We developed a real-time river stage prediction model using a hybrid deep neural network and physically based distributed rainfall-runoff model. The main component of the hybrid model was a four-layer feed-forward artificial neural network. Using the predicted flow of the rainfall-runoff model as the input data of the neural network, we integrated the two models into the hybrid model. The input data of the hybrid model included upstream water level, hourly change in water level, and estimated hourly change in catchment storage. The output was the change in water level at the prediction point. In the training phase, input data and supervised data were formed using the observed data. In the prediction phase, input data were formed using a combination of the observed data and flowrate calculated using the distributed model.

The result of the hybrid model outperformed those of the ANN and distributed models. Especially in the largest flood event, the performance of the hybrid model was significantly stronger.

Key Words: river stage, real-time prediction, deep learning, artificial neural network, hybrid model

1. INTRODUCTION

To reduce frequent floods and inundation disasters, the flood forecasting system has been operating in class-A rivers throughout Japan\(^1\). Therefore, there is a need to improve the instantaneity and accuracy of flood forecast information.

As shown in Table 1, flood forecasting methods are categorized into two types. The first type is the runoff model, which expresses rainfall runoff processes physically or conceptually. The other type is statistical models, which are based on the historical records of floods. Though both these models are widely used in practical applications, their prediction accuracy is not always sufficient\(^1\). Therefore, several studies have been conducted to improve their accuracy. For runoff model-based prediction, a conceptual model that expresses rainfall-runoff processes conceptually, such as through storage function\(^3\) and tank models\(^3\), and a distributed model, which expresses the basin physically, have been developed\(^3, 5\). These models have been applied to class-A rivers throughout Japan. These runoff models have several existing problems, including modeling approximation error, parameter setting error, and rainfall data error. Therefore, in flood prediction using runoff models, the models are continually corrected through comparison of past prediction results and current observations.

Thus far, various approaches have been proposed as corrective techniques. Typical methods include simple error compensation by subtracting the previous error from the current predicted value or data assimilation techniques such as the particle filter\(^6, 7, 8, 10, 11\) and ensemble Karman filter\(^9\) approaches. However, such correction techniques are still under development for practical use; in many cases, sufficient accuracy cannot be guaranteed\(^10, 11\).

On the other hand, statistical methods make predictions based on previous observations of water level or flowrate. One classical method is based on the auto-regressive (AR) model\(^12\). This method does not introduce physical processes but makes predictions simply by analyzing the time series of the data. Recently, linear regression and artificial neural networks (ANN) have also been applied to flood prediction\(^13, 14, 15, 16\). These models learn the patterns of past floods and then predict the future water level or
flowrate based on the current flood status. Such models learn the actual flood records to allow them to express the nonlinear relationship between rainfall and runoff.

Many authors have focused on the ANN model and evaluated its applicability and validity in class-A rivers in Japan\(^\text{17, 18, 19}\). The ANN model exhibits excellent real-time performance because its calculations are completed immediately and it has good prediction accuracy. Furthermore, a flood prediction method based on a new learning method, deep learning\(^\text{20}\), has been proposed as an advanced form of the ANN model (hereafter, deep-learning model). Deep-learning models have been shown to be more accurate than ANN models based on conventional learning methods (hereinafter, conventional ANN models)\(^\text{21, 22}\). In the hydrological field, drought prediction\(^\text{23}\), reservoir inflow prediction\(^\text{24}\), and soil moisture distribution estimation\(^\text{25}\) have been reported applications of deep learning. However, additional future applications are possible.

The challenge of the ANN model is that it does not incorporate the essential physical mechanism. As the prediction result of the model is constrained by the past flood history used for learning, the prediction accuracy for the inexperienced scale of floods is not guaranteed\(^\text{16}\). This is a significant problem from the viewpoint of disaster prevention.

This paper proposes a hybrid model to improve prediction accuracy by combining the ANN model and the runoff model to compensate for the lack of physical mechanisms addressed in statistical models. Using this method, ANN can be incorporated into the runoff model currently adopted in the flood forecasting system applied for many class-A rivers in Japan and improve the model’s accuracy.

This paper first outlines the proposed hybrid model. Next, the ANN that is included in the hybrid model and the target flood are described. Then, a specific procedure for building a hybrid model combining the ANN and runoff model is presented. Finally, as an example of the model’s application, 1)
the deep ANN model, 2) the dis-tributed model, and 3) the hybrid model that combines these models is verified and validated for the Ooyodo river system.

2. CONCEPT OF THE HYBRID RIVER STAGE PREDICTION MODEL

(1) Overview of the hybrid model
The hybrid prediction model presented in this paper is an ANN model that incorporates the runoff model. Figure 1 shows the procedure for developing an ordinary ANN model and for de-veloping and conducting prediction calculations using the hybrid model.

