
5/31/2000	6.47	
6/18/2001	5.30	
8/5/2003	6.79	
8/27/2004	9.80	2nd highest peak
10/17/2004	7.70	
9/3/2005	10.65	The highest peak
6/21/2006	4.99	
7/19/2006	5.28	
8/15/2006	5.20	
7/8/2007	5.58	
7/11/2007	7.11	
9/28/2008	5.18	
6/17/2010	6.30	
6/30/2010	9.16	
6/13/2011	4.84	
6/18/2011	4.77	
6/19/2012	5.13	
6/25/2012	4.60	
7/10/2012	5.86	
6/25/2014	5.31	
7/28/2014	6.61	
8/6/2014	6.08	
7/19/2015	5.05	
8/22/2015	4.62	
6/25/2016	6.09	
7/6/2016	5.52	
7/11/2016	6.21	
9/17/2016	7.91	
8/4/2017	5.13	

1 Date and time is denoted as month/day/year for a flood event; 2 Maximum value at each event was observed at the predicted points (Figure 1).

Table2. Flood events in the target watershed

Events in the source watershed[1]	**Maximum water level (m) during each event[2]**	**Remarks**
4/9/2000	3.81	
9/9/2001	4.84	2nd highest peak
9/30/2002	3.38	
8/7/2003	4.28	
4/18/2006	3.30	
8/17/2006	4.04	
10/6/2006	4.77	3rd highest peak
10/7/2009	3.33	
9/20/2011	3.20	Moderate peak
4/5/2013	3.71	
9/14/2013	4.33	
10/6/2015	4.15	
8/15/2016	4.17	
8/19/2016	5.03	The highest peak
8/28/2016	3.46	
9/7/2016	4.15	
8/7/2019	3.22	Provisional value

1 and 2 are the same as Table 1.

2.3 A conventional neural network model

A multiple layer (MLP) model, one of the conventional neural network models, consists of three layers: input layer, hidden layer, and output layer, and makes a network structure with nodes (Figure 2). The hidden layer can be extended to multiple layers. Each node has an activation function that filters input variable x with weighted coefficients w into an output variable y. If a layer has I nodes, a certain node j in a subsequent layer receives I input values. These inputs weighted by coefficients are integrated and added by a bias b_j. The activation function $F(*)$ outputs z_j from the node of j. These variables are defined in the following equations.

$$y_j = \sum_{i=1}^{I} w_{i,j} x_i + b_j, \quad (1)$$

$$z_j = F(y_j). \quad (2)$$

A CNN model is a deep neural work model. We created a relatively simple algorithm of a CNN model based on past studies (e.g., Suzuki et al., 2018). The variables in Eqs. (1) and (2) are mapped to a two-dimensional (2D) spatial image in the CNN model, which consists of a convolution layer (convolution) and a max-pooling layer (pooling). We consider that the input variable x has the 2D source image with pixel size ($L \times L$), which is

filtered with a $H \times H$ small window of convolution with a table of weighted values. The convolution takes the same pixel size of the small window from the 2D source image, and then multiplies the weighted values of the $H \times H$ window to the values of the clipped pixels. This filtering way is repeated over the source image by shifting the window. Components of the input and convolution kernel (filter) are defined as $x_{i,j}$ ($1 \le i \le L$, $1 \le j \le L$) and $w_{k,l,n}$ ($1 \le k \le H$, $1 \le l \le H$, $1 \le n \le N$), respectively, where N is the number of various filters. This process is shown in Figure 3a. Note that a zero-padding technique is introduced to ensure that the size of input is equivalent to that of the output. That is, the $x_{i,j}$ was extended with $0 \le i \le L+1$, $0 \le j \le L+1$ so that $x_{0,j}$, $x_{i,0}$, $x_{L+1,j}$, and $x_{i,L+1}$ are set to zero. The $x_{i,j}$ is multiplied by $w_{k,l,n}$ while shifting a grid at (i, j) one by one on the image with the filter. This equation, including the convolution form and bias b_n in the function is defined as follows:

$$y_{i,j,n} = \sum_{l=1}^{H} \sum_{k=1}^{H} w_{k,l,n} x_{i+k-H/2-1, j+l-H/2-1} + b_n, \tag{3}$$

$$z_{i,j,n} = f(y_{i,j,n}), \tag{4}$$

where $H/2$ is an integer value with decimals truncated. We utilize a rectified linear unit (ReLU) as the activation function (f), which can select positive input values due to an improvement in the conversion of a matrix, as expressed in the following equation.

$$f(y_{i,j,n}) = \max(y_{i,j,n}, 0) \tag{5}$$

By filtering in the convolution, common features among images are extracted when the filter pattern is similar to a portion of each image.

