Human Activity Prediction using Long Short Term Memory

Jaka Septiadi^{1}, Budi Warsito², Adi Wibowo³*

¹Master Program of Information System, School of Postgraduate Studies, Diponegoro University, Semarang, Indonesia

²Department of Statistics, Faculty of Science and Mathematics, Diponegoro University, Semarang, Indonesia

³Department of Informatics, Faculty of Science and Mathematics, Diponegoro University, Semarang, Indonesia

Abstract. Early symptoms of dementia is one of the causes decrease in quality of life. Human activity recognition (HAR) system is proposed to recognize the daily routines which has an important role in detecting early symptoms of dementia. Long Short Term Memory (LSTM) is very useful for sequence analysis that can find the pattern of activities that carried out in daily routines. However, the LSTM model is slow to achieving convergence and take a long time during training. In this paper, we investigated the sequence of actions recorded in smart home sensors data using LSTM model, then the model will be optimized using several optimization methods. The optimization methods were Stochastic Gradient Descent (SGD), Adagrad, Adadelta, RMSProp, and Adam. The results showed that using Adam to optimized LSTM is better than other optimization methods.

1 Introduction

Human activity recognition (HAR) is one of the efforts that can be completed in developing an intelligent environment that can be utilized for monitoring daily activities [1]. HAR has an important role in human interaction or interpersonal, because it can provide a better understanding of one's personality and psychological condition [2]. Generally, HAR is used to identify abnormal behaviour [3]. According to historical data, abnormal behaviour is defined as a decrease in physical activity in carrying out daily routines [4]. Abnormal behaviour is refers to abnormal activity that caused by cognitive decline and it cause early symptoms of dementia at 30-50 years old [5].

Early detection of dementia is refers to finding the pattern of activities that are incompatible with normal or abnormal behaviour, such as finding a pattern of dementia patients. However, availability of the real world data of dementia patients is a big challenge for non-medical diagnosis of the patient [6]. Early symptoms of dementia can be identified by monitoring behavioral variations such as disturbance in sleep, difficulty in walking and inability to complete an activity. Such variations can provide important information about

* Corresponding author: jakaseptiadi37@gmail.com

memory, mobility, and cognition of a person, so they can find a pattern of activities that carried out in their daily lives [7]. The research area of HAR shows that modeling activities sequentially over time or time series modeling can be used as indicators to early detection of dementia [8].

Long Short Term Memory networks, usually just called LSTM is a special kind of RNN, capable of learning long-term dependencies. They work very well on a large variety of problems, and are now widely use in sequence analysis [9]. The author of [10] analyzed the sequence of actions recorded in smart home environment with sensors by introducing the concept of LSTM model that allows us to predict the user's next actions and to identify anomalous user behaviour. The research [7, 8] shows that their experiment uses LSTM to modeling activities have a clear advantage in accuracy. However, the LSTM training method is slow to achieving convergence and take a long training time [11]. One way to overcome this issue is to apply optimization methods. Gradient based algorithms use the gradient of the performance function to determine how to adjust the weights to minimize the performance [12], and it is aim to improve the performance, the accuracy and training time [13].

In this paper we investigated activity recognition using LSTM networks model on publicly available smart homes dataset [14]. Using several optimization methods, such as SGD [15], Adagrad [16], Adadelta [17], RMSProp [18], and Adam [19], as optimization methods to optimized the LSTM model for time series analysis is good idea to get a good results. Such variations of the optimization has been done to optimized Neural Network with MicroRNA as a feature, and showed very good results [20]. Therefore, in this study such variations of the optimization method were used to optimized the LSTM model and the results were compared to find out the best optimization for activity prediction.

2 Method Review

2.1 Word Embedding

A lot of word representation techniques are used in machine learning and deep learning area and generally use one-hot encoding. However, the disadvantage of using this type of representation is not possible to compute how similar words are. The solutions is to use word embedding [21]. Word embedding have the ability to generalize words that have semantic similarities, capture semantic meaning, and able to achieve similar words have the same vectors, which is not in one-hot encoding. Hence, word embedding can better model semantic complexity. In our model, we use Word2Vec representation to define each action's embedding matrix.