In a general ANN model, learning and predic-tion are conducted using water level and rainfall as the main input data (Fig. 1 left). However, in the hybrid prediction model, the actual or predicted flowrate data are considered in the training and application of the model. Specifically, a value corresponding to a change in water storage (a value obtained by sub-tracting flowrate at the downstream end of the basin from rainfall) is used for the input layer of the hybrid prediction model instead of rainfall. The construction and real-time prediction procedures for these two models are shown on the right side of Fig. 1. In the model development phase, an input/output dataset is prepared using the observed water lev-el/rainfall/flowrate data, and ANN is trained. Then, in the real-time prediction phase, the trained ANN is used in combination with the runoff model. First, flowrate is predicted using a runoff model. Next, the estimated change in storage volume at a future time is estimated using the predicted flowrate and rainfall, and input into the trained ANN input data along with the water level and water-level change. The predicted water level is obtained from the ANN output via the above procedure.

(2) Theoretical background of the hybrid model
According to the rainfall-runoff theory of storage function2) Eq.(1) or the tank model, it is not the rainfall but the water storage in the basin that primarily determines the runoff into the river channel.

\[
\frac{dS}{dt} = R(t-T_L) - q(t), \quad S = kq^p \tag{1}
\]

Here, \( S \) is storage, \( R \) is rainfall, \( k \) and \( p \) are runoff coefficients, \( q \) is flowrate, \( t \) is time, and \( T_L \) is lag time. When using the storage function method, the interception loss of rainfall and separation of runoff components are often considered, but these factors are not discussed here. From Eq.(1), runoff \( q \) depends on storage \( S \). Therefore, the change in flowrate \( \Delta q \) from the current time to several hours later corre-sponds to the change in storage volume \( \Delta S \).

3. OVERVIEW OF ANN

(1) Composition of ANN
In this paper, we use a feed-forward neural net-work, which is composed of an input layer, a hidden layer, and an output layer, as shown in Fig. 2. The architecture of a single neuron is shown in Fig. 3. Each neuron computes the data according to the following equations:

\[
u = \theta + \sum_{i=1}^{K} w_i x_i \tag{2}
\]
\[
z = f(u) \tag{3}
\]

Here, \( u \) is the weighted sum of all inputs to a neuron, \( x \) is the input, \( w \) is the network weight, \( \theta \) is the bias, \( K \) is the number of inputs to each neuron, \( f(u) \) is the activation function, and \( z \) is the output of the neuron.
In this paper, we redefine that the bias is included in the parameter vector \( w \). While there are many types of activation functions, we use the sigmoid function, which has been most frequently used in conventional ANN. Identity mapping, \( f(u) = u \), is used as the activation function of the output layer.

(2) Training of ANN

In the learning process, the ANN optimizes all network weights to minimize the error in the network’s output and the training data – normally observation data. In this paper, the root mean error \( E \) is used to evaluate the error.

\[
E(w) = \frac{1}{2} \sum_{n=1}^{N} (d_n - y(x_n; w))^2 \tag{4}
\]

Here, \( N \) is the number of sample datasets, \( d \) is the desired data, and \( y \) is the network output. By iteratively applying the gradient descent method to the randomly preset initial network weight \( (w) \), we obtain the optimized \( w \), which minimizes \( E \).

We apply the stochastic gradient descent method\(^{26} \) for optimization.

In the “normal” gradient descent method, the gradient of the objective function \( E \) is calculated from all sample data. In the stochastic gradient descent method, on the other hand, the gradient of the objective function \( E \) is approximated by a small part of the training data (called mini-batch). Compared to the normal gradient descent method, it has advantages of speeding up learning and avoiding convergence to the local minima. In the gradient descent method, \( w \) is updated by the following equation:

\[
w^{(t+1)} = w^{(t)} - \varepsilon \nabla E \tag{5}
\]

Here, \( \varepsilon \) is the learning rate, which decides the amount of updating of \( w \), and subscript \( t \) is the number of iterations. In this paper, we apply the backpropagation method\(^{27} \) to obtain \( \nabla E \). To set the learning rate, we apply AdaGrad\(^ {28} \), which calculates the learning rate by using the following equation:

\[
\varepsilon = \frac{\varepsilon_0}{\sqrt{\sum_{i=1}^{T} (\nabla E_i)^2}} \tag{6}
\]

Here, \( \varepsilon_0 \) is the initial value of the learning rate, and \( T \) is the calculation step number. To improve the convergence of the gradient descent method, momentum\(^ {29} \) is introduced over consecutive iterations of updating \( w \). Updating of \( w \) is described by the following equation:

\[
w^{(t+1)} = w^{(t)} - \varepsilon \nabla E + \mu \Delta w^{(t-1)} \tag{7}
\]

Here, \( \mu \) is a hyperparameter that controls the ratio of momentum and is often set in the range of 0.5–0.9. To sufficiently learn the relationships between the training data and to prevent performance degradation due to the overfitting, dropout\(^ {29} \) is applied.