The pooling does not have weighted coefficients and activation functions. It is usually utilized to produce 2D data, and is defined for a 2D image ($x_{i,j}$) in the following equation:

$$z_{p,q} = \max(x_{i,j}), (i, j) \in U_{p,q}, \tag{6}$$

where $U_{p,q}$ is the square unit domain with the size $R \times R$ and (p,q) are the horizontal and vertical components. $U_{p,q}$ strides over a 2D image with a non-overlap approach. As horizontal and vertical scales are shrunk 1/R times, the maximum value of $x_{i,j}$ in $U_{p,q}$ provides output with a data size of $1/R^2$ times of the input 2D data (Figure 3b). The features extracted by the convolution can satisfy the consistency by the pooling even if the same features in reality are recognized as different features due to searching errors that consider shifted and rotated features. The CNN model in this study had two convolution layers and two pooling layers. After going through these layers, smaller-size 2D image data are generated and then passed to fully connected layers, which has the same function of Eqs. (1) and (2). In the fully connected layers, these 2D image data are converted into numerous 1D digital datasets that involve information about classification of the original image. The outputs are generally classified with a softmax that converts outputs to probability quantities. Instead, our model employs the sigmoid function that can assign the magnitude from the normalized value (0 to 1) between the maximum and minimum values to predict a specific magnitude of water level.

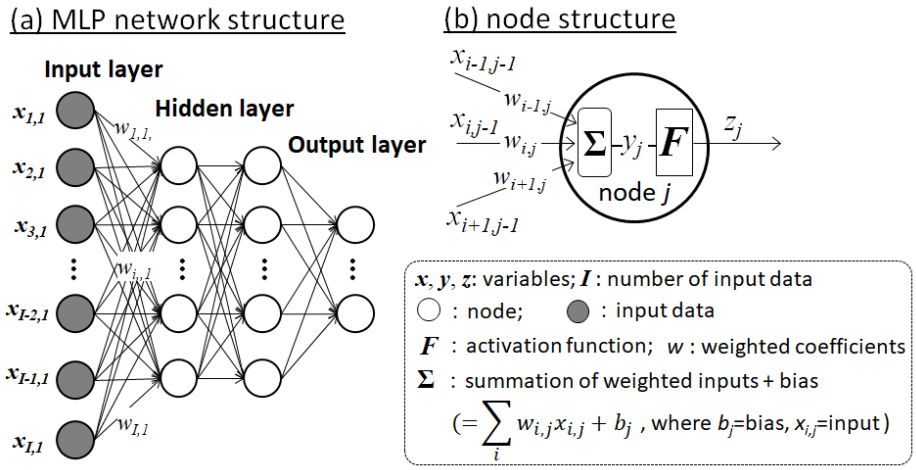


Figure 2. Structures of MLP model that show data flows (a) in network with deep learning and (b) the inner structure of a neuron of jth with i-1th, ith, and i+1th inputs with weighed coefficients.

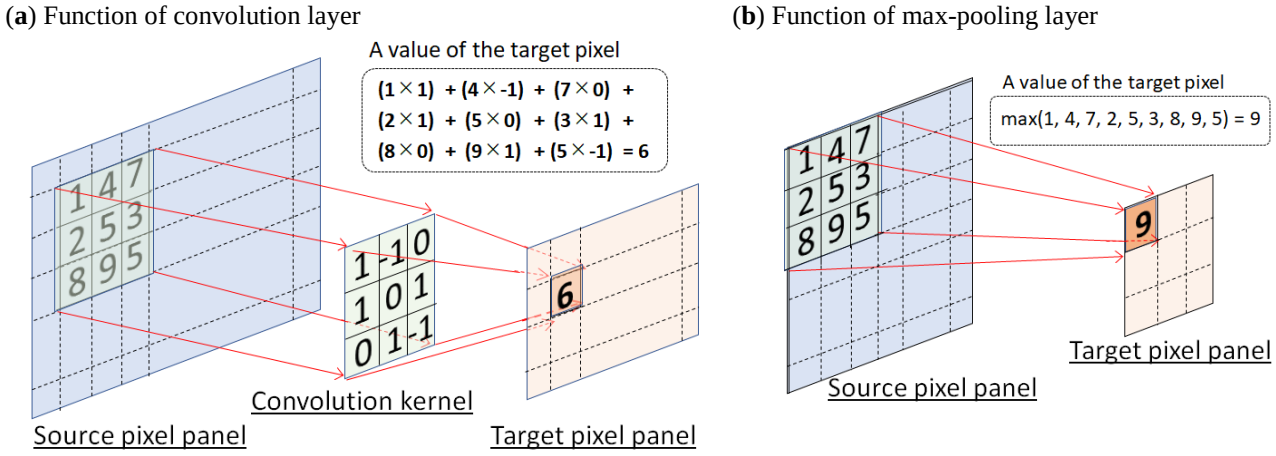


Figure 3 Sketches of the CNN functions: (a) convolution layer and (b) max-pooling layer.

2.4 Transfer learning approach

We assume two different (source and target) watersheds that may have similar features in both sets of observed data. When a model is trained in the source watershed with large datasets, it usually takes ample time to have an accurate prediction. If the model is applied to the target watershed without any connection with the source watershed, training the model also causes ample time. A transfer learning approach (Pratt, 1992) is one of the solution methods for improvement of the efficient prediction (e.g., reduction of run time) in the target watershed through the pre-training in the source watershed. It can reuse the common knowledge obtained from the source watershed for the target watershed. The CNN model coupled with a transfer learning approach (CNN transfer learning) is a well-accepted method in image classification. As our model should perform time-series predictions, CNN transfer learning is utilized with a conversion tool from image data to time-series data. First, the CNN model is run in the source watershed (Figure 4a). Second, the CNN model that fixes parts of the hidden layers is reused in the target watershed. Last, "the fully connected layer 1" and "the fully connected layer 2" in the deep layers of the CNN model are retrained using the datasets of the target watershed (Figure 4b).

