


## Article

# DERN: Deep Ensemble Learning Model for Short- and Long-Term Prediction of Baltic Dry Index

Imam Mustafa Kamal <sup>1</sup>, Hyerim Bae <sup>2,\*</sup>, Sim Sunghyun <sup>2</sup> and Heesung Yun <sup>3</sup><sup>1</sup> Department of Big Data, Pusan National University, Busan 46241, Korea; imamkamal@pusan.ac.kr<sup>2</sup> Department of Industrial Engineering, Pusan National University, Busan 46241, Korea; ssh@pusan.ac.kr<sup>3</sup> Korea Maritime Institute, Busan 49111, Korea; heesung@kmi.re.kr

\* Correspondence: hrbae@pusan.ac.kr; Tel.: +82-51-510-2733 or +82-51-512-7603

Received: 8 January 2020; Accepted: 20 February 2020; Published: 22 February 2020



**Abstract:** The Baltic Dry Index (BDI) is a commonly utilized indicator of global shipping and trade activity. It influences stakeholders' and ship-owners' decisions respecting investments, chartering, operational plans, and export and import activities. Accurate prediction of the BDI is very challenging due to its volatility, non-stationarity, and complexity. To help stakeholders and ship-owners make sound short- and long-term maritime business decisions and avoid market risk, we performed short- and long-term predictions of BDI using an ensemble deep-learning approach. In this study, we propose to apply recurrent neural network models for BDI prediction. The state-of-the-art of sequential deep-learning models such as RNN, LSTM, and GRU are employed to predict one- and multi-step-ahead BDI values. In order to increase the accuracy, we assemble the models. In experiments, we compared our results with those of traditional methods such as ARIMA and MLP. The results showed that our proposed method outperforms ARIMA, MLP, RNN, LSTM, and GRU in both short- and long-term prediction of BDI.

**Keywords:** time-series; forecasting; baltic dry index; ensemble method; deep learning

## 1. Introduction

The Baltic Dry Index (BDI) is a freight index created by the London-based Baltic Exchange. This index indicates shipment costs for dry bulk cargoes consisting of commodities such as grain, coal, iron, ore, and copper. The BDI is a composite of three sub-indices, namely Capesize, Panamax, and Supramax. Those indices have different bulk-carrier capacities, 180,000, 74,000, and 58,000 dwt, respectively. The BDI has been widely used as a world-trade economic indicator [1]. Many stakeholders make serious efforts to forecast it, precisely, so as to be able to make smart investment and trading decisions. However, the volatility, non-stationarity, and complexity of the BDI is known to be more intractable than stock prices. Therefore, it is a challenging task to perform predictions against BDI values.

The BDI is regarded as a barometer not only of the shipping industry and international trade, but also of the global economy [2]. Investors, speculators, and researchers have long found it to be useful, theoretically challenging, and relevant when projecting future profits. However, because many managerial decisions are based on future prospects, forecasting accuracy is essential for organizations and companies in order to avoid market risk. Recent advances in both analytical and computational methods have resulted in a number of new ways of mining freight-index time-series data.

Ship owners, stakeholders and investors need to be concerned about not only short-term prediction of time-series data but also long-term prediction. Predicting a long-term sequence of time-series data is more difficult than short-term prediction [3]. For example, in making a decision on a vessel, there are multiple options available to the vessel's owners. If the BDI trend is increasing,

the vessel's owners wait for the right time to sell the vessel to get the maximum profit. Instead, they will immediately sell it if the BDI index tends to decline. In other cases, if the BDI index has an upward trend, the vessel's owner will not rent the vessel to the market. Instead, they will operate the ship by themselves. Conversely, if the BDI trend decreases, they will rent the vessel. Therefore, in this study, we developed an analytic method for accurate prediction of short- and long-term values of the BDI for helping vessel's owners.

The rest of this paper is organized as follows. Section 2 provides the background of our research and discusses the relevant BDI-related work. Section 3 introduces our proposed method. Section 4 analyzes the experimental results and compares them against econometrics and machine learning methods. Finally, Section 5 draws conclusions and anticipates future work.

## 2. Background

In this section, we present the background on our BDI-related research and discuss and some time-series prediction models available in the literature.

### 2.1. Related Works

Cullinane et al. [4] were the pioneers in conducting research on BDI prediction using the ARIMA model. In the past several years, there has been some research done on BDI prediction. Cho and Lin used a fuzzy neural network model to analyze and forecast BDI [5], and Kamal et al. [6] forecast BDI as a high-dimensional multivariate regression problem by using deep neural networks. Sahin et al. [7] predicted one-step-ahead BDI values by their proposed three artificial neural networks, specifically a univariate model and two bivariate models, by harnessing historical BDI data and the world price of crude oil. Qingcheng et al. [8] proposed a decomposition technique for BDI data, and then used a neural network for prediction. Zhang et al. [9] compared econometric models such as ARIMA and GARCH with artificial neural network models such as BPNN, RBFNN, and ELM. The majority of the previous research has treated BDI prediction as regression and short-term prediction tasks. In the present study, by contrast, we conducted research on both short- and long-term prediction of BDI to facilitate ship-owners' short- and long-term decision-making. In addition, in terms of models, the majority of the previous studies have harnessed artificial neural network models and statistical models. Note that in terms of sequential learning for prediction of time-series data, the recurrent neural network is the state-of-the-art method. Moreover, nowadays, deep learning with a deep architecture is a promising approach for accurate prediction.

Over the course of the past few decades, there have been many outstanding approaches to the prediction of time-series data, such as ARIMA [10], Support Vector Regressor (SVR) [11], fuzzy systems [12], and deep learning [13–16]. Nonetheless, by those methods alone, accurate prediction of real data is unobtainable, since real, time-series data commonly is volatile and non-stationary. For enhanced prediction, some researchers have proposed data transformation [17], decomposition [18–20], and even ensemble methods [21,22]. In the present study, we harnessed and combined deep-learning approaches including a deep recurrent neural network (Deep RNN), a long-short-term memory network (LSTM) and a gated rectified unit neural network (GRU) to obtain more accurate prediction results.

### 2.2. Sequential Model

Recurrent Neural Network (RNN) is a type of neural network with loops that allow for retention of information from the past. Specifically, the loops enable RNN to use information from past time slices to produce output for the current time slice  $t$ . Thus, we can say that the decision made at a time slice  $t - 1$  affects the decision to be made at  $t$ . Therefore, the response of the network to the new

data depends on the current input as well as the output from the recent past data. The RNN output calculation is based on iterative calculation of the output of the following two equations:

$$h_t = H(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (1)$$

$$y_t = w_{hy}h_t + b_y \quad (2)$$

In Equations (1) and (2),  $x_t$  is the input sequence,  $y_t$  is the output sequence, and  $h_t$  represents the hidden vector sequence at time slice  $t$  ( $t = 1, 2, \dots, T$ );  $W$  and  $b$  represent weight matrices and biases, respectively; and lastly,  $H$  is an activation function used for the hidden layer. The back-propagation through time (BPTT) technique is usually used to train RNNs [23]. However, it is difficult to use BPTT to train traditional RNNs, due to the gradient-vanishing and exploding problem [24]. Errors from later time steps are difficult to propagate back to previous time steps for proper updates of network parameters. To address this problem, the long short-term memory (LSTM) unit has been developed [25].