Here, overfitting is an excessive adaptation to the noise included in the training data. The key idea of dropout is to randomly drop units with probability \( p \), along with their connections, from the neural network during training. It aims to forcibly reduce the degree of freedom of the network and to prevent overfitting. To develop an ideal ANN model, it is important to avoid overlearning and to learn general relationships between the input and output data\(^ {26} \).

(3) Overview of deep learning

Deep learning is a type of ANN that uses multiple hidden layers, and various methodologies have been proposed\(^ {26} \). One of the advantages of deep learning is its high expressive ability. In conventional learning using a single-hidden-layer ANN, if the input data increase, the number of unknown parameters to be optimized becomes enormous, which makes training difficult. Therefore, it is necessary to reduce the number of parameters as much as possible by narrowing down the number of input datasets by selecting important input data for the target network output\(^ {14, 15, 16} \). However, there are various problems in the data selection process, and it is not easy to narrow down the truly necessary data. On the other hand, deep learning exhibits the ability to automatically narrow down important information in the learning process by inputting considerable learning data\(^ {30} \).

(4) Procedure of deep learning

In conventional ANN, \( w \) is randomly initialized and then optimized by the gradient descent method. In such a procedure, the so-called “vanishing gradient problem” is known in deep networks, so learning
is difficult. In this paper, we use a stacked autoencoder, which is a commonly used method in deep learning. By introducing pre-training, we attempt to solve the vanishing gradient problem. A schematic image of the pre-training conducted by the autoencoder is shown in Fig. 4. First, the deep ANN is divided into an autoencoder that is folded at each layer. Each autoencoder optimizes (train) the network weight to equalize the input $x$ and output $\hat{x}$.

In the example shown in Fig. 4, training is first conducted using the autoencoder folded in the first and second layers (autoencoder 1), and then, $w$ is determined. The $w$ between layers 1 and 2 is used as the initial value of the deep ANN. Next, the output from the hidden layer of autoencoder 1 is regarded as the training data, and training of the autoencoder folded in layers 2 and 3 (autoencoder 2) is conducted. The $w$ of the obtained autoencoder 2 is used as the initial value between the second and third layers of the deep ANN. By repeating the same procedure, pre-training for networks with deeper layers can be conducted. After pre-training, $w$ is optimized by the gradient descent method as in the conventional ANN method. By the above procedure, learning progresses more effectively than that using the conventional method.

In addition, a more accurate learning result has been reported when a denoising autoencoder, an extended version of an autoencoder, was employed. When using the denoising autoencoder, noise is added to the learning data as follows:

$$\tilde{x} = x + \delta x$$

Here, $\delta$ is random noise following Gaussian distribution with mean 0. After training by the autoencoder, the network is expected to have the ability to not only reproduce the input but also remove the noise of the input data.

### 4. STUDY BASIN AND TARGET FLOOD OF THE HYBRID RIVER STAGE PREDICTION MODEL

(1) Overview of the study basin and target flood data

The study area is the Hiwatashi station catchment in Ooyodo river basin, in the Kyushu Island, Japan, as shown in Fig. 5. The basin area is 861 km$^2$ and the length of the mainstream is 52 km. Fourteen rain-gauge stations and five river-stage gauging stations are installed in this basin. There is no big dam and other flood control facilities in the upstream, and many river-stage stations and rain gauges are installed, so this basin is suitable for the study. The observed river-stage and rainfall data are obtained from the Water Information System database of the Ministry of Land, Infrastructure, Transport and Tourism in Japan. The target floods for the model development are 19 out of 24 flood events that exceed the Evacuation Warning Water Level (6.0 m) during 1990–2014. Five events that have considerable missing data are excluded. One flood period is defined as 121 h, from 72 h before to 48 h later around the flood peak. Consequently, we use 121 h...
×19 floods = 2299 datasets. Among the 19 target floods, the top four events (1990, 1993, 2004, and 2005), which exceed the Hazardous Water Level (9.2 m), are selected as the validation events. Among the four floods, the 2005 flood is the largest in terms of both water level and total rainfall.

