IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.13, No.1, January 2019, pp. 31~42
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: https://doi.org/10.22146/ijccs.39071

Adaptive Moment Estimation On Deep Belief Network For Rupiah Currency Forecasting

Abram Setyo Prabowo*[1], Agus Sihabuddin[2], Azhari SN[3]
[1]Program Studi S2 Ilmu Komputer,FMIPA UGM, Yogyakarta, Indonesia
[2,3]Departemen Ilmu Komputer dan Elektronika, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1]abram.s@mail.ugm.ac.id**, [2]a_sihabudin@ugm.ac.id, [3]arisn@ugm.ac.id

Abstrak

Salah satu pendekatan yang sering digunakan pada peramalan adalah jaringan syaraf tiruan (JST), namun JST mempunyai permasalahan dalam menentukan nilai bobot awal antar koneksi, waktu yang lama untuk mencapai konvergen, dan permasalahan lokal minimum.

Model Deep Belief Network (DBN) diusulkan untuk meningkatkan kemampuan JST dalam meramalkan nilai tukar. DBN tersusun dari tumpukan Restricted Boltzmann Machine (RBM). Struktur DBN secara optimal ditentukan melalui eksperimen. Metode Adam diterapkan untuk mempercepat pembelajaran dalam DBN karena mampu mencapai hasil yang baik dengan cepat dibandingkan metode optimasi stokastik lain seperti Stochastic Gradient Descent (SGD) dengan mempertahankan tingkat pembelajaran untuk setiap parameter .

Pengujian dilakukan pada data nilai tukar harian USD/IDR dan empat kriteria evaluasi diadopsi untuk mengevaluasi kinerja metode yang diusulkan. Model DBN-Adam menghasilkan RMSE 59.0635004, MAE 46.406739, MAPE 0.34652, dan Directional Accuracy sebesar 66,67%. DBN-Adam juga mampu mencapai titik konvergen dengan cepat, di mana hasil ini mampu mengungguli model DBN-SGD.

Kata kunci—*DBN, Deep Belief Network, Adam, Optimasi Gradient Descent,* Peramalan

Abstract

One approach that is often used in forecasting is artificial neural networks (ANN), but ANNs have problems in determining the initial weight value between connections, a long time to reach convergent, and minimum local problems.

Deep Belief Network (DBN) model is proposed to improve ANN's ability to forecast exchange rates. DBN is composed of a Restricted Boltzmann Machine (RBM) stack. The DBN structure is optimally determined through experiments. The Adam method is applied to accelerate learning in DBN because it is able to achieve good results quickly compared to other stochastic optimization methods such as Stochastic Gradient Descent (SGD) by maintaining the level of learning for each parameter.

Tests are carried out on USD / IDR daily exchange rate data and four evaluation criteria are adopted to evaluate the performance of the proposed method. The DBN-Adam model produces RMSE 59.0635004, MAE 46.406739, MAPE 0.34652. DBN-Adam is also able to reach the point of convergence quickly, where this result is able to outperform the DBN-SGD model.

Keywords—*DBN, Deep Belief Network, Adam, Gradient Descent Optimazation, Forecasting*

Received September 24th,2018; Revised January 23th, 2019; Accepted January 28th, 2019

1. INTRODUCTION

Currency exchange transactions occur because of the difference between demand and supply for a particular currency at the same time which causes the currency to fluctuate. The difference in this difference is then used to gain profit through currency exchange forecasting. Predicting the volatility of currency exchange rates accurately is difficult to obtain, therefore many approaches are taken, one of which is the use of artificial neural networks. Extensive adaptation and learning abilities in the context of non-linear functions that can approach continuous functions with the desired accuracy make artificial neural networks (ANN) an option in performing currency exchange forecasting approaches sing. Artificial neural networks have the best accuracy compared to statistical methods such as ARIMA (Autoregressive Integrated Moving Average) and Regression [1]. However, ANN has problems in determining the initial weight value between connections, a long time to reach convergent, and minimum local problems. The deep learning is an option to overcome the limitations of classical artificial neural networks, where classical neural networks are difficult to determine the initial weight between connections, a long time to reach convergent, and minimum local problems. One of the models used is using the DBN (Deep Belief Network) model by using a stack of unsupervised learning layers, namely RBM (Restricted Boltzmann Machines) as a pre-training process and supervised learning as a fine-tuning process and has succeeded in solving problems such as classification, Dimension reduction, forecasting, and information retrieval [2][3][4][5].