LSTM is a special type of recurrent neural network with memory cells. These memory cells are the essential component for handling of long-term temporal dependencies in the data. LSTM has the option to add or delete information from this cell state. This operation is done by special structures in LSTM, which are called gates. The three types of gate are *input gate* ( $i_t$ ), *forget gate* ( $f_t$ ), and *output gate* ( $o_t$ ), shown in Equations (3) to (8).  $\tilde{C}_t$  is a “candidate” hidden state that is computed based on the current input and the previous hidden state.  $C_t$  is the internal memory of the unit. It is a combination of the previous memory, as multiplied by the forget gate, and the newly computed hidden state, as multiplied by the input gate.  $h_t$  is the output hidden state, and is computed by multiplying the memory by the output gate [26].

$$h_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (3)$$

$$h_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (4)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (6)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (7)$$

$$h_t = o_t * \tanh(C_t) \quad (8)$$

GRU is another type of RNN with memory cells [27]. It is similar to LSTM but with a simpler cell architecture. GRU also has a gating mechanism to control the flow of information through the cell state, but has fewer parameters and does not contain an output gate. It consists of two gates,  $r$  being a reset gate, and  $z$  an update gate. The reset gate regulates the flow of new input to the previous memory, and the update gate determines how much of the previous memory to keep. The following equations are used in the GRU output calculations:

$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z) \quad (9)$$

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r) \quad (10)$$

$$\tilde{h}_t = \tanh(W_{xh}x_t + W_{hh}(r_t \odot h_{t-1}) + b_h) \quad (11)$$

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

In previous studies [28,29], it has been noted that GRU is comparable to, or even outperforms, the LSTM. Regarding the obtainment of high accuracy in prediction of the BDI, in this study, we combined RNN, LSTM, and GRU into an ensemble method. The idea was to combine the predictions from multiple different sequential models. Each model has different strength and weakness, meaning that its predictions are better than any other in a certain condition. Importantly, the models must

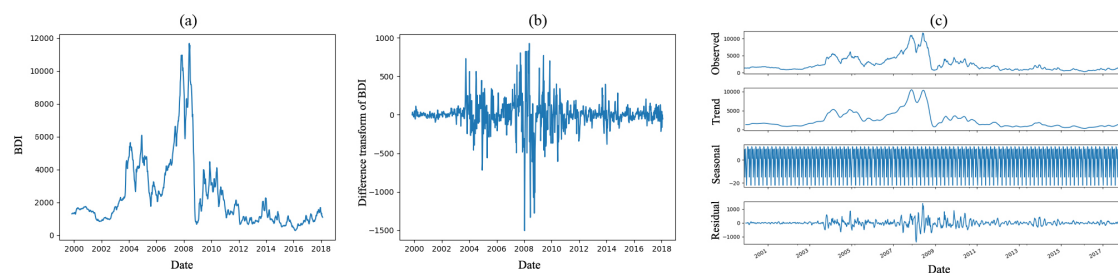
be good in different ways: they must make different prediction errors. In addition to reducing the variance in the prediction, our ensemble can also result in better predictions than any single best model. For instance, Krizhevsky et al. used model averaging across multiple well-performing CNN models to achieve outstanding results [30].

### 3. Method

In this section, our data pre-processing technique, followed by the system design of our proposed method and the metric measure used to assess the accuracy of our approach are explained.

#### 3.1. Data Pre-Processing and Analysis

The BDI data plotted in Figure 1a shows a sharp increase and a dramatic decrease between 2007 and 2008. Therefore, we pre-processed the data in a way to make it more stationary. By using a decomposition technique, the BDI data is separated into three components: trend, seasonality, and noise, as depicted in Figure 1c. The trend can be observed as increasing or decreasing the trend value in the time-series data; however, the BDI data does not show any significant increasing or decreasing trend, but rather a peak in 2008 and two long tails. The seasonality repeats the short-term cycle in the BDI, and the noise corresponds to random variation in the series.



**Figure 1.** (a) BDI index data; (b) Difference transform of BDI data; (c) Decomposition of BDI data.

Due to the complexity of the bulk shipping market and the non-linear nature of freight rates series [31], in this study, some data transformation techniques, including difference transform, power transform, log transform, standardization and normalization, were employed. As indicated in Table 1, for each transformation technique, we conducted a Dickey-Fuller stationary test to ensure its effectiveness.

**Table 1.** Dickey-Fuller test of pre-processing data.

Data	Test Statistics	<i>p</i> -Value	Critical Value 1%	Critical Value 5%	Critical Value 10%
Original data	−2.190442	0.209733	−3.437455	−2.864676	−2.568440
Normalization	−2.190442	0.209733	−3.437455	−2.864676	−2.568440
Standardization	−2.190442	0.209733	−3.437455	−2.864676	−2.568440
Log transform	−2.434086	0.132346	−3.437348	−2.864630	−2.568415
Power transform	−2.546063	0.104665	−3.437348	−2.864630	−2.568415
1st difference transform	−7.690264	$1.42 \times 10^{-11}$	−3.430000	−2.864676	−2.568440

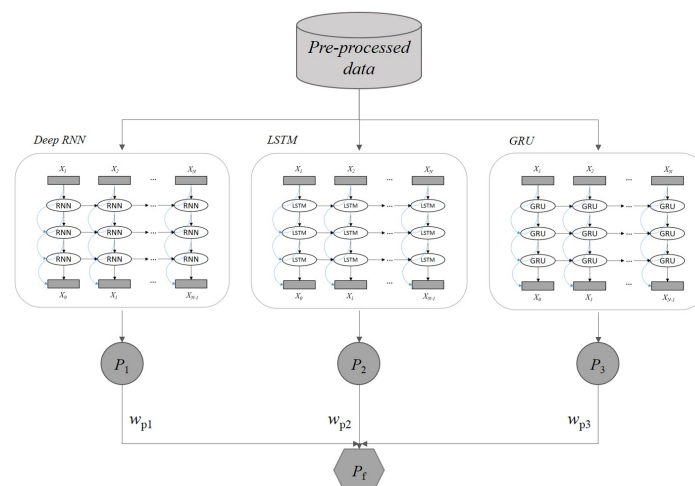
Normalization is a rescaling of data so that all values are within a certain range. As shown in Equations (1), (8) and (11), the value range of the  $\tanh$  function is between  $-1$  and  $1$ ; therefore, we shrunk the BDI to this range. Different from normalization, the standardization technique rescales the dataset according to the distribution of values so that the mean of the observed values is  $0$  and the standard deviation is  $1$ . Further, simple transformation techniques such as power transform and log transform are performed. The 1st difference transform applied to a time series  $x$  creates a new series  $z$  whose value at time  $t$  is the difference between  $x(t+1)$  and  $x(t)$ . This method works very well in removing trends and cycles. As shown in Table 1, ‘1st difference transform’ results in the

smallest  $p$ -value, which indicates that it generates more stationary time-series data than the original data. Therefore, in this research, we transformed the BDI data into a more stationary form by using ‘1st difference transform’, before feeding it into the model. The 1st-difference-transformed data is plotted in Figure 1b.

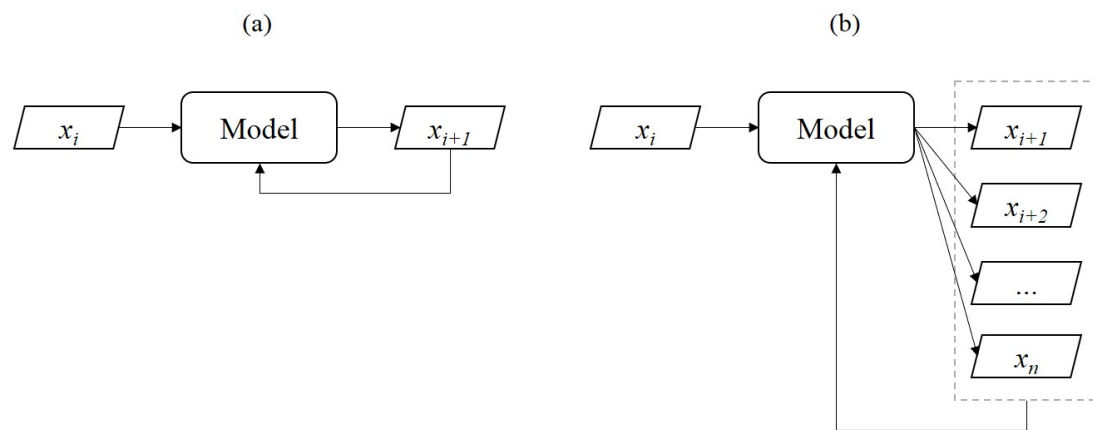
### 3.2. Deep Ensemble Recurrent Network