2.2 Long Short Term Memory

Long Short Term Memory units, or LSTMs, proposed by [22] is a variant of Recurrent Neural Network architecture. LSTM has the advantage to solving sequential data problems, especially in time series data. The LSTM networks has cell memory blocks. Each cell contains gates: an input gate, a neuron with self-recurrent connection, a forget gate and an output gate. The cell operates on the input sequence, each gate uses sigmoid activation, where the output of sigmoid activation is between 0 and 1, indicating whether the information will be forwarded or stopped. In brief, a sigmoid activation is to control whether they are triggered, making the change state and the addition of information flowing through the cell conditional. The procedures of LSTM network at time step t are as follows [22]:

- (a) The first step LSTM is to decide what information we're going to throw away from the cell state, previous cell state C_{t-1} in the forget gate f_t .

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (1)$$

- (b) Decide what new information of input x_t should be stored in the cell state C_t , where i_t and $\tilde{C}_t$ should be updated.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (2)$$

$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (3)$$

- (c) Update the old cell state, C_{t-1} , into the new cell state C_t which combine the candidate memory $\tilde{C}_t$ and the old cell state C_{t-1} .

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (4)$$

- (d) Decide what we're going to output h_t in the output gate where the output information o_t and the cell state C_t are used.

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + \tilde{C}_t) \quad (5)$$

$$h_t = o_t * \tanh(C_t) \quad (6)$$

where W is the input weight, U is the recurrent weight, and b is the bias. The terms f , i , o represent the forget gate, input gate, and output gate. The terms x_t is the input information to the memory cell layer at time step t , h_{t-1} is the result calculated by previous hidden state and h_t is the output of hidden state at time step t . σ being the sigmoid which enables the valueable between 0 and 1, then $\tanh$ is the hyperbolic tangent activation function which squeezes the value between -1 and 1. Both of the two activation functions are expressed as:

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (7)$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (8)$$

The steps above are feedforward training process. In order to perform the backpropagate steps, it is necessary to calculate the derivative of each gate at timestep as follows [22]:

$$\delta h_t = \Delta_t + U_f^T \delta f_{t+1} + U_i^T \delta i_{t+1} + U_{\tilde{C}_t}^T \delta \tilde{C}_{t+1} + U_o^T \delta o_{t+1} \quad (9)$$

$$\delta o_t = \delta h_t \cdot \tanh(C_t) \cdot o_t (1 - o_t) \quad (10)$$

$$\delta C_t = \delta h_t \cdot o_t [1 - \tanh^2(C_t)] + C_{t+1} + f_{t+1} \quad (11)$$

$$\delta \tilde{C}_t = \delta C_t \cdot i_t [1 - \tanh^2(\tilde{C}_t)] \quad (12)$$

$$\delta i_t = \delta C_t \cdot \tilde{C}_t \cdot i_t (1 - i_t) \quad (13)$$

$$\delta f_t = \delta C_t \cdot C_{t-1} \cdot f_t (1 - f_t) \quad (14)$$

$$\delta x_t = W_f^T \cdot \delta f_t + W_i^T \cdot \delta i_t + W_{\tilde{c}}^T \cdot \delta \tilde{C}_t + W_o^T \cdot \delta o_t \quad (15)$$

where Δ_t is the derivative of error it formally corresponds to $\frac{\partial E}{\partial y}$, but not including the recurrent dependencies [22]. The term $t + 1$ indicates that need to compute the gate operation from next time step. In the last steps, the gradient for the weights are calculated as in equation (17), (18), and (19), where δg_t is the gradient from all gates at time step, δW is the gradient of weight, δU is the gradient of recurrent weight, and δb is the gradient of bias:

$$\delta W = \sum_{t=0}^T (\delta g_t \cdot x_t) \quad (16)$$

$$\delta U = \sum_{t=0}^{T-1} (\delta g_{t+1} \cdot h_t) \quad (17)$$

$$\delta b = \sum_{t=0}^T (\delta g) \quad (18)$$

2.3 Optimizer

Optimizer is an algorithm used to change the attributes or parameters of neural network in order to reduce the error. In neural network, the goal is to optimize the network with reduce the difference between the predicted output and the actual output. In this work, the backpropagation through LSTM are using Adam and SGD, then the results obtained will be compared.

2.3.1 Gradient Descent

Gradient descent is a first-order optimization algorithm and the most used optimization algorithm in neural network. The algorithm works by updating the parameters which are to be optimized to gradient of the value function. Stochastic gradient descent or SGD, is a variant of gradient descent optimization method that each iteration minimizes the error by considering the error for each data point [24]. The iterations of SGD can be described as [15]:

$$w_k = w_{k-1} - \alpha_{k-1} \hat{\nabla} f(w_{k-1}) \quad (19)$$

where w_k is iterate, α_k is a learning rate, and $\hat{\nabla} f(w_k)$ denotes the stochastic gradient computed at w_k .