Estimating the currency exchange rate's accuracy is difficult to obtain so that it affects economic players to make decisions, but estimating the direction of a forecasting model is a little easier by utilizing the increase in value from the Directional Accuracy (DA). The net increase always depends on the size of the change as well, but the accuracy generated from a forecasting model of more than 60% can be used as a profitable forecasting model [6][7].

Research using DBN has been done and evaluates conjugate gradient as a fine tuning algorithm on weekly data exchange rate forecasting [8], do comparison some methods like Feed Forward Neural Network (FFNN), Random Walk (RW) and Auto-Regressive and Moving Average (ARMA). The DA value generated by DBN-CG at the exchange rate of GBP /USD is 0.636 and is a high value compared to other models. At the INR / USD exchange rate, DBN-CG produces a DA value that is slightly smaller than the conventional DBN which is 0.567, while the conventional DBN is 0.587. However, the two DBNs have the best value compared to the FFNN, ARMA, or RW methods. The error values generated by the two DBNs are also relatively small compared to other codes. Conventional DBN values are 1.6505E-2, DBN-CG 1.7000E-2, FFNN 1,8899E-2, ARMA 8,7135E-2 and RW 9,8488E-2. This study agrees that DBN has good predictive accuracy, but also high stability than FFNN, ARMA, or RW. DBN development continues to be done by combining many algorithms as weight updates [9][10].

A DBN model is needed that produces a good-fitting model and is able to achieve convergence with consistent loss. There are several ways to improve learning in deep learning such as improving architecture, finding optimal parameters, playing with data representation, choosing the best optimization algorithm and so on. The optimization that is often used in deep learning is Stochastic Gradient Descent (SGD) as a derivative of Gradient Descent [11]. However, there is a problem with SGD that causes the gradient to be stuck at a certain minimum point because weight updates use data one by one, causing fluctuations in the cost function. One way to optimize SGD is the use of Adam's algorithm for deep learning because it achieves good results quickly compared to other stochastic optimization methods [12]. Basically, the Adam algorithm maintains the learning level for each network weight (parameter) and is adapted separately when learning takes place from the estimated first and second moments of the gradient.

Initiation of weighting and convergent models are the most important thing in a learning network. Therefore, this study utilizes Adam's algorithm as a fine-tuning process in DBN so that a forecasting model is obtained which has good learning ability and the desired accuracy value.

2. METHODS

2.1 Deep Belief Network

Deep Belief Network (DBN) is a multilayer and generative neural network composed of RBM (Restricted Boltzmann Machines) layers, each layer has a greedy layer-wise algorithm as a DBN training algorithm and is a graphical model that learns to extract hierarchical representations in depth from a training data so that it can extract features high-dimensional data [2] . The DBN training process includes two stages: pre-training with unsupervised learning and fine-tuning with supervised learning. **Figure 1** shows the DBN architecture.