The design system of our approach is depicted in Figure 2, and the training process is expressed in Algorithm 1. Firstly, the pre-processed data is independently trained by Deep RNN, LSTM, and GRU. After those models converge in learning BDI data, in the testing phase, each of the predictions of RNN, LSTM, and GRU, represented as  $P_1$ ,  $P_2$  and  $P_3$ , are summarized using weighting  $w_{p1}$ ,  $w_{p2}$  and  $w_{p3}$ , respectively, to predict the  $P_f$  that represents the next value of the BDI. The values of  $w_{p1}$ ,  $w_{p2}$  and  $w_{p3}$  are learned in supervised learning using basic forward- and back-propagation techniques as in the neural network. As explained in Section 2.2, all of them are powerful sequence models. Deep RNN is the best recurrent model, which is able to learn time dependency to predict the next value; however, it cannot be combined with long-term dependency. LSTM is equipped with complex cell memory to handle long-term dependency. As for GRU, whereas it harnesses cell memory as well, it uses a simpler cell than LSTM. Therefore, for obtainment of more accurate results, we combined all of those models in an ensemble called deep ensemble recurrent network (DERN). The purpose behind this was to minimize the error rate, since in terms of memory cells, RNN is the simplest (under-fit) model, while GRU is a simpler model than LSTM albeit more complex (over-fit) than RNN, which complexity can be denoted as  $RNN < GRU < LSTM$ . Moreover, in machine-learning theory, there is no method that is universally better than any other method (the “no free lunch” theorem), and each method may make mistakes in different facets of operation. Stacking of multiple different sequential models may lead to performance improvement over individual models. The multi-model ensemble is a technique by which the predictions of the collection of models are given as inputs to a second-stage learning model. The second-stage model is trained to combine the predictions from the first-stage models optimally in order to obtain a final set of predictions.

The mechanism that we used to make short- and long-term predictions in the present study is depicted in Figure 3. Figure 3a is one-step-ahead (short-term) prediction model. Therefore, we transform the univariate time-series data of BDI into  $x_i$  as a predicted variable and  $x_{i+1}$  as a response variable. Afterwards, to predict the  $x_{i+2}$ ,  $x_{i+1}$  is appended to the training data. In Figure 3b meanwhile, the model predicts multi-values of BDI at a time; thus, it predicts multiple sequences from  $x_{i+1}$  to  $x_n$  at once, where  $n$  is the number of step predictions. The challenging part of this technique is creating a model that performs long-sequence prediction at once.



**Figure 2.** Ensemble deep RNN, LSTM and GRU model.



**Figure 3.** (a) Short-term (one-step-ahead) prediction; (b) Long-term (multi-steps-ahead) prediction.

---

**Algorithm 1: DERN**

```

Input :  $D = \{x_i, y_i\}_{i=1}^m$ 
output:  $P_f$ 
for  $t \leftarrow 1$  to  $T$  do
    | Learn  $M_1$  based on  $D$                                 //  $M_1$  is RNN model with its hyper parameter
end
for  $t \leftarrow 1$  to  $T$  do
    | Learn  $M_2$  based on  $D$                                 //  $M_2$  is LSTM model with its hyper parameter
end
for  $t \leftarrow 1$  to  $T$  do
    | Learn  $M_3$  based on  $D$                                 //  $M_3$  is GRU model with its hyper parameter
end
for  $t \leftarrow 1$  to  $m$  do
    |  $P_1 = D(M_1)$                                          // Predict  $P_1$  given data  $D$  and model  $M_1$ 
    |  $P_2 = D(M_2)$                                          // Predict  $P_2$  given data  $D$  and model  $M_2$ 
    |  $P_3 = D(M_3)$                                          // Predict  $P_3$  given data  $D$  and model  $M_3$ 
end
for  $t \leftarrow 1$  to  $T$  do
    | Learn  $w_{P_1}, w_{P_2}, w_{P_3}$  based on  $D$            //  $w_{P_1}, w_{P_2}, w_{P_3}$  is weight of  $P_1, P_2$  and  $P_3$ ,
    | respectively
end
for  $t \leftarrow 1$  to  $m$  do
    |  $P_f = P_1(w_{P_1}) + P_2(w_{P_2}) + P_3(w_{P_3})$ 
end

```

We conducted some extensive experiments to decide the hyper-parameters of our models. After some trials and errors, the optimal architecture of RNN, LSTM, and GRU are described in Table 2. We utilized two stacked of recurrent layers, such as two-layer stacked of RNN, LSTM, and GRU layer in RNN, LSTM, and GRU model, respectively. Each layer consist of 500 hidden unites with 20% dropout units and tanh as gate activation. We set the output of the recurrent layers to return a sequence. The length of the sequence corresponds to the number of steps of prediction. For instance, if the task is a five-steps-ahead prediction, the sequence length is five. The sequence will be wrapped by one TimeDistributed layer with the number of hidden units correspond to the sequence length, and will be activated by Sigmoid function. Each model is trained independently until converge by using Mean Square Error (MSE) as a loss function and Adam as an optimizer. In the ensemble layer,



the output of each model is trained by a standard neural network with one dense layer, Sigmoid activation function, and Adam optimizer is employed to decide the final prediction.

**Table 2.** Model configuration.

Hyper-Parameter	RNN	LSTM	GRU
Recurrent layer	2 RNN layer	2 LSTM layer	2 LSTM layer
Hidden units	500 of each layer	500 of each layer	500 of each layer
Gate activation	tanh	tanh	tanh
Dropout	20% of each layer	20% of each layer	20% of each layer
Wrapper layer	1 TimeDistributed layer	1 TimeDistributed layer	1 TimeDistributed layer
Hidden units	# of step prediction	# of step prediction	# of step prediction
Activation	Sigmoid	Sigmoid	Sigmoid
Loss function	MSE	MSE	MSE
Optimizer	Adam	Adam	Adam
Ensemble layer		1 Dense layer	
Activation		Sigmoid	
Loss function		MSE	
Optimizer		Adam	

To assess how well our prediction predicted short- and long-term BDI, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Error (MAPE), denoted in Equations (13)–(15), respectively, were employed, where  $n$  corresponds to the number of data,  $y_t$  is the actual value of BDI at time  $t$ , and  $\hat{y}$  corresponds to the predicted BDI at time  $t$ .

$$RMSE = \sqrt{\frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t)^2} \quad (13)$$

$$MAE = \frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t) \quad (14)$$

$$MAPE = \frac{100\%}{n} \sum_{t=1}^n \left| \frac{(y_t - \hat{y}_t)}{y_t} \right| \quad (15)$$