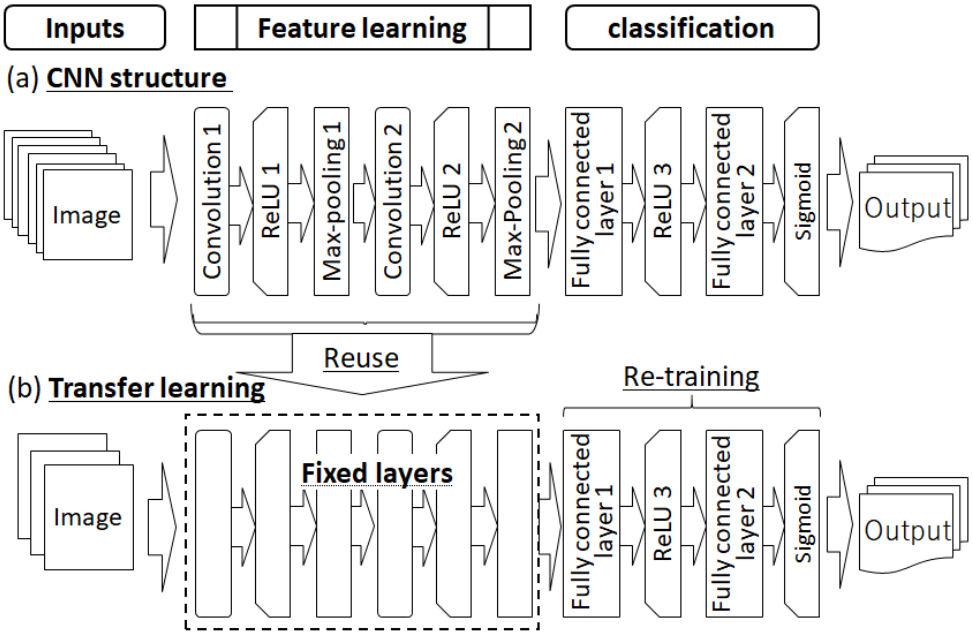
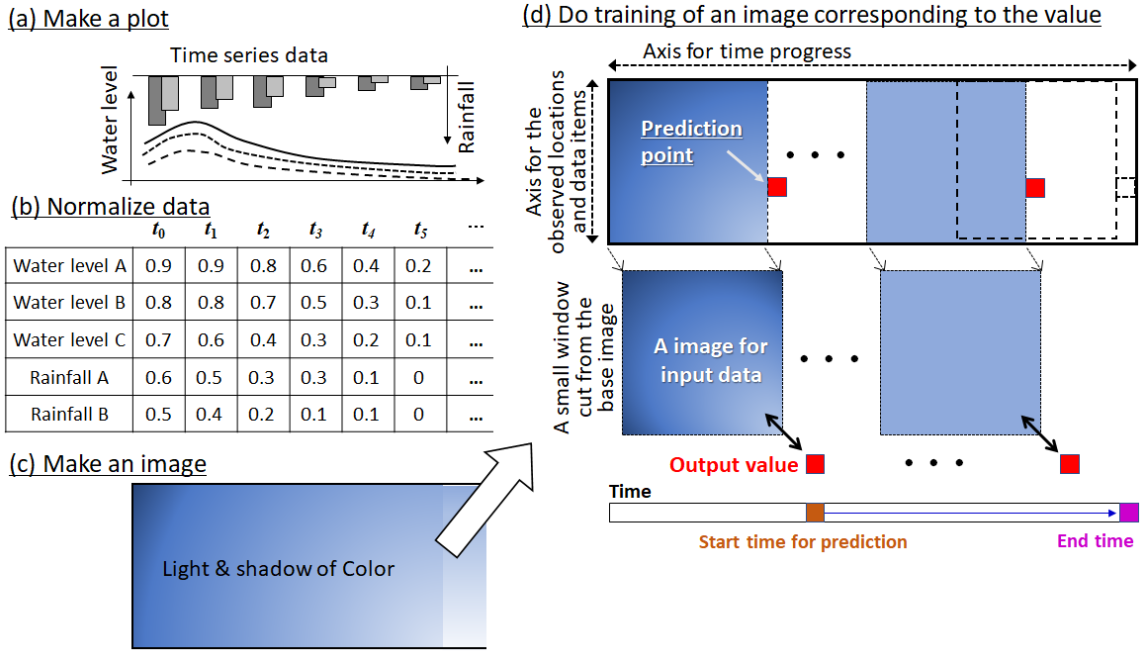


Figure 4. Structures of (a) CNN model and (b) CNN with transfer learning.