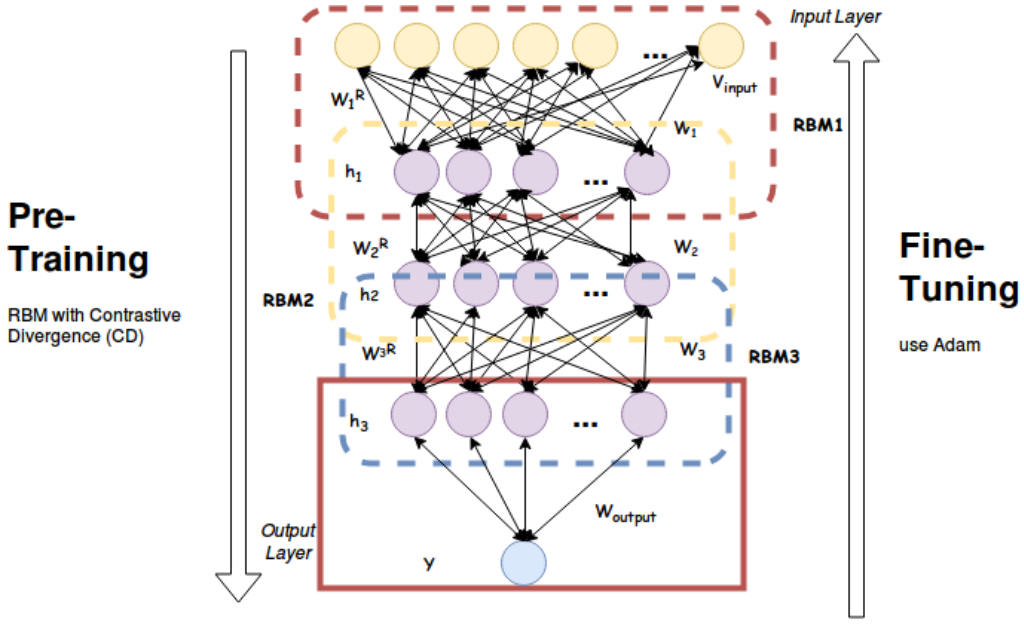

Figure 1 DBN Architecture

2.1.1 Pre-Training

The pre-training is the process of initiating network parameters namely connection weights and biases between each layer using unsupervised learning in each layer. The steps in this pre-training process can also be called a greedy learning step on DBN. The steps from the pre-training stage are as follows [2] :

a) Train RBM by entering data and improving RBM parameters
b) Use the output as a second RBM input by sampling or by calculating the activation of hidden units.
c) The steps above can be done again to get DBN with several layers whose parameters are suitable for extracting features from the type of data entered.

A Restricted Boltzmann Machines (RBM) consists of two different layers of interconnected units. RBM consists of one visible node layer and one hidden unit layer. **Figure 2** shows the RBM structure.

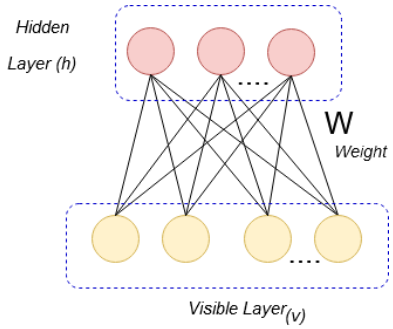

Figure 2 RBM Structure [10]

Equation (1) and **Equation (2)** is the evolution of RBM on each node that randomly evaluates 1 or 0 that adheres to the posterior probability$P$, which w is represented to represent the connection weights between hidden units (h) and visible units (v), while $b$and a are the values passed from each layer and f is an energy function.

$$P(h_i = 1|v) = f(b_i + W_i v) \quad (1)$$
$$P(v_i = 1|h) = f(a_i + W_i h) \quad (2)$$

RBM can be as a classification, regression or generative model by adding either a single regression label or Softmax class label to the visible unit allowing for supervised learning, and a trained model can produce representative samples of data distribution that flank the visible unit [10]. If v_i and hj are a representation of the visible unit and hidden unit and w_{ij}=w_{ji} is the bidirectional weight. So the probability of each unit is like the **Equation (3)** - **Equation (6)**.

$$v_i = \frac{1}{e^{-\Delta E_j}} \quad (3)$$
$$\Delta E_j = \Sigma^{n} i = 1 w_{ij} v_i + b_j \quad (4)$$
$$h_i = \frac{1}{e^{-\Delta E_i}} \quad (5)$$
$$\Delta E_i = \Sigma^{m} j = 1 w_{ij} h_i + b_j \quad (6)$$

RBM training is carried out for the first time in the visible unit (v_i) according to equation (3) so that the hidden unit (h_i) will be passed on to **Equation (5)**. This process is repeated once again to update the visible unit (v_i) and hidden unit (h_i) to produce a reconstruction step $v'_i h'_j$ so that the update weights w_{ij} can be seen in **Equation (7)**.

$$\Delta w_i = \eta(vh^T - v'^T h') \quad (7)$$

Equation (7) can be explained where η is the learning rate refers to the mean of all training data. The calculation Δw_{ij} to calculate the difference in changes in weight to be carried out. The calculation is done from the learning rate multiplication η with the average data of values v and value h minus the reconstructed value $v'^T h'$. RBM reconstruction trained using Contrastive Divergence (CD). In this experiment use as much RBM as the number of hidden layers in DBN to build the DBN model. Because each new layer is added, the overall generative model gets better. This learning process continues until a number of hidden layers required in the DBN have been trained.

2.1.2 Fine-Tuning

At the end of pre-training, each layer of RBM can obtain initialized parameters, which consist of the main structure of the DBN. So, DBN fine-tuning the entire structure. The first process in fine-tuning is the advanced stage of calculating the output of each unit based on input to the last layer. The second process is the calculation of parameters in the backward stage where the weight is updated as much as the error that occurs in the output with the data label. This step is done until the number of epochs is met. This study applies supervised learning, namely Adam's algorithm to improve the final layer to the initial layer of DBN.

