Journal of Reliable Intelligent Environments (2024) 10:177–195
https://doi.org/10.1007/s40860-023-00214-8

ORIGINAL ARTICLE

Tsunami tide prediction in shallow water using recurrent neural networks: model implementation in the Indonesia Tsunami Early Warning System

Willy Dharmawan[1,2] · Mery Diana[1,3] · Beti Tuntari[1] · I. Made Astawa[1] · Sasono Rahardjo[1] · Hidetaka Nambo[2]

Received: 10 June 2023 / Accepted: 7 October 2023 / Published online: 16 November 2023

Abstract
Near-field tides prediction for tsunami detection in the coastal area is a significant problem of the cable-based tsunami meter system in north Sipora, Indonesia. The problem is caused by its shallow water condition and the unavailability of an applicable model or research for tsunami detection in this area. The problem foundation of shallow water area is its ambient noise level-dependent property that requires preprocessing to improve its feature representation. Moreover, because this shallow water is close to the land area, we must consider a model that can accommodate low prediction time for a Tsunami Early Warning System. Therefore, we propose a recurrent neural network (RNN) model because of its reliable performance for time series forecasting. Our report evaluates variants of the RNN model (the vanilla RNN, LSTM and GRU models) in tides prediction and z-score analysis for tsunami identification. The GRU model overwhelms the other two variants in error scores and time processed (training and prediction). It can achieve median error score distribution of 7.8×10^{-5} on the L1000-P250 configuration with time prediction under 0.1 s. This lower-time prediction is necessary to ensure the early warning system is going well. Moreover, the GRU model can identify all synthetic tsunami tide spikes (compared with the ground truth result) from magnitude 7.2–8.2 by applying a z-score on the GRU's prediction.

Keywords Recurrent neural network · Deep neural network · Shallow water body · Tides prediction · Tsunami early warning system

1 Introduction

There is already quite much proof of how devastating the tsunami impacted the land of Indonesia, which brought in significant loss of material and human lives. Take one of the recent tsunami events as an example [1], tsunami Palu Dongala (2018) records a total loss of 20.89 trillion rupiahs and 4340 people died. tsunami Palu Dongala (2018) records a total loss of 20.89 trillion rupiahs, and 4340 people died. Moreover, the impact will be worsened because of the stop of the economic growth in the post-disaster.

One of the attempts to reduce the number of tsunami strike victims, Indonesia has developed a Tsunami Early Warning System (TEWS), which started in 2005 [2] (post-Aceh's tsunami). Recently, predicated on the president's instruction, article 5 number 93 2019, as part of strengthening disaster mitigation, the Agency for the Assessment and Applica-

Mery Diana, Beti Tuntari, I. Made Astawa, Sasono Rahardjo and Hidetaka Nambo contributed equally to this work.

✉ Willy Dharmawan
will003@brin.go.id

Mery Diana
mery003@brin.go.id

Beti Tuntari
beti001@brin.go.id

I. Made Astawa
imad001@brin.go.id

Sasono Rahardjo
sasono.rahardjo@brin.go.id

Hidetaka Nambo
nambo@blitz.ec.t.kanazawa-u.ac.jp

[1] Centre of Electronics, BRIN, Puspiptek, Serpong, South Tangerang 15314, Indonesia

[2] Electrical Engineering and Computer Science, Kanazawa University, Kakuma, Kanazawa, Ishikawa 9201192, Japan

[3] Computer Science and Electrical Engineering, Kumamoto University, Kurokami City, Kumamoto 8600862, Japan

tion of Technology or BPPT Indonesia developed Cable Based Tsunami-meter (CBT) [3]. This system will adopt the SMART concept, Scientific Monitoring, And Reliable Telecommunication, which incorporates a monitoring function of the tsunami, earthquake, climate, ocean condition, and sea level with telecommunication capabilities.

Concerning the threat of the giant tsunami caused by megathrust earthquakes in the Mentawai island area [4], BPPT initiated the deployment of the CBT system in Sipora. The subduction zone of this area has a characteristic of a low water depth of about 80 m, which consider having considerable ambient noise [5, 6]. Multiple occurrences of bottom bounce path in the sound channel and uncertain seafloor properties, including sound speed, density, and attenuation, make continental shelves environment has significant external noise, which muddles up the data measurement retrieved from Bottom Pressure Recorder (BPR). This ambient noise level-dependent feature makes shallow water environments more challenging to analyze and model [5].

Meanwhile, tides prediction is a time series problem in which the output is the sequence prediction within some margin of error. The traditional modelings are mainly parametric based, such as AutoRegressive (AR) [7], exponential smoothing [8, 9] or structural time series model [10]. However, it has also been found that many of these real-time series modelings seem to follow nonlinear behavior [11] and are insufficient to represent their dynamics [11–13]. Therefore, another approach using a different mathematical representation of the nonlinearity present in the data is suggested to overcome this problem [11, 13, 14].

Notably, the emergence of artificial neural networks (ANN) adopting this approach have been widely used for the prediction of various complex system [15–17]. They can identify and learn the complicated nonlinear relationship between system variables, showing more accurate results than linear regression techniques [18].

Among these various techniques, recurrent neural network (RNN) can detect a pattern in the data sequence [19]. This ability differentiates from Feedforward Neural Networks, which pass information through the network without cycles. The RNN has cycles and transmits information back into itself, which extends Feedforward Networks to account for previous information. Despite this advantage, RNN suffers from vanishing or exploding gradient in long-term dependency [19]. This problem motivated the introduction of long–short-term memory units (LSTMs) [20] for handling the vanishing gradient problem. LSTM has become popular in time series forecasting [20]. Compared to deep Boltzmann machines, graph-structured recurrent neural networks, and convolutional neural networks, LSTM-NN-based deep learning performs better [21] for time series forecasting. It can extract robust patterns for an input feature space and effectively handle Multiple Input Multiple Output System (MIMO) systems in Deep Neural Networks (DNN). Moreover, the LSTM system can take nonlinear systems due to their specialized LSTM cell that performs better after learning. However, LSTM has some drawbacks related to its complicated unit and more data necessary to learn effectively [22]. Therefore Gated Recurrent Unit is proposed as a simpler hidden unit to compute and implement [22].

Nonetheless, recent research on time forecasting shows unforeseeable CNN [23–25] to solve time series problems. However, this problem is still limited to the classification as output, not time sequences which is the output of tides prediction. In addition, the wide range variability of the data set must also be experimented with for a solid model hyperparameter.

Based on all these studies, we select RNN as our model foundation. Some factors that support this are as follows:

- Tides prediction is a univariate time forecasting problem relevant to the efficacy of RNN [26, 27].
- The output of tides prediction is temporal-dependent sequence data. RNN is suitable for sequence learning from the features [28, 29].
- Though for the training phase, RNN still cannot take advantage of parallel computation in GPU [30], RNN can still achieve a closed real-time prediction in the inference time, which is around one second or less, depending on the GPU.

Thus, our work is considered novel since the unavailability of tides prediction model and study in shallow water areas for tsunami prediction purposes. Our contribution will be as follows:

- Combination of multistage preprocessing and RNN-based deep-neural network on tide data for solving tide prediction modeling in shallow water cases. This model is intended to get a better tide prediction and human interpretation.
- The suitable tides preprocessing for shallow water cases that can reduce the noise of the tides data and accommodate the neural networks input.
- Empirical insight of various RNN models, vanilla RNN, LSTM, and GRU, approach on a case of tsunami detection based on tide prediction.
- Experimentations on defining look-back and forward parameter scenarios on shallow water tides prediction models.
- Z-score analysis toward variability of the synthetic tsunami triggered by earth magnitude. This analysis evaluates the sensitivity of the current model.

Finally, solving this near-field tsunami forecasting in the coastal area is urgently required to reduce casualties. Indone-

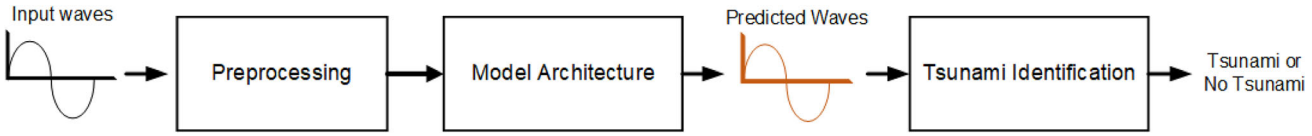


Fig. 1 General block diagram of designed system

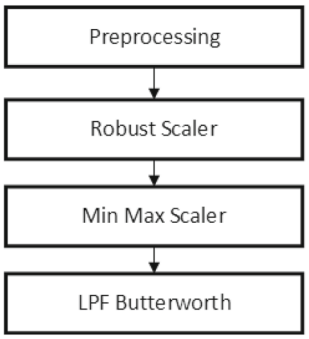


Fig. 2 Preprocessing block diagram

sia experienced a tsunami caused by coastal volcano eruptions in 2018 [31]. Furthermore, Sumatra island's coastal region, especially Mentawai island, has considerable potential for megathrust earthquakes and landslides [32]. Therefore, the RNN tides prediction model proposed in this paper can become the required solution to mitigate this problem.

2 Related works

National oceanic and atmospheric administration (NOAA) developed the tsunami detection algorithm under the deep-ocean assessment and reporting of tsunamis (DART) project using cubic polynomial [33]. While in [34], Beltrami tried to find a more efficient alternative tsunami detection algorithm by proposing an artificial neural network (ANN). Both of these algorithms use the data from the bottom pressure recorder (BPR) as a sensor to collect the sea level in the deep sea. Based on the comparison [34], ANN methodology can predict tide and other regular patterns in the wave better than the DART.

Before [34], Barman et al. [35] utilized non-linear regression in ANN to calculate the estimation time arrival (ETA) for predicting the tsunami travel time in the Indian Ocean. The ANN model could perform the rapid computation for ETA. The model proved its robustness in developing a real-time tsunami warning system for the Indian Ocean.

These efficacies of ANN encourage data-driven forecasting tsunami [36]. Romano et al. [36] utilized spatial values of maximum tsunami heights and tsunami arrival times (snapshots) computed through the TUNAMI-N2-NUS model. They achieved good accuracy and near-instantaneous forecasting of the maximum tsunami heights and arrival times for the entire computational domain.

Another variant of ANN is also adapted to estimate tsunami inundation [37]. Fauzi and Mizutani applied two machine learning models, a convolutional neural network and a multilayer perceptron, for real-time tsunami inundation forecasting in the Nankai region of Japan. They experimented using the hypothetical future Nankai megathrust earthquake with Atashika and Owase Bays in Japan as the study cases. The results show that the proposed methods are high-speed (less than 1 s) and comparable with nonlinear forward modeling.

Besides tsunami mitigation, another natural disaster, such as an earthquake, is also predicted using ANN [38–40]. In the most recent [41], Kishore et al. used the LSTM to model the sequence of earthquakes. They used the trained model to predict the future trend of earthquakes and compared the LSTM with an ordinary Feed Forward Neural Network (FFNN) solution for the same problem. The result showed that the LSTM neural network was found to outperform the FFNN in the task of modeling the sequence of earthquakes.