2.3.2 Adagrad Optimizer

Adagrad is an algorithm for gradient based optimization and modified SGD algorithm with per-parameter learning rate, it means adapting the learning rate to the parameters. Adagrad can improved the SGD algorithm it is because increases the learning rate for sparses parameters and decreases the learning rate for ones that are less sparse [16]. The update rule for Adagrad is as follows:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \cdot g_{t,i} \quad (20)$$

2.3.3 Adadelta Optimizer

The main idea of Adadelta is from Adagrad algorithm that seeks reduce its aggressive which the learning rate continuously decreases in each iteration [17]. Adadelta is an extension of Adagrad that adapts learning rates based on a moving window of the gradient instead of accumulating the sum of squared gradients over time [17]. Compared to Adagrad, Adadelta does not have to set an initial learning rate. The steps of Adadelta is as follows:

1. Calculate the gradient:

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1 - \rho)g_t^2 \quad (21)$$

2. The resulting parameter update:

$$\Delta\theta_t = -\frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} \cdot g_t \quad (22)$$

3. Accumulate update values:

$$E[\Delta\theta^2]_t = \rho E[\Delta\theta^2]_{t-1} + (1 - \rho)\Delta\theta_t^2 \quad (23)$$

4. Apply the updates:

$$\theta_{t+1} = \theta_t + \Delta\theta_t^2 \quad (24)$$

2.3.4 RMSProp Optimizer

RMSProp is an adaptive learning rate method that has found much success in practice and the method was proposed by [18]. RMSProp was developed to overcome Adagrad's issues, which the learning rate countinously decrease in each iteration. The steps of RMSProp is as follows:

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1 - \rho)g_t^2 \quad (25)$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t \quad (26)$$

where the term γ is to be set to 0.9 and for the learning rate η value is 0.001.

2.3.5 Adam Optimizer

Adam is an adaptive learning rate method which computes individual learning rates for different parameters. Adam is derived from adaptive moment estimation, it is because the algorithm uses both first and second moments of the gradient to update the learning rate. Adam method combines the advantages of two methods, AdaGrad and RMSProp. The method first updates the exponential moving averages of the gradient m_t and also the squared gradient v_t , which is the estimates of the first and second moment. The Adam optimization steps as follows [19]:

1. Initialize the decaying rate $\beta_1 = 0.9$, $\beta_2 = 0.99$, and $\epsilon = 10^{-8}$, initialize time $t = 1$.

2. Calculate the gradient g_t at time t :

$$g_t = \frac{\partial L}{\partial W} \quad (27)$$

3. Calculate the first moment:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (28)$$

4. Calculate the second moment:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (29)$$

5. Calculate the bias correction for the first moment:

$$\hat{m}_1 = \frac{m_t}{1 - \beta_1^t} \quad (30)$$

6. Calculate the bias correction for the second moment:

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (31)$$

7. Update the parameters:

$$W_t = W_{t-1} - \alpha \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}} \right) \quad (32)$$

2.4 Evaluation of Model Performance

To evaluate the performance of our model, the evaluation used is top-k categorical accuracy. The formula for accuracy is expressed as [10]:

$$acc_at_k = \frac{1}{N} \sum_{i=1}^N 1 [a_i \in C_i^k]$$

where N is the number of queries, a_i is the expected action, C_i^k is the top-k predicted actions and k is usually set default from 1 to 5.

3 Implementation

In order to create a time series model for activity prediction, we have used a LSTM network for the proposed prediction architecture, as shown in Figure 1. The process consists of data

acquisition, word embedding representation, LSTM model for activity prediction, then using several optimization methods during training to optimize the LSTM network, and we have a model prediction that will be evaluated.

Fig. 1. The proposed architecture.

In our proposed architecture, the acquisition of data was carried out using data that was already present in previous studies, as explained in the next section. The data is emanated from monitoring activities using sensors and further labelled as the sequences of activities. These labels are then fed to the embedding layer and represented as Word2Vec vectors. These vectors are then processed by LSTM as sequence modelling module, then the model will be optimized with several optimization methods to help get a good result. Finally, the results obtained will be evaluated.