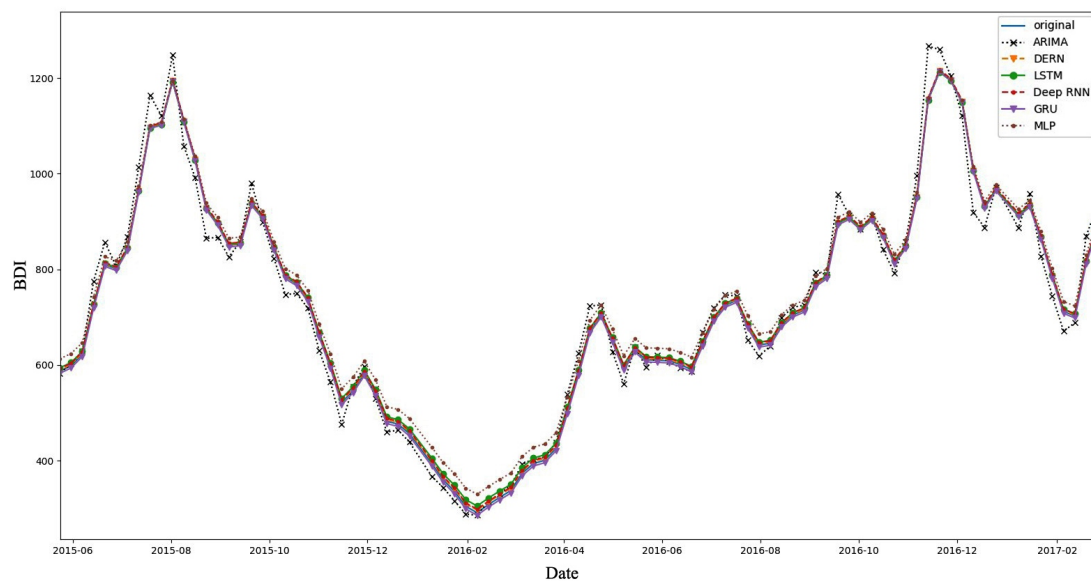
## 4. Experiments

The original BDI data is stored on a daily basis starting from November 1999 and extending to February 2018. To make our prediction simpler in performing long-term prediction, we sampled the data on a weekly basis by using average values. The summary statistics of BDI data is shown in Table 3. In the experiments, the data-shuffling technique was not implemented; instead, we employed a sliding window technique to split our training and testing data. The data was divided into 70–30%, 80–20% and 90–10% slices for training and testing. We compared the results with deep-learning models such as Deep RNN, LSTM, and GRU. Further, we compared our proposed method with ARIMA and Multi-Layer Perceptron (MLP). To obtain the best parameters and architectures, the grid search technique was employed.

### 4.1. Short-Term Prediction

Short-term prediction finds one-step-ahead BDI data; the predicting mechanism follows the model in Figure 3a, due to the fact that the data is weekly. Therefore, it predicts one week ahead of the BDI. Figure 4 depicts the prediction of BDI in the testing phase using ARIMA, MLP, LSTM, Deep RNN, GRU, and DERN. As we can see, all of them approximately predict the BDI correctly. In this case, the worst method is the ARIMA model. As shown in Table 4, the ARIMA error rate is three times higher than DERN in terms of RMSE. Notice that by changing the portion of training data, the error

rate is uncertain since the BDI data is volatile, as indicated in Table 3: the swings the data takes relative to the variance are enormously increased in the interval from 25 to 75%.



**Figure 4.** Short-term or one-step-ahead prediction of BDI in testing phase.

**Table 3.** Summary statistics of BDI data.

Parameter	Value
Count	4567.00
Mean	2417.19
Standard Deviation	2127.62
Minimum	290.00
25%	993.00
50%	1588.00
75%	3020.00
Maximum	11,793.00

**Table 4.** Experimental results of one-step-ahead (short-term) prediction of BDI.

Method	70–30%			80–20%			90–10%		
	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE
ARIMA	47.46	33.34	3.29	37.69	28.99	3.24	39.75	31.84	3.19
MLP	5.99	4.56	0.47	18.84	16.11	2.58	27.22	27.20	2.94
Deep RNN	5.92	4.45	0.46	4.23	3.89	0.59	<b>3.31</b>	2.84	0.36
LSTM	5.55	4.43	0.55	5.36	4.31	0.70	10.27	8.84	0.99
GRU	4.75	3.74	0.47	3.62	3.39	0.50	3.95	3.57	0.44
DERN	<b>2.53</b>	<b>2.13</b>	<b>0.44</b>	<b>1.98</b>	<b>1.58</b>	<b>0.45</b>	3.52	<b>2.99</b>	<b>0.31</b>

Table 4 shows that DEEP RNN, LSTM, and GRU roughly have a similar error rate. GRU slightly outperforms LSTM, while LSTM outperforms Deep RNN. This was due to GRU and LSTM having a cell memory gate to handle long-term dependency. It is known that GRU and LSTM have the same mechanism for effective tracking of long-term dependencies while mitigating the vanishing/exploding gradient problems. The LSTM uses more complex gates than GRU. Therefore, in this case, the LSTM model, relative to GRU, tended to over-fit more. The Figure 5 shows that DERN is the nearest result to the original data, and this is an average value among RNN, GRU, and LSTM, due to our having combined them into an ensemble using a weighted technique. Moreover, in its short-term prediction



of BDI, our approach outperforms the previous, Artificial Neural Network (ANN) model, for which the average of MAPE was never lower than five [5]. In the ensemble layer, we obtain the weight  $w_{p1}$ ,  $w_{p2}$ , and  $w_{p3}$  of RNN, LSTM and GRU, respectively. The value of  $w_{p1}$ ,  $w_{p2}$ , and  $w_{p3}$  are 0.337, 0.330, and 0.333, respectively. We can infer that in short-term prediction, each of the models has an approximately equal effect to the prediction value.

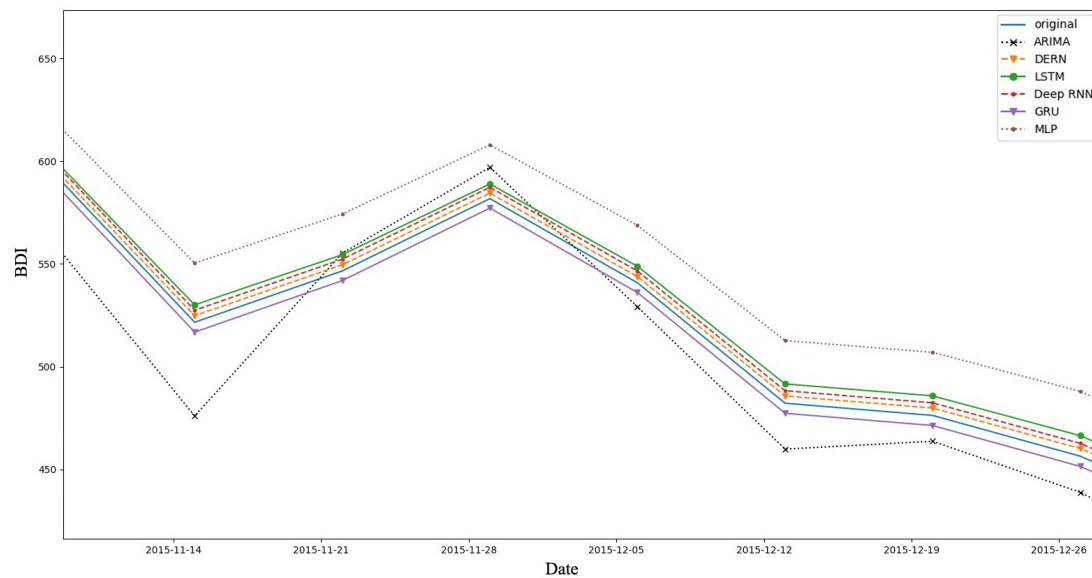


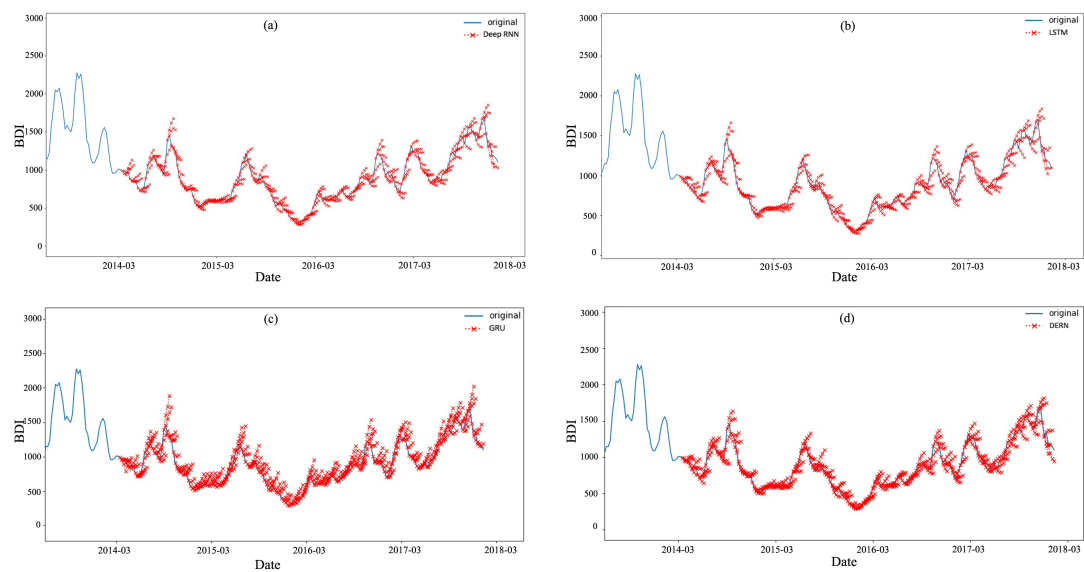
Figure 5. Zoom-in of Figure 4: DERN result in more accurate prediction.