Compared to all the previous work, our study regarding tsunami detection in shallow water case is considered a premiere. The challenging part of BPR data in shallow water areas is the muddled ambient noise, which requires signal processing to filter these out from the expected features. Various RNN networks are evaluated, serving as context learners that forecast the upcoming tides. Finally, the z-score will identify the tsunami spikes from the set of predicted tides.

3 Methodology

3.1 General algorithm design

This project defines the primary solution through a block diagram comprising preprocessing, training model architecture, and tsunami identification. Preprocessing sequences involve feature scaling, vector shape matching, and denoising. The training model architecture consists of the RNN stacked model and the dense layer, which map the features into a serial data prediction. Finally, the system will identify the tsunami from the prediction sequence of tides by smoothed z-score methodology, as shown in Fig. 1.

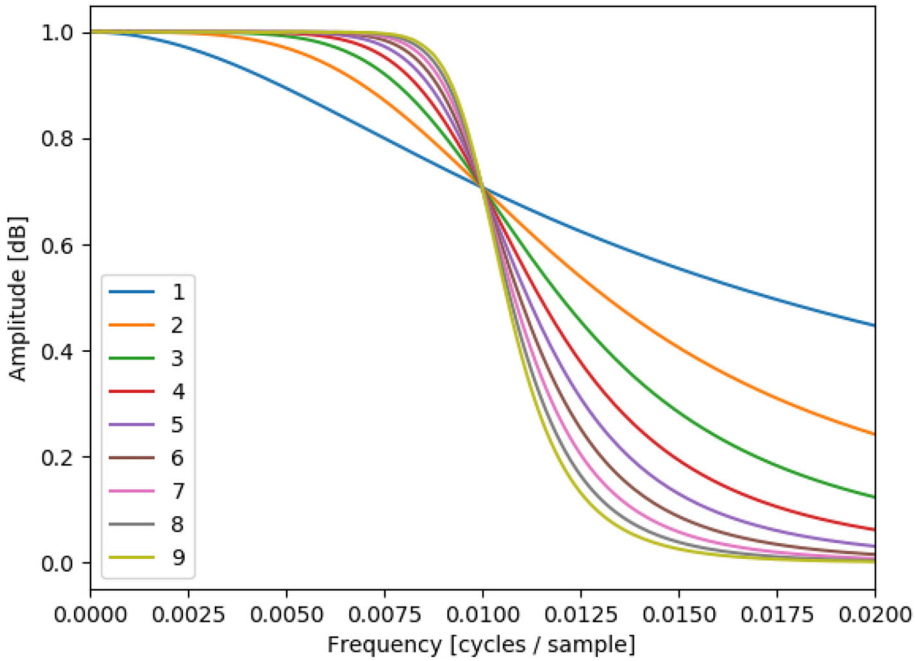


Fig. 3 Butterworth filter frequency response on 0.01 Hz cutoff frequency

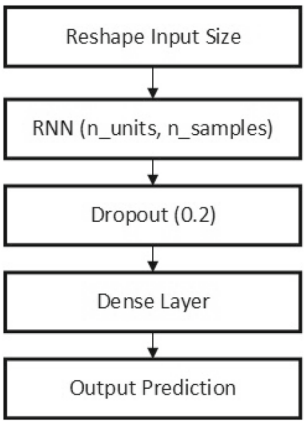


Fig. 4 Model network layers design

3.2 Preprocessing

The first part of the section (Fig. 2) is a scaler based on a percentile that will improve distribution data scaling. The process is unaffected by significant marginal outliers, which commonly occur in a noisy data environment. From this section onward, 0–1 normalization is required to match the RNN input layer. Eventually, the processing system applies a low-pass filter to reduce ambient noise. The LPF design parameter follows [42] Oceanographical Engineering Textbook allowing tsunami data to be captured with a frequency less than or equal to 0.01 Hz. The order of the filter is also set to 9 to reduce stopband ripple maximally, as shown in Fig. 3.

On the other hand, based on the general block diagram section (Fig. 1), we can write the system in pseudocode. Before utilizing the function, training data are fitted into the scaler to capture data traits (mean, variance, interquartile range, etc.). This trait can be saved into.bin format and loaded in the function.

Algorithm 1 Preprocessing

$data \Leftarrow tides$
$N \Leftarrow filter\_order$
$C \Leftarrow cutoff\_coefficient$
$trait_1 \Leftarrow robust\_trait$
$trait_2 \Leftarrow normal\_trait$
$robust\_data = trait_1.transform(tides)$
$normal\_data = trait_1.transform(robust\_data)$
$filter\_data = filter(N, C, normal\_data)$

3.3 Model architecture

The target of the design model (Fig. 4) is a vector of future predictions. This mechanism can be achieved by applying a stack of RNNs, followed by a dense layer. A drop-out layer shall be attached to the sequence to reduce overfitting cases.

Some variables define each of the functions. The input x needs to be in three-dimensional size, in which the vector should be reshaped into $(height \times weight \times 1)$. The n_units represents the number of hidden units denoting the number of dimensional output space while the $n\_samples$ symbolizes the number of data input that becomes the previous data context. Finally, the predicted output is an array with time sequence size mapped using a dense layer.

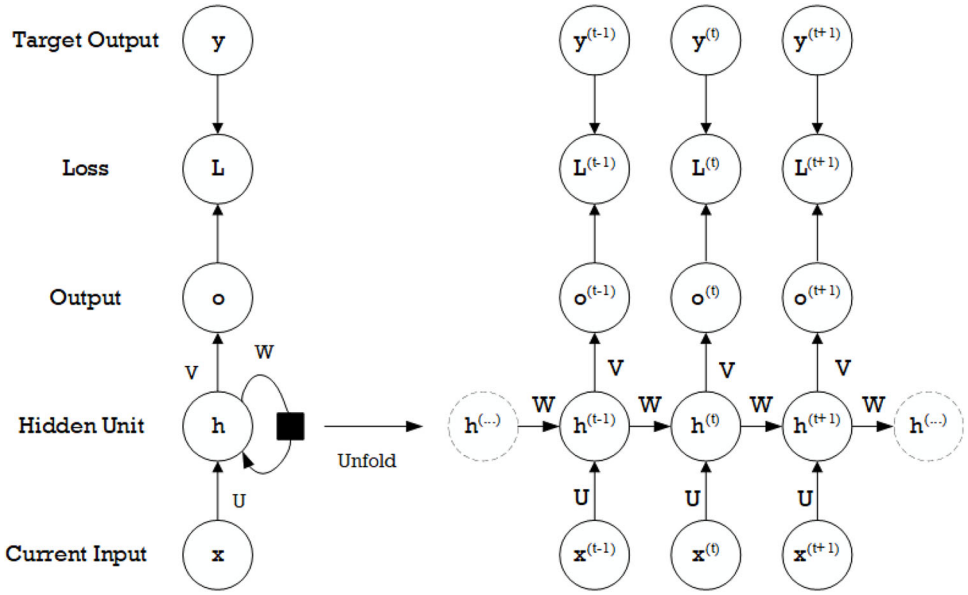


Fig. 5 Computational graph representation of RNN basic form, including training loss computation

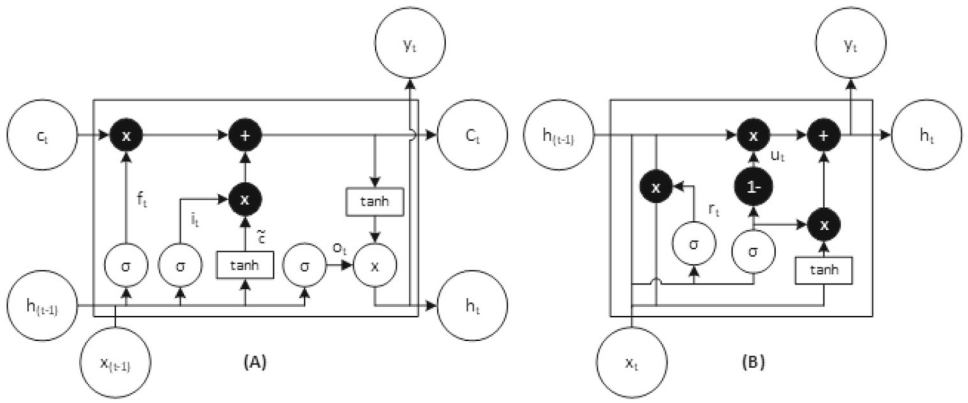


Fig. 6 General block diagram of **A** LSTM and **B** GRU

3.4 Model prediction algorithm

This part (Algorithm 2) defines the basic algorithm to train and test the model prediction. The windowing LSTM with look-back variation value becomes the core of the algorithm. The program's first segment defines the number of $look-backs$, the number of predictions and reshapes input x into three-dimensional input ($samples$, $timesteps$, $features$), and y into two-dimensional information. The $look-back$ parameter is the number of data points in prior timesteps, which become part of this project analysis. Finally, the model predicts the tide, then $data\_input$ and $predictions$ variables are updated.

Algorithm 2 Prediction

```
data_input ⇐ [number_of_lookback]
index ⇐ 0
model ⇐ load('model')
while true do
    index++
    if index < number_of_lookback then
        data_input.append(new_data)
    else
        prediction ⇐ model.predict(data_input)
        predictions.append(prediction)
        data_input.pop(0).append(new_data)
    end if
end while
```

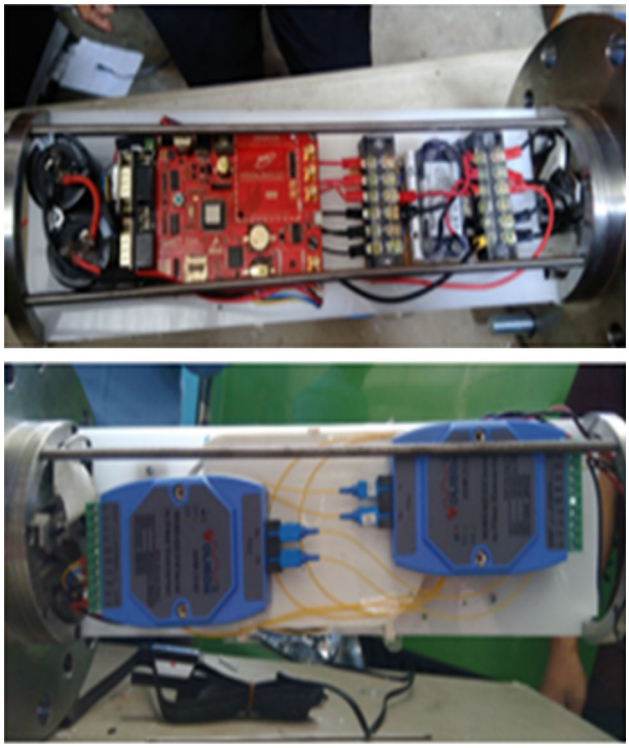


Fig. 7 Component of OBU Sipora

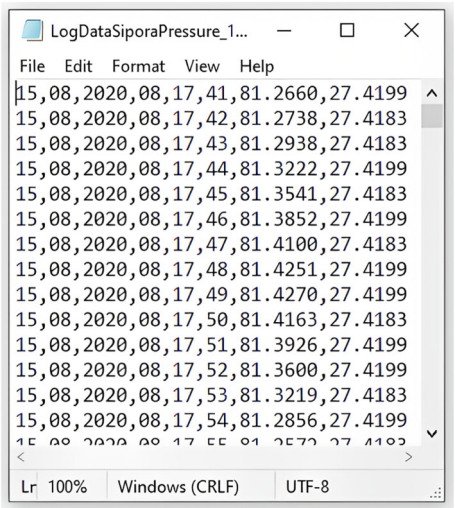


Fig. 8 Periodic tide data acquisition

4 Technical background

4.1 Recurrent neural network

In most of the literature, a hidden unit in RNN can be formulated as follows [43]:

$$h^{(t)} = f(h^{(t-1)}, x^{(t)}; \theta) \tag{1}$$

Referring to the (1) equation, h is a hidden unit function, $x^{(t)}$ is the current input, and θ is the parameter of the function f. This equation is recurrent because h at time t refers to the same definition at time $t-1$. There are several examples of the design pattern of RNN, yet to ease the exposition, we focus on the basic form of recurrent networks, hidden-to-hidden recurrent connection, which refers to Fig. 5 [44]

From Fig. 5, we can see that the idea of RNN is to pass or connect previous information to the present task. This process is beneficial because it needs sequence information to be processed, such as a video frame. However, in this "long-term dependencies" case, Hochreiter [45] and Bengio [46] found some fundamental reasons why it is difficult. Therefore, basic RNNs fail to learn "long-term dependencies". Nevertheless, the variants of RNN architecture called gated RNNs, including LSTM and Gated Recurrent Units (GRUs), are introduced to tackle this problem. Especially, LSTM, which was introduced by Hochreiter and Schmidhuber [20], has been quite popular nowadays, as many researchers use it because of the efficacy in many different applications [44].

