京都大学防災研究所年報 第 65 号 B
DPRI Annuals, No. 65 B, 2022

Capturing Atmospheric Signatures with Convolutional Neural Networks to Predict Occurrences of Rainfall Events

Sunmin KIM(1), Tsuguaki SUZUKI(2) and Yasuto TACHIKAWA(1)

(1) Dept. of Civil & Earth Resources Eng., Graduate School of Eng., Kyoto University, Japan
(2) Fukushima Daiichi Decommissioning Company, Tokyo Electric Power Company Holdings, Inc., Japan

Synopsis

Rainfall occurrence prediction models were developed based on a convolutional neural network algorithm, which is one of the most representative machine learning algorithms for image recognition. Spatiotemporal map of atmospheric movement was prepared as image data based on ground gauged data and satellite image data. The spatiotemporal map contained information regarding the last 30 min of atmospheric movement to predict rainfall occurrence at the target area with a lead time of 30 min. The proposed model was developed for on-off (rain or no rain) forecasting at the target area to maximize the image classification functions of the CNN algorithm. Various forms of input combinations and hyperparameters were tested to evaluate the performance and applicability of the model. The evaluation index from the best model showed promising results with a detection probability of 0.836 and a critical success index of 0.456. This paper illustrates the concept of the developed model and summarizes the results from various model structures and input data combinations.

Keywords: Convolutional Neural Network, Rainfall Prediction, Atmospheric variables

1. Introduction

Rainfall prediction is essential to hydrological forecasting, and extensive research efforts have been made to achieve high accuracy in rainfall prediction. These efforts are conventionally based on numerical weather prediction (e.g., Golding, 2000; Cloke and Pappenberger, 2009; Shrestha et al., 2013), radar image extrapolation (e.g., Bowler et al., 2004; Kim et al., 2009; Mandapaka et al., 2012, Thorndahl et al., 2017), or a blend of these two schemes (e.g., Ganguly and Bras, 2003; Cuo et al., 2011; Yu et al., 2015; Wu and Lin, 2017). However, precise quantitative precipitation forecasting (QPF) remains one of the most challenging tasks.

Recent progress in machine learning algorithms has attracted the attention of many researchers and provides another approach to hydrological research, including rainfall prediction. For example, monthly rainfall prediction can be achieved by training long-term historic rainfall data and large-scale climate indices within a fully connected feed-forward neural network (e.g., He et al., 2015; Alizadeh et al., 2017) or long short-term memory (LSTM) algorisms (e.g., Ni et al., 2020; Tao et al., 2021).

Owing to the increased computing power, accumulated observation data, and improved training techniques, recent machine learning algorithms are being actively tested for storm cell detection (Kim et al., 2019), downscaling of satellite images (Pan et al., 2019), and even mimicking numerical weather predictions (Scher, 2018; Weyn et al., 2019; Matsuoka et al., 2020).

In addition to the conventional form of fully

connected artificial neural network (ANNs), there are two types of representative machine learning algorithms: convolutional neural networks (CNNs) and recurrent neural networks (RNNs). The RNN was originally developed for natural language analysis, and the concept has evolved into the LSTM algorithm to be specialized in time series analysis and modeling sequential data (Pascanu et al., 2014). The time series of rainfall or streamflow data can be effectively reconstructed using the trained LSTM algorithm for short-term forecasting (e.g., Ni et al., 2020; Tao et al., 2021; Nguyen et al., 2020; Cheng et al., 2020). The CNN was originally introduced for image recognition (LeCun et al., 1998), and owing to its specialized function in feature extraction from spatial data, it is now actively utilized in hydrological forecasting, such as flood mapping (Wang et al., 2019; Khosravi et al., 2020; Kabir et al., 2020) and spatial rainfall estimation (Wu et al., 2020; Tu et al., 2021). Owing to the flexibility of these deep learning algorithms and improved computing power, it is now possible to test and propose many new types of modeling approaches in hydrological forecasting (Shen et al., 2018; Nearing et al., 2021).

The CNN algorithm is specialized in the function of image recognition by extracting features with a small set of parameters, known as a filter, by scanning the image several times. The algorithm can be applied to any form of data that can be expressed in two- or three-dimensional images (e.g., color images with RGB channels). Spatiotemporal data of atmospheric movement can also be expressed in two- or three-dimensional data arrays, and CNN can be applied to capture specific atmospheric patterns, for example, atmospheric patterns before rainfall occurs. (Suzuki et al., 2018; Park et al., 2019; Kim et al., 2020).

Suzuki et al. (2018) tested three-dimensional input data composed of ground gauged observations (precipitation, temperature, wind speed, etc.) and fed into a CNN algorithm to classify whether the atmospheric movement resulted in rain at the predicted target area. In the proposed CNN algorithm, the three-dimensional input data were prepared by arranging the time series (vertical) of multiple atmospheric variables (depth) at multiple observation points (horizontal). The CNN algorithm was extensively tested with various model structures and training techniques in three different regions of Japan to evaluate its applicability (Kim et al., 2020). The best model performances in terms of the probability of detection (POD) were 0.665, 0.780, and 0.550 in Kyoto, Osaka, and Tokyo, respectively, for a 30-min prediction lead time. However, regarding overall performance, the upper limits of the critical success index (CSI) were 0.271, 0.318, and 0.226 respectively, in the three cities, because of the high false alarm ratio (FAR). It was found that the model tends to falsely forecast rainfall right after the rainfall events, showing difficulty in identifying the atmospheric pattern between ‘during the rain’ and ‘after the rain’ phases (Kim et al., 2020).

There was a limitation in the first version of the CNN rainfall detection model proposed by Suzuki et al., 2018, because the input was only from the ground gauged data and it was not possible to consider the vertical movement of the atmosphere (Kim et al., 2020). To improve the prediction accuracy of the algorithm, it is necessary to integrate additional atmospheric information. In this paper, we introduce an improved version of the rainfall occurrence prediction model with the help of additional atmospheric information from satellite images.

We have attempted to incorporate satellite images observed from Himawari-8 to predict rainfall occurrence in the Kyoto region of Japan. Himawari-8 is a geostationary meteorological satellite operated by the Japan Meteorological Agency (JMA) since 2015, which provides 16 observation bands (3 for visible, 3 for near-infrared, and 10 for infrared). These observation bands provide information on vertical cloud conditions with a 1 km resolution every 2.5 min around Japan, and all the observed data are available to the public through the Data Integration and Analysis System (DIAS) of Japan.

In this study, to improve the rainfall prediction accuracy using the CNN algorithm, we first tested four types of input data sets: two sets were based on ground gauged data, and the other two sets were based on satellite image data (plus ground gauged precipitation data), which are in two different types of time series settings. After sensitivity analysis with the four input data sets, the best input data combination was selected to build the best model.