2.2 Adam

Adam (Adaptive Moment Estimation) is a combination of AdaGrad and Rmsprop [12], which combines the advantages of both algorithms that are able to maintain an adaptive level of learning for each parameter. The idea from Adam to further optimize the performance of Stochastic Gradient Descent (SGD) is to adjust the level of learning per parameter based on the average of the first moment that runs exponentially and utilizes the average second-moment gradient. Adam stores the result of the exponential mean decay from the square gradient as applied to Rmsprop, and saves the exponential mean decay from the gradient and this process is the same as using momentum.

$$m_t = \beta_1 m_t - 1 + (1 - \beta_1) g_t. \quad (8)$$

$$v_t = \beta_2 v_t - 1 + (1 - \beta_2) g_t^2 \quad (9)$$

Equation (8) and **Equation (9)** are the values of the first moment estimation$m_t$and the estimated values in the second moment v_t, which are then initiated as vectors 0, these values are biased towards zero especially during the initial steps and when the decay value is small. To overcome the refraction problem, a step is needed to calculate the bias correction of the first moment estimation and second-moment estimation. The bias correction can be seen in **Equation (10)** and **Equation (11)**.

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \quad (10)$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \quad (11)$$

After correcting the bias, the next step is to repair the weight using **Equation (12).**

$$\theta_t + 1 = \theta_t - \frac{\eta}{\sqrt{v_t}+\varepsilon} \hat{m}_t \quad (12)$$

The recommended value as the default value is 0.9 for β_1 and 0.999 for β_2, and 10e-8 for ε epsilon. The value of this parameter gives results and can work well compared to other optimizations such as Rmsprop and Adagrad [11] while η is a measure of the learning step.

2.3 Testing and Performance Measures

In this paper, four criteria are used to evaluate the performance of DBN in forecasting exchange rates. The four criteria are Root Mean Square error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and Direction Accuracy (DA). The formula of the four prediction accuracy measures is listed as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{T}(y_i - y'_i)^2}{T}} \quad (13)$$

$$MAE = \frac{\sum_{i=0}^{T}|y_i - y'_i|}{T} \quad (14)$$

$$MAPE = \frac{1}{T}\sum_{i=0}^{T}\left|\frac{y_i - y'_i}{y_i}\right| x100\% \quad (15)$$

$$DA = \frac{1}{T}\sum_{i-1}^{T} a_i \, x100\% \quad (16)$$

Equation (16), where $a_i = 1$ if $(y_{i+1} - y_i)(y'_i - y_i) > 0$, $a_i = 0$ is the opposite. The notation a_i is the error difference (actual value y_i - prediction value y'_i). The higher the DA value of a model, the better the performance of the model. The financial industry standard for a forecasting model is that the model can achieve an accuracy value equal to 60% or more [13].

2.4 Data Preparation

This study uses Rupiah / US Dollar (USD/IDR) exchange rate data. All data used are daily exchanges taken from the period of 20th October 2014 to 24thOctober 2017 with a total of 754 daily data and are univariate one variable. Data sources are downloaded from the "Pacific Exchange Rate Service" (http://fx.sauder.ubc.ca/data.html). Daily data obtained from related sources does not include data for holidays.

2.4.1 Data Distribution

This study keep 80% for training data, 5% for validation data and 15% for testing data [14][7] . In addition, this study uses a sliding windows technique to separate data [8] to recognize patterns based on the order of time values. The number of sliding windows affects the level of recognition of features by DBN but there is no standard rule in determining the number of sliding windows. The input unit to be tried is 3,4,5,6,7 days and the output is one day after.

2.4.2 Normalization

, The normalize the sliding windows data is calculated using min-max, and values are between [0,-1] or [-1-1]. Normalization calculations can be done with the following rules:

$$y_{\text{new}} = \frac{y - y_{\min}}{y_{\max} - y_{\min}} \quad (17)$$

The notation y shows the currency exchange rate in the period t while y_{new} the value after normalization as input into the DBN. The notation $y_{\max}$ is the maximum data value and $y_{\min}$ is the minimum value of data. After data normalization, the predicted output data will be converted to its original value.

2.5 Model Architecture Design

Determining the number of input units, hidden units, and hidden layers are one way to determine the architecture used. Units that are too small will result in poor forecasting, while too many units will produce poor forecasting. The architectural design of this study includes the following:

a) Input Layer.