LSTM introduces a new element called cell state c, which comprises the forget gate (f_t), input gate (u_t), and output gate (o_t). According to its name, forget gate determines whether the previous data is diminished. In contrast, the input gate evaluates the information to be carried over in the sequence, and the output gate decides the next hidden state value from the previous data. We can define each of the gates in the following equation:

$$u_t = \sigma(W_u h_{t-1} + I_u x_t + b_u) \tag{2}$$
$$f_t = \sigma(W_f h_{t-1} + I_f x_t + b_f) \tag{3}$$
$$o_t = \sigma(W_o h_{t-1} + I_o x_t + b_o) \tag{4}$$

Each of the formulae at the time step t, W_f, W_u, W_o, I_f, I_u and I_o are weight parameters on the corresponding gate, while variables, b_f, b_u and b_o, are *bias* alongside the gate. Thus, the cell candidate (c), current hidden state (h_t), and current cell state (c_t) can be formulated as below:

$$\tilde{c}_t = \tanh(W_c h_{t-1} + I_c x_t + b_c) \tag{5}$$
$$c_t = f_t \odot c_{t-1} + u_t \odot \tilde{c}_t \tag{6}$$
$$h_t = o_t \odot \tanh(c_t). \tag{7}$$
$$y_t = \sigma(W_y h_t + b_y) \tag{8}$$

where variables, W_c and I_c, represent weight parameters on the cell and variable b_c is bias alongside the cell.

In comparison with LSTM (shown in Fig. 6), GRU replaces the three's LSTM gates into two gates: the update z_t and reset r_t gates. The update gate helps the model control the new state's number from a copy of the previous state, while the reset gate intuitively controls how much past information to forget. The GRU unit is defined as the set of the equation

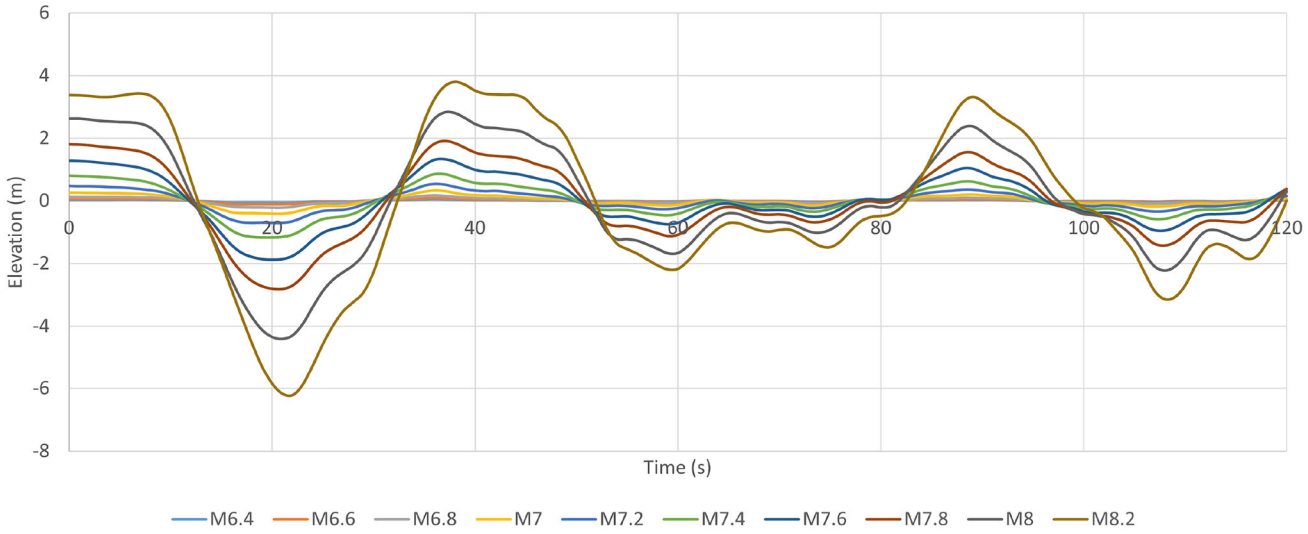


Fig. 9 Tsunami caused by an earthquake with the variability of magnitude

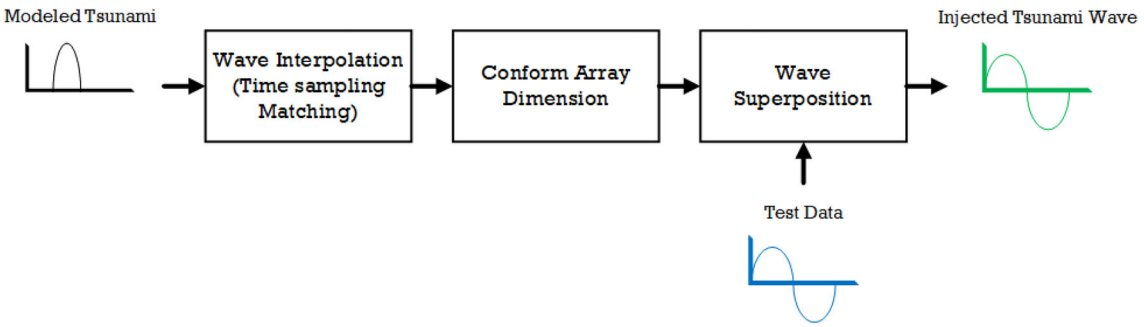
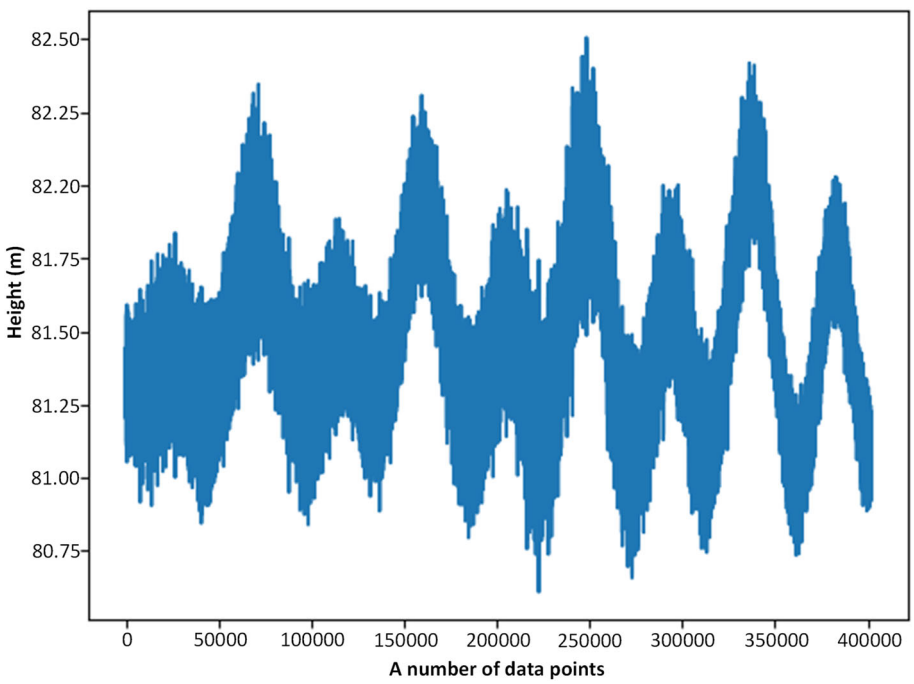