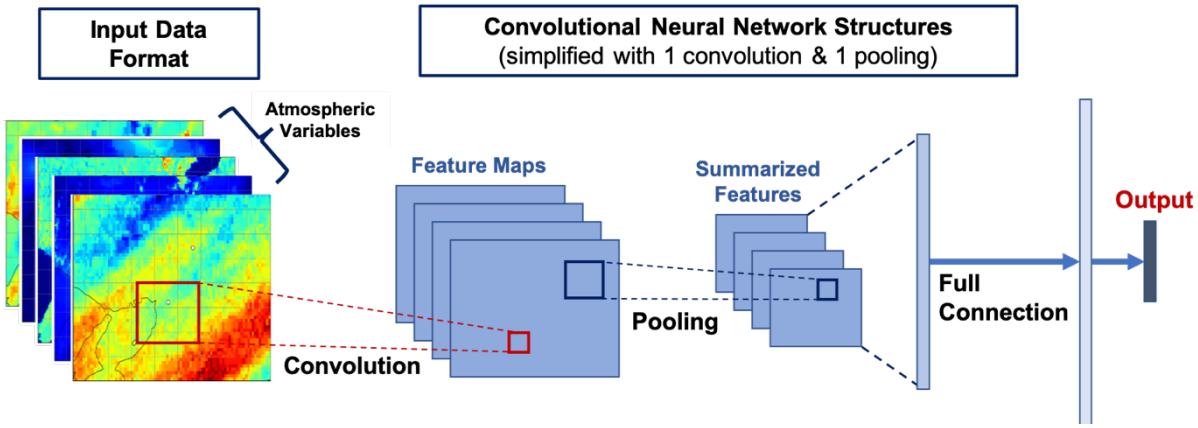

Figure 1. Schematic drawing of the CNN structure with a convolution layer for filtering, pooling layer for sampling, and fully connected layer to finally classify the given image input.

We have tested four types of CNN algorithms: two algorithms based on the conventional structure that has convolution layers and pooling layers for two or three layers, and the other two algorithms with a simplified structure that has only convolution layers for two or three layers. After the sensitivity analysis for these four model structures that considered various training conditions, the best model structure was selected to build the ideal model. In summary, four types of input data sets and four types of CNN model structures were tested (16 combinations) to select the best model structure and input combination. Furthermore, some hyper-parameters were also tested (e.g., batch sizes and learning rates) for every 16 model combinations, as well as the finalized best model.

This paper summarizes the sensitivity analysis results with various combinations of input data and model structures to build up the rainfall occurrence model with the CNN algorithm in the Kyoto region of Japan. The remainder of this paper proceeds as follows. Section 2 provides a brief overview of the CNN algorithm (2.1), the concept of the developed model (2.2), the details of the model structures and input data (2.3), and the details of experimental design. Section 3 provides sensitivity analysis results for the model structures (3.1) and for each variable in the input data (3.2) to build up the best model and to check the model performance (3.3). Finally, section 4 summarizes the results with concluding remarks.

2. Data and Methodology

2.1 Convolutional Neural Network

The CNN algorithm was originally introduced for document recognition and has evolved to its present form to specialize in image recognition. The algorithm has three data processing layers, as shown in Fig. 1: the convolution layer for extracting features, the pooling layer for summarizing the features, and the fully connected layer to classify the features for the final output.

First, the convolutional layer extracts the image features by applying a set of weight factors, $w_{p,q,k}$, known as a filter, for a certain input area, $x_{i,j,k}$, where $(p,q,k) \in [0,H-1] \times [0,H-1] \times [1,K]$ and $(i,j,k) \in [1,L] \times [1,L] \times [1,K]$. Input data have a two- or three-dimensional array format, such as an image with $L \times L$ pixels and K channels (e.g., RGB channel), and the filter also has a two- or three-dimensional array format ($H \times H \times$ K).

$$y_{i,j,k'} = F\left(\sum_{k=1}^{K}\sum_{p=0}^{H-1}\sum_{q=0}^{H-1} w_{p,q,k}\, x_{i+p,j+q,k} + b_{k'}\right) \quad (1)$$

The extracted feature, $y_{i,j,k'}$, was estimated using Eq. (1). The convolutional process of the function $F(\cdot)$ is applied to the entire input range by shifting (i,j) by a definite stride (in general, the stride is one pixel). The result of the convolutional process is an $(L-H+1) \times (L-H+1) \times M$ feature map, where M is the number of convolutional processes. The size can be controlled to $L \times L \times M$ by adding zero values surrounding the input data, which is called

zero padding. The activation function $F(\cdot)$ converts the filtered results, and the most common activation function in the CNN is the rectified linear unit (ReLU).

Second, the pooling layer summarizes and reduces the size of the extracted features by taking the maximum values within a given window (max pooling). As shown in Eq. (2), max pooling takes only one maximum value within the given area, $U_{s,t}$, where *s* and *t* are the vertical and horizontal sizes of the given area, respectively. If the given area of the input data is 2×2 for $L \times L \times M$, then the pooling layer provides an $(L/2) \times (L/2) \times M$ output. By conducting the max pooling process, only significant information is delivered to the next process, and unnecessary features are eliminated, improving the model performance, and reducing computational resources.

$$y_{i,j,k} = \max_{(i,j) \in U_{s,t}} x_{i,j,k} \quad (2)$$

Third, a fully connected layer rearranges the three-dimensional feature map into a one-dimensional array and connects it to the output layer for classification. The SoftMax function is often utilized as an activation function to emphasize the classification task, and a cross-entropy function is the most commonly used error function in the output layer. More details of the CNN algorithm can be found in Rawat and Wang (2017).

The main advantage of using the CNN algorithm compared to the conventional fully connected neural network is the processing efficiency with a small number of parameters. The CNN algorithm can efficiently extract specific features from the input image by focusing on a limited region of the image using a filter. There are several hyperparameters for maximizing the learning process in the CNN algorithm, such as the number of layers, filter size, number of epochs, and batch size. In this study, some hyperparameters were fixed (e.g., filter size was set as 3×3, and pooling window was set as 2×2), while some hyperparameters were examined (e.g., batch size and learning rate) to maximize the model performance. The details of the model settings are given in Section 2.3.

2.2 Modeling Concept for Rainfall Occurrence Prediction

Rainfall is generated when precipitable water in the air is raised up and is condensed under various conditions. Air can be raised up due to frontal effects, orographic effects, or massive atmospheric movements caused by typhoons. However, these sophisticated atmospheric movements and rainfall generation are difficult to simulate correctly even with a very fine numerical weather prediction model because of the difficulty in the initial setting of the model and the chaotic movement of the atmosphere.

Another type of forecasting approach can be used with a machine-learning algorithm by utilizing various types of observations. Even though the observations are not detailed enough to reformulate the atmospheric movement in our conventional numerical simulation model, accumulated observation data from various sources, such as ground gauge and satellite images, is valuable, and the machine learning approach allows us to find a shortcut to link those different types of observations and extract a possible linkage from the observed natural phenomena.

We have developed a rainfall prediction model based on the CNN algorithm to forecast rainfall occurrence at a specific target area. The developed model is only for on-off (rain or no rain) prediction for a target area with a certain lead time (e.g., 30 min or 1 h), similar to the image classification process of the CNN algorithm. The CNN algorithm has been developed for efficient image recognition, and once it is trained with a sufficient number of data sets, the algorithm can classify images by extracting the necessary features. The application of this algorithm is not restricted to image recognition. If we can prepare the necessary spatiotemporal data of atmospheric movement, the CNN can be trained to capture and extract the internal features of atmospheric movements before it rains.