Make sliding windows of varying sizes, as discussed earlier that there is no standard rule in determining the input so that it will be done by experiment. The input layer is used as a data receiver for forecasting needs.

b) RBM layer

This layer is used for the initial process of weight initialization in the unsupervised training process. The number of RBM layers affects the ability of DBN to recognize features that are in the data. The number of layers more than one can increase the ability to recognize the DBN feature because the features generated from the previous RBM will be recognized by the next RBM layer.

c) Output Layer

This layer is the prediction layer of the currency exchange rate. In this layer, the fine-tuning process will be carried out, the prediction results are calculated error values using the Adam algorithm.

3. RESULTS AND DISCUSSION

3.1 Hyperparameter Selection in the Model

The testing phase in this study consists of finding the optimal value in the form of a network architecture consisting of input units and units in the hidden layer, learning rate search, and testing of the model using test data. Hyperparameter selection until the testing process is carried out using the DBN-Adam model.

3.1.1 Input Unit and Hidden Unit Selection in the Layer

Determining the number of input units and the number of layers does not have a specific rule. The selection of input units uses 5 variations of input units, namely 3,4,5,6, and 7, while for hidden unit variations using 5 variations namely 4,8,12,16, and 20 for the first layer.

Table 1 Effects of DBN factors: input and hidden nodes in first layer

Input	Hidden1	RMSE	MAE	MAPE	DA (%)
3	4	54.50136	42.31136	0.318034	54.285714
3	8	50.80322	37.59514	0.28235	65.714286
3	12	51.63197	38.2115	0.287084	57.142857
3	16	51.50844	37.88215	0.284335	62.857143
3	20	51.46384	37.79782	0.283667	62.857143
Avg		51.98177	38.75961	0.291094	60.5714286
4	4	52.55554	39.47751	0.296253	55.882353
4	8	53.22364	39.95021	0.299712	55.882353
4	12	53.66837	41.14105	0.309161	52.941176
4	16	52.16483	38.85447	0.291775	61.764706
4	20	53.37667	39.29181	0.294839	50
Avg		52.99781	39.74301	0.298348	55.2941176
5	4	54.53372	43.60793	0.327839	51.515152
5	8	52.82718	40.24626	0.302425	60.606061
5	12	54.26512	41.54559	0.311888	51.515152
5	16	54.76357	40.70297	0.305573	57.575758
5	20	54.98723	40.00192	0.300261	60.606061
Avg		54.27537	41.22093	0.30959	56.36363
6	4	51.09957	38.59629	0.28975	59.375
6	8	52.55881	40.59982	0.30487	56.25
6	12	52.65667	40.64565	0.30523	59.375
6	16	56.01914	41.34404	0.31037	53.125
6	20	64.64951	52.04165	0.39130	46.875
Avg		55.39674	42.64549	0.32030	55
7	4	56.89655	44.25508	0.33256	51.61290
7	8	54.27837	40.950725	0.30722	54.83871
7	12	55.94016	42.434769	0.31868	54.83871
7	16	57.26383	43.793992	0.32917	54.83871
7	20	59.50985	46.292817	0.34800	54.83871
Avg		56.77775	43.5454782	0.32713	54.19354

Table 1 shows the values of RMSE, MAE, MAPE, and Direction Accuracy (DA) from the results of the combination of input units and hidden units for the first layer. Among the four measurements above, DA is one of the most important evaluation criteria. DA is considered capable of representing the possible direction of correct prediction. An economic actor is more interested in knowing changes in the direction of the exchange rate in the future because it helps them to carry out a trading strategy. Therefore, in determining the number of input units, DA is chosen as an assessment by taking the highest average value of DA on the number of inputs. The results above show the best average DA value occurs in the number of inputs 3 with a value

of 60.57142. The next step is to determine hidden units, based on **Table 1**, the best DA value in input 3 occurs in hidden unit 8, which is 65.71428. Therefore, the first layer is composed of 3 input units and 8 hidden units.

Table 2 Effects of DBN factors: input and hidden nodes in second layer

Input	Hidden1	Hidden2	RMSE	MAE	MAPE	DA (%)
3	8	4	52.15580	39.46019	0.29617	68.57142
3	8	8	50.50988	37.35169	0.28039	65.71428
3	8	12	53.29481	38.89791	0.29177	54.28571
3	8	16	54.52670	43.06564	0.32371	48.57142
3	8	20	54.31358	41.14505	0.30855	60

After having 3 input units and 8 hidden units in the first layer, the next step is to determine the hidden unit in the second layer. The proposed hidden unit is the same as in the first layer, using 5 hidden unit variations namely 4,8,12,16, and 20, the results of this combination can be seen in **Table 2**. The results show the best DA values occur in hidden units 4 with values amounting to 68.571429 so that in this second layer the network architecture obtained is 3-8-4. The next experiment redefined the hidden unit in the third layer using the same variation as the previous experiment, it is shown in **Table 3.**