(2) Correlation analysis between change in river stage of prediction point and each station dataset
A correlation analysis is conducted to confirm the water-level change at the predicted point. The target variable of the correlation analysis is the water-level change per hour, \( \Delta H \), at the predicted point. The explanatory variables are the water-level change at the upstream site and the hourly rainfall \( R \). Figures 6a and 6b show the calculated correlation coefficients. As for the water-level change, the closer the distance to Hiwatashi station, the higher is the correlation and the smaller is the time lag. As for the rainfall, each station shows similar behavior, but Takeshita station has the highest correlation. As the correlation between rainfall and water-level change is highest, with a time lag of 1 to 2 h, the flood arrival time is about 1 to 2 h.

(3) Correlation analysis between change in river stage of prediction point and basin storage volume
To confirm the strength of correlation between \( \Delta H \) at the predicted point and the change in the basin storage volume, a correlation analysis is conducted with \( \Delta H \) as the target variable and the storage volume change \( \Delta H \) as the explanatory variable. Figure 6c shows a comparison of the correlation coefficients for Takeshita site, which shows the strongest correlation.

According to Fig. 6c, the highest correlation coefficient is obtained for a time lag of 2 h. The correlation coefficient for hourly rainfall is 0.642. The correlation coefficient for changes in storage volume is 0.718 when lag time \( T_L = 0 \) and 0.732 when lag time \( T_L = 1 \). The correlation with the water-level change is higher for the storage volume change than for the hourly rainfall. Further, the correlation of these factors with the water level is higher at \( T_L = 1 \).

5. DEVELOPMENT OF THE HYBRID RIVER-_STAGE PREDICTION MODEL

(1) Composition of the hybrid model
The hybrid model comprises a combination of an ANN model and a runoff model. Any runoff model can be applied to the hybrid model. Therefore, the accuracy can be improved by hybridizing the existing flood-forecasting system. Although it is possible to construct a hybrid model using a simple conceptual model, the prediction accuracy of the hybrid model can be improved by using a runoff model with higher prediction accuracy. In this study, we apply a runoff model that is composed of the distributed runoff model with the data assimilation technique and a particle filter. As an ANN model, a deep ANN model based on deep learning is applied.

(2) Development of flowrate prediction model with distributed runoff model and particle filter
a) Overview of the distributed model
As shown in Fig. 1, in the hybrid water-level prediction model, the predicted flowrate based on the physically based model is added to the input layer of the ANN water-level prediction model. The distributed runoff model used in this study comprises a one-dimensional unsaturated infiltration model, a saturated subsurface flow model, a surface flow model, and a one-dimensional shallow water flow model.
Table 2 Main setting parameters of the distributed model.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Setting value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertical hydraulic conductivity, Upstream*</td>
<td>$3.0 \times 10^{-2}$ (cm/s)</td>
</tr>
<tr>
<td>Vertical hydraulic conductivity, Middle basin*</td>
<td>$5.0 \times 10^{-2}$ (cm/s)</td>
</tr>
<tr>
<td>Vertical hydraulic conductivity, Downstream*</td>
<td>$1.0 \times 10^{-1}$ (cm/s)</td>
</tr>
<tr>
<td>Thickness of surface soil (m)</td>
<td>4.0</td>
</tr>
<tr>
<td>Saturated volume water content (porosity)</td>
<td>0.6</td>
</tr>
<tr>
<td>Residual volume water content</td>
<td>0.1</td>
</tr>
<tr>
<td>Equivalent roughness coefficient (s/m$^{1/3}$)</td>
<td>1.0</td>
</tr>
</tbody>
</table>

*Hydraulic conductivity of urban area was 1/1000

Table 3 Setting condition of particle filter.

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective of correlation</td>
<td>State quantity of model</td>
</tr>
<tr>
<td>Number of particles</td>
<td>96</td>
</tr>
<tr>
<td>Estimation error of observed water level</td>
<td>10%</td>
</tr>
<tr>
<td>Upper and lower limits of correction factor</td>
<td>0.995–1.01</td>
</tr>
<tr>
<td>Resampling noise</td>
<td>Normal distribution with 0.3% variance</td>
</tr>
</tbody>
</table>

Table 4 Input and output of the hybrid model for $t$ hours prediction.