The first version of our prediction model was based on ground-gauged data only (Suzuki et al., 2018; Kim et.al., 2020). The spatiotemporal data of atmospheric movement were prepared by attaching the time series of an atmospheric variable from multiple observation points side by side. By overlapping this spatiotemporal data with multiple atmospheric variables, we produced a three-

dimensional data array containing atmospheric movement around the target area. Based on extensive experiments, we confirmed that the CNN algorithm with this data array can be successfully trained to classify the spatiotemporal movements of the atmosphere to determine whether there will be rain in the next 30 min.

However, even though the detection ratio was good at 0.665 in the case of Kyoto, the CSI was limited to 0.271 because of the high FAR (Kim et al., 2020). Many false alarms were noticed immediately after the rain stopped, and the reason for this high FAR could be traced to two factors. One of the factors is of the gradually changing atmospheric conditions. Even after the rain stops at the target area, the surrounding atmospheric conditions still have a pattern similar to the conditions during rainfall; thus, the CNN algorithm tends to falsely forecast rain immediately after the event, resulting in a high FAR.

Another reason is the limited information available in our previous model. The first version of our model only utilized ground-gauged atmospheric variables (precipitation, temperature, wind speed, wind direction vectors, and sunshine ratio), and it was not enough to integrate the information of air mass that is moving vertically. Thus, we have been looking for additional information sources to integrate three-dimensional air mass movements so that the CNN algorithm can classify the slight differences in atmospheric conditions during, and after the event.

In our second version of the prediction model introduced in this paper, we utilize satellite images to incorporate additional atmospheric information observed from the upper layers of air mass. In this study, we utilized the fine resolution of satellite images observed from Himawari-8, which is a geostationary meteorological satellite operated by the Japan Meteorological Agency (JMA). Himawari-8 provides 16 observation bands (3 for visible, 3 for near-infrared, and 10 for infrared) in 0.5-2 km resolution every 2.5 minutes around Japan. Among the many observation bands, we selected six bands to represent the vertical conditions of cloud and precipitable water and prepared 4 km spatial resolution data at 10 min intervals. A summary of the selected observation bands is presented in Table 1.

Table 1. Utilized Bands from Himawari-8

B.N.	**Wave length**	**Observation target**
B07	3.9 μm	Cloud distribution
B08	6.2 μm	Precipitable water amount in high altitude (350 hpa)
B09	6.8 μm	Precipitable water amount in middle-high altitude (450 hpa)
B10	7.3 μm	Precipitable water amount in middle altitude (550 hpa)
B13	10.4 μm	Cloud height and types
B15	12.4 μm	Aerosol

Subsequently, we prepared a spatial map of atmospheric variables from the ground gauged observations to be compatible with the satellite images. The target region is shown in Fig. 2, and there are 44 observation points from the Automated Meteorological Data Acquisition System (AMeDAS) of Japan. By utilizing the inverse distance weight method, we generated 4 km spatial resolution data for the target region with 34 x 34 grids format that will be the same format to the satellite image data. The prepared 6 parameters of atmospheric data from the ground gauged observations were precipitation, temperature, wind speed, wind direction (u & v), and sunshine ratio at 10 min intervals. Further details on the input data sets are given in the next section.

2.3 Input Data and Model Structure

We tested four different types of input data sets and four different types of model structures to select the best input combination and model structures. All four types of input data sets were tested in four types of model structures, and the best model structure was selected based on the overall model performance. During this test, we also examined several hyperparameters (batch size and learning rate). In this section, the details of the input data sets and model structures are described.

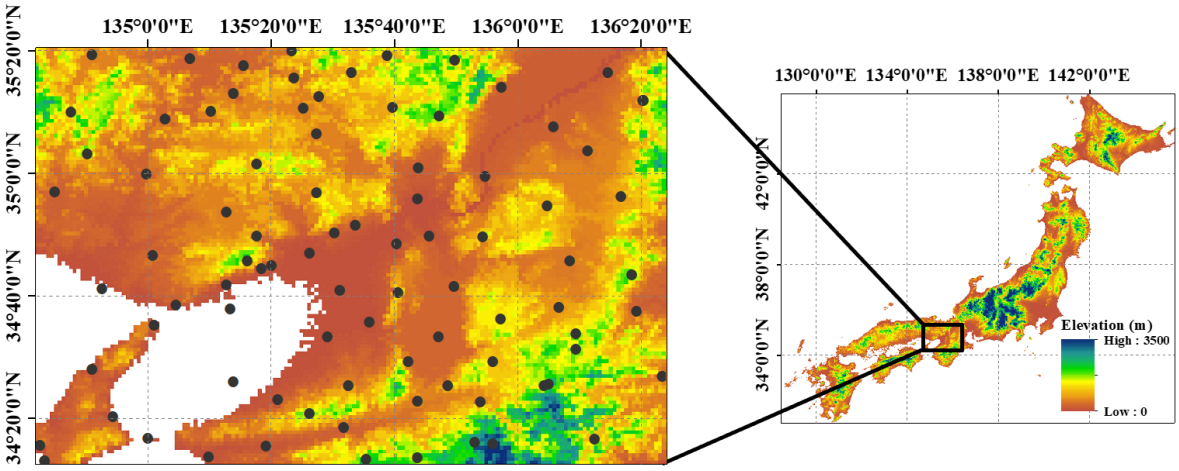

Figure 2. Location map of target area (134.7°E ~136.4°E; 136-km, 34.21°N ~35.34°N; 136-km). Black dots represent the locations of 44 ground gauge stations.

Table 2. Details of Input Data Sets

Data Sets	**Data Source**	**Data Format**
A-TS	**AMeDAS**, 6 variables; precipitation, temperature, wind speed, wind direction (u & v), and sunshine ratio	Time series of t_0 & t_{-30} (6 variables at 2-time steps; 12 channels)
A-DF		Difference between t_0 & t_{-30} (6 channels)
H-TS	**Himawari**-8, 6 bands; B07, B08, B09, B10, B13, B15 + Precipitation (AMeDAS)	Time series of t_0 & t_{-30} (7 variables at 2-time steps; 14 channels)
H-DF		Difference between t_0 & t_{-30} (7 channels)

Table 3. Details of Model Structure and Training Setup

CATEGORY	**ITEM**	**VALUE**
PREDICTION TARGET	**R**ainfall threshold (mm/10 min)	1.0
	Lead-time (min)	30
MODEL STRUCTURE	**C**onvolution filter size	3×3
	Conv. filtering number	**32-64 (2 Conv.)** or **32-64-128 (3 Conv.)**
	Pooling window	**No pooling** or **2×2**
ACTIVATION FUNCTION	**C**onvolution layer	Leaky ReLU ($\alpha = 0.3$)
	Fully connected layer	Sigmoid
TRAINING OPTION	**O**ptimizer	ADAM
	Learning rate	1.0×10^{-3} or 1.0×10^{-4}
	Mini-batch size	**32** or **96**
	Epoch number (max)	50

For the two input data sets, Himawari-8 and AMeDAS, we prepared two different types of data formats: the time series (TS) and the difference (DF). The TS data format utilizes the atmospheric data at the current time (t_0) and the data from 30 min ago (t_{-30}) together, and thus the input data have 12 channels in the case of the AMeDAS data sets (two channels for six atmospheric variables). The DF data format utilizes the atmospheric data with the difference between the current time and 30 min ago ($t_0 - t_{-30}$), and thus the input data have six channels in the case of the AMeDAS data sets. For the Himawari-8 data sets, in addition to the selected six bands, we included precipitation data as well, which is from the AMeDAS data set, to include the rainfall information clearly. Thus, the Himawari-8 input data sets have 14 channels in the TS format and 7 channels in the DF format. A summary of the input data sets is shown in Table 2.