#### 4.2. Long-Term Prediction

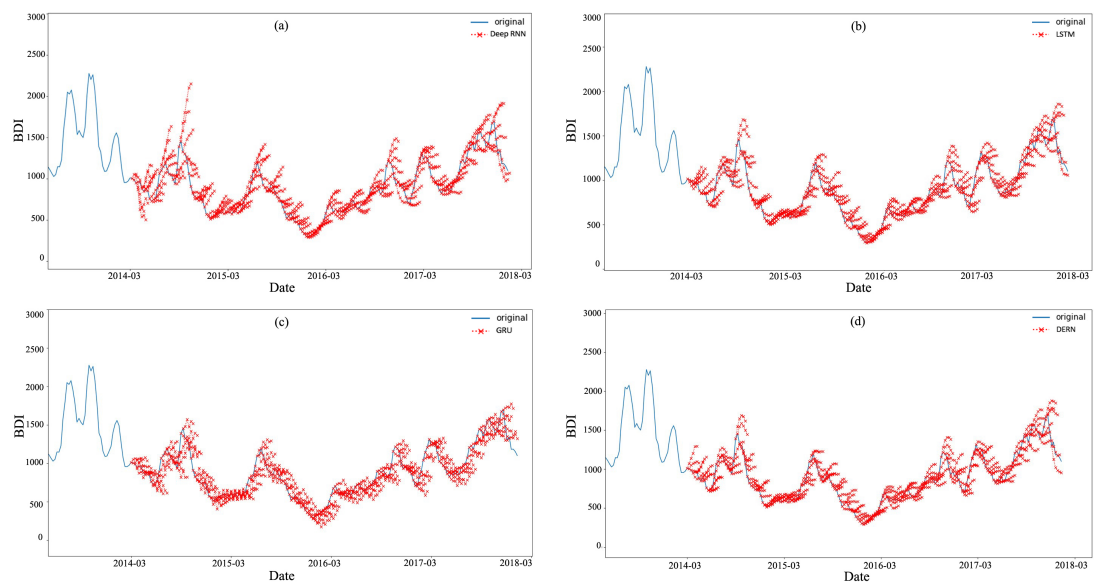
In long-term prediction, we try to predict the BDI more than one step ahead. In the present study, value three, five and seven weeks ahead of BDI were predicted. The experiment showed that long-term prediction resulted in a higher error rate than short-term prediction in terms of RMSE, MAE, and MAPE; therefore, we conducted the experiment up to seven-steps-ahead prediction. The experimental results of the long-term prediction are shown in Tables 5–7 for three-, five- and seven-steps-ahead prediction of BDI. As indicated in all of those tables, DERN obtained the best error rate among the methods. In this study, ARIMA failed to predict long-term BDI, its average error rate being more than seven times higher than that of our proposed method. A visualization of the comparison of RNN, LSTM, and GRU with DERN is provided in Figures 6–8 for three-, five-, and seven-ahead prediction of BDI, respectively. Notice that the error rate trend is increasing over time. Even though the error rate grows with the increasing number of steps, the models follow the trend of the testing data. The overall error rate averages in terms of RMSE, MAE and MAPE, respectively, are plotted in Figure 9. Note that ARIMA was omitted due to its large error. From the data, we could infer that one-step-ahead prediction results in a much lower error rate than that of long-term prediction. Therefore, long-term prediction is more challenging than short-term prediction; nonetheless, ship-owners and stakeholders commonly are more interested in long-term prediction. The weight of ensemble layer in three-steps-ahead prediction are 0.281, 0.357, and 0.362 for RNN ( $w_{p1}$ ), LSTM ( $w_{p2}$ ), and GRU ( $w_{p3}$ ) respectively. In the five-steps-ahead prediction is 0.300, 0.350, and 0.350 for  $w_{p1}$ ,  $w_{p2}$ , and  $w_{p3}$ , respectively. While in seven-steps-ahead prediction are 0.294, 0.362, and 0.344 for  $w_{p1}$ ,  $w_{p2}$ , and  $w_{p3}$ , respectively. Unlike in short-term prediction, RNN has a smaller effect on the final prediction, while LSTM and RNN are approximately the same effects to it.

**Table 5.** Experimental results of three-steps-ahead prediction of BDI.

Portion	Accuracy Measure	Model	$t_1$	$t_2$	$t_3$
70–30%	RMSE	ARIMA	701.29	789.60	989.10
		MLP	121.19	199.15	229.12
		Deep RNN	94.09	141.05	208.98
		LSTM	71.79	139.65	208.80
		GRU	70.93	133.71	<b>204.14</b>
		DERN	<b>64.12</b>	<b>123.98</b>	234.25
	MAE	ARIMA	693.11	732.51	932.43
		MLP	89.62	148.18	189.19
		Deep RNN	57.72	117.78	168.06
		LSTM	<b>55.62</b>	107.58	166.01
		GRU	56.63	108.76	146.91
		DERN	58.00	<b>99.96</b>	<b>126.03</b>
	MAPE	ARIMA	98.81	101.08	131.43
		MLP	15.31	17.91	25.93
		Deep RNN	9.08	13.61	20.93
		LSTM	6.45	12.73	20.46
		GRU	6.13	11.99	20.09
		DERN	<b>4.05</b>	<b>9.87</b>	<b>18.46</b>
80–20%	RMSE	ARIMA	694.93	872.92	991.80
		MLP	152.71	195.15	299.92
		Deep RNN	101.01	155.05	274.97
		LSTM	91.79	149.11	278.82
		GRU	91.10	134.31	269.22
		DERN	<b>89.04</b>	<b>132.93</b>	<b>258.21</b>
	MAE	ARIMA	901.97	1002.22	1032.43
		MLP	68.92	199.98	208.01
		Deep RNN	65.02	147.38	196.91
		LSTM	63.92	127.38	186.91
		GRU	62.97	128.48	176.91
		DERN	<b>60.57</b>	<b>126.35</b>	<b>167.76</b>
	MAPE	ARIMA	103.91	140.08	181.13
		MLP	17.78	25.71	36.44
		Deep RNN	12.28	17.73	33.04
		LSTM	9.78	15.73	23.99
		GRU	<b>9.62</b>	14.95	22.09
		DERN	10.01	<b>11.85</b>	<b>20.17</b>
90–10%	RMSE	ARIMA	894.03	1012.42	1300.10
		MLP	119.89	197.04	309.92
		Deep RNN	99.91	179.14	292.82
		LSTM	99.11	169.14	288.81
		GRU	101.07	159.25	278.91
		DERN	<b>98.22</b>	<b>147.50</b>	<b>268.62</b>
	MAE	ARIMA	981.07	1302.22	1687.03
		MLP	108.12	187.01	206.11
		Deep RNN	72.32	160.01	199.61
		LSTM	64.32	145.33	189.51
		GRU	63.92	<b>144.99</b>	189.01
		DERN	<b>60.61</b>	154.19	<b>175.11</b>
	MAPE	ARIMA	114.21	180.18	221.90
		MLP	16.95	24.95	33.91
		Deep RNN	11.65	18.55	29.20
		LSTM	<b>10.64</b>	16.63	24.01
		GRU	11.80	15.03	24.91
		DERN	10.71	<b>12.01</b>	<b>19.82</b>