4 Experiments

4.1 Data Description

We used the popular dataset collected by [14]. The dataset is the result of monitoring an adult man in a three room where the binary sensors (DM 1810 RFM) were installed. These sensors were installed in different locations and each sensors were mapped into an action or action mapping. In this case, action mapping has been done by [10], resulting in the following set of actions: “UseMicrowave”, “UseDishwasher”, “UseHallToiletDoor”, “UseHallBathroomDoor”, “ToiletFlush”, “UseHallBedroomDoor”, “UseWashingmachine”, “OpenGroceriesCupboard”, “OpenFreezer”, “OpenCupsCupboard”, “OpenFridge”, “OpenPlatesCupboard”, “UseFrontDoor”, and “OpenPansCupboard”.

The data captures daily activities in less than a month and the annotated activities were the following: “LeaveHouse”, “UseToilet”, “TakeShower”, “GoToBed”, “Prepare Breakfast”, “Prepare Dinner”, and “Get Drink”.

4.2 Results and Discussion

In order to implement the prediction model, the LSTM algorithm was implemented using Keras Deep Learning library’s. The experiment consists of 5 training experiments with several optimization methods, as show in Table 1. The LSTM model was built with 1 LSTM layer of 512 neurons, 2 dense layers with different activation function for each experiments with dropout 0.8, and in the last layer is 1 dense layer with softmax activation. For each optimization experiment, the number of epochs trained are 100, 300, and 500, and for the learning rate are 0.001, 0.01, 0.05, and 0.1. The dataset divided into training dataset, and validation dataset. The data used as training is 80% of whole dataset, and 20% for validation dataset. The training was performed using Nvidia Geforce GTX 1050Ti.

Table 1. Optimization experiment.

ID	Optimizer	Activation Function
Exp-1	Adagrad	Sigmoid
Exp-2	Adadelata	Tanh

Exp-3	SGD	ReLU
Exp-4	RMSProp	ReLU
Exp-5	Adam	ReLU

Table 2. Experiment 1 results.

Exp-1		Accuracy at k (%)				
Epoch	Learning Rate	top-1	top-2	top-3	top-4	top-5
100	0.001	35.47	53.41	67.09	76.06	81.19
	0.01	37.17	58.11	66.66	76.06	80.76
	0.05	38.04	57.69	67.94	74.35	80.34
	0.1	35.47	58.97	67.09	76.92	80.34
300	0.001	36.32	54.27	66.66	76.92	80.34
	0.01	38.46	58.54	66.66	76.06	82.05
	0.05	41.02	58.97	68.37	75.64	79.05
	0.1	35.89	47.43	62.82	74.35	80.34
500	0.001	37.17	52.56	63.67	75.64	80.34
	0.01	41.45	62.39	70.51	76.49	81.62
	0.05	35.47	57.26	67.94	75.21	81.19
	0.1	36.75	54.27	66.66	76.49	80.76

Table 3. Experiment 2 results.

Exp-2		Accuracy at k (%)				
Epoch	Learning Rate	top-1	top-2	top-3	top-4	top-5
100	0.001	46.15	61.53	70.94	79.05	82.90
	0.01	45.29	61.53	73.50	77.77	83.33
	0.05	45.29	61.11	70.51	78.63	83.33
	0.1	44.87	62.39	71.36	77.35	82.05
300	0.001	45.72	58.54	69.65	77.35	83.33
	0.01	46.15	59.82	69.23	77.77	84.18
	0.05	44.31	64.52	73.07	76.06	84.61
	0.1	45.72	62.39	70.51	78.20	82.47

500	0.001	45.29	60.68	71.79	78.63	82.05
	0.01	45.29	58.54	70.51	79.05	81.62
	0.05	44.44	57.26	70.51	78.63	81.69
	0.1	45.19	57.69	70.52	77.77	84.18

Table 4. Experiment 3 results.

Exp-3		Accuracy at k (%)				
Epoch	Learning Rate	top-1	top-2	top-3	top-4	top-5
100	0.001	20.51	32.05	39.74	45.72	54.27
	0.01	20.71	29.91	35.04	55.55	56.83
	0.05	20.51	29.91	45.29	50	55.12
	0.1	20.51	28.63	33.76	45.29	61.96
300	0.001	28.20	39.74	53.84	64.10	67.94
	0.01	28.20	39.74	53.41	64.10	68.10
	0.05	26.92	38.88	51.70	63.24	67.94
	0.1	27.77	39.31	54.27	64.10	68.37
500	0.001	33.33	45.30	58.97	67.52	75.19
	0.01	33.33	44.44	56.41	67.09	75.21
	0.05	32.90	42.73	56.41	66.66	73.07
	0.1	33.33	45.29	58.97	67.09	74.78

Table 5. Experiment 4 results.