We prepared these two different types of data formats to check the effect of seasonal variation in our model. The summer season in Japan is very humid with high atmospheric temperatures, and this particular seasonal condition results in frequent rainfall events in summer. However, rainfall can occur in other seasons, as well as at lower temperatures and mild wind conditions. We decided to test two versions of the data format to understand the behavior and efficiency of our model under this seasonal effect. The DF data format (difference between t_0 and t_{-30}) eliminates the seasonal effect and provides information only for the most recent atmospheric change within the last 30 min. In contrast, the TS data format (data at t_0 and t_{-30}) contains information about the different seasonal atmospheric patterns.

For all input data sets, we prepared 10 min intervals in 4 km spatial resolution data (34 x 34 grids format). We collected data from March 2015 to July 2019, and we excluded winter season (from November to February) to avoid any possible noise from snowfall events in winter season. Of this data, the values from March 2015 to March 2019 were allocated for training and testing, and the testing data was randomly selected among this duration with the portion of 12.5%, which is about 4 months long data. Validation data is another 4 months long data from April to July 2019. The purpose of testing is to pick up the best training option while avoiding the overfitting problem, and the purpose of validation is to evaluate the trained model performance with a new data set. In summary, we have prepared 159,927 input data altogether, and it is divided into 124,581 data for training, 17,798 data for testing, and 17,548 data for validation. And, every atmospheric variable was normalized to have a zero mean and one standard deviation, before we divided them into training, testing, and validation.

For the model configuration, we tested four different types of model structures, which were combinations of “with” or “without” a pooling process, and two or three convolutional processes. In other words, we tested Type 1: C1-P1-C2-P2, Type 2: C1-P1-C2-P2-C3-P3, Type 3: C1-C2, and Type 4: C1-C2-C3, where C stands for the convolutional process and P stands for the pooling process. For the filtering numbers of each convolutional process, we set 32 times for C1, 64 times for C2, and 128 times for C3. The filter size in all filtering processes was set as 3×3. These values were chosen based on our results in the first model test. We also need to check the efficiency of the hyperparameters to maximize the training efficiency. Two types of learning rates (1.0×10^{-3} or 1.0×10^{-4}) and two types of mini-batch sizes (32 or 96) were tested to determine the best training performance. The maximum epoch number of the training was set as 50 times, and, in every 10 epochs, the cross-entropy was checked with the testing data and the training process was terminated if the overfitting pattern appeared. A summary of the model setup and the tested model structure is presented in Table 3.

2.4 Experimental Design and Evaluation Criteria

In this study, we tested our rainfall prediction model to detect rainfall events of more than 1.0 mm/10 min for 30 min of prediction lead time. This threshold corresponds to a rainfall intensity of 6.0 mm/h. We set this threshold to identify the definite rainfall events and to secure a sufficient number of events for the proper training of the CNN algorithm. We are trying to develop a rainfall forecasting model for hydrological purposes, and thus, we might want to set a higher threshold. However, if we set a high threshold (e.g., 30 mm/h), the number of rainfall events to train the model is limited drastically. Once

we successfully confirm the model performance and efficiency with a sufficient amount of training data, it is able to test various thresholds in the following research, as well as considering various forecasting lead times.

We developed our model in the following three steps: First, we prepared four types of input data sets and fed them into four types of CNN algorithms. Each CNN algorithm was trained with four types of hyperparameter combinations (two learning rates and two batch sizes) as well as two different random seed numbers. The random seed number is used to generate the initial weight and bias in each algorithm, and we trained every option twice with different random seed numbers to avoid any randomness in the training results. Based on these results, we are going to selected the best CNN structure.

Second, with the results to reference from the first step, we carried out a sensitivity analysis for each input variable. In our proposing sensitivity analysis, we eliminated the effect of each input variable by setting it to zero in the input and ran the model again with the validation data. If the accuracy is drastically lower than the first-step results, we regard the input variable as a key factor in the model. By checking the changes with the zero-input test one by one, we can identify higher impact input variables (that reduce the accuracy drastically) and lower impact input variables (that do not reduce the accuracy).

Third, we composed the best model setup with the best model structure selected in the first step and the best input combination selected in the second step. For a comprehensive evaluation, the best model setup was also trained with four types of training options (based on hyperparameter combinations) and two random seed numbers. Finally, we show the extent of the proposed algorithm accuracy.

To evaluate model performance, we adopted four indices: accuracy (ACC), CSI, POD, and FAR. As shown in Table 4, CSI is the correct prediction ratio among the sum of model predictions and observed events, POD is the ratio of correct predictions among observed events, and FAR is the ratio of false predictions among the model predictions. These three criteria exclude the correct prediction for no-rainfall events, to emphasize the model performance for rarely occurring rainfall events, and they are often utilized in the evaluation of the atmospheric prediction model. ACC is the overall ratio of correct predictions, including rainfall and no-rainfall events.

Table 4. Evaluation Indices

Event Category		**Observation**	
		Rain (O)	**No rain (X)**
Model Pred.	**Rain (P)**	**PO** Correct	**PX** False
	No rain (X)	**XO** Missing	**XX** Correct

$$CSI = \frac{PO}{PO + PX + XO}$$

$$POD = \frac{PO}{PO + XO}$$

$$FAR = \frac{PX}{PO + PX}$$

$$ACC = \frac{PO + XX}{PO + PX + XO + XX}$$

3. Results and Discussions

3.1 Selecting the Best Model Structure

We tested four input data sets, four CNN structures, and four hyperparameter sets, with two initial random seed numbers. First, to analyze the performance of each model structure, the results from each input data set are separately illustrated in Fig. 3. In the figure, each panel shows 32 testing results from each input data set with the indices of POD and CSI. The 32 testing results are from the combination of four model structures, which is a combination of pooling options (CNN1: with pooling process, CNN2: without pooling process) and convolution number (N=2, N=3), the four hyperparameter sets (two learning rates and two batch sizes), and two initial random seed numbers at the beginning of the training.