Table 3 Effects of DBN factors: input and hidden nodes in third layer

Input	Hidden1	Hidden2	Hidden3	RMSE	MAE	MAPE	DA (%)
3	8	4	4	50.82998	38.25094	0.28734	60
3	8	4	8	50.52119	37.74079	0.28342	65.71428
3	8	4	12	51.44099	39.74157	0.29858	60
3	8	4	16	50.44826	37.64935	0.28274	65.71428
3	8	4	20	62.42777	49.41928	0.37154	45.71428

The best DA values occur in hidden units occur in hidden units 8 and 16, have the same value of 65.71428, but the smallest error value is generated by the hidden unit 16 with the RMSE value of 50.44826. After trying several combinations, the optimal architecture that has been found is 3-8-4-16-1 which consists of 3 input units, 8 units of RBM1, 4 units of RBM2, 16 units of RBM3, and 1 unit of output.

3.1.2 Learning Rate Selection

Selection of Learning rate that is too small leads to slow convergence, while learning rate that is too large can inhibit convergence and cause the loss function to fluctuate around the minimum or even deviate so that the selection of learning rate hyperparameter according to the architecture is obtained, then it will be compared between iteration and value loss generated by the model as much as the iteration done. This experiment uses 100 iterations.

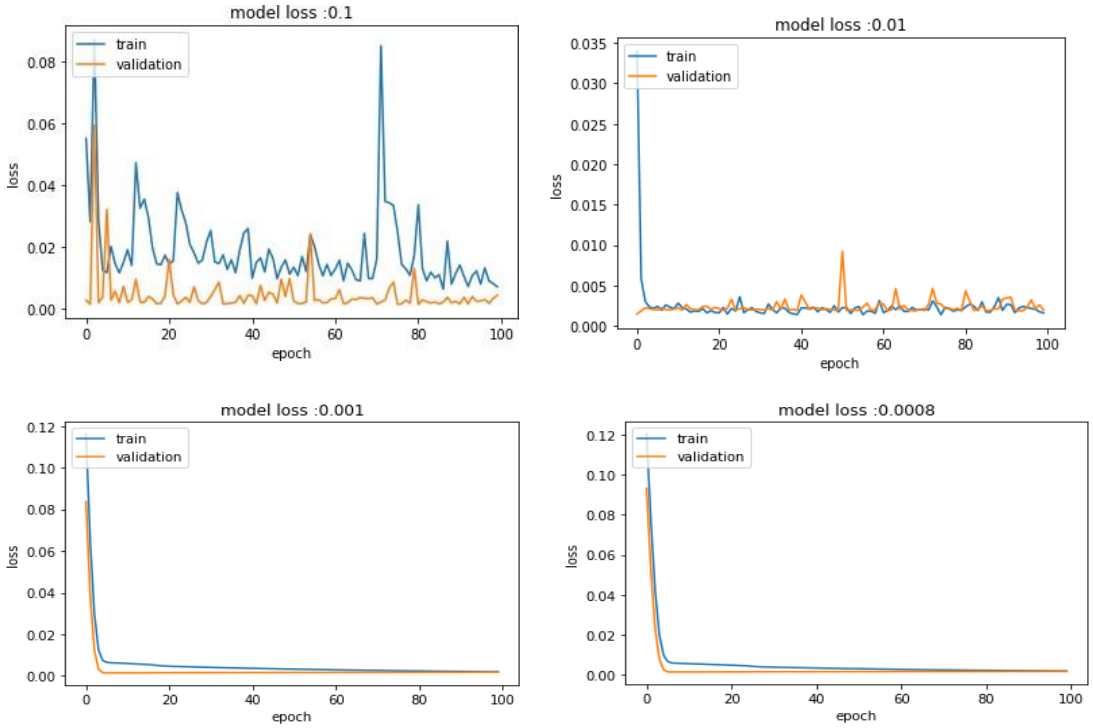

Figure 3 Learning rate graph train loss

Figure 3 is a graph of the decrease in the value of the loss to the lowest point of a learning rate, not fluctuating, and no overfitting. Generally, the value used is starting from 0.1, and then progressively decreasing the value (for example being 0.01, 0.001 and so on) until the perceived model produces a good loss graph to the lowest point without fluctuations. this study takes 0.001 as parameter.