<table>
<thead>
<tr>
<th>Type</th>
<th>Number of stations</th>
<th>Time</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>River stage</td>
<td>1</td>
<td>-1, 0</td>
<td>Change in river stage at Hiwataishi in $t$ hours.</td>
</tr>
<tr>
<td>Hourly change in river stage</td>
<td>5</td>
<td>-2, -1, 0</td>
<td></td>
</tr>
<tr>
<td>Hourly change in storage (rainfall – discharge)</td>
<td>14</td>
<td>$t-5, t-4, t-3, t-2, t-1$</td>
<td></td>
</tr>
</tbody>
</table>

Figure 7 shows the model’s conceptual diagram. The topographic shape is modeled using an unstructured grid, which is divided such that the river channel is composed of grid edges.

b) Development of the distributed model

The unstructured grid used in the distributed model has 4000 nodes, 7687 elements, and 4 vertical mesh divisions. The average grid size is 500 m on one side of the triangle. For elevation data, 5- and 10-m digital elevation maps from Fundamental Geospatial Data$^{34}$ are used. For the land use data, mesh data of National Land Numerical Information$^{35}$ are used, which are reflected in the setting of hydraulic conductivity. Using the developed distributed model, the reproduction calculation is conducted for the four validation floods and the model parameters are calibrated (Table 2). Forcing rainfall data are the hourly rain gauge data, and they are obtained for each node of the mesh by using the Thiessen polygon method. Details of the study are described in a previous study$^{19}$.

c) Application of data assimilation by particle filter

To reduce the prediction error by reflecting the real-time observation data in the model (feedback), a particle filter$^{6}, 7, 8, 10, 11$, which has been intensively studied in recent years, is used.

In the process of a particle filter, many calculation models are executed in parallel and the degree of fit (likelihood) between the calculation result and the observed value is evaluated. In this way, calculation proceeds while selecting and duplicating (resampling) models with appropriate calculation conditions and state quantities. Each calculation model is called a particle, and each particle has different calculation conditions (state quantities, parameters, etc.).

Table 3 shows the calculation condition setting in this study. Only the model state quantities (water saturation of unsaturated infiltration model, water level, and flowrate of unsteady flow model) are corrected by filtering. At each hour, the likelihood of each particle is calculated from the observed water level at the current time and the predicted water level, one-hour prediction water level before 1 h. Next, a particle filter is applied to select and duplicate particles with high reproducibility to the observed values. The distributed model parameters are treated as constant values, as shown in Table 2.

As the number of particles increases, the calculation accuracy becomes stable but the calculation cost increases. In previous studies$^{6}, 7, 8, 11$, the number of particles required to obtain sufficient performance has been reported to be 200–1000. However, it may vary depending on the model characteristics, uncertainty of the observation data, and condition setting of particles.

In this study, performance reaches a peak at 60 particles by trial and error. Considering the possibility of accuracy improvement, the final setting is considered as 96 particles. In this study, a fully calibrated and physically well-modeled distributed model is used. Therefore, it can be considered that the calculation uncertainty is small and stable performance is obtained with fewer particles, as compared to previous studies, which used a conceptual model.

The estimated error of the observed water level is used to evaluate the likelihood of each particle with the measured value. Meanwhile, if only the same
particles are used in the resampling process, the diversity of computation will be lost and filtering will not work. To maintain particle diversity, noise is added to the state quantity, i.e., soil water saturation, of each particle in the resampling process. The resampling noise is given as a magnification of the water saturation $S_w$ of the subsurface layer grid of thickness 1.0 m. When $S_w$ exceeds 1.0, excess water is supplied to the surface water. The noise magnitude and upper and lower limits are set by trial and error based on calculation stability and filtering performance. If future rainfall data are required, the observed data are used in place of the predicted rainfall.

(3) Development of deep ANN model for hybrid model

We use a four-layer network comprising an input layer, two hidden layers, and an output layer. Networks with three or more hidden layers may also be promising but have been excluded from this paper. The calculation program is coded using C ++ and a shell script without using the existing library.

The input-and-output combination is shown in Table 4. The prediction time was one to six hours, using which six ANN models are developed. For example, the input data for the three-hour prediction network are as follows: River water level of Hiwatashi (time lag is 0 to 1 h), hourly change in river water level of five stations (time lag is 0 to 2 h), and hourly change in storage (rainfall – discharge) calculated at 14 rain gauges (time lag is -2 to 2 h, namely, containing two future times).

Based on the concentration time studied by the correlation analysis in chapter 4, the input data are set up to include as much data as possible about location and time series that can affect the river-stage prediction. Thus, the input layer of one study dataset comprises 87 data items.

In a conventional ANN model, due to learning-ability limitations, it is common to narrow down the important input data and simplify the network as much as possible\textsuperscript{14, 15, 16}. However, to improve the accuracy of water-level prediction, it is important to reflect the distribution and time series of the rainfall as much as possible, and thus it is desirable to use more input data. In this study, by using deep learning, as many as 87 input datasets could be effectively used in training and the model’s prediction accuracy could be improved\textsuperscript{21}.

If future rainfall data are required, the observed data are used to approximate the predicted rainfall. As the future river stage is unknown, the input river-stage dataset is formed up to the current time. The use of the predicted rainfall will be studied in the future.