Fig. 10 Block diagram process of tsunami injection

Fig. 11 Training data set



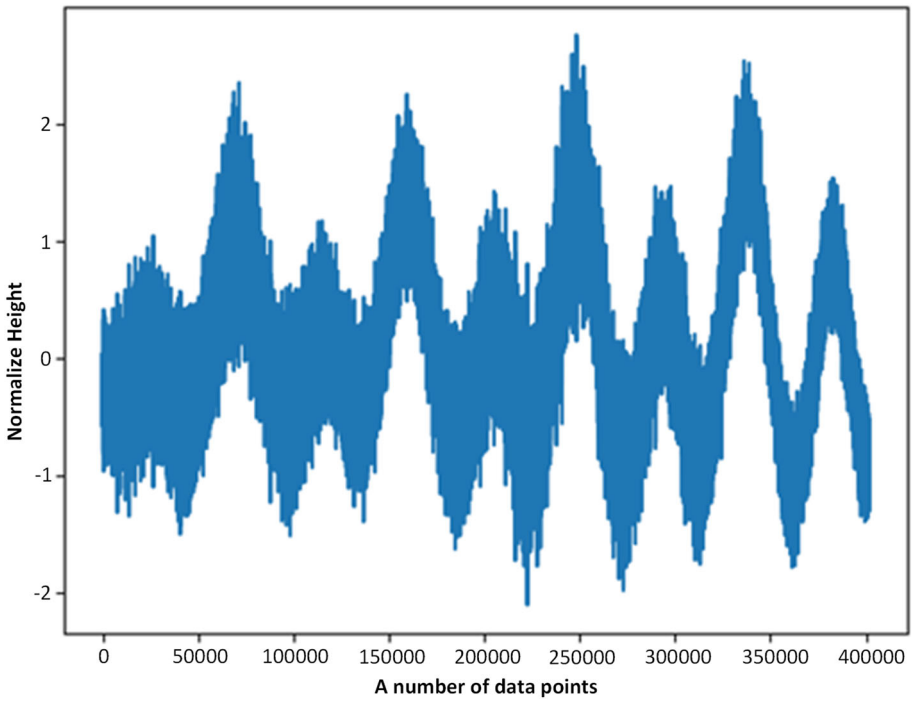


Fig. 12 Robust scaler input transformation

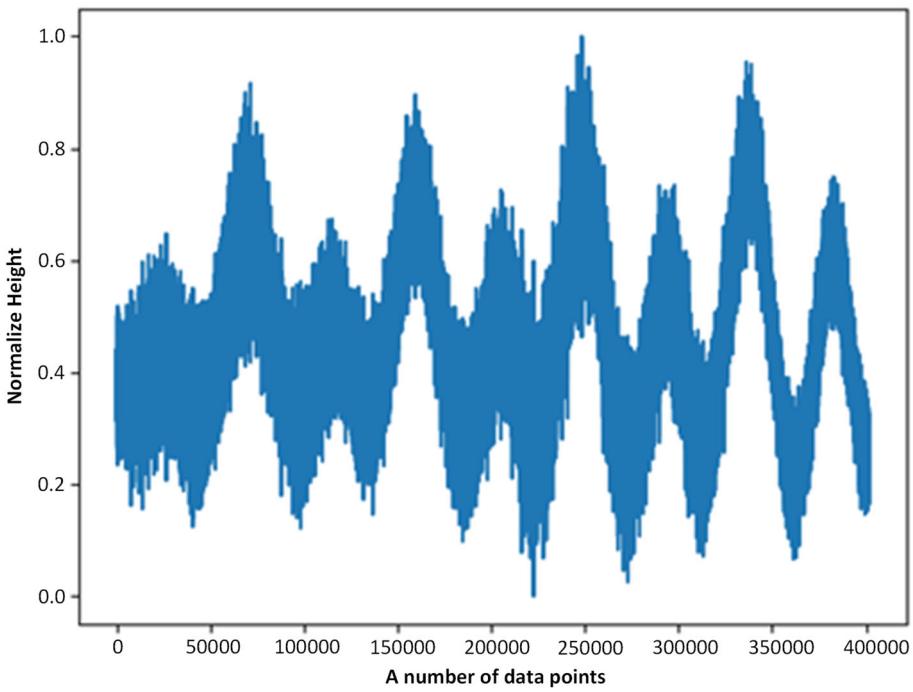


Fig. 13 Min–max scaler input transformation

below:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{9}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{10}$$

$$\tilde{h_t} = \tanh(W_h x_t + (r_t \odot h_{t-1}) U_h + b_h) \tag{11}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h} \tag{12}$$

$$y_t = \sigma(W_y h_t + b_y) \tag{13}$$

From empirical insight [47], GRUs overcome LSTM network performance for low complexity sequences and vice versa. This performance [47] corresponds to the size of the learning rate for each complexity rate (low and high) of seed strings. LSTM networks perform better for similar forecasting on higher complexity of seed strings.

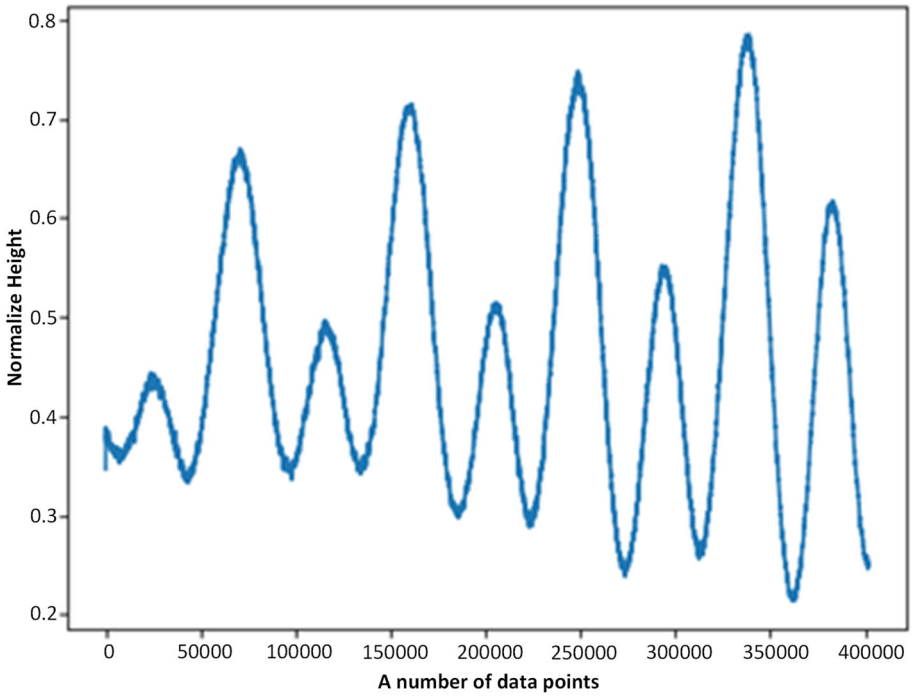


Fig. 14 Low-pass filter Butterworth output

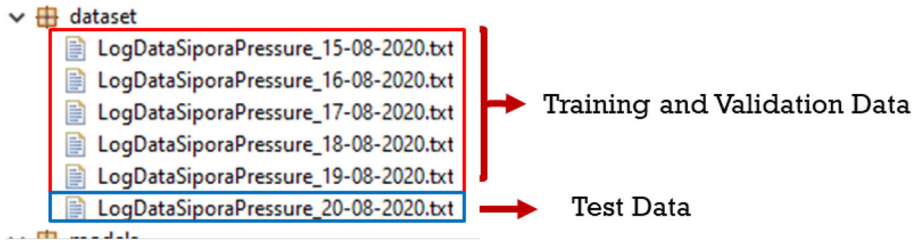


Fig. 15 Data set composition

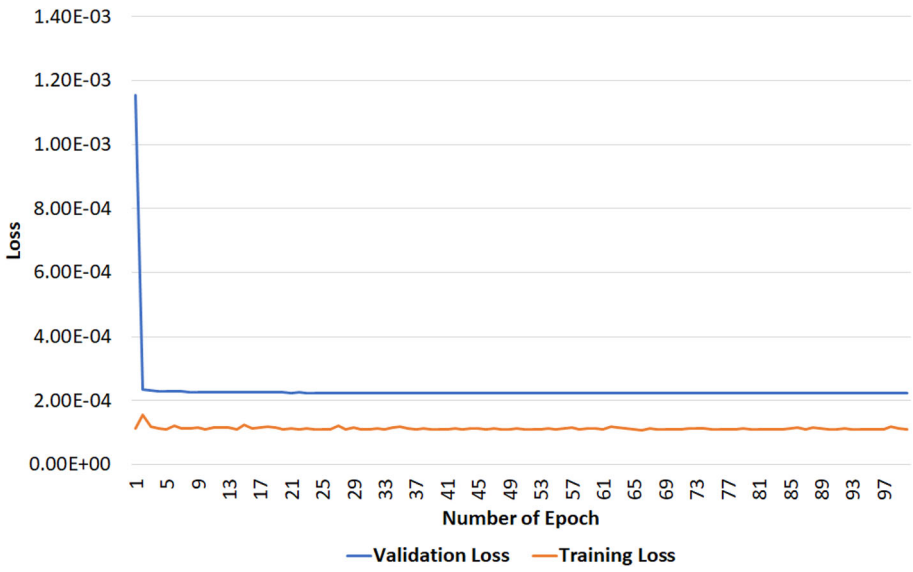
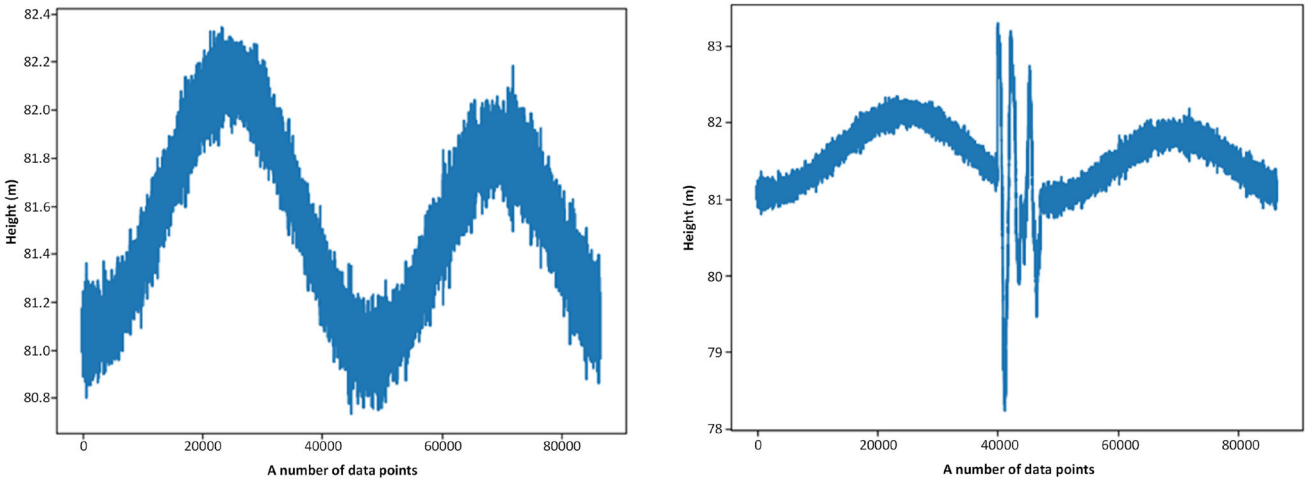


Fig. 16 Model train versus validation loss (from LSTM training process)



(a) Without tsunami (b) Injected synthetic tsunami

Fig. 17 Data test

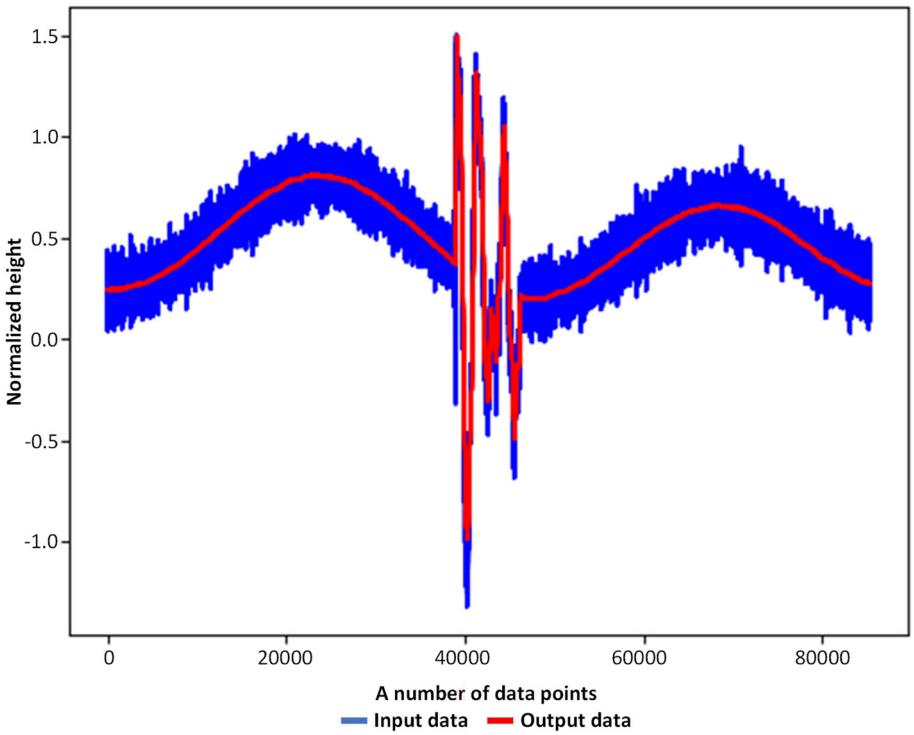


Fig. 18 Sequence prediction test on tsunami data injected induced by earthquake on mag. 7.8

4.2 Data acquisition

INA CBT Sipora consists of two sensors on Ocean Bottom Unit (OBU) and an optical cable under the sea (Fig. 7). One sensor is the Bottom Pressure Recorder (BPR), a pressure transducer measuring tide periodically. Three parameters are captured per second, DateTime, water column height, and temperature (Fig. 8).