2.5 Conversion from image data to time-series data

The conversion method from image data to time-series data is necessary to perform time-series predictions with the CNN model. We employ a simple method to arrange plural variables (rainfall and water level) at all gauge stations from upstream to downstream on the vertical (totally 16 sets), and to arrange the temporal variation of each variable on the horizontal (123 time steps) in Figures 5a and b. The digital values are converted to an image using the light and shade of color that ranges from 0–1 (Figure 5c). The observed data are normalized by the maximum and minimum values of all observed datasets of water level and rainfall. The spacing between two values on the horizontal axis is one hour. The interval of the vertical axis is one for simplicity.

The 2D image in the CNN model has to correspond to the time-series data of water level when the model is trained and predicted (Figure 5d). A 16 × 16-pixel image (a small window) is selected from the 2D image of a flood event. The number of pixels of the small window corresponds to the total sets of observed locations and variables of observed data. The information of the small window, including spatial characteristics of a watershed, is used as input data. We assume that the information from the input data must be associated with a 1 × 1 image (i.e., value) as output at the predicted point and that the value is assigned at the next step (i.e., 1h-lead time), at the next three steps (3h-lead time), or at the next six steps (6h-lead time). Note that the predicted point indicates a location where water level is predicted. This treatment was separately repeated in time progress as shifted from the head of the 2D image to the tail (Figure 5d). The CNN model tries to find spatial patterns in each small window. The number of the input set (i.e., four for water-level gauges and nine for rain gauges) in the target watershed differs from that of the source watershed. To force the image size of the target watershed to the same image size of the source watershed, the image size is expanded from 123 × 13 to 123 × 16 using the interpolation to maintain high-resolution images with the Lanczos algorithm (Lanczos, 1950).



	t_0	t_1	t_2	t_3	t_4	t_5	...
Water level A	0.9	0.9	0.8	0.6	0.4	0.2	...
Water level B	0.8	0.8	0.7	0.5	0.3	0.1	...
Water level C	0.7	0.6	0.4	0.3	0.2	0.1	...
Rainfall A	0.6	0.5	0.3	0.3	0.1	0	...
Rainfall B	0.5	0.4	0.2	0.1	0.1	0	...

Figure 5. Procedures for a data creation and a training process with simplified data and conception. (a) plots for temporal variables, (b) data normalization of each variable, (c) creation of an image, and (d) a method of learning an image corresponding to output value in time series.

2.6 Model evaluation and computational setups

Accuracy of the CNN model pre-trained in the source watershed is evaluated using a leave-one-out cross validation by a root mean square error (RMSE) between predicted and observed outputs. This cross validation selects top three largest flood events that have the first-, second-, and third-highest peaks, and one mid-level (moderate) peak as a representative of the mid-level peak events (Table 1). One of the four flood events is used as a test dataset to compare it with the predicted output by the model after the model is trained with the datasets of the other flood events. It is repeated for the other flood events. Then, the RMSEs from the four tests are calculated and are considered as degree of accuracy for the pre-trained model. On the other hand, in the target watershed, training, validation, and testing processes (James et al., 2013) are employed as a typical evaluation of CNN model accuracy. The top three largest flood events and a moderate-peak flood event (one testing dataset) are selected for the testing process. The four cases are set up as the testing process for the evaluation of model accuracy. The testing process is conducted using one from the selected four events one by one. For validation process, another moderate-peak flood event is added to these four flood events as a reference event. These data are defined as four validation datasets. The other flood events (12 training dataset) are utilized for the training process. The trained datasets exclude the top three largest flood events to achieve accurate prediction even without larger flood events. The flood events selected in each process are listed in Table 3. The procedures of the CNN transfer leaning (Figure 1) are as follow: first, the CNN model is trained and evaluated with three tests in the source watershed with 47 datasets, then creating the pre-trained model, trained with all the datasets; secondly, the pre-trained model is retrained (or fine-tuned) with training datasets in the target watershed, then being checked to see whether or not a loss function is overfitted among validation datasets; and finally the testing process is performed using one of the top three largest flood datasets or the moderate-peak flood dataset.

The programs for the models (CNN and CNN transfer learning) were created by Python (version 3.6.4, www.python.org) incorporated with Keras libraries (keras.io/ja) on the Window-OS PC that has the Intel(R) Core(TM) i7-4770K CPU @ 3.50GHz. The setups of several parameters such as batch size, epoch number and activation functions are shown in Table 4.

Table 3. Datasets for training, validation, and testing in the target watershed.

Name	Training datasets	Verification datasets	Test dataset
Case 1	9/30/2002, 8/7/2003, 4/18/2006, 8/17/2006, 10/7/2009, 4/5/2013, 9/14/2013, 10/6/2015, 8/15/2016, 8/28/2016, 9/7/2016, 8/7/2019	4/9/2000, 9/9/2001, 10/6/2006, 9/20/2011	8/19/2016
Case 2	Same as above	4/9/2000, 10/6/2006, 9/20/2011, 8/19/2016	9/9/2001
Case 3	Same as above	4/9/2000, 9/9/2001, 9/20/2011, 8/19/2016	10/6/2006
Case 4	Same as above	4/9/2000, 9/9/2001, 10/6/2006, 8/19/2016	9/20/2011

Table 4. Setups of CNN parameters and functions.