(4) Condition setting of the deep-learning model

The basic condition setting of training is shown in Table 5. Other setting conditions, including number of training iterations, number of neurons in each layer, and dropout rate, are separately examined through case studies. The size of a mini-batch in Table 5 is the number of training datasets required to evaluate the error function in the stochastic gradient method. Although there is no fixed way of determining the size, the number of input datasets of the mini-batch has not been established, and is often set 10 to 100 from the viewpoints of learning efficiency and computational cost\textsuperscript{26}. In this study, the mini-batch size is set to 100. The method of selecting data samples in the mini-batch is as follows: All data are assigned a number in order of the water level, and

<table>
<thead>
<tr>
<th>Case No</th>
<th>Number of training iterations</th>
<th>Number of neurons</th>
<th>Dropout rate</th>
<th>Noise adding</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>300</td>
<td>10-5</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1000</td>
<td>3000</td>
<td>87</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3000</td>
<td>10000</td>
<td>87</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>10000</td>
<td></td>
<td>87</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case No</th>
<th>Number of training iterations</th>
<th>Number of neurons</th>
<th>Dropout rate</th>
<th>Noise adding</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>300</td>
<td>10-5</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1000</td>
<td>3000</td>
<td>87</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3000</td>
<td>10000</td>
<td>87</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>10000</td>
<td></td>
<td>87</td>
</tr>
</tbody>
</table>

Table 5 Learning data and basic condition setting of ANN model.

<table>
<thead>
<tr>
<th>Type</th>
<th>Setting condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>training data</td>
<td>24 floods during 1990 to 2014</td>
</tr>
<tr>
<td>type of data</td>
<td>River stage, hourly change in river stage, hourly change in storage</td>
</tr>
<tr>
<td>number of datasets</td>
<td>2299</td>
</tr>
<tr>
<td>mini-batch size</td>
<td>100</td>
</tr>
<tr>
<td>prediction time</td>
<td>1 to 6 h</td>
</tr>
<tr>
<td>momentum</td>
<td>0.5</td>
</tr>
<tr>
<td>learning rate</td>
<td>initially 0.1. Automatically adjusted by AdaGrad</td>
</tr>
</tbody>
</table>

Table 6 Case study for condition setting of training.
100 datasets are extracted in this order. For example, the first mini-batch comprises the 1st to 100th dataset and the second mini-batch comprises the 101st to 200th dataset. Then, a mini-batch is similarly created with 100 datasets. In case of the last dataset in the last dataset in the iterative calculation process, we return to the first dataset and extract it, so that the total number becomes 100.

Statistical biases of the training data reduce the efficiency of learning. In this study, the input data are normalized before training, as follows:

$$x_{n_i} \leftarrow \frac{x_i - \bar{x}}{\sigma}$$  \hspace{1cm} (9)

Here, $\bar{x}$ is the average data, $\sigma$ is the standard deviation, $x_i$ is the regularized data, and subscript $i$ expresses the sample set. At the end of the calculation process, the output is converted inversely to obtain the results.

As there is considerable room for improvement in training conditions and methods, it is necessary to accumulate knowledge in the future.

6. CASE STUDY OF THE HYBRID WATER-LEVEL PREDICTION MODEL

(1) Optimization of network by case study

It is unclear how various learning settings affect prediction accuracy. Here, the study cases are set as shown in Table 6 for the number of training iterations, number of neurons in each layer, dropout rate ($p$), and noise addition to the learning data, and the case with the smallest error is adopted. In this study, different ANN models are used for 1–6 h prediction, but the settings in Table 6 are common. If the number of training iterations is too small, network optimization will be insufficient, and if it is too large, overfitting will be a problem, both of which will decrease the accuracy. To set an appropriate number of learning, we set four cases between 300 and 10000.

The number of neurons in each layer is set as follows: The number of neurons in the input layer is fixed at 87, because it is equal to the number of input datasets. As for the hidden layer, if the number of neurons is too small, the model’s ability to express the relationship between the input and output data will be insufficient, and if the number is too large, overfitting will become a problem. However, according to a previous study, it is important to set a sufficient number of neuron when a mechanism to suppress overfitting is introduced\(^\text{37}\). In this study, the number of neurons is set in four stages, as shown in Table 6.

The dropout rate $p$ is set to three cases: 0.1, 0.3, and 0.5. Noise addition is set to two cases: “with” and “without.” When adding noise, normal random numbers with an average of 0 and variance of 3% are used, and noise is added to the training data at the time of pre-training by the denoising autoencoder.

From the above, 96 cases are set by combining 4 cases for the number of learning iterations, 4 cases for the number of neurons in each layer, 3 cases for the dropout rate, and 2 cases for noise addition. In Table 6, for example, if the number of learning iterations is 300, the number of neurons in hidden layer 1 is 10, and that in hidden layer 2 is 5; then, the case number is 11. In addition, there are six subcases depending on the dropout rate and whether noise is added.