4.3 Testing data

The testing data preparation consists of modelling shallow water tsunamis using the Tunami-F1 model [48] and injecting the tsunami model into actual capture data. The work of Infrastructure Technology Centers Ports and Coastal Dynamics BPPT Indonesia helped the tides prediction model for tsunami identification by providing shallow water tsunami data tests. They simulate dummy tsunamis generated by an earthquake ranging from 6.4 to 8.2 magnitude (Fig. 9).

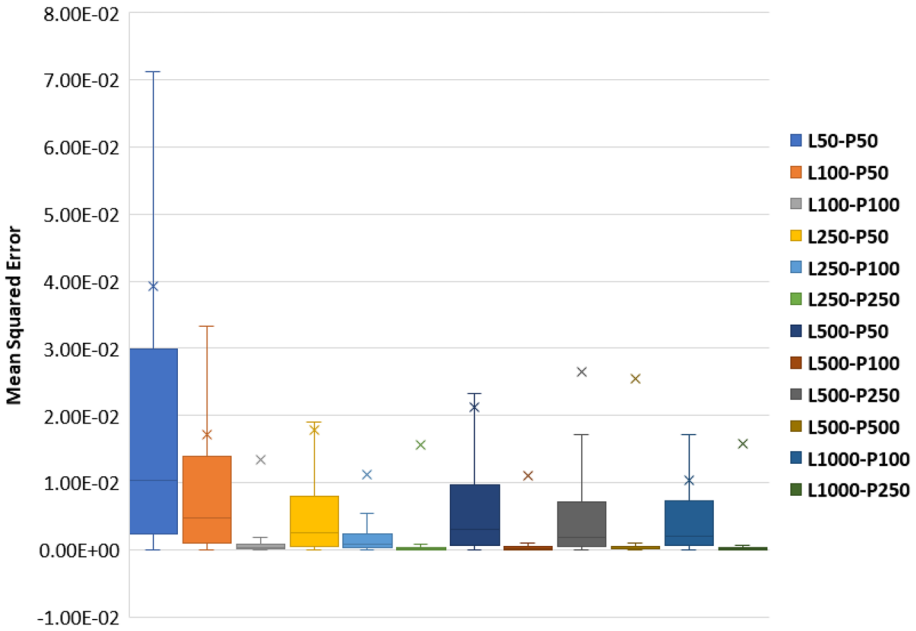


Fig. 19 Error distribution of tides prediction of Vanilla RNN model

After the tsunami-generated by earthquake variabilities are produced, the modeled tsunami is injected into the test data. Some steps are to follow for injecting tsunami data into data tests and real-time captured tide data. The process starts by interpolating the tsunami-generated data by matching the model time sampling. The resulting wave will have zero paddings conforming to the array dimension. Finally, the modeled tsunami superposes test data, which yields injected tsunami waves, as presented in Fig. 10.

4.4 Tsunami identification

Because of the low amount of the actual shallow water tsunami data set (only one was generated), the model cannot be expected to solve the classification problem. Instead, the generated data prediction will use a smoothed z-score to determine the tides as a tsunami or not. Z-score is a standard methodology used in forecasting problems to identify the trend from the prediction. This score indicates how many standard deviations an observation on each i is above or below the mean:

$$Z_i = (x_i - \bar{x})/\sigma \tag{14}$$

5 Result and analysis

All test procedures are performed through Python 3.8 with Keras, Tensorflow, numpy, pandas, scikit-learn, and matplotlib third-party library. On top of that, these program specifications are supported by GPU Nvidia A6000 as part of the computer platform in Artificial Intelligence Laboratory Kanazawa. After obtaining the results, we performed two analyses to evaluate our designed performances. Those are $look-back$ prediction and z-score tsunami identification analysis.

5.1 Preprocessing procedural testing

In Fig. 11, the training data set comprises input data, a periodic tidal wave captured continuously every second for 5 days. Later, these data go into two scalers, $robust$, and *min–max* scaler. A robust scaler transforms the data input by removing the median and scaling the data according to the interquartile range (IQR) (Fig. 12). The IQR ranges between the $1stquartile$ and $3rdquartile$. This process makes the distribution of data robust to outliers. A $min-max$ scaler is then applied to adjust the value range from 0 to 1, which is required for LSTM input data.

After that, the Butterworth filter refines the normalized waves to reduce data noise form (shown in Fig. 13). From Fig. 14, data noise is decreased heavily into a smoother appearance.

5.2 Model and validation data set

We sample six daily Sipora tides log data from Sipora OBU and divide them into a 5:1 ratio of a data set for training and testing. This testing data become the basis of dummy data for the superposition of tsunami-generated earthquake variation.

There are 401,696 data points for the training set and validation. These data are split into 0.8 training and 0.2 test data (Fig. 15). Arbitrarily, we choose a configuration from a particular model to represent the "Training loss VS Validation

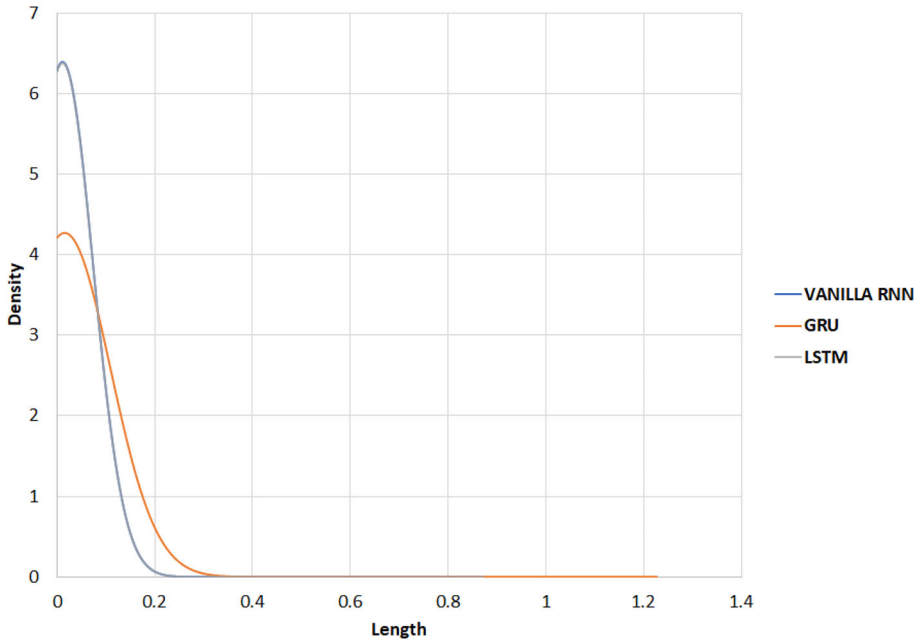


Fig. 20 Density distribution normalized graph of MSE distribution on L1000-P250

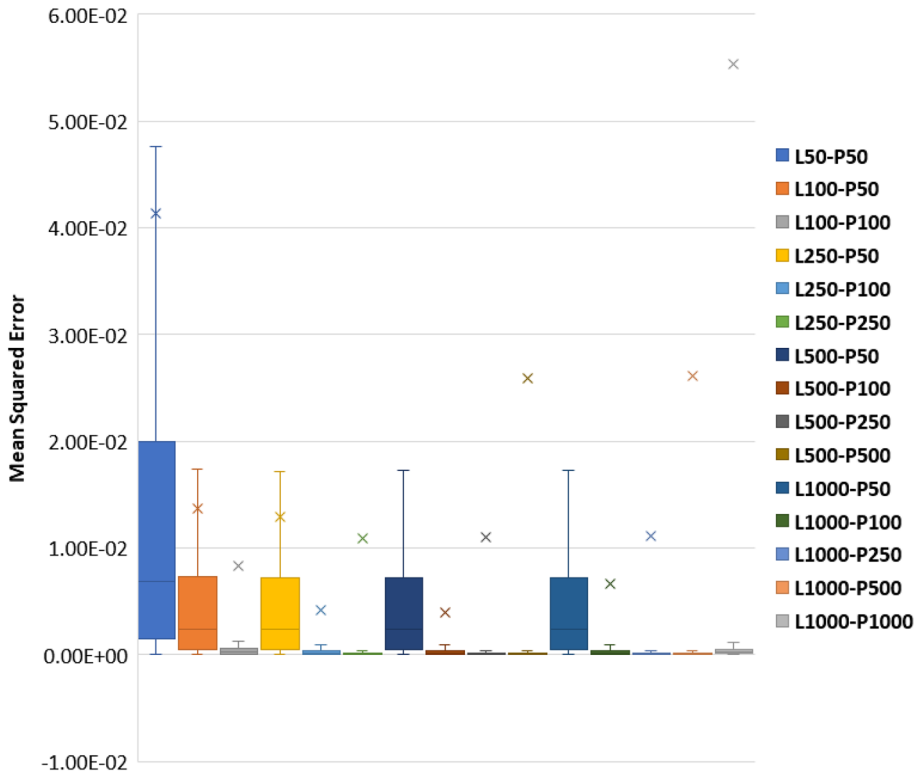


Fig. 21 Error distribution of tides prediction of LSTM model

loss" output. The training process uses Mean Squared Error (MSE) as a loss function. It converges with a training loss of 1.11×10^{-4} and a validation loss of 2.23×10^{-4}. The difference in these numbers indicates that the model is neither underfitting nor overfitting because of its small margin in the region 10^{-4} (refer to Fig. 16).

This model training is finished with 100 epochs and batch size 256-this process executes 100 units of the RNN stacked model. Adam optimizer is also applied with the learning rate $1e^{-3}$ and decay rate $1e^{-5}$ to improve convergence speed.