Parameters/ functions		Values /formula	Remarks
Convolutional layer	Filter size	3 × 3	
	Filter number	5	
Pooling layer	Filter size	2 × 2	
Fully connected layer 1	Neuron number	16	
Fully connected layer 2	Neuron number	1	
Learning process	Batch size	100	
	Epoch number	100	
	Learning rate	0.001	
	Optimizer	Adam	
Activation function 1 to 3		ReLU	See Figure 4a
Activation function 4		Sigmoid	See Figure 4a
Loss function		Mean square error = $\frac{1}{N1}\sum_{i=1}^{N1}(V_{ci}-V_{oi})^2$	c_i=model, o_i= observation, $N1$=the number of data
Error evaluation		Root mean square error (RMSE) = $\sqrt{\frac{1}{N1}\sum_{i=1}^{N1}(V_{ci}-V_{oi})^2}$	Same as above
		Nash–Sutcliffe coefficient (NS) = $1-\sum_{i=1}^{N1}(V_{ci}-V_{oi})^2/\sum_{i=1}^{N1}(V_{oi}-\langle V_{oi}\rangle)^2$	Same as above, and <*> = average

3. RESULTS & DISCUSSION

We first performed the verification of the CNN model as a pre-trained model to predict water levels in the top three largest flood events and a moderate flood event in the source watershed using the leave-one-out cross validation. Figure 6 shows the predicted water levels compared with the observation in the 1h-, 3h- and 6h-lead times in the four flood events. For the error evaluation, the NSEs ranged from 0.98–0.99 in the 1h-lead time, 0.95–0.99 in the 3h-lead time, and 0.76–0.96 in the 6h-lead time, respectively. The predicted water level for the third-largest flood event in the 6h-lead time was the poorest. The RMSEs of the 1h- and 3h-lead time in the largest, and the second largest flood event were similar with 0.35 m for the 1h-lead time and 0.20 m for the 3h-lead time. On the other hand, the RMSEs of the other events in the 3h-lead time were greatly worse than that of the 1h-lead time (i.e., 0.45 vs. 0.28 m for the third-largest event, and 0.23 vs. 0.10 for the moderate event). The worse result for the third-largest flood event may be caused by a more complicated wave shape including clear two peaks, which is hard to be captured due to a few training patterns that occurred in past datasets. Another worse result was caused by time-lags when the wave started up. Past study by Kimura et al (2020) reported that the RMSEs ranged from 0.14–0.73 m in the 1h-lead time in the same flood events for the source watershed. The different error validation in this study showed the error reduction by 25–60% in the 1h-lead time. Furthermore, the conventional MPL model with deep learning developed by Hitokoto at al. (2016) reported that mean RMSEs for the four large flood events in water level prediction were 0.12 m in the 1h-lead time and 0.27 m in the 3h-lead time. Although their datasets were quite different from ours due to the data selection, a simple comparison of both RMSEs is shown here. Their mean RMSE was greatly better than the RMSEs (0.19–0.34 m) for the top three largest events from our model in the 1h-lead time; however, their mean RMSE for the 3h-lead time was partly similar to the RMSEs (0.22–0.45 m) by our model. The predicted water level in the 6h-lead time had the worst reproductions with several unstable oscillations when the wave shapes were peaks. The RMSEs (0.64–1.09 m) in the 6h-lead time from our study were dramatically increased by twice to three times from those in

the 3h-lead time for the four flood events. As a result, the model may not be utilized for a longer time-series prediction like the result that the model predicted in the 6h-lead time.