Exp-4		Accuracy at k (%)				
Epoch	Learning Rate	top-1	top-2	top-3	top-4	top-5
100	0.001	45.29	62.39	70.94	78.20	84.61
	0.01	44.44	59.40	70.51	76.92	80.34
	0.05	44.87	59.40	70.51	77.35	83.76
	0.1	44.01	59.82	70.94	76.92	82.47
300	0.001	44.87	61.53	70.51	79.05	82.90
	0.01	44.01	60.68	72.22	77.35	81.62
	0.05	45.51	61.11	67.94	77.77	84.61
	0.1	45.72	63.24	71.36	79.48	84.76
500	0.001	44.87	62.39	69.65	77.35	81.19

	0.01	45.29	58.97	73.21	77.35	82.05
	0.05	43.58	61.11	70.51	77.77	81.62
	0.1	45.29	59.82	71.36	77.35	81.19

Table 6. Experiment 5 results.

Exp-5		Accuracy at <i>k</i> (%)				
Epoch	Learning Rate	top-1	top-2	top-3	top-4	top-5
100	0.001	46.15	63.24	73.50	79.48	84.18
	0.01	46.58	62.39	70.08	78.63	83.76
	0.05	46.15	61.53	70.51	79.91	84.18
	0.1	46.58	61.53	72.22	80.34	82.90
300	0.001	47.43	62.82	73.93	79.48	85.47
	0.01	47.00	63.24	73.07	79.05	83.76
	0.05	46.15	63.67	72.64	79.91	84.61
	0.1	47.43	61.53	71.79	79.91	85.04
500	0.001	46.15	63.67	70.08	78.63	82.47
	0.01	46.58	60.25	73.50	79.48	85.04
	0.05	46.15	69.65	69.65	79.48	85.04
	0.1	47.00	60.25	71.36	80.34	86.32

Tables 2-6 show the results for each optimization experiment. All experiments use the same LSTM architecture, same learning rate, and same the number of epochs, as explained previously. In the case of the first experiment (Exp-1), table 2, shows that using Adagrad optimizer with sigmoid activation function achieved the best accuracy at top-5, with learning rate value is 0.01 at 300 epochs.

In the case of the second experiment (Exp-2), table 3, shows that using Adadelata optimizer with Tanh activation function achieved the best accuracy at top-5, with learning rate value is 0.05 at 300 epochs.

In the case of the third experiment (Exp-3), table 4, shows that using SGD optimizer with ReLU activation function achieved the best accuracy at top-5, with the learning rate value is 0.01 at 500 epochs.

For the fourth experiment (Exp-4), table 5, shows that using RMSProp optimizer with ReLU activation function achieved the best accuracy in almost top-*k* with learning rate value is 0.1 at 300 epochs.

Finally, in the last experiment (Exp-5), table 6, the best results were achieved using Adam optimizer with ReLU activation function with the values of 80.34% at top-4 and 86.32% at top-5, with learning rate value is 0.1 at 500 epochs. Based on the results, Adam has the highest accuracy. However, the accuracy results can be influenced by several factors, such as data used, good preprocessing results, good architecture used, good tuning parameter, etc.

5 Conclusion

In this paper, we showed a study of human activity prediction using LSTM as a prediction model. The proposed methodology consists of five steps, the acquisition of data, word embedding representation, LSTM layer, optimization, and evaluation of the model. Based on the results showed that using Adam with ReLU activation function to optimized the LSTM model works well and produced the best accuracy compared than other methods. Adam achieved the best accuracy value of 86.32% at top-5, with learning rate value is 0.1 at 500 epochs. By this result, there is still a potential improvement for the accuracy. The combination of Adam and SGD by switching from Adam to SGD during middle of training to optimized LSTM is an excellent idea for future work, because Adam showing huge performance gains in terms of speed of training, and switching from Adam to SGD in the middle of training, they've been able to obtain better generalization power than when using Adam alone [15].