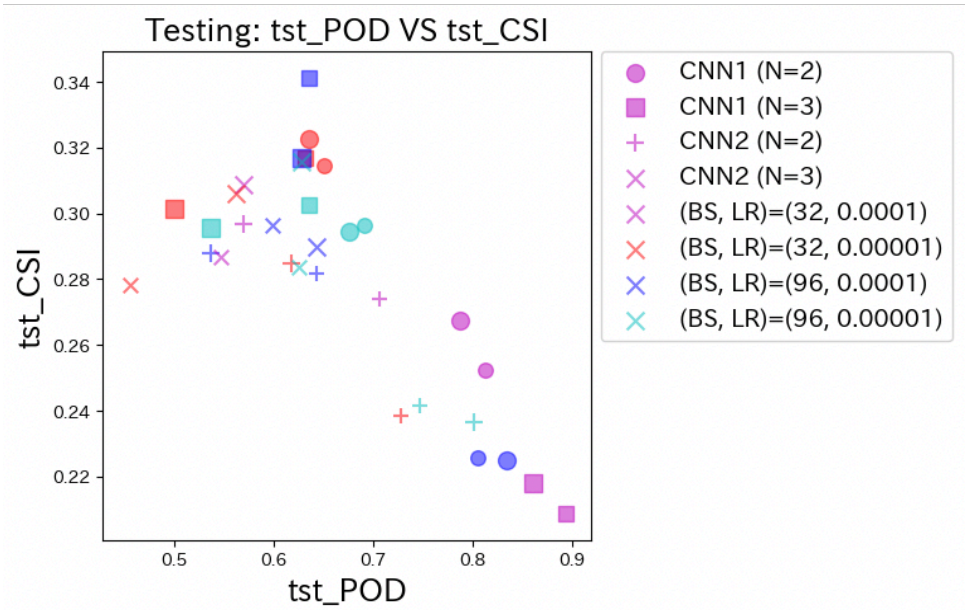

Figure 3. Accuracy indices (POD & CSI) from the testing results with four input data sets; A-TS. In each panel, 32 results are shown with the combination of four model structures (CNN1: convolution and pooling, CNN2: convolution only; N=2: two times of convolution, N=3: three times of convolution; these are marked with the symbol of □, ○, +, ×), four hyperparameter combinations (two learning rates and two batch sizes; these are marked with four different colors with purple, orange, blue, and green), and two initial random seed numbers (marked with the same color and the same shape).

Table 5. Evaluation indices for validation based on the best three indices in training

Input	Testing					Validation			
	Index	CSI	POD	FAR	ACC	CSI	POD	FAR	ACC
A-TS	Best **CSI**	**0.450**	0.651	0.406	0.988	**0.240**	0.399	0.623	0.985
	Best Loss	0.379	0.688	0.542	0.983	0.237	0.448	0.665	0.983
	*Best **POD***	0.289	***0.853***	0.696	0.968	0.237	***0.764***	0.744	0.972
A-DF	Best **CSI**	**0.341**	0.636	0.576	0.981	**0.219**	0.419	0.686	0.983
	Best Loss	0.306	0.563	0.598	0.981	0.211	0.384	0.680	0.983
	*Best **POD***	0.209	***0.893***	0.786	0.948	0.148	***0.793***	0.847	0.947
H-TS	Best **CSI**	**0.460**	0.691	0.422	0.988	**0.291**	0.458	0.557	0.987
	Best Loss	0.422	0.684	0.476	0.986	0.290	0.443	0.543	0.987
	*Best **POD***	0.308	***0.824***	0.670	0.972	0.236	***0.665***	0.733	0.975
H-DF	Best **CSI**	**0.346**	0.621	0.562	0.982	**0.204**	0.438	0.724	0.980
	Best Loss	0.269	0.618	0.678	0.974	0.194	0.483	0.754	0.977
	*Best **POD***	0.225	***0.846***	0.766	0.955	0.167	***0.734***	0.822	0.958

Figure 4. Venn diagram of model performance from the testing results of A-DF: the best CSI (left), the best Loss (middle), and best POD (right) model performances, respectively. The number outside of the colored area (e.g., 17331) represents the events number of no-rain that is observed and correctly forecasted, the number in red color area (e.g., 99) is the events number of rain but failed to predict, the number in blue color area (e.g., 235) is the event number of no rain but model falsely predict, and the number in the intersection (e.g., 173) is the event number of rain that is correctly predicted.

If we focus on the results from the A-TS input (AMeDAS data with time series of t_0 & t_{-30}) shown in the upper panel of Fig. 3, the results from CNN1 (marked with ○ or □) show high values in CSI and POD compared to the results from CNN2 (marked with + or ×). We can see that the results from CNN1 show a wide range of model performances depending on the training options. For example, the results from CNN1 with three convolutional processes show high CSI with low POD when it is trained with a batch size of 96 and learning rate of 0.0001 (blue colored □), and they show high POD with low CSI when it is trained with a batch size of 32 and learning rate of 0.001 (purple □). Most results with high POD show low CSI due to high FAR, and the results with high CSI do not show significantly good POD. In general, CSI is a comprehensive index that considers the concepts of POD and FAR, but we also want to consider POD as a valuable index separately to check how well it can detect rainfall events.

The results from other input data sets show a pattern similar to the results from A-TS. The result from the A-DF input (AMeDAS data with the difference between t_0 and t_{-30}; down left panel in Fig. 3) shows a similar pattern to the results from A-TS. The results from H-TS (Himawari-8 data; down middle panel) as well as the results from H-DF (bottom right panel) also show a similar pattern but with a slightly more diverse POD and CSI combination. For every input data set, we can see that the results from CNN2 (marked with + or ×), which are the results from the convolution process only, are stable regardless of the training option. However, the results are not good compared to those from CNN1 (marked with ○ or □) with respect to the CSI or POD. It seems that the pooling process in CNN1 provides more flexible training results compared to that without the pooling process. CNN1 is more likely to be trained to give a high CSI or POD. The superiority of CNN1 to CNN2 is apparent from the results of the H-DF input (low right panel in Fig. 5).

When the results from CNN1 were examined for the number of convolution processes (N=2 or N=3), CNN1 with N=3 (marked with □) performs slightly better than the one with N=2 (marked with ○). For every input data set, the best CSI was achieved from CNN1 with N=3, and the best POD was also achieved with N=3. Thus, we decided to select the best model structure in our prediction model as CNN1 with N=3, which is the model with three convolution and pooling processes.

Regarding the training option, it is difficult to decide which hyperparameter set is the best one. In the case of the A-TS input, it is apparent that high CSI values were achieved with a batch size of 96 and a learning rate of 0.0001 (e.g., blue color □), and high POD values were achieved with a batch size of 32 and a learning rate of 0.001 (purple color □). However, considering the results from other input data sets, it is difficult to find a clear trend; four colors (purple, orange, blue, and green) show the results from four different hyperparameter sets. To consider this behavior of our CNN model, we decided to consider every hyperparameter

combination for every training, and picked up the best test results based on three parameters: a model with the best CSI, a model with the best POD, and a model with the best loss. Here, the best loss represents the test result with the minimum cross-entropy values, which shows a balanced CSI and POD.

The accuracy indices for the validation data are shown in Table 5 based on the best three indices in the tests (best CSI, best loss, and best POD). The best CSI during training was 0.460 with the input of H-TS, and the best POD was 0.893 with the A-DF input data set. These indices are slightly degraded to 0.291 and 0.793 with the validation data. To illustrate the model performance, a Venn diagram with the training results of A-DF is shown in Fig. 4. The best CSI model performance (Fig. 4, left) shows a CSI of 0.341, which is the ratio of correct predictions (173) to the sum of model predictions and observed events (99 + 173 + 235). In the case of the best POD model performance (Fig. 4, right), the POD is very high at 0.893 = 243/ (29+243), but the CSI is rather low at 0.209 = 243/ (29+243+892) owing to the high FAR, which is 0.789 = 892/ (243+892). However, considering the correct prediction for the no-rain events (16674), the number of false alarms (892) was not significant. In this case, the accuracy of all prediction performances (ACC) was still very high, at 0.948.