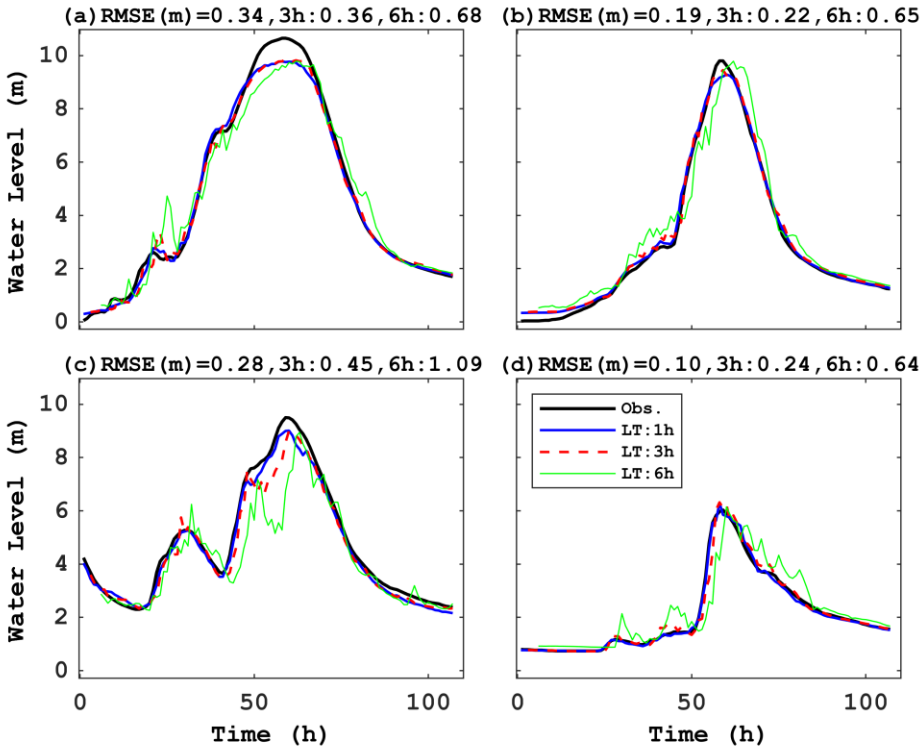


Figure 6. Comparisons between observed and predicted water levels in the 1h-, 3h-, and 6h-lead times in the source watershed simulated by the pre-trained model in (a) the largest flood event, (b) the second-largest flood event, (c) the third-largest flood event, and (d) a moderate-peak flood event. Obs. = observation and LT = lead time.

Using the pre-trained model, the model predictions for the four cases (Table 3) were performed in the target watershed as a transfer learning approach. Two fully connected layers of the pre-trained model (Figure 4b) were retrained separately with 0, 10, 20, and 40 times in the epoch number in 1h- and 3h-lead times. In the error evaluation, the RMSEs in four epoch numbers of the retraining were compared with the RMSE of no pre-train model (i.e., no transfer learning) with a default epoch number (=100) (Figure 7). The RMSEs of the top three largest flood events were greatly reduced from the RMSE of no pre-trained model in the 1h-lead time by 10–40 of the epoch number. The retrained model for the moderate-peak flood event had a similar RMSE to no pre-trained model from 10–40 of the epoch number (Figure 7a). For the 3h-lead time, the RMSEs of the second and third largest flood events (Cases 2 and 3) up to 40-time retraining were significantly reduced from the RMSE of no pre-trained model (Figure 7b). The RMSEs of the highest- and moderate-peak flood events (Cases 1 and 4) were unable to reduce but were similar to those of no pre-trained model when the retraining was done by 40 times (Figure 7b). For the 6h-lead time, the RMSE of the third largest flood event was slightly higher than that of no pre-trained model even for 60-time retraining, but the others were improved by up to 40-time retraining. In addition, no retraining (zero time) of the pretrained model provided the worst error in any cases.

Figure 8 shows the flood waves with 40-time retraining in 1h-, 3h-, and 6h-lead times for the top three largest and moderate-peak events to compare with the observed data. The NSEs ranged from 0.98–0.99 in the 1h-lead time, from 0.96–0.99 in the 3h-lead time, and from 0.93–0.96 in the 6h-lead time, respectively, for the model accuracy. The RMSEs of the 3h-lead time for the top three largest and moderate-peak flood events were worse by 31–73% than those of the 1h-lead time. The RMSE of the highest-peak flood event (Case 1) in the 3h-lead time was larger than that of no pre-trained model. This implies that a complicated wave shape, such as the shape of the highest-peak flood event, might be hard to capture accurately in the model due to a few patterns in past datasets. As a result, the model accuracy in transfer learning was greatly improved in the 1h-lead time from the previous study due to the accurate pre-trained model. According to the quantitative evaluation shown in Table 4, the RMSEs in the 3h-lead time for 40-time retraining were improved by 20–36% reduction from those of no pre-trained model in Cases 2 and 3, but the RMSEs in Cases 1 and 4 were slightly worse by 16–20% than those of no pre-trained model. For the 1h- and 3h-lead times, the computational cost can be reduced till 2/5 of that of no pre-trained model. All predicted flood waves in the 6h lead time were the worst in accuracy (Figure 8). Predicted water levels were unstably under- and over-estimated from the observed water levels possibly due to the features such as oscillations of wave shapes (i.e., zigzag shapes) from the pre-trained model (Figure 6). Note

that for the validation process of four cases, the RMSEs were 0.072–0.082 m in the 1h-lead time, 0.110–0.124 m in the 3h-lead time, and 0.158–0.181 m in the 6h-lead time, respectively, which were approximately within the ranges of the RMSEs predicted by the CNN transfer learning with 40-time retraining (Table 4).