5.3 Testing data preparation

After we collect the necessary model data, the training model is saved into *.h5* format. Then, the model is tested with an external data set referred to as the B section. Before the

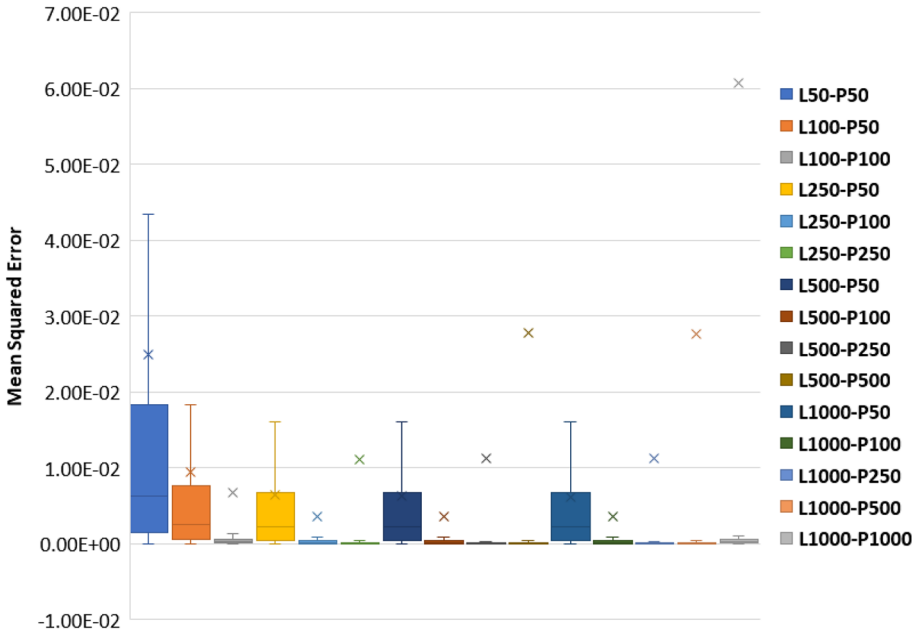


Fig. 22 Error distribution of tides prediction of GRU model

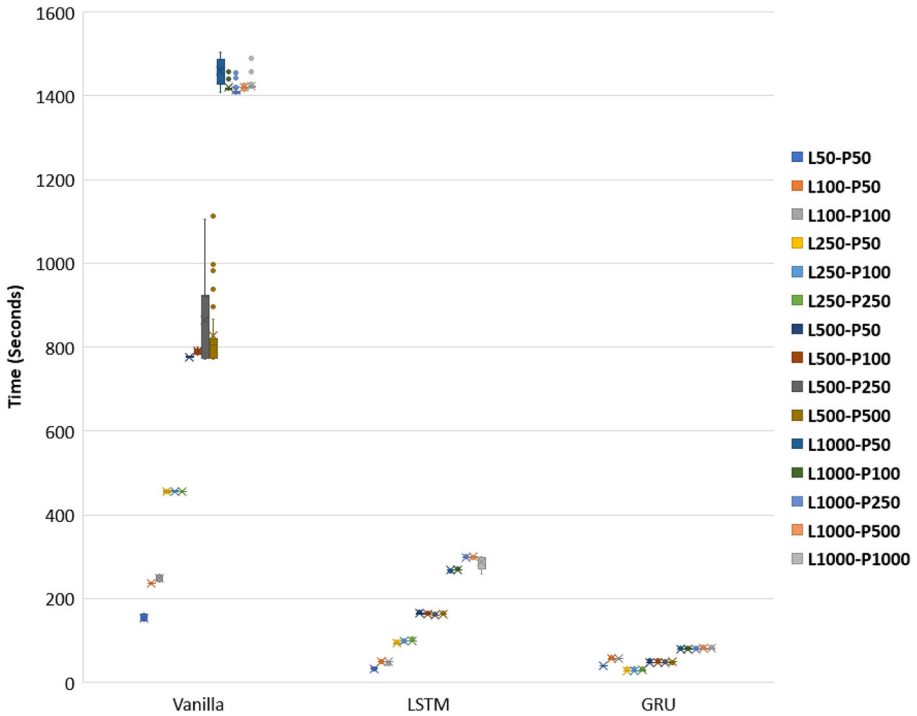


Fig. 23 Vanilla RNN VS LSTM VS GRU training time for one epoch distribution

prediction, the data test embedded tsunami synthetic are prepared by using the explained algorithm in the *methodology part*, *section 5* (Fig. 17).

When all necessary data tests are gathered, the upcoming process will be a prediction. From this prediction, we can compute the error rate for performance evaluation. Moreover, prediction and data input are also plotted to know how well the RNN model filters out the ambient noise (Fig. 18).

5.4 Look-back variation analysis

A set of look-back and prediction parameter configurations is tested. Moreover, to choose the look-back parameter, we need to empirically assess the computer's capability to execute multiple arrays related to GPU resources. In this experiment, the composition number of look-back and prediction points that can be executed maximally is 1000 nodes of look-back and 1000 predictions. Higher configuration points can raise Out of Memory (OOM) errors caused by insufficient mem-

Table 1 Median of MSE distribution for each of model configuration

Model	L50-P50	L100-P50	L100-P100	L250-P50	L250-P100
Vanilla RNN	1.02×10^{-2}	4.69×10^{-3}	4.15×10^{-4}	2.54×10^{-2}	8.28×10^{-4}
LSTM	6.91×10^{-3}	2.42×10^{-3}	2.33×10^{-4}	2.35×10^{-3}	1.6×10^{-4}
GRU	6.35×10^{-3}	2.54×10^{-3}	2.82×10^{-4}	2.25×10^{-3}	1.65×10^{-4}
	L250-P250	L500-P50	L500-P100	L500-P250	L500-P500
Vanilla RNN	1.88×10^{-4}	3.1×10^{-3}	1.95×10^{-4}	1.78×10^{-3}	2.28×10^{-4}
LSTM	8.29×10^{-5}	2.35×10^{-3}	1.62×10^{-4}	7.96×10^{-5}	1.13×10^{-4}
GRU	8.36×10^{-5}	2.25×10^{-3}	1.59×10^{-4}	7.92×10^{-5}	1.2×10^{-4}
	L1000-P50	L1000-P100	L1000-P250	L10000-P500	L1000-P1000
Vanilla RNN	NaN	2.03×10^{-3}	1.14×10^{-4}	NaN	NaN
LSTM	2.36×10^{-3}	1.58×10^{-4}	8.14×10^{-5}	1.18×10^{-4}	3.01×10^{-4}
GRU	2.26×10^{-3}	1.57×10^{-4}	7.8×10^{-5}	1.17×10^{-4}	3.09×10^{-4}

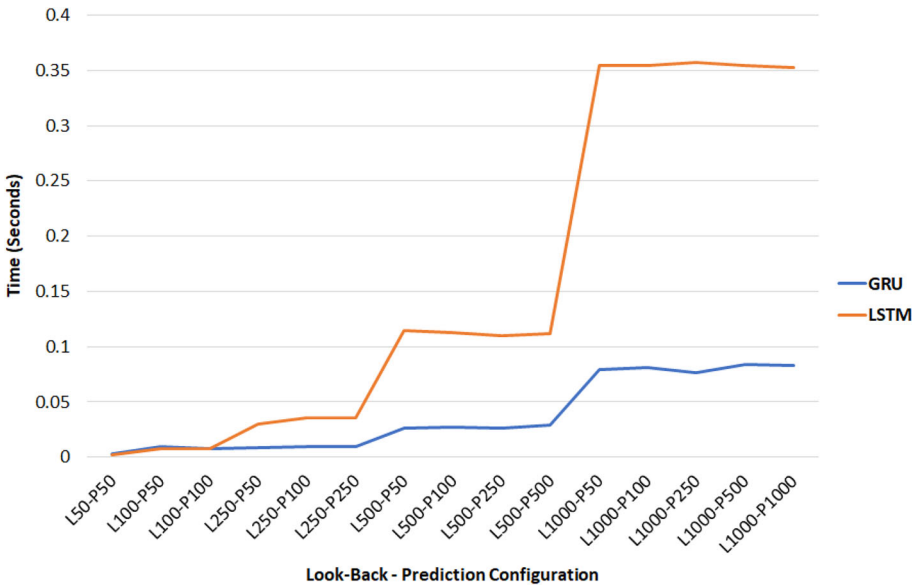


Fig. 24 GRU VS LSTM time prediction

ory. Initially, vanilla RNN networks are implemented to see how classic RNN work in the tide prediction application. In this experiment, as the input data will be continuous tides with temporal dependent, we apply a stateful setting on the RNN networks. Consequently, the network can learn the previous batches. Mini batches are also done for the input to ensure all the sequences are processed.

Vanilla RNN shown in Fig. 19 can achieve a median from MSE score distribution of 1.14×10^{-4} on Look-back 1000 and Prediction 250 configuration. The median of the MSE score distribution is used as a pointer because the mean of the MSE score is skewed as a result of outliers, as shown in Fig. 19. It also can be seen that each ratio L250-P250, L500-P100, and L500-P500 has the same range of 10^{-4} as the most minimum median in the MSE distribution. On the other hand, the other's ratio shows MSE scores in the range of 10^{-2} and 10^{-3}. Besides its performance, from Fig. 19, L1000-P50, L1000-P500, and L1000-P1000 configurations are missing because "NaN" errors occur during prediction. We consider this instability to be caused by vanilla RNN's insufficient representability for capturing the complexity of the tides.

Surprisingly, according to Fig. 20, LSTM has a similar MSE distribution on our tides prediction with vanilla RNN on the ratio of L1000-P250. However, compared to vanilla RNN, the LSTM model tremendously improved the median MSE score (Fig. 21) by reaching 7.96×10^{-5} on a ratio of L500-P250. On the same ratio, L1000-P250, it also shows improvement to 8.14×10^{-5}. Other ratios denote better scores than the vanilla RNN model.

Finally, the GRU model shows the best performance compared to the others. It can pull off the median in the error score distribution to 7.8×10^{-5} on the L1000-P250 configuration (Fig. 22), which is also smaller than the other two mod-

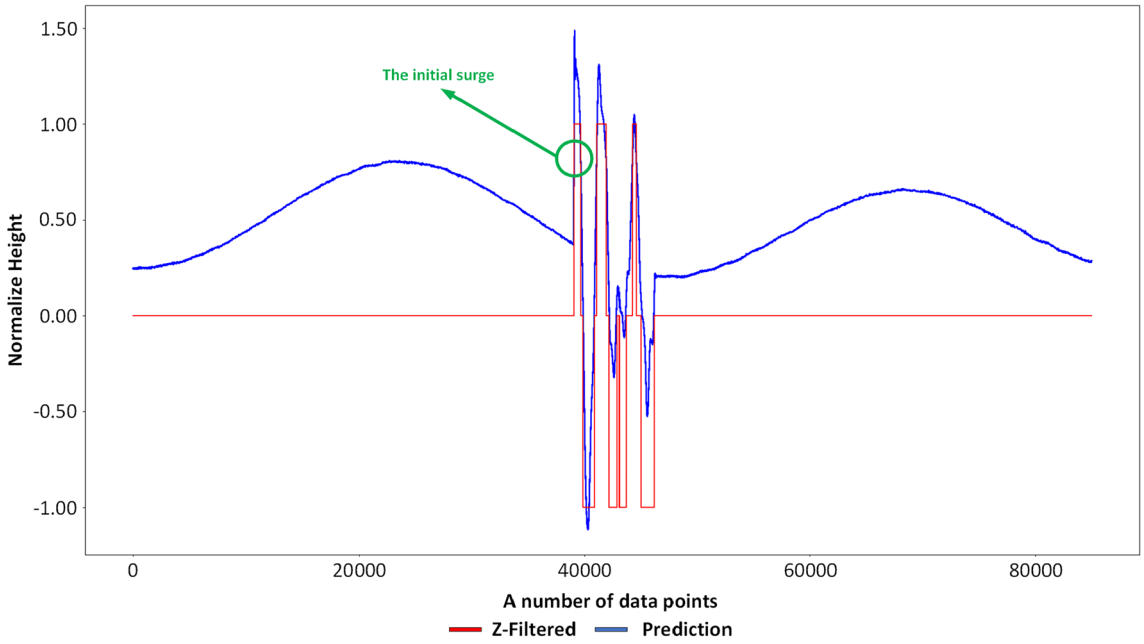


Fig. 25 Z-score tsunami spike identification on the data test with magnitude 7.8

Table 2 Tsunami tides identification for each model

Magnitude	Ground Truth Det	Vanilla RNN Det	LSTM Det	GRU Det
7.2	2	2	2	2
7.4	5	3	5	5
7.6	6	6	6	6
7.8	7	7	7	7
8	6	6	6	6
8.2	6	6	6	6