A more detailed analysis of the model performance for the input data difference is presented in the next section. The performance indices shown in Table 5 are utilized as a reference value for the sensitivity analysis of the input variables.

3.2 Selecting the Best Input Data Set

With respect to the input data format, the TS data set (values at t_0 and t_{-30}) shows better results than the DF data set (difference between t_0 and t_{-30}) in terms of the CSI (0.450 vs. 0.341 and 0.460 vs. 0.346) as shown in Table 5. With respect to the POD, the DF data set shows better results than the TS data set (0.893 vs. 0.853 and 0.846 vs. 0.824). Even though the difference is not very significant, we decided to adopt the TS data format for the best input data set.

Regarding the data source, it is not clear which source provides better results; input only with the AMeDAS ground gauge data and input with the Himawari-8 data show approximately the same model performance (0.450 vs. 0.460 and 0.341 vs. 0.346 for CSI). It is reasonable to combine these two data sources, and we tried to select the best input variables from them based on a sensitivity analysis of each input variable.

For the sensitivity analysis on the input variables, we designed a new concept of sensitivity analysis on a neural network. Because the ANN structures are highly depending on the input variable numbers and every ANN should be trained again based on various hyper parameter options, it may not be a good idea to build every different ANN model for every different input variable combination. Rather than that, we eliminated the effect of each input variable one by one from the already trained ANN by setting the values as zero and running the trained model again with the validation data. Because all input variables were normalized with a zero mean and one standard deviation, if a certain input value is set as zero and given to the trained neural network, the effect of the input to the network will be eliminated in the prediction performance since the network is not able to get any information from the zero-input variable.

If the accuracy is drastically decreasing compared to the reference results shown in Table 5 when a certain input variable was given with zero values, we regard the input variable as a key factor in the model, and vice versa. By checking the changes with the zero-input test, one by one, we can understand higher impact input variables (that bring the accuracy down drastically) and lower impact input variables (that do not reduce the accuracy).

Fig. 5 illustrates the result of the sensitivity analysis by showing the change in accuracy indices for each zero-setting variable in the case of the Himawari-8 data (H-TS input data set). It was observed that precipitation input is the most valuable information because the prediction accuracy decreased drastically when the precipitation input value was set to zero. The CSI value decreased by 0.42 on average, compared to the original accuracy (the left-most of the bottom row in Fig. 5).

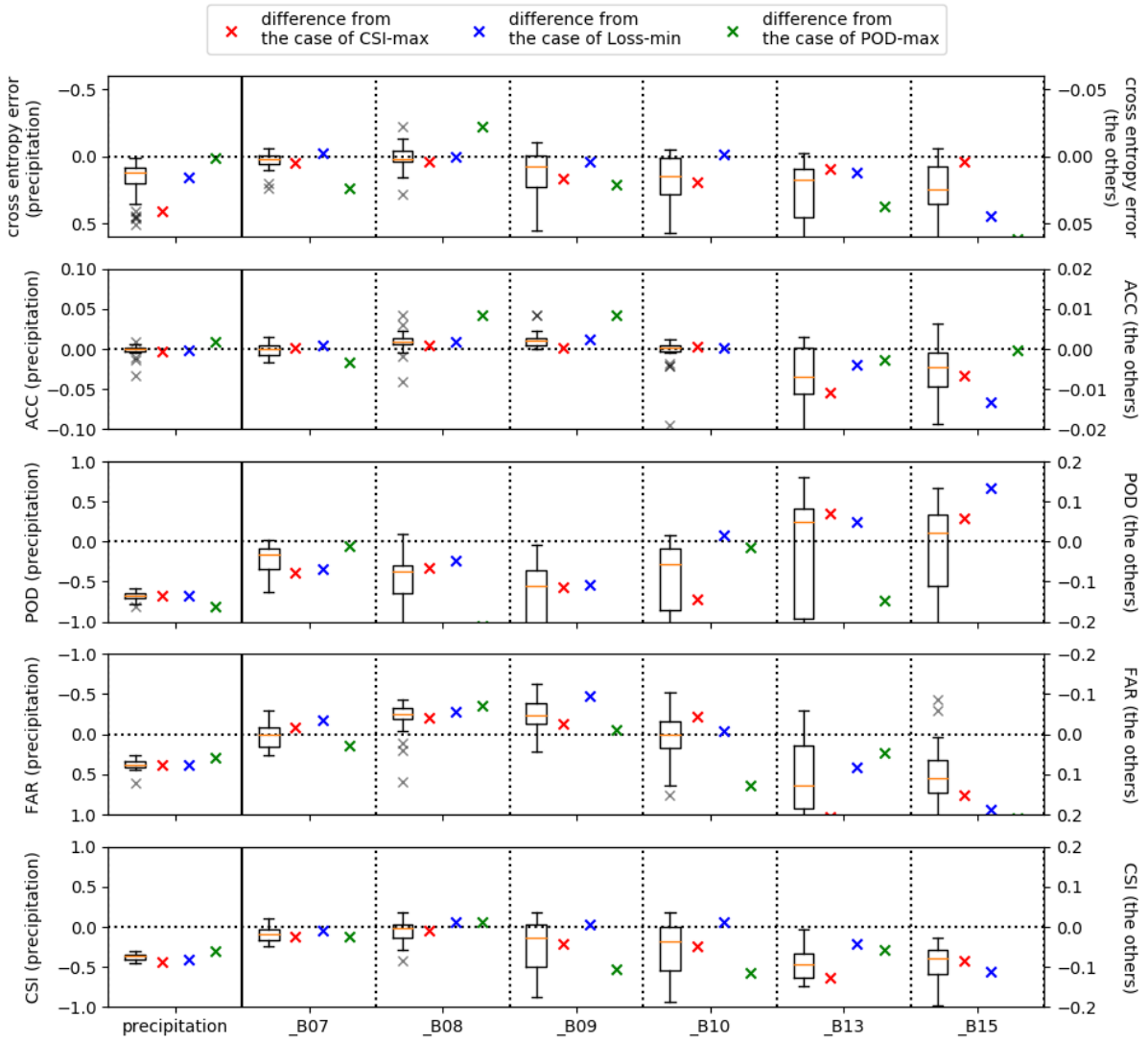

Figure 5. Sensitivity analysis on input variables. Vertical axis shows the index change compared to the reference values shown in Table 4, when one of the input variables was set as zero. Box plot is the result of all model combinations, while the red, blue, and green marks are from the best model on the three parameters (best CSI, best Loss, best POD, respectively).

Among the five bands adopted in our model, the B07 and B08 bands are not very sensitive to the model performance, and the B10, B13, and B15 bands show sensitivity when these values are eliminated from the input data (see the CSI changes in Fig. 5). The effect of the B09 band is ambiguous, but we decided to exclude this band in the best input data set because it has a negative effect on the accuracy (ACC; second row in Fig. 5) when it is eliminated.