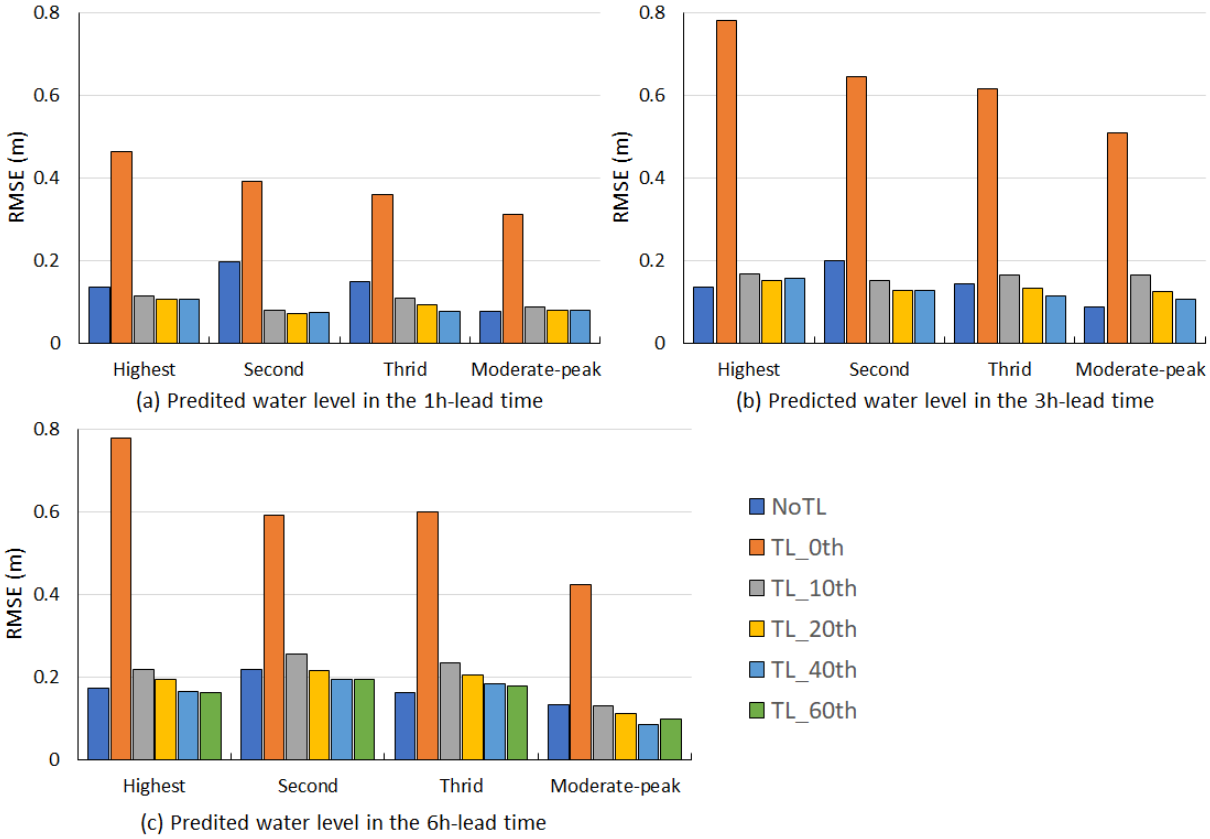


Figure 7. Comparisons of the RMSEs without transfer learning (NoTL) and with retraining done by 0 (TL_0th), 10 (TL_10th), 20 (TL_20th), and 40 (TL_40th) times in transfer learning, in (a) 1h-, (b) 3h-, and (c) 6h-lead times for the highest, second-largest, third-largest, and moderate peaks. 60-time retraining (TL_60th) was done only in the 6h-lead time. Note that TL = transfer learning.

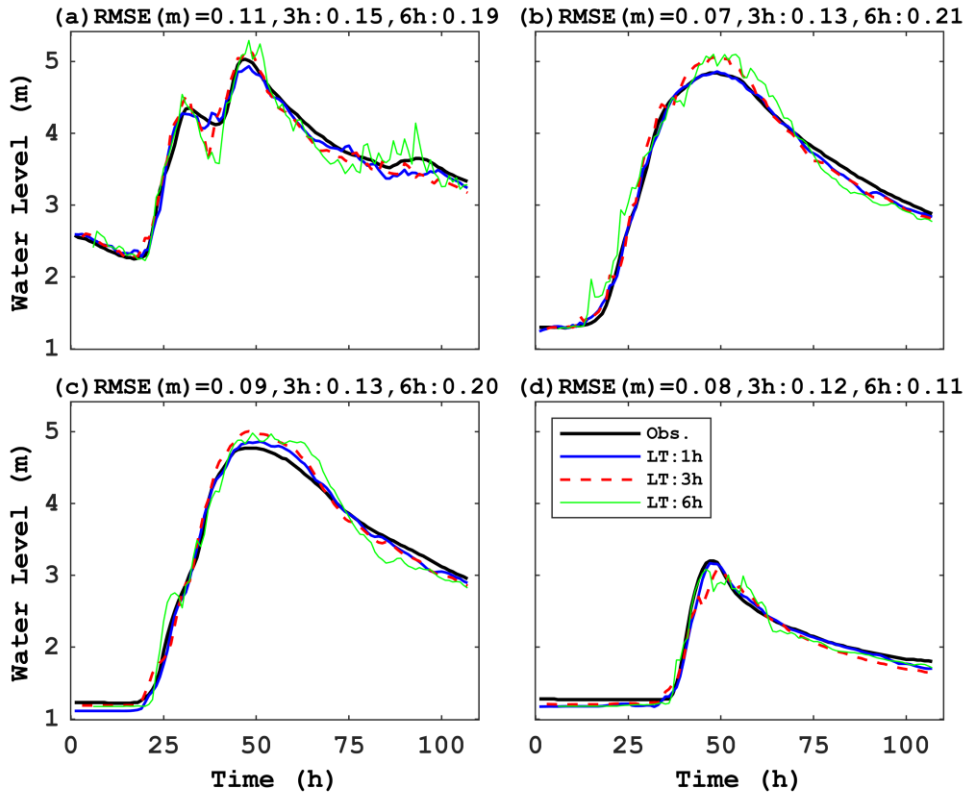


Figure 8. Comparison between observed and predicted water levels. The predictions are shown in the 1h, 3h- and 6h-lead times by the retrained model with 40 times in the target watershed. The explanations of (a) to (d) are the same as figure 6.