3.2 Testing

The test is carried out using parameters that have been previously obtained. Based on **Table 4** the model results in a small error value with RMSE 59.063, MAE 46.406, MAPE 0.3468 and able to provide DA value of 66.66% and this result has exceeded the minimum standard of industry which is 60%. The performance obtained from the DA value is important for financial industry players who are more concerned with predictive value so that actors in this field are able to predict whether the exchange rate will rise or fall compared to how large or small the error value is in the RMSE.

Table 4 Performance results of the DBN-ADAM model for forecasting

RMSE	MAE	MAPE	DA (%)
59.06300	46.40673	0.34685	66.66667

3.3 fects of Eta η

The value of Adam's algorithm is a hyperparameter that controls the learning step or known as step size or learning rate. In this test, the value is configured with a predetermined value of 0.1.0.01.0.001.0.0008 to see the effect generated on the forecasting model.

Table 5 Effects of Eta η on Adam algorithm

	0.1	0.01	0.001	0.0008
RMSE	153.14624	90.63583	59.06300	58.97763
MAE	142.00963	75.721548	46.40673	46.24006
MAPE	1.06121	0.565318	0.34685	0.34564
DA (%)	49.54955	49.54955	66.66667	63.06306

Table 5 shows step size η 0.001 which has been previously selected as the initial parameter produces RMSE, MAE, and MAPE values slightly higher than the step size value which is reduced to 0.0008. However, the step size η that is derived does not result in a high DA value. Step size η 0.0008 is only able to produce DA value of 63%, while Step size η 0.001 is able to achieve a DA value of 66.67%.

3.4 Effects of β_1 dan β_2

The value β_1 and β_2 on Adam algorithm are parameters that can be configured according to need, recommended values are β_1 0.9 and β_2 0.999, where the initial value of the moving average is close to 1.

Table 6 Effects of β_1 and β_2 on Adam algorithm

β_1-β_2	RMSE	MAE	MAPE	DA (%)
0.5-0.5	61.903139	47.746362	0.356505	61.261261
0.5-0.99	59.216446	46.475698	0.347339	64.864865
0.5-0.999	59.097268	46.416483	0.346915	66.666667
0.5-0.8	59.216446	46.475698	0.347339	64.864865
0.9-0.5	60.920378	47.886804	0.357948	66.666667
0.9-0.99	60.219723	47.588526	0.355654	64.864865
0.9-0.999	59.063004	46.406739	0.346852	66.666667
0.9-0.8	59.445405	46.589538	0.348148	64.864865
0.99-0.5	60.906071	47.430999	0.35421	62.162162
0.99-0.99	60.379884	47.123137	0.352015	63.063063
0.99-0.999	60.354751	47.090758	0.35177	62.162162
0.99-0.8	61.15617	47.900378	0.357773	62.162162

Table 6 is the result of the influence of the configuration and the USD / IDR data, where the value of 0.9-0.999 is the best value by producing the RMSE 59.063004, MAE 46.406739, MAPE 0.346852, and DA 66.7% values.

3.5 Model Comparison

The purpose of this study is to evaluate the performance of DBN-Adam in forecasting exchange rates.The next experiment is to compare the DBN-Adam model with the DBN-SGD and DBN-SGD + Momentum models. The parameters used are the same as the DBN-Adam parameter, except that the learning rate is determined by hyperparameter.

Table 7 Model performance comparison

Model	RMSE	MAE	MAPE	DA (%)
DBN-Adam	59.06300	46.40673	0.34685	66.66667
DBN-SGD	59.79602	47.25604	0.35318	63.86486
DBN-Momentum	59.75597	47.23662	0.35305	64.86486

Based on **Table 7**, DBN-Adam has the smallest error value with RMSE 59.06. The largest RMSE value occurs in the DBN-SGD model with a value of 59.79, but with the help of the use of the DBN-SGD momentum, it is able to reduce the value of the error even though it is not significant, namely 59.75. Evaluation of the model also does not only consider measurement values as above, but the resulting model is also expected to have a good decrease in train loss value. The decrease in train loss from the three models above can be seen in **Figure 4**.