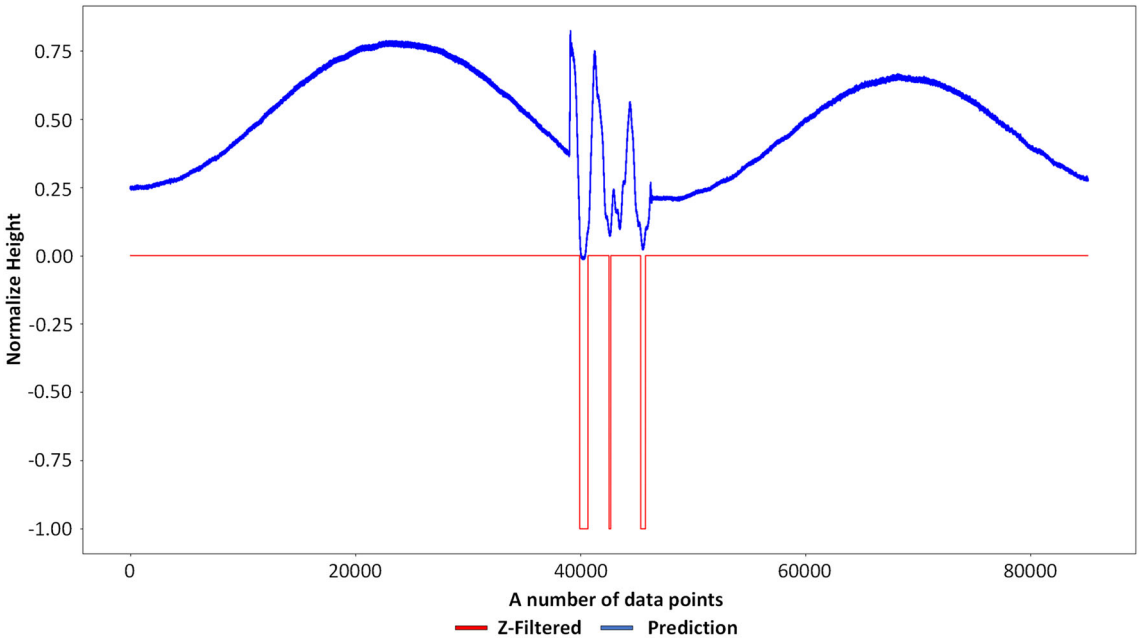
els (vanilla RNN and LSTM). Overall performance, GRU exhibits a higher but close error score to the LSTM model (9 configurations higher than the LSTM model (Table 1)). This improvement in error score indicates that the tides prediction problem has a low complexity sequence which, in this case, GRU has better performance and efficiency (Table 1 and Fig. 23).

This efficiency refers to the training and prediction time. As for the context of tides prediction in the tsunami application (Fig. 22), our model should predict as fast as possible to ensure enough time for the information to be conveyed on the shore. Nonetheless, the training time for one epoch depends on the layer type, number of hidden units, network depth, input data dimension, and model hyperparameter. From our experiments, GRU shows remarkable efficiency in training and predicting time. All GRU model configuration accomplishes the training process for under 90 s per epoch (Fig. 23). This result is also directly proportional to the prediction time of one sequence output which set off all the configurations under 0.1 s (Fig. 24). Second best in efficiency, The LSTM model finishes the training process for one epoch up to 302 s and prediction of 0.352 s (Fig. 24). This performance evaluation is relevant to the [47] for less complex sequence problem. The last model, vanilla RNN, is the palest in time performance compared to others. It takes up to 1490 s to finish one epoch training and 1.14 s to predict the sequence of tides. Nevertheless, this performance showcases validation on RNN model comparison, which is relevant to the previous research [19, 22, 44, 47].

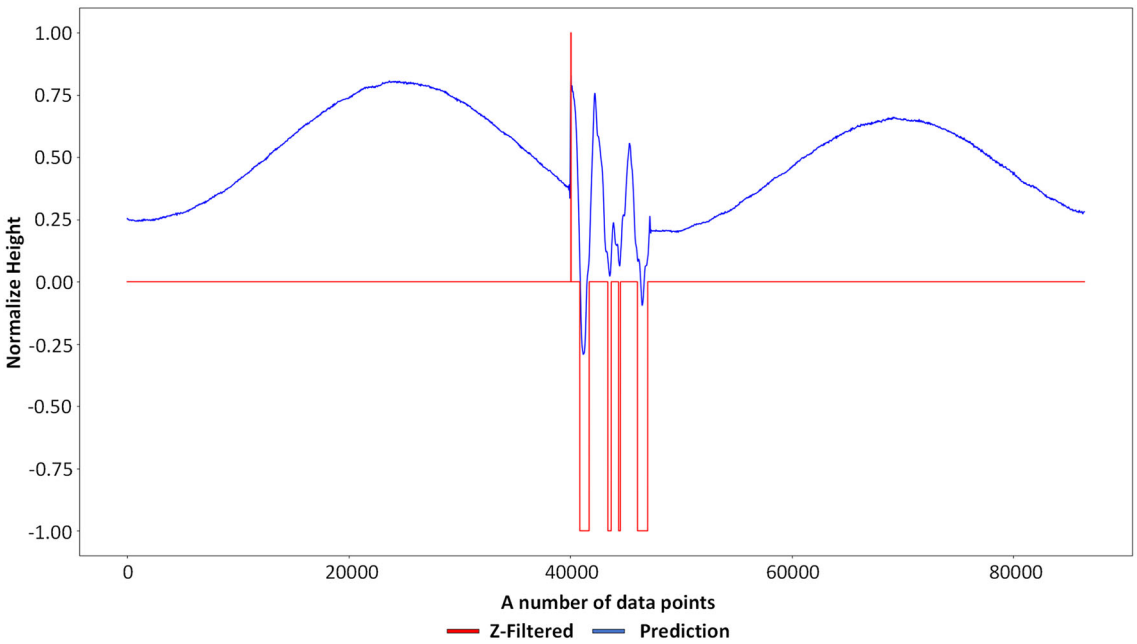
5.5 Z-score analysis

Z-score detects significant variations of the spikes from the expected tides. This methodology can work using the means and standard deviation of normal tides. Then, the threshold value is applied to the standard deviation as a margin to the expected tidal wave distribution.

This methodology is tested on the divergence of tsunami synthetic. In this experiment, we empirically set the threshold of 2.7 of standard deviation as it shows the test's false error. The fluctuation can be identified by "1" as a rising tide and "−1" as a downward spike (Fig. 25).



(a) Vanilla RNN



(b) Ground Truth

Fig. 26 Comparison of z-score processed

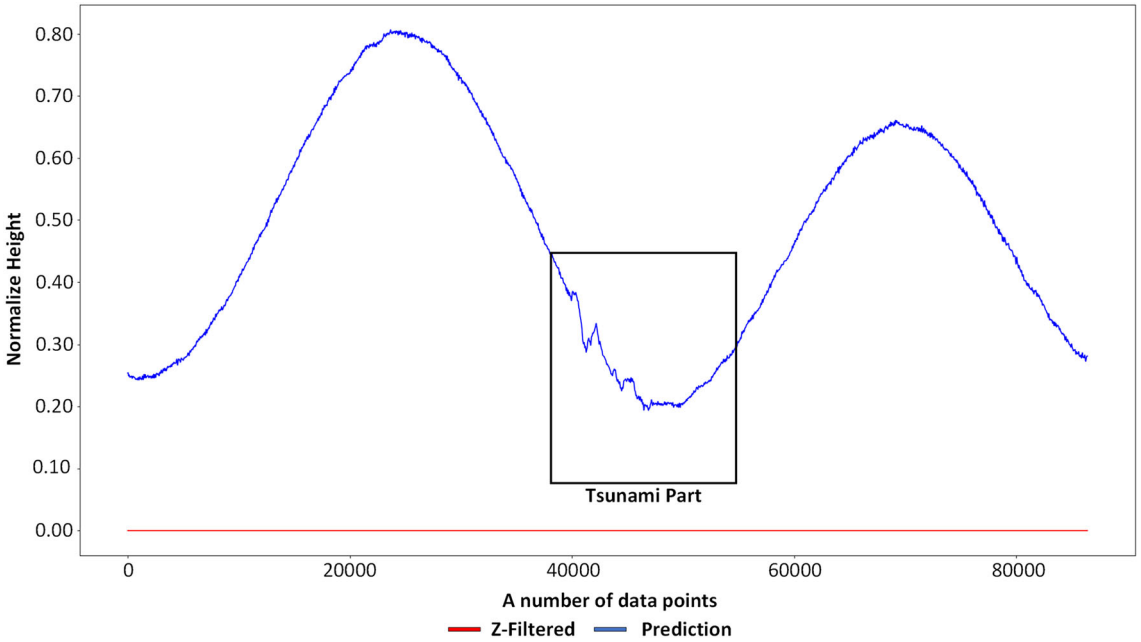


Fig. 27 Tides embedded synthetic tsunami induced by earthquake with a magnitude of 6.4

Our experiment also evaluates the z-score on each model prediction. Table 2 shows the performance of the tides prediction model processed in z-score to identify a surge of tides or tsunami caused by an earthquake of various magnitude. The shallow water tsunami of LSTM and GRU prediction can be completely identified in corresponding to the number of peaks detected in ground truth prediction. Still, vanilla RNN prediction misses two tsunami tides on magnitude 7.4 as shown in Fig. 26.

Furthermore, the z-score methodology cannot identify tsunami spikes in the magnitude 6.4–7 range. This outcome is supported by the fact that the waves embedded in the tsunami on that range are blended exceptionally well; the waveform is hardly noticed. The z-score can determine the number of fluctuations caused by the synthetic tsunami in the other magnitude span. The higher the magnitude cause, the easier z-score can recognize the sudden change of tides. Moreover, it can detect the tsunami's initial surge on the majority of magnitude except for magnitude 7.2 and 7.4 (only for vanilla RNN), which is vital to know when the tsunami starts (shown in Fig. 27).

6 Conclusion and future works

Applying a time series model to tide prediction problems on shallow water requires multiple mechanisms to identify tsunami spikes. The procedures start from the preliminary operation on the input waves. Two-step features scaling, robust, and min–max scaler are applied to capture the wave's distribution and adjust to the RNN variant input array requirement. Then, the Butterworth filter with 9 order and 0.01 Hz cutoff frequency work in the sequence to filter out the ambient noise. This process continues to the time series model prediction of RNN and its variation. Finally, the z-score will determine whether these waves are possibly tsunamis.