With this sensitivity analysis, we decided to pick up B10 (7.3 μm; precipitable water amount in middle altitude), B13 (10.4 μm; cloud height and types), and B15 (12.4 μm; aerosol) for the best input data. A sensitivity analysis was also carried out for the AMeDAS data (A-TS input data set), and we confirmed that precipitation data is very important, and that the sunshine ratio and temperature are not critical information in the model performance. Thus, we selected precipitation, wind speed, and wind direction (in u & v) for the best input data.

3.3 The Best Model Combination

In the last two sections, we evaluated four types of model structures, four types of input data formats, and examined the sensitivity of each input variable. Based on the results, we adopted the best model structure as C1-P1-C2-P2-C3-P3, which is a CNN

with three convolution and pooling processes. The best input data format is adopted as the time series of the last 30 min data, that is, the spatial map of atmospheric variables at t_0 and t_{-30}. The most significant input variables were selected based on precipitation, wind speed, wind direction (u & v) from the AMeDAS data, and B10, B13, and B15 bands of the Himawari-8 satellite images. A summary of the best model setup is presented in Table 6.

We trained the best model with four types of training options (two types of batch sizes and two types of learning rates) and two different initial random seed numbers. Among the eight trained results, the best CSI, the best POD, and the best loss of testing results were selected, and the model performances are summarized in Table 7 along with the validation results. In the table, the results of A-TS (input only with ground gauged data) shown in Table 5 are also shown again for comparison. We can easily find that the best model set up with the AH-TS input data set provides improved results in most cases. For example, the validation result from the best CSI model improved from 0.240 to 0.296 with improved the POD (from 0.399 to 0.468) and decreased the FAR (from 0.623 to 0.554). The validation results from the best POD model and the best loss model also show improved CSI with the AH-TS input data set. Only the POD from the best POD model did not improve; the POD of the best POD model with A-TS was 0.764, and it was 0.616 with AH-TS. However, the FAR is significantly improved (from 0.744 to 0.630), and thus the AH-TS model shows an improved CSI compared to the A-TS option (from 0.237 to 0.300).

Table 6. Details on the Best Model Set-up

Input	**Data Source**	**Data Format**
AH-TS	**AMeDAS**; precipitation, wind speed, wind direction (u & v) and **Himawari-8**; B10, B13, B15	Time series of t_0 & t_{-30} (7 variables at 2-time steps; 14 channels altogether)
Model Structures: C1-P1-C2-P2-C2-P3 Convolution: filter size: 3×3, filtering numbers: 32-64-128, Pooling: max pooling in 2×2 window		

Table 7. Evaluation indices for validation based on the best three indices in training

Input	Testing					Validation			
	Index	CSI	POD	FAR	ACC	CSI	POD	FAR	ACC
A-TS	Best **CSI**	**0.450**	0.651	0.406	0.988	**0.240**	0.399	0.623	0.985
	Best Loss	0.379	0.688	0.542	0.983	0.237	0.448	0.665	0.983
	*Best **POD***	0.289	***0.853***	0.696	0.968	0.237	***0.764***	0.744	0.972
AH-TS	Best **CSI**	**0.459**	0.722	0.442	0.987	**0.296**	0.468	0.554	0.987
	Best Loss	0.428	0.698	0.475	0.985	0.273	0.483	0.614	0.985
	*Best **POD***	0.417	***0.836***	0.545	0.982	0.300	***0.616***	0.630	0.983

We tested several options of model structures, input data formats, and training set-ups to build a rainfall prediction model using the CNN algorithm. It is clear that the CNN algorithm can successfully detect rainfall occurrence for a definite forecasting lead time as an image classification function when it is properly trained with historic atmospheric data. In addition, the model performance was improved when the input data were carefully selected after a sensitivity analysis of each input variable.

There might be no absolute answer for the best model option, such as a specific model structure or an absolute input data combination for a specific target area. The performance of the machine learning algorithm may differ regionally, and on various input data. However, once we extensively examine possible options with several model structures and carefully select the best input data combinations, the machine learning algorithm can provide an efficient way of hydrological modeling from a different aspect. The only thing we need to do is to test and evaluate the model performance repeatedly to improve the accuracy until we are satisfied.

4. Concluding Remarks

In this paper, we illustrated the development process of a rainfall occurrence prediction model using a CNN algorithm. By feeding the spatiotemporal information of atmospheric variables into the CNN algorithm, it was possible to train the algorithm to classify the atmospheric conditions, whether they are conducive for rain or not, within a definite forecasting lead time. In this study, we tested four types of CNN structures, four types of input data formats, and four types of hyperparameter sets, to build up the best model for our target area.

Based on our modeling test on the Kyoto area of Japan, a CNN with three convolution and pooling processes provides good performance. However, we found that model performance was also controlled by the training options. Our solution to this performance behavior was to select a good model structure with respect to the overall performance and utilize several model options based on various training options. Subsequently, we checked the model performance on three parameters: the best CSI index, the best POD index, and the best loss index.

For the best input data selection, we introduced a sensitivity analysis on the input variables rather than utilizing a correlation-based input selection. Based on the model performance with sufficient input data setting, we checked the change in accuracy when any one of the input data was missing. If the model accuracy decreases drastically when any one of the input variables is eliminated, we conclude that the input variable is a major variable in our model. If the accuracy does not change much even without a certain input, we decide that the input is not an important one in the model. Based on this input data sensitivity analysis, we selected the best input candidates for precipitation, wind speed, and wind direction (u and v) from ground gauged data, and B10, B13, and B15 bands of the Himawari-8 satellite image data.

The best model shows promising performance with a CSI of 0.296 and POD of 0.616. Considering the no-rain events, the prediction accuracy was as high as 0.987. It is clear that the CNN algorithm can classify the atmospheric conditions to predict rainfall occurrence at a definite lead time (in this study, 1 mm/10 min of rainfall intensity for 30 min ahead of time). It was also clear that the satellite image data can provide further information on the atmospheric conditions and provide improved model performance, rather than only using ground-gauged atmospheric data. However, there is more room to improve prediction accuracy with additional tests with other input combinations and training options.

The main purpose of this paper is not to show the best model structures but to show the possibility of a machine learning algorithm by suggesting a standardized modeling procedure. It will be of great help if this kind of modeling process is applied in numerous other regions with distinct options and the results are shared.

Acknowledgements

This research was supported by KAKENHI of JSPS (Project No:22K04332), Development of Hybrid Flood Forecasting System based on Machine Learning of Rainfall Forecasting.

References

Alizadeh, M.J., Kavianpour, M.R., Kisi, O., Nourani, V., 2017. A new approach for simulating and forecasting the rainfall-runoff process within the next two months. Journal of Hydrology. 548, pp. 588-597.

Bowler, N.E.H., Pierce, C.E., Seed, A., 2004. Development of a precipitation nowcasting algorithm based upon optical flow techniques. Journal of Hydrology. 288, pp. 74-91.

Cheng, M., Fang, F., Kinouchi, T., Navon, I.M., Pain, C.C., 2020. Long lead-time daily and monthly streamflow forecasting using machine learning methods. Journal of Hydrology. 590, 125376

Cloke, H.L. and Pappenberger, F., 2009. Ensemble flood forecasting: A review. Journal of Hydrology. 375, pp. 613-626.