**Figure 6.** Long-term prediction: three-steps-ahead prediction of BDI: (a) Deep RNN; (b) LSTM; (c) GRU; (d) DERN.



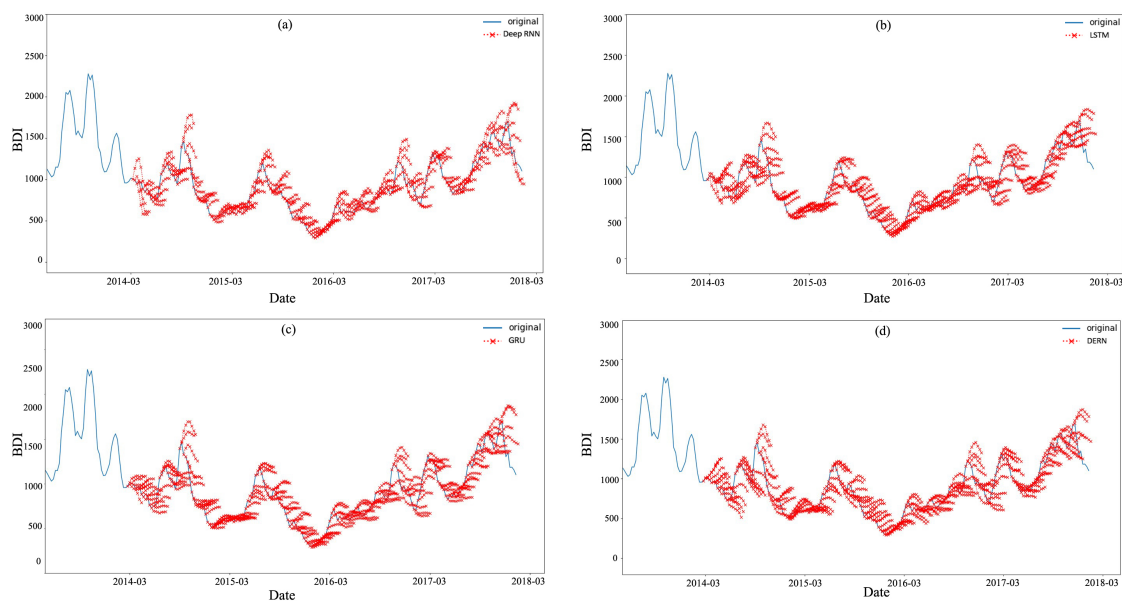
**Figure 7.** Long-term prediction: five-steps-ahead prediction of BDI: (a) Deep RNN; (b) LSTM; (c) GRU; (d) DERN.

**Table 6.** Experimental results of five-steps-ahead prediction of BDI.

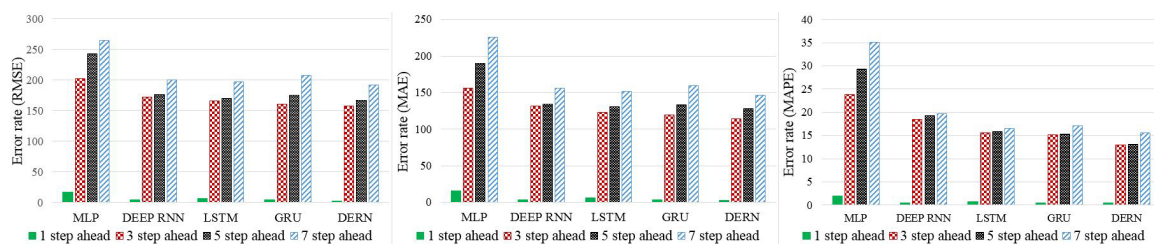
Portion	Accuracy Measure	Model	$t_1$	$t_2$	$t_3$	$t_4$	$t_5$
70–30%	RMSE	ARIMA	731.07	799.10	999.08	1009.98	1020.05
		MLP	145.10	209.31	241.97	255.06	259.99
		Deep RNN	91.96	160.28	214.43	257.35	290.34
		LSTM	79.40	142.33	201.69	248.45	293.33
		GRU	79.92	155.84	219.02	276.65	325.09
		DERN	<b>75.11</b>	<b>139.05</b>	<b>200.15</b>	<b>245.23</b>	<b>293.31</b>
	MAE	ARIMA	698.81	772.81	952.69	978.01	981.97
		MLP	94.92	158.19	191.09	201.44	210.03
		Deep RNN	64.38	111.64	153.25	188.69	214.41
		LSTM	58.09	99.40	141.41	180.80	212.31
		GRU	<b>57.54</b>	108.50	154.50	198.63	231.78
		DERN	57.90	<b>99.00</b>	<b>136.03</b>	<b>178.81</b>	<b>212.05</b>
	MAPE	ARIMA	100.01	121.18	141.03	152.87	164.09
		MLP	17.93	19.08	23.93	27.87	29.65
		Deep RNN	9.46	13.19	19.30	25.75	27.46
		LSTM	5.73	10.86	15.90	19.64	22.89
		GRU	5.58	10.62	15.17	19.51	22.95
		DERN	<b>4.95</b>	<b>9.85</b>	<b>12.96</b>	<b>17.00</b>	<b>19.32</b>
80–20%	RMSE	ARIMA	724.03	842.92	1000.80	1010.75	1024.62
		MLP	155.29	197.45	309.62	320.74	350.97
		Deep RNN	98.81	138.35	167.67	212.59	224.87
		LSTM	67.56	122.51	<b>160.39</b>	190.65	213.23
		GRU	62.96	119.82	160.87	191.39	214.23
		DERN	<b>62.04</b>	<b>118.98</b>	160.99	<b>190.01</b>	<b>210.00</b>
	MAE	ARIMA	900.17	1005.21	1042.33	1050.39	1072.11
		MLP	73.48	201.08	218.01	240.55	255.12
		Deep RNN	60.96	104.67	129.80	167.03	179.80
		LSTM	53.18	92.75	123.78	151.54	171.19
		GRU	49.10	88.78	122.20	148.33	170.18
		DERN	<b>48.56</b>	<b>87.39</b>	<b>121.76</b>	<b>144.72</b>	<b>170.00</b>
	MAPE	ARIMA	123.31	143.18	183.83	189.11	192.99
		MLP	19.18	26.17	35.41	37.11	39.98
		Deep RNN	9.07	14.24	20.76	24.53	27.81
		LSTM	6.93	11.87	17.74	20.26	23.93
		GRU	<b>6.54</b>	10.98	16.94	19.39	21.48
		DERN	6.50	<b>8.81</b>	<b>14.17</b>	<b>16.98</b>	<b>19.52</b>
90–10%	RMSE	ARIMA	890.01	1042.12	1370.07	1490.10	1640.97
		MLP	126.39	207.45	259.11	298.55	301.54
		Deep RNN	93.99	137.53	166.94	175.15	221.83
		LSTM	71.00	131.49	178.91	216.40	241.35
		GRU	70.43	127.65	172.92	210.40	243.12
		DERN	<b>67.29</b>	<b>127.51</b>	<b>168.92</b>	<b>209.55</b>	<b>240.76</b>
	MAE	ARIMA	991.47	1402.21	1707.09	1811.87	1991.08
		MLP	118.22	197.53	216.10	223.09	245.12
		Deep RNN	58.25	98.50	132.02	162.15	182.85
		LSTM	55.67	103.69	142.64	174.16	<b>194.45</b>
		GRU	56.07	<b>102.74</b>	139.42	172.02	205.66
		DERN	<b>55.61</b>	101.19	<b>135.19</b>	<b>171.04</b>	200.11
	MAPE	ARIMA	124.01	189.08	231.10	256.98	279.14
		MLP	18.35	27.91	36.04	39.43	42.76
		Deep RNN	8.99	13.98	20.28	25.15	28.20
		LSTM	6.73	11.95	16.01	22.25	24.06
		GRU	6.88	10.97	17.90	20.21	25.67
		DERN	<b>5.01</b>	<b>9.13</b>	<b>14.87</b>	<b>17.02</b>	<b>19.09</b>

Table 7. Experimental results of seven-steps-ahead prediction of BDI.