We find that tides prediction is a low-complexity sequence problem corresponding to the performance evaluation of GRU, which is better than the LSTM model. GRU score lowest MSE median of 7.8×10^{-5} on the L1000-P250 configuration. It also exhibits the best efficiency by accomplishing the training process for under 90 s per epoch and the prediction process for under 0.1 s for all test configurations. Besides, in z-analysis, GRU and LSTM prediction show complete identification of tides. This result indicates that the GRU model suits the tides prediction problem.

Incorporating a z-score in the surge of tides identification is due to the limitation of actual tsunami data in shallow water areas on newly deployed CBT Sipora, in which the model needs sufficient data for classification problems. In the future, incorporating an accelerometer in time series data input and prediction will improve the tides prediction model and cover the lack of capability in determining tsunami spikes caused by lower earthquake magnitude.

In addition, it is also worth mentioning recent developments of a deep transformer model [49–51], which has shown state-of-the-art performance in time series forecasting problems. This algorithm introduces the self-attention method, which can overcome the "short-term memory" problem over infinite long sequences [50]. This approach should also be

included in the next study to find a better tides prediction model to improve efficiency and accuracy.

Acknowledgements We express our gratitude to Infrastructure Technology Centers Ports and Coastal Dynamics BPPT and task force 5 of the Indonesian Tsunami Early Warning System, Widjo Kongko. Dr.-Ing, for providing tsunami synthetic evoked by earthquake with the variability of magnitude. Furthermore, we also convey our thankfulness to the photonics laboratory for facilitating us with the said computer in the experiment. Finally, we appreciate all of the Indonesian CBT team's hard works in supplying the data set for the training and the test.

Author Contributions Mr WD, Ms MD, and Prof HN wrote the main manuscript text. Ms BT prepared the data set for training and testing. Mr IMA and Mr SR prepared all the figures and tables. All authors reviewed the manuscript.

Funding DIPA–BPPT–PTE Indonesia financially supports the main research for Tsunami Early Warning System Project 2020–2022. Partial financial support was received from AI Laboratories Kanazawa University for publication and review.

Availability of data and materials All the data sets used in the report are unavailable online. It requires permission to access the data because it is part of the government project (Some documents agreement are necessary to use the data sets).

Code availability The code is not freely accessible online. It also requires permission to use the code because it is part of the government project (Some documents agreement are necessary to use the code).

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

1. Indonesian National Board for Disaster Management (2018) Laporan kinerja tahun 2018. https://bnpb.go.id/uploads/24/laporan-kinerja-bnpb-2018.pdf
2. Harig S, Immerz A, Griffin J, Weber B, Babeyko A, Rakowsky N, Hartanto D, Nurokhim A, Handayani T, Weber R et al (2019) The tsunami scenario database of the Indonesia Tsunami Early Warning System (InaTEWS): evolution of the coverage and the involved modeling approaches. Pure Appl Geophys 177:1379–1401
3. Center of Electronic (2020) BPPT: term of reference: program penguatan dan pengembangan Indonesia Tsunami Early Warning System (Ina-TEWS). Puspiptek Serpong, South Tangerang, Indonesia
4. Singh SC, Hananto N, Mukti M, Robinson DP, Das S, Chauhan A, Carton H, Gratacos B, Midnet S, Djajadihardja Y et al (2011) Aseismic zone and earthquake segmentation associated with a deep subducted seamount in Sumatra. Nat Geosci 4(5):308–311
5. Zkrausk (1986) Ambient noise in shallow water: a survey of the unclassified literature. https://apps.dtic.mil/sti/tr/pdf/ADA167696.pdf
6. Knobles D, Joshi S, Gaul R, Graber H, Williams N (2008) Analysis of wind-driven ambient noise in a shallow water environment with a sandy seabed. J Acoust Soc Am 124(3):157–162
7. Box GE, Jenkins GM (1976) Time series analysis, control, and forecasting. Holden Day, San Francisco, CA 3226(3228):10
8. Gardner ES Jr (1985) Exponential smoothing: the state of the art. J Forecast 4(1):1–28
9. Winters PR (1960) Forecasting sales by exponentially weighted moving averages. Manag Sci 6(3):324–342
10. Harvey AC (1990) Forecasting, structural time series models and the Kalman filter. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9781107049994
11. Tealab A (2018) Time series forecasting using artificial neural networks methodologies: a systematic review. Fut Comput Inform J 3(2):334–340
12. Granger CW, Terasvirta T et al (1993) Modelling non-linear economic relationships. OUP Catalogue, Oxford
13. Clements MP, Franses PH, Swanson NR (2004) Forecasting economic and financial time-series with non-linear models. Int J Forecast 20(2):169–183
14. Teräsvirta T (1994) Specification, estimation, and evaluation of smooth transition autoregressive models. J Am Stat Assoc 89(425):208–218
15. Haider SA, Naqvi SR, Akram T, Umar GA, Shahzad A, Sial MR, Khaliq S, Kamran M (2019) LSTM neural network based forecasting model for wheat production in Pakistan. Agronomy 9(2):72
16. Naqvi SR, Akram T, Iqbal S, Haider SA, Kamran M, Muhammad N (2018) A dynamically reconfigurable logic cell: from artificial neural networks to quantum-dot cellular automata. Appl Nanosci 8(1):89–103
17. Naqvi SR, Akram T, Haider SA, Kamran M, Shahzad A, Khan W, Iqbal T, Umer HG (2018) Precision modeling: application of metaheuristics on current-voltage curves of superconducting films. Electronics 7(8):138
18. Rather AM, Agarwal A, Sastry V (2015) Recurrent neural network and a hybrid model for prediction of stock returns. Expert Syst Appl 42(6):3234–3241
19. Schmidt RM (2019) Recurrent neural networks (RNNs): a gentle introduction and overview. https://doi.org/10.48550/arXiv.1912.05911
20. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
21. Tan M, Santos Cd, Xiang B, Zhou B (2015) LSTM-based deep learning models for non-factoid answer selection. arXiv preprint arXiv:1511.04108
22. Cho K, Merrienboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder–decoder for statistical machine translation. https://doi.org/10.48550/arXiv.1912.05911
23. Lai G, Chang W-C, Yang Y, Liu H (2018) Modeling long- and short-term temporal patterns with deep neural networks. https://doi.org/10.48550/arXiv.1703.07015

24. Shih S-Y, Sun F-K, Lee H-y (2019) Temporal pattern attention for multivariate time series forecasting. https://doi.org/10.48550/arXiv.1809.04206
25. Dudek G, Smyl S, Pełka P (2022) Recurrent neural networks for forecasting time series with multiple seasonality: a comparative study. https://doi.org/10.48550/arXiv.2203.09170
26. Petneházi G (2019) Recurrent neural networks for time series forecasting. https://doi.org/10.48550/arXiv.1901.00069
27. Smyl S (2020) A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. Int J Forecast 36(1):75–85. https://doi.org/10.1016/j.ijforecast.2019.03.017
28. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. https://doi.org/10.48550/arXiv.1409.3215
29. Yin W, Kann K, Yu M, Schütze H (2017) Comparative study of CNN and RNN for natural language processing. https://doi.org/10.48550/arXiv.1702.01923
30. Bradbury J, Merity S, Xiong C, Socher R (2016) Quasi-recurrent neural networks. https://doi.org/10.48550/arXiv.1611.01576
31. Grilli ST, Tappin DR, Carey S, Watt SF, Ward SN, Grilli AR, Engwell SL, Zhang C, Kirby JT, Schambach L et al (2019) Modelling of the tsunami from the December 22, 2018 lateral collapse of Anak Krakatau volcano in the Sunda Straits, Indonesia. Sci Rep 9(1):1–13
32. McCloskey J, Antonioli A, Piatanesi A, Sieh K, Steacy S, Nalbant S, Cocco M, Giunchi C, Huang J, Dunlop P (2008) Tsunami threat in the Indian Ocean from a future megathrust earthquake west of Sumatra. Earth Planet Sci Lett 265(1–2):61–81
33. Mofjeld H (1997) Tsunami detection algorithm. http://nctr.pmel.noaa.gov/tda_documentation.html
34. Beltrami GM (2008) An ANN algorithm for automatic, real-time tsunami detection in deep-sea level measurements. Ocean Eng 35(5–6):572–587
35. Barman R, Prasad Kumar B, Pandey PC, Dube SK (2006) Tsunami travel time prediction using neural networks. Geophys Res Lett **33**(16). https://doi.org/10.1029/2006GL026688
36. Romano M, Liong S-Y, Vu MT, Zemskyy P, Doan CD, Dao MH, Tkalich P (2009) Artificial neural network for tsunami forecasting. J Asian Earth Sci 36(1):29–37
37. Fauzi A, Mizutani N (2020) Machine learning algorithms for real-time tsunami inundation forecasting: a case study in Nankai region. Pure Appl Geophys 177(3):1437–1450. https://doi.org/10.1007/s00024-019-02364-4
38. Negarestani A, Setayeshi S, Ghannadi-Maragheh M, Akashe B (2002) Layered neural networks based analysis of radon concentration and environmental parameters in earthquake prediction. J Environ Radioact 62(3):225–233
39. Ozerdem MS, Ustundag B, Demirer RM (2006) Self-organized maps based neural networks for detection of possible earthquake precursory electric field patterns. Adv Eng Softw 37(4):207–217
40. Ni Y, Zhou X, Ko J (2006) Experimental investigation of seismic damage identification using PCA-compressed frequency response functions and neural networks. J Sound Vib 290(1–2):242–263
41. Bhandarkar T, Satish N, Sridhar S, Sivakumar R, Ghosh S et al (2019) Earthquake trend prediction using long short-term memory RNN. Int J Electr Comput Eng 9(2):1304–1312
42. Wiegel RL (1964) Oceanographical engineering. Prentice-Hall, Inc., Englewood Cliffs, NJ
43. Le Cun Y, Jackel LD, Boser B, Denker JS, Graf HP, Guyon I, Henderson D, Howard RE, Hubbard W (1989) Handwritten digit recognition: applications of neural network chips and automatic learning. IEEE Commun Mag 27(11):41–46
44. Goodfellow I, Bengio Y, Courville A, Bengio Y (2016) Deep learning, vol 1. MIT Press, Cambridge
45. Hochreiter S (1991) Untersuchungen zu dynamischen neuronalen netzen. Diploma, Technische Universität München **91**(1)
46. Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw 5(2):157–166
47. Chen RC, Güttel S (2021) A comparison of LSTM and GRU networks for learning symbolic sequences. CoRR arxiv:2107.02248
48. Imamura F, Yalciner A, Ozyurt G (2006) Tsunami modelling manual (tunami model). IOC Manuals and Guides (30)
49. Wu N, Green B, Ben X, O'Banion S (2020) Deep transformer models for time series forecasting: the influenza prevalence case. CoRR arxiv:2001.08317
50. Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H, Zhang W (2020) Informer: Beyond efficient transformer for long sequence time-series forecasting. CoRR arxiv: 2012.07436
51. Grigsby J, Wang Z, Qi Y (2021) Long-range transformers for dynamic spatiotemporal forecasting. CoRR arXiv:2109.12218