To evaluate the prediction accuracy, a leave-one-subject-out cross-validation was conducted for the four floods. In one study, for example, the validation dataset was one of the target flood datasets out of the 19 target floods and the learning data comprised the remaining 18 flood datasets, excluding the validation flood. By repeating the same procedure for each validation flood, the accuracy of the prediction model is evaluated. Considering the importance of evacuation judgments during floods, the evaluation period starts from the noticeable increase in water level and continues up to six hours after the peak is reached. The root mean square error (RMSE) is used as the accuracy index.

The average of RMSE for one to six hours prediction (four floods average) of the case study is

---

**Table 7** RMSEs (m) of observed and predicted river water levels by each model and average of four floods.

<table>
<thead>
<tr>
<th>Model</th>
<th>Prediction time (h)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed model &amp; particle filter</td>
<td></td>
<td>0.52</td>
<td>0.62</td>
<td>0.70</td>
<td>0.77</td>
<td>0.80</td>
<td>0.82</td>
<td>你缺失数据</td>
</tr>
<tr>
<td>Deep ANN</td>
<td></td>
<td>0.12</td>
<td>0.19</td>
<td>0.27</td>
<td>0.39</td>
<td>0.54</td>
<td>0.65</td>
<td>你缺失数据</td>
</tr>
<tr>
<td>Hybrid</td>
<td></td>
<td>0.16</td>
<td>0.20</td>
<td>0.27</td>
<td>0.33</td>
<td>0.43</td>
<td>0.52</td>
<td>你缺失数据</td>
</tr>
<tr>
<td>Distributed model</td>
<td></td>
<td>0.96*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSE at reproduction calculation for distributed model</td>
</tr>
</tbody>
</table>

*RMSE at reproduction calculation for distributed model

---

**Table 6**

<table>
<thead>
<tr>
<th>Case number(caseXYの X is the training iteration number, Y is the number of neurons.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE (m)</td>
</tr>
<tr>
<td>Case 1</td>
</tr>
<tr>
<td>Case 2</td>
</tr>
<tr>
<td>Case 3</td>
</tr>
<tr>
<td>Case 4</td>
</tr>
<tr>
<td>Case 5</td>
</tr>
<tr>
<td>Case 6</td>
</tr>
</tbody>
</table>

---

**Fig. 8** RMSEs of hybrid model for each case (average of four floods for one to six hours prediction).
shown in Fig. 8. For convenience, the case numbers in the figure correspond to those in Table 6.

As a result of the comparison, in case 44, having 10000 training iterations, the number of neurons in each layer is 87, 60, 30, and 1, and the dropout rate is 0.1 without noise addition. This is the most accurate result. Therefore, case 44 was set as the final optimum setting.

The effect of each setting on the prediction accuracy is as follows: Regarding the number of neurons in each layer, there is no clear difference when the learning iterations are 1000 or less. However, when the learning iterations are 3000 or more, the accuracy tends to increase with the number of neurons. Looking at the case with the largest number of neurons, there is no clear difference between the number of learning iterations 3000 and 10000, so it can be considered that learning has almost converged at about 3000 times.

Thus, the number of learning and the number of elements influence each other, and each must be set to a sufficiently large value. However, it is confirmed that the performance reaches a limit at a certain stage.

As for the dropout rate (p), 0.1 obtains the best result, followed by 0.3 and then 0.5. Although the recommended value of the dropout rate was 0.2–0.5 in a previous study, in this study, a smaller network and different conditions are used.

There is almost no difference between the case of with and without noise addition. As for the dropout rate and noise addition, the tendencies differ from those observed previously. Thus, further studies need to be conducted under various conditions.

(2) Case study of the hybrid model

A hybrid model prediction calculation is conducted using a deep ANN optimized by a case study. The accuracy of the predicted water level is evaluated by the leave-one-out cross-validation similarly to that in the case study. At the beginning of training deep ANN, the weighting factors are initialized by a random number. Therefore, even if the data and various settings are the same, the learning result, i.e., the optimized weighting factor, varies with each calculation. Generally, the prediction accuracy can be improved by arithmetically averaging multiple prediction results using multiple ANNs. This is called model averaging and is an effective method not only for ANN but for other machine learning methods as well. In this study, 10 hybrid models are constructed, and the final predicted water level is calculated by model averaging. A total of 60 models are used to calculate an average of 10 models for each prediction time of 1–6 h. Furthermore, as the leave-one-subject-out cross-validation is applied to the four floods, 240 models are used to evaluate the prediction accuracy. In the actual operation stage, it is only necessary to prepare the model by the number of prediction time multiplied by the number of models used for model averaging.