References

1. E.D. Franco, P.A. Colpas, J.M. Quero, M. Espinilla, Sensor Based Datasets for Human Activity Recognition – A Systematic Review of Literature, *International Journal of Computer Science*, 2169-3536, (2018)
2. M. Vrigas, C. Nikou, A. Kakadiaris, A Review of Human Activity Recognition Methods, *Frontiers in Robotic and Artificial Intelligent*, (2015)
3. Z.A. Khan, W. Sohn, A Hierarchical Abnormal Human Activity Recognition System Based on R-transform and Kernel Discriminant Analysis for Elderly Health Care, *Computing* **95**, 109-127, (2013)
4. J.L. Candas, V. Pelaez, G. Lopez, M.A. Fernandez, E. Alvarez, G. Diaz, An Automatic Data Mining Method to Detect Abnormal Human Behaviour using Physical Activity Measurements, *Pervasive and Mobile Computing* **15**, 228-241, (2015)
5. L. Xiantao, Y. Sun, L. Gong, L. Zheng, K. Chen, Y. Zhou, Y. Gu, Y. Xu, Q. Guo, Z. Hong, D. Dhing, J. Fu, Q. Zhao, A Novel Homozygous Mutation in TREM2 found in a Chinese early-onset Dementia Family with Mild Bone Involvement, *Neurobiology of Aging*, (2019)
6. C. Dhiman, D.K. Vishwakarma, A Review of state-of-the-art Techniques for Abnormal Human Activity Recognition, *Engineering Applications of Artificial Intelligence* **77**, 21-45, (2019)
7. D. Arifoglu, A. Bouchechia, Activity Recognition and Abnormal Behaviour Detection with Recurrent Neural Network, *International Conference on Mobile Systems and Pervasive Computing*, **110**, 86-93, (2017)
8. D. Singh, E. Merdivan, S. Hanke, J. Kropf, M. Geist, A. Holzinger, Convolutional and Recurrent Neural Networks for Activity Recognition in Smart Environment, *Integrative Machine Learning*, (2017)
9. Z. Chang, Y. Zhang, W. Chen, Electric Price Prediction based on Hybrid Model of Adam Optimized LSTM Neural Network and Wavelet Transform, *International Journal of Energy Research*, (2019)
10. A. Almeida, G. Azkune, Predicting Human Behaviour with Recurrent Neural Networks, *Applied Sciences* **8**, 2, (2018)
11. W. Yu, X. Li, J. Gonzalez, Fast Training of Deep Learning LSTM Networks, *International Symposium on Neural Networks*, 3-10, (2019)

12. B. Warsito, H. Yasin, A. Prahutama, Particle Swarm Optimization versus Gradient Based Methods in Optimizing Neural Network, *Journal of Physics: Conference Series* 1217, (2019)
13. A. Wibowo, D. Setianto, P.W. Wisnu, W. Jatmiko, Optimization of Backpropagation Using Nguyen-Widrow and Stimulus-Sampling for Breast Cancer Classification and Feature Selection, *Journal of Engineering Science and Technology* **6**, 3437-3456, (2019)
14. V. Kasteren, A. Noulas, G. Englebiennne, B. Krose, Accurate Activity Recognition in a Home Setting, *Proceeding of the 10th International Conference on Ubiquitous Computing* (2008)
15. N.S. Keskar, R. Socher, Improving Generalization Performance by Switching from Adam to SGD, (2017)
16. J. Duchi, E. Hazan, Y. Singer, Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, *Journal of Machine Learning Research* **12**, 2121-2159, (2011)
17. M.D. Zeiler, ADADELTA: An Adaptive Learning Rate Method, *arXiv preprint arXiv:1212.5701*, (2012)
18. T. Tieleman, E. Hinton, COURSERA: Neural Networks for Machine Learning, (2012)
19. D. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, *Proceeding of International Conference on Learning Representations*, (2015)
20. A. Wibowo, P.W. Wiryawan, N.I. Nuqoyati, Optimization of Neural Network for Cancer MicroRNA Biomarkers Classification, *Journal of Physics: Conference Series*, 1217, (2019)
21. T. Mikolov, I. Sutskever, K. Chen, G. Corrado, Dean, D. Jeffrey, Distributed Representations of Words and Phrases and their Compositionality, In *Advanced in Neural Information Processing Systems*, 3111-3119, (2013)
22. S. Hochreiter, J. Schmidhuber, Long Short Term Memory, *Neural Computation*, **9**, 1735-1780, (1997)
23. K. Greff, K. Rupesh, Srivastava, J. Koutnik, R. Bas, Steunebrink, J. Schmidhuber, LSTM: A Search Space Odyssey, *Transactions on Neural Networks and Learning Systems*, (2017)
24. R.D. Shirwaikar, D.U. Acharya, K. Makkithaya, Optimizing Neural Networks for Medical Data Sets: A Case Study on Neonatal Apnea Prediction, *Artificial Intelligent In Medicine* **98**, 59-76, (2019)