Portion	Accuracy Measure	Model	$t_1$	$t_2$	$t_3$	$t_4$	$t_5$	$t_6$	$t_7$
70–30%	RMSE	ARIMA	739.21	889.88	1008.98	1208.98	1568.72	1759.66	1973.26
		MLP	131.09	197.15	229.82	267.11	299.56	337.98	353.31
		Deep RNN	90.57	157.58	212.35	255.03	288.60	310.13	324.06
		LSTM	73.67	143.11	200.43	249.52	290.83	317.09	329.46
		GRU	77.74	150.71	214.46	270.16	323.02	364.61	390.65
		DERN	<b>69.53</b>	<b>143.08</b>	<b>198.09</b>	<b>240.92</b>	<b>287.43</b>	<b>300.41</b>	<b>320.04</b>
	MAE	ARIMA	699.93	734.08	962.73	974.23	999.00	1020.87	1127.11
		MLP	90.62	147.18	179.69	199.04	244.42	261.81	280.00
		Deep RNN	63.38	109.58	151.40	186.30	212.95	232.66	250.13
		LSTM	57.11	101.85	142.55	181.92	211.23	230.28	243.53
		GRU	56.63	107.01	153.64	196.62	232.58	264.67	290.33
		DERN	<b>56.00</b>	<b>99.93</b>	<b>136.03</b>	<b>167.39</b>	<b>201.04</b>	<b>225.01</b>	<b>237.33</b>
	MAPE	ARIMA	98.01	112.78	134.49	156.03	175.53	187.67	199.90
		MLP	14.91	15.91	18.93	22.09	25.00	26.22	29.34
		Deep RNN	7.45	12.17	16.48	19.99	21.93	24.14	31.27
		LSTM	5.37	<b>10.34</b>	14.40	18.44	21.51	<b>23.41</b>	24.79
		GRU	5.52	10.81	15.62	20.07	23.82	27.05	29.73
		DERN	<b>4.05</b>	10.87	<b>13.96</b>	<b>18.01</b>	<b>20.11</b>	23.48	<b>23.69</b>
80–20%	RMSE	ARIMA	699.23	887.92	1004.21	1220.89	1400.67	1502.77	1701.12
		MLP	159.01	181.25	209.92	220.90	250.11	270.99	300.86
		Deep RNN	87.46	123.43	166.68	192.01	219.95	237.83	255.86
		LSTM	62.74	117.33	155.29	186.05	209.26	227.93	242.47
		GRU	61.07	117.77	156.74	188.74	214.68	237.32	257.57
		DERN	<b>60.94</b>	<b>102.33</b>	<b>154.21</b>	<b>178.00</b>	<b>208.11</b>	<b>217.43</b>	<b>240.12</b>
	MAE	ARIMA	900.06	1001.22	1202.49	1442.11	1558.00	1872.71	1991.91
		MLP	69.91	198.18	218.31	236.16	257.12	277.09	296.66
		Deep RNN	69.08	93.11	129.07	153.14	177.12	191.42	211.68
		LSTM	55.76	86.98	118.25	147.24	168.10	182.84	194.88
		GRU	47.63	87.48	<b>118.09</b>	146.51	170.46	187.73	207.20
		DERN	<b>46.51</b>	<b>84.35</b>	118.76	<b>146.13</b>	<b>165.10</b>	<b>180.33</b>	<b>194.00</b>
	MAPE	ARIMA	102.90	141.08	185.03	198.01	210.77	240.11	290.19
		MLP	17.68	26.71	38.14	40.11	43.03	45.32	47.71
		Deep RNN	8.48	13.71	16.04	20.88	22.03	26.12	30.80
		LSTM	5.52	10.19	14.02	17.87	20.77	<b>22.84</b>	24.71
		GRU	5.52	10.27	14.09	17.95	21.30	23.67	26.11
		DERN	<b>5.51</b>	<b>10.12</b>	<b>13.17</b>	<b>17.05</b>	<b>19.09</b>	23.08	<b>24.50</b>
90–10%	RMSE	ARIMA	880.53	1002.41	1040.90	1321.11	1509.97	1610.11	1891.09
		MLP	129.80	199.19	319.95	342.65	364.12	390.17	402.11
		Deep RNN	71.73	126.45	164.00	195.55	219.68	243.05	264.41
		LSTM	67.87	127.67	172.43	207.99	230.10	253.48	276.82
		GRU	66.25	127.80	173.42	207.92	232.08	254.53	276.37
		DERN	<b>65.21</b>	<b>126.20</b>	<b>161.23</b>	<b>206.01</b>	<b>241.08</b>	<b>251.53</b>	<b>267.11</b>
	MAE	ARIMA	981.07	1312.12	1487.03	1700.76	1880.12	1976.32	2019.19
		MLP	109.12	180.31	233.11	258.88	298.12	320.11	379.11
		Deep RNN	56.73	97.33	129.47	<b>160.01</b>	180.24	<b>197.67</b>	220.86
		LSTM	53.68	100.67	139.20	168.14	188.49	201.61	217.05
		GRU	51.61	<b>98.23</b>	134.01	165.39	189.76	209.58	231.24
		DERN	<b>50.71</b>	99.45	<b>133.11</b>	160.09	<b>180.01</b>	195.11	<b>202.11</b>
	MAPE	ARIMA	124.91	190.38	251.93	271.33	296.39	301.11	329.81
		MLP	14.45	24.05	33.31	43.11	56.65	71.12	84.49
		Deep RNN	6.91	13.01	17.13	20.16	22.14	27.01	33.34
		LSTM	5.57	10.61	14.71	17.91	19.91	21.10	22.46
		GRU	<b>5.22</b>	9.95	13.47	16.60	19.14	21.15	23.23
		DERN	5.67	<b>9.13</b>	<b>11.82</b>	<b>15.84</b>	<b>18.01</b>	<b>19.22</b>	<b>21.21</b>



**Figure 8.** Long-term prediction: seven-steps-ahead prediction of BDI: (a) Deep RNN; (b) LSTM; (c) GRU; (d) DERN.



**Figure 9.** Averages of RMSE, MAE and MAPE for one-to-seven-steps-ahead prediction of BDI using MLP, Deep RNN, LSTM, GRU, and DERN, respectively.

## 5. Conclusions and Outlook

The Baltic Dry Index (BDI) is a parameter representative of international shipping activities. It is an essential tool with which ship-owners and stakeholders plan their maritime businesses and avoid market risk. Unlike common time-series data, the BDI index is characterized by volatility, non-stationarity, and complexity; therefore, its prediction is very challenging indeed. In previous work, most researchers have used the Artificial Neural Networks (ANN) and statistical models. In keeping with the popularity of deep learning in this decade, in this paper, we propose a deep-learning approach whereby the deep sequential models (RNN, LSTM, and GRU) are combined in an ensemble called Deep Ensemble Recurrent Networks (DERN) for accurate prediction of short- and long-term BDI. In short-term prediction using the RMSE indicator, DERN had an error rate roughly a half of GRU, LSTM, Deep RNN and MLP, and approximately a third of ARIMA. However, in long-term prediction, the error rate was not as good as the short-term prediction. Specifically, the results showed that with increasing prediction steps, the error rate grew. Therefore, the long-term prediction is more challenging than short-term prediction. Nonetheless, DERN still outperforms the conventional methods in long-term prediction. Ship-owners and stakeholders, not to mention investors, prevalently are more interested in long-term prediction. In future work, we will propose a more fine-grained approach entailing sequence-to-sequence learning for more accurate long-term prediction.