Cuo, L., Pagano, T.C., Wang, Q.J., 2011. A review of quantitative precipitation forecasts and their use in short- to medium-range streamflow forecasting. Journal of Hydrometeorology. 12(5), pp. 713-728

Ganguly, A.R., Bras R.L., 2003. Distributed Quantitative Precipitation Forecasting using Information from Radar and NWP Models. Journal of Hydrometeorology. 4, pp. 1168-1180.

Golding, B.W., 2000. Quantitative Precipitation Forecasting in the UK. Journal of Hydrology. 239, pp. 286-305.

He, X.H., Guan, H., Qin, J., 2015. A hybrid wavelet neural network with mutual information and particle swarm optimization for forecasting monthly rainfall. Journal of Hydrology. 527, pp. 88-100.

Kabir, S., Patidar, S., Pender, G., 2020. A deep convolutional neural network model for rapid prediction of fluvial flood inundation. Journal of Hydrology. 2020, 125481.

Khosravi, K., Panahi, M., Golkarian, A., Keesstra, S.D., Saco, P.M., Bui, D.T., Lee, S., 2020. Convolutional neural network approach for spatial prediction of flood hazard at national scale of Iran. Journal of Hydrology. 591, 125552.

Kim, S., Tachikawa, Y., Sayama, T., Takara, K., 2009. Ensemble Flood Forecasting with Stochastic Radar Image Extrapolation and a Distributed Hydrologic Model. Hydrological Processes. 23 (4), pp. 597-611.

Kim, S., Kim, H., Lee, J., Yoon, S., Kaoru, S., Kashinath, K., 2019. Deep Hurricane Tracker: Tracking and Forecasting Extreme Climate Events. 2019 IEEE (WACV), pp. 1761-1769.

Kim, S., Suzuki, T., and Tachikawa, Y., 2020. Rainfall Occurrence Prediction with Convolutional Neural Network. Journal of JSCE, Ser. B1(Hydraulic Engineering). 76(2), pp. I_379-I_384.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient Based Learning Applied to Document Recognition. Proceedings of the IEEE. 86 (11), pp. 2278-2324.

Mandapaka, P.V., Germann, U., Panziera, L., Hering, A., 2012. Can Lagrangian extrapolation of radar fields be used for precipitation nowcasting over complex Alpine orography? Weather and Forecasting. 27(1), pp. 28-49.

Matsuoka, D., Watanabe, S., Sato, K.,Kawazoe, S., Yu, W., Easterbrook, S., 2020. Application of deep learning to estimate atmospheric gravity wave parameters in reanalysis data sets. Geophysical Research Letters. 47, e2020GL089436.

Nearing, G. S., Kratzert, F., Sampson, A. K., Pelissier, C. S., Klotz, D., Frame, J. M., et al., 2021. What Role Does Hydrological Science Play in the Age of Machine Learning? Water Resources Research. 57(3), e2020WR028091

Nguyen, D.H., Bae, D.H., 2020. Correcting Mean Areal Precipitation Forecasts to Improve Urban Flooding Predictions by Using Long Short-Term Memory Network. Journal of Hydrology. 584, 124710.

Ni, L., Wang, D., Singh, V.P., Wu, J., Wang, Y., Tao, Y., Zhang, J., 2020. Streamflow and rainfall forecasting by two long short-term memory-based models. Journal of Hydrology. 583, 124296.

Park, M., Kim, S., Suzuki, T., and Tachikawa, Y., 2019. Sensitivity analysis on data array and model structure of convolutional neural network for rainfall occurrence prediction. Journal of JSCE, Ser. B1(Hydraulic Engineering). 75(2), pp. I_1183-I_1188.

Pan, B., Hsu, K., AghaKouchak, A., Sorooshian, S., 2019. Improving Precipitation Estimation Using Convolutional Neural Network. Water Resources Research. 55(3), pp. 2301-2321.

Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y., 2014.

How to Construct Deep Recurrent Neural Networks, arXiv:1312.6026v5

Rawat, W., Wang, Z., 2017. Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. Neural Computation, 29(9), pp. 2352–2449.

Scher, S., 2018. Toward Data-Driven Weather and Climate Forecasting: Approximating a Simple General Circulation Model with Deep Learning. Geophysical Research Letters. 45(22), pp. 12616-12622.

Shen, C., Laloy, E., Elshorbagy, A., Albert, A., Bales, J., Chang, F.-J., Ganguly, S., Hsu, K.-L., Kifer, D., Fang, Z., Fang, K., Li, D., Li, X., and Tsai, W.-P., 2018. HESS Opinions: Incubating deep-learning-powered hydrologic science advances as a community. Hydrology and Earth System Sciences, 22(11), pp. 5639-5656.

Shrestha, D.L., Robertson, D.E., Wang, Q.J., Pagano, T.C., Hapuarachchi, H.A.P., 2013. Evaluation of numerical weather prediction model precipitation forecasts for short-term streamflow forecasting purpose, Hydrology and Earth System Sciences 17(5), pp. 1913-1931.

Suzuki, T., Kim, S., Tachikawa, Y., 2018. Application of Convolutional Neural Network to Occurrence Prediction of Heavy Rainfall. Journal of JSCE, Ser. B1(Hydraulic Engineering). 74(4), pp. I_877-I_882. (Japanese with English Abstract)

Tao, L., He, X., Li, J., Yang, D., 2021. A multiscale long short-term memory model with attention mechanism for improving monthly precipitation prediction. Journal of Hydrology. 602, 126815.

Thorndahl, S., Einfalt, T., Willems, P., Nielsen, J. E., ten Veldhuis, M.-C., Arnbjerg-Nielsen, K., Rasmussen, M. R., Molnar, P., 2017. Weather radar rainfall data in urban hydrology, Hydrology and Earth System Sciences. 21, pp. 1359–1380.

Tu, T., Ishida, K., Ercan, A., Kiyama, M., Amagasaki, M., Zhao, T., 2021. Hybrid precipitation downscaling over coastal watersheds in Japan using WRF and CNN. Journal of Hydrology: Regional Studies. 37, 100921.

Wang, Y., Fang, Z., Hong, H., Peng, L., 2020. Flood susceptibility mapping using convolutional neural network frameworks. Journal of Hydrology. 582, 124482.

Weyn., J.A., Durran, D.R., Caruana, R., 2019. Can machines learn to predict weather? Using deep learning to predict gridded 500-hPa geopotential height from historical weather data. Journal of Advances in Modeling Earth Systems. 11, pp. 2680-2693.

Wu, H., Yang, Q., Wang, G., 2020. A spatiotemporal deep fusion model for merging satellite and gauge precipitation in China, Journal of Hydrology. 2020, 124664.

Wu, M.-C., Lin, G.-F., 2017. The very short-term rainfall forecasting for a mountainous watershed by means of an ensemble numerical weather prediction system in Taiwan. Journal of Hydrology. 546, pp. 60-70.

Yu, W., Nakakita, E., Kim, S., Yamaguchi, K., 2015. Improvement of rainfall and flood forecasts by blending ensemble NWP rainfall with radar prediction considering orographic rainfall. Journal of Hydrology. 531, pp. 494-507.

(Received August 31, 2022)