Table 4. Error evaluations of the CNN transfer leaning with 40-time retraining and no pretrained model.

Case	RMSE (m) in the 1h LT[1]	RMSE (m) in the 3h LT[1]	RMSE (m) in the 6h LT[1]	RMSE (m) of CNN without TL[2] in the 1h LT	RMSE (m) of CNN without TL[2] in the 3h LT	RMSE (m) of CNN without TL[2] in the 6h LT
Case 1	0.1057	0.1564	0.1633	0.1347	0.1352	0.1731
Case 2	0.0732	0.1267	0.1945	0.1965	0.1992	0.2188
Case 3	0.0763	0.1146	0.1819	0.1480	0.1427	0.1622
Case 4	0.0800	0.1044	0.0842	0.0751	0.0868	0.1334

1 LT = lead time; 2 TL= transfer learning.

4. CONCLUSIONS

This study improved and extended the DNN-based flood model with a transfer learning approach, developed in the previous study (Kimura et al., 2020), for the model accuracy and longer lead-time prediction. The model accuracy was greatly improved by a change of the pre-training datasets with a leave-one-out cross validation. The lead time of the model prediction was extended to 6 hours. As a result, the CNN model with the transfer learning approach are still beneficial up to the 3h-lead time for accuracy improvement as well as computational-time reduction when compared with the CNN model without transfer learning. For the model prediction in the 6h-lead time was poor in accuracy, which indicates that a longer time-series prediction could not be utilized for a flood forecast.

ACKNOWLEDGMENTS

This research received a funding of the director discretionary supports in the NARO. We greatly appreciate MLIT Japan and JMA for the observed data acquisition.

REFERENCES

Bengio, Y., Courville, A. and Vincent, P. (2013). Representation learning: a review and new perspectives. *Institute of Electrical and Electronics Engineers* (*IEEE) Transactions on Pattern Analysis and Machine Intelligence*. 35 (8):1798–1828. doi:10.1109/TPAMI.2013.50.

Hitokoto, M., Sakuraba, M. and Sei, Y. (2016). Development of the real-time river stage prediction method using deep learning. *Journal of Japan Society of Civil Engineers* (*JSCE*), *Series B1* (*hydraulic engineering*), 72(4):I_187–I_192. (In Japanese) doi:10.2208/journalofjsce.5.1_422. doi:10.2208/journalofjsce.5.1_422

Hochreiter, P. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780. doi:10.1162/neco.1997.9.8.1735.

Hsu, M.-H., Lin, S.-H., Fu, J.-C., Chung, S.-F. and Chen, A.-S. (2010). Longitudinal stage profiles forecasting in rivers for flash floods. *Journal of Hydrology*, 388(3-4):426–437. doi:10.1016/j.jhydrol.2010.05.028.

Hu, C., Wu, Q., Li, H., Jian, S., Li, N. and Lou, Z. (2018). Deep learning with a long short-term memory networks approach for rainfall-runoff simulation. *Water*, 10:1543. doi:10.3390/w10111543.

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). An Introduction to statistical learning: with applications in R, 1st ed. Springer, pp. 176. ISBN-13: 978-1461471370.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of mathematical biophysics*, 5(4):115–133. doi:10.1007/BF02478259.

Kimura, N., Yoshinaga, I., Sekijima, K., Azechi, I. and Baba, D. (2020). Convolutional neural network coupled with a transfer-learning approach for time-series flood predictions. *Water*, 12(1):96. doi:10.3390/w12010096.

Lanczos, C. (1950). An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45(4):255–282. doi:10.6028/jres.045.026.

LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998) Gradient-based learning applied to document recognition. Proc. *the IEEE*, 86(11), 2278–2324, November 1998. doi:10.1109/5.726791.

Pratt, L.Y. (1992). Discriminability-based transfer between neural networks. Proc. the conference of the Advances in NIPS 5, Denver, Colorado, USA, Nov.30-Dec.3, 1992, 154–196, Hanson, S. J., Giles, C. L., Cowan, J. D. Eds.; Morgan Kaufmann Publishers Inc., San Francisco, USA.

Suzuki, T., Kim, S., Tachikawa, Y., Ichikawa, Y. and Yorozu, K. (2018). Application of convolutional neural network to occurrence prediction of heavy rainfall. *Journal of JSCE Series B1* (*hydraulic. engineering*), 74(5), I_295–I_300. (In Japanese) doi:10.2208/jscejhe.74.5_I_295.

Yamada, K., Kobayashi, Y., Nakatsugawa, M. and Kishigami, J. (2018). A case study of flood water level prediction in the Tokoro River in 2016 using recurrent neural networks. *Journal of JSCE Series B1* (*hydraulic. engineering*), 74(5):I_1369–I_1374. (In Japanese) doi:10.2208/jscejhe.74.5_I_1369.