**Author Contributions:** Conceptualization, I.M.K. and H.B.; Data curation, I.M.K., S.S. and H.Y.; Formal analysis, I.M.K., H.B., S.S. and H.S.; Methodology, I.M.K.; Funding acquisition, H.Y.; Investigation, S.S.; Supervision, H.B.



and H.Y.; writing—original draft, I.M.K.; Writing—review & editing, H.B. and H.Y.; Project administrator, S.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by principal cooperative research of KMI (the Korea Maritime Institute) and the project titled ‘Development of IoT Infrastructure Technology for Smart Port’ funded by the Ministry of Oceans and Fisheries, Korea.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Bakshi, G.; Panayotov, G.; Skoulakis, G. The Baltic Dry Index as a Predictor of Global Stock Returns, Commodity Returns, and Global Economic Activity. In *Proceedings of the AFA 2012 Chicago Meetings*; Austin, TX, USA, **2011**. doi:10.2139/ssrn.1787757. [\[CrossRef\]](#)
2. Jacks, D.S.; Pendakur, K. Global Trade and the Maritime Transport Revolution. *Rev. Econ. Stat.* **2010**, *92*, 745–755, doi:10.1162/REST\_a\_00026. [\[CrossRef\]](#)
3. Sovilj, D.; Sorjamaa, A.; Yu, Q.; Miche, Y.; Séverin, E. OPELM and OPKNN in long-term prediction of time series using projected input data. *Neurocomputing* **2010**, *73*, 1976–1986. [\[CrossRef\]](#)
4. Cullinane, K.; Cape, M.K. A Comparison of Models for Forecasting the Baltic Freight Index: Box-Jenkins Revisited. *Int. J. Marit. Econ.* **1999**, *1*, 15–39, doi:10.1057/ijme.1999.10. [\[CrossRef\]](#)
5. Chou, C.; Lin, K.S. A Fuzzy Neural Network Model for Analysing Baltic Dry Index in the Bulk Maritime Industry. *Int. J. Marit. Eng.* **2017**, *159*, doi:10.3940/rina.ijme.2017.a2.410. [\[CrossRef\]](#)
6. Kamal, I.M.; Bae, H.; Sim, S.; Kim, H.; Kim, D.; Choi, Y.; Yun, H. Forecasting High-Dimensional Multivariate Regression of Baltic Dry Index (BDI) using Deep Neural Networks (DNN). *ICIC Express Lett.* **2019**, *13*, 427–434, doi:10.24507/icicel.13.05.427. [\[CrossRef\]](#)
7. Sahin, B.; Gürgeç, S.; Ünver, B.; Altın, I. Forecasting the Baltic Dry Index by using an artificial neural network approach. *Turk. J. Electr. Eng. Comput. Sci.* **2018**, *26*, 1673–1684, doi:10.3906/elk-1706-155. [\[CrossRef\]](#)
8. Zeng, Q.; Qu, C.; Ng, A.K.; Zhao, X. A new approach for Baltic Dry Index forecasting based on empirical mode decomposition and neural networks. *Marit. Econ. Logist.* **2016**, *18*, 192–210, doi:10.1057/mel.2015.2. [\[CrossRef\]](#)
9. Zhang, X.; Xue, T.; Eugene Stanley, H. Comparison of Econometric Models and Artificial Neural Networks Algorithms for the Prediction of Baltic Dry Index. *IEEE Access* **2019**, *7*, 1647–1657, doi:10.1109/ACCESS.2018.2884877. [\[CrossRef\]](#)
10. Ohhyver, M.; Pudjihastuti, H. Arima Model for Forecasting the Price of Medium Quality Rice to Anticipate Price Fluctuations. *Procedia Comput. Sci.* **2018**, *135*, 707–711, doi:10.1016/j.procs.2018.08.215. [\[CrossRef\]](#)
11. Henrique, B.M.; Sobreiro, V.A.; Kimura, H. Stock price prediction using support vector regression on daily and up to the minute prices. *J. Financ. Data Sci.* **2018**, *4*, 183–201, doi:10.1016/j.jfds.2018.04.003. [\[CrossRef\]](#)
12. Yolcu, O.C.; Lam, H.K. A combined robust fuzzy time series method for prediction of time series. *Neurocomputing* **2017**, *247*, 87–101, doi:10.1016/j.neucom.2017.03.037. [\[CrossRef\]](#)
13. Sagheer, A.; Kotb, M. Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing* **2019**, *323*, 203–213, doi:10.1016/j.neucom.2018.09.082. [\[CrossRef\]](#)
14. Sun, X.; Li, T.; Li, Q.; Huang, Y.; Li, Y. Deep belief echo-state network and its application to time series prediction. *Knowl. Based Syst.* **2017**, *130*, 17–29, doi:10.1016/j.knosys.2017.05.022. [\[CrossRef\]](#)
15. Wang, L.; Wang, Z.; Qu, H.; Liu, S. Optimal Forecast Combination Based on Neural Networks for Time Series Forecasting. *Appl. Soft Comput.* **2018**, *66*, 1–17, doi:10.1016/j.asoc.2018.02.004. [\[CrossRef\]](#)
16. Tealab, A. Time series forecasting using artificial neural networks methodologies: A systematic review. *Future Comput. Inform. J.* **2018**, *3*, 334–340, doi:10.1016/j.fcij.2018.10.003. [\[CrossRef\]](#)
17. Salles, R.; Belloze, K.; Porto, F.; Gonzalez, P.H.; Ogasawara, E. Nonstationary time series transformation methods: An experimental review. *Knowl. Based Syst.* **2019**, *164*, 274–291, doi:10.1016/j.knosys.2018.10.041. [\[CrossRef\]](#)
18. Mabrouk, A.B.; Abdallah, N.B.; Dhifaoui, Z. Wavelet decomposition and autoregressive model for time series prediction. *Appl. Math. Comput.* **2008**, *199*, 334–340, doi:10.1016/j.amc.2007.09.067. [\[CrossRef\]](#)
19. Joo, T.W.; Kim, S.B. Time series forecasting based on wavelet filtering. *Expert Syst. Appl.* **2015**, *42*, 3868–3874, doi:10.1016/j.eswa.2015.01.026. [\[CrossRef\]](#)

20. Godfrey, L.B.; Gashler, M.S. Neural Decomposition of Time-Series Data for Effective Generalization. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 2973–2985, doi:10.1109/TNNLS.2017.2709324. [[CrossRef](#)]
21. Galicia, A.; Talavera-Llames, R.; Troncoso, A.; Koprinska, I.; Martinez-Alvarez, F. Multi-step forecasting for big data time series based on ensemble learning. *Knowl. Based Syst.* **2019**, *163*, 830–841, doi:10.1016/j.knosys.2018.10.009. [[CrossRef](#)]
22. Adhikari, R. A neural network based linear ensemble framework for time series forecasting. *Neurocomputing* **2015**, *157*, 231–242, doi:10.1016/j.neucom.2015.01.012. [[CrossRef](#)]
23. Williams, R.J.; Peng, J. An Efficient Gradient-Based Algorithm for On-Line Training of Recurrent Network Trajectories. *Neural Comput.* **1990**, *2*, 490–501, doi:10.1162/neco.1990.2.4.490. [[CrossRef](#)]
24. Bengio, Y.; Simard, P.; Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **1994**, *5*, 157–166, doi:10.1109/72.279181. [[CrossRef](#)]
25. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780, doi:10.1162/neco.1997.9.8.1735. [[CrossRef](#)] [[PubMed](#)]
26. Deng, Y.; Jiao, Y.; Lu, B.L. Driver Sleepiness Detection Using LSTM Neural Network. In *Neural Information Processing*; Cheng, L., Leung, A.C.S., Ozawa, S., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 622–633.
27. Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*; Association for Computational Linguistics: Doha, Qatar, 2014; pp. 1724–1734, doi:10.3115/v1/D14-1179. [[CrossRef](#)]
28. Chung, J.; Gülçehre, Ç.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv* **2014**, arXiv:1412.3555.
29. Wang, Y.; Liao, W.; Chang, Y. Gated Recurrent Unit Network-Based Short-Term Photovoltaic Forecasting. *Energies* **2018**, *11*, 2163, doi:10.3390/en11082163. [[CrossRef](#)]
30. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems—Volume 1*; Curran Associates Inc.: Red Hook, NY, USA, 2012; NIPS’12, pp. 1097–1105.
31. Goulielmos, A.M.; Psifia, M.E. Forecasting weekly freight rates for one-year time charter 65,000 dwt bulk carrier, 1989–2008, using nonlinear methods. *Marit. Policy Manag.* **2009**, *36*, 411–436, doi:10.1080/03088830903187150. [[CrossRef](#)]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).