(3) Result of the hybrid model

For each of the four floods, the result of prediction is shown in Fig. 9 and the prediction error is shown in Fig. 10. The prediction error of the four floods’ average is shown in Table 7. Here, the deep-learning model is constructed and optimized separately from the hybrid model. The input data of deep-learning model include rainfall and water level. Although the accuracy of the hybrid model is slightly different from case to case, there is significant improvement in the 2005 flood. In the four-flood average, the accuracy is slightly decreased in the 1–2 h prediction, but significantly improved in the 4–6 h prediction, as shown in Table 7.

Figure 11 shows a comparison between the hybrid model and deep-learning model in the 2005 flood, which was the largest in the study period. The vertical axis on the right side of Fig. 11 is consistent with the input value of the model, the upper figure shows the change in storage volume (rainfall minus runoff depth), and the lower figure shows the rainfall. Around the peak of the 2005 flood, the rainfall is 25 mm/h, while the change in storage is relatively small at 13 mm/h. The prediction results of the deep-learning model with rainfall as an input are overestimated, and the result of the hybrid model with changes in storage as an input reproduce the observation well. The relation between the input layer (rainfall and change in storage) and the magnitude of prediction result is consistent with an intuitive understanding of the rainfall-runoff behavior, indicating the correctness of the developed model. In addition, even for the 2005 flood, which exceeds the training flood, the water level is predicted appropriately with the hybrid model by using the change in storage. This is because the generality of the rainfall-runoff mechanism can be learned more appropriately by adding the change in storage to the input layer of the deep-learning model. Although the problem of the statistical model, including the conventional ANN model, is insufficient accuracy of the large flood that exceeds the training event, the proposed method can improve the accuracy. Meanwhile, for the 1990 flood, the hybrid model is less accurate than the deep-learning model. Such issues need to be analyzed in the future.

Compared to the distributed model, the accuracy of the hybrid model exceeds that for all floods at all prediction times. In addition, the state updating of the distributed model by the particle filter may not follow the rising limb of the water level because there is
Fig. 9 Comparison of river-stage prediction results of each method.

Fig. 10 Comparison of RMSEs of observed and predicted river stages by each method for 1 to 6 h prediction.
a lag time between the basin to the river channel and that to the prediction point. The accuracy of the rising limb is a problem of the prediction method with distributed model and state updating.

(4) Calculation time of the hybrid model

As an applicability verification of the real-time prediction using the hybrid model, computation time is tested. The computer used for testing comprises two CPUs with Intel® Xeon®X5690 (6-Core 3.46GHz) and 96GB (DDR3) memory. The compiler is Intel Composer XE 2011 Linux Edition (C++). Computation time for six hours prediction by the particle filter, where 96 particles are allocated to 12 cores by MPI (openmpi-1.4.3), is 74.0 s. The prediction calculation of the deep-learning model is 0.1 s. Core by MPI (openmpi-1.4.3), is 74.0 s. The pre-

7. CONCLUSION

In this study, we developed a hybrid water-level prediction model, called the ANN model, which incorporated the runoff model. The developed model was applied to the Hiwatashi station, Ooyodo river, and its prediction accuracy was confirmed. In previous major floods, the hybrid model showed higher prediction accuracy than the original deep learning and distributed models.

In particular, the hybrid model had a significant effect of improving accuracy for the largest flood in this study period. The prediction accuracy for an inexperienced magnitude of the flood has been regarded as a problem of statistical models, but the proposed method showed the possibility of improvement. As for applicability to real-time systems in the future, the calculation time of the hybrid model was about 74 s in the 6-h prediction, so it exhibited enough applicability from the viewpoint of calculation speed. Most of the calculation time was occupied by the computation of the distributed model and the calculation cost of the ANN was negligible. Therefore, a hybrid model can be implemented using the existing flood forecasting based on the runoff model. In the calculation procedure of the hybrid model, the result of the runoff model was used as the input value of the ANN, so each model could be calculated independently and easily incorporated into the existing system. As future issues for operation in a real system, it is important to confirm the prediction accuracy in the case of combination of lumped models and various other runoff models. It is also important to confirm the prediction accuracy in case of using the predicted rainfall.

In addition, accuracy verification under the following various conditions is required.

- More complex basin shapes
- Deformation of riverbed
- Complex multimodal rainfall-runoff waveform
- Small basins and urban rivers

REFERENCES

9) Moradkhani, H., Sorooshian, S., Gupta, H. V. and Houser, P. R.: Dual state-parameter estimation of hydrological models using ensemble Kalman filter, Advances in Water


(Received November 28, 2019)
(accepted January 15, 2020)