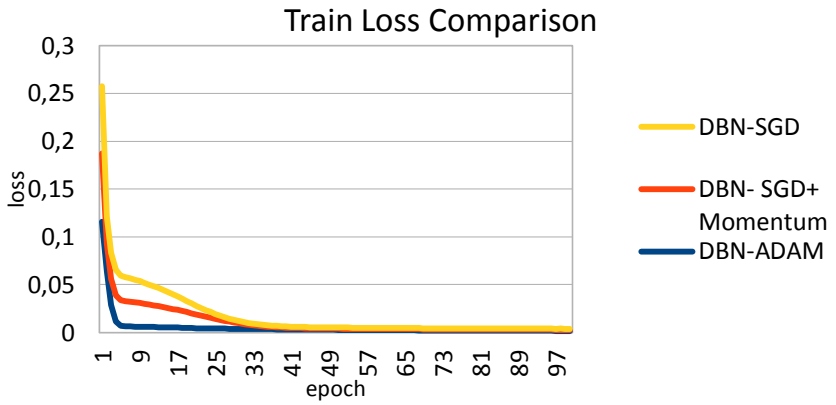

Figure 4 Train loss graph comparasion

Figure 4 shows that the Adam DBN-model has consistently decreased rapidly in 100 epochs, while the other two models are still trapped to experience a decline in some early epochs.

4. CONCLUSIONS

In this paper, the Deep Belief Network (DBN) is enhanced by applying the Adam algorithm in the fine-tuning process as an alternative to the use of stochastic gradient descent. DBN-Adam capable Empirical results clearly show that the DBN-Adam model outperforms the DBN-SGD model with four measurement units. DBN-Adam is able to reach the point of convergence quickly and can produces RMSE 59.0635004, MAE 46.406739, MAPE 0.34652, and Directional Accuracy at 66.67%.

5. FUTURE WORKS

The ability of this model can be enhanced by adding acceleration capabilities such as adding the *nesterov* method, can try to add regularization to the Adam algorithm, or can use other adaptive optimizations.

REFERENCES

[1] D. Semaan, A. Harb, and A. Kassem, "Forecasting Exchange Rates: Artificial Neural Networks Vs Regression," *Third Int. Conf. e-Technologies Networks Dev.*, vol. 2, pp. 156–161, 2014.

[2] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets.," *Neural Comput.*, vol. 18, no. 7, pp. 1527–54, 2006.

[3] T. Hirata, T. Kuremoto, M. Obayashi, S. Mabu, and K. Kobayashi, "Time Series Prediction Using DBN and ARIMA," *Proc. Int Comput. Appl. Technol. Conf*, pp. 24–29, 2015.

[4] T. Kuremoto, M. Obayashi, K. Kobayashi, T. Hirata, and S. Mabu, "Forecast chaotic time series data by DBNs," *2014 7th Int. Congr. Image Signal Process.*, pp. 1130–1135, 2014.

[5] I. Zulfa and E. Winarko, "Sentimen Analisis Tweet Berbahasa Indonesia Dengan Deep Belief Network," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 11, no. 2, p. 187, 2017.

[6] A. Sihabuddin and S. Hartati, "Exchange rates forecasting using nonlinear autoregressive," *AIP Conf. Proc.*, vol. 1755, 2016.

[7] A. Sihabuddin, D. Rosadi, and S. Utami, "An Empirical Comparative Forecast Accuracy of Exponential Smoothing and Nonlinear Autoregressive Models on Six Major Rates," *Int. J. Comput. Sci. Inf. Secur.*, vol. 15, no. 1, pp. 670–673, 2017.

[8] F. Shen, J. Chao, and J. Zhao, "Forecasting exchange rate using deep belief networks and conjugate gradient method," *Neurocomputing*, vol. 167, pp. 243–253, 2015.

[9] T. Hirata, T. Kuremoto, M. Obayashi, S. Mabu, and K. Kobayashi, "Forecasting Real Time Series Data using Deep Belief Net and Reinforcement Learning," *Int. Conf. Artif. Life Robot.*, no. Icarob, pp. 658–661, 2017.

[10] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using a deep belief network with restricted Boltzmann machines," *Neurocomputing*, vol. 137, no. Jsps 23500181, pp. 47–56, 2014.

[11] S. Ruder, "An overview of gradient descent optimization algorithms," *CoRR*, vol. abs/1609.04747, 2016.

[12] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *Int. Conf. Learn. Represent.*, pp. 1–13, Dec. 2015.

[13] K. Maris, K. Nikolopoulos, K. Giannelos, and V. Assimakopoulos, "Options trading driven by volatility directional accuracy," *Appl. Econ.*, 2007.

[14] A. M. Oyewale, "Evaluation of Artificial Neural Networks in Foreign Exchange Forecasting," *Am. J. Theor. Appl. Stat.*, vol. 2, no. 4, pp. 94–101, 2013.