International Journal of Multidisciplinary and Current Research ISSN: 2321-3124
Research Article Available at: http://ijmcr.com

Modified Long Short-Term memory and Utilizing in Building sequential model

Hozan K. Hamarashid*

Computer Science Institute, Sulaimani Polytechnic University, Sulaimani, Iraq

Received 10 March 2021, Accepted 01 May 2021, Available online 02 May 2021, **Vol.9 (May/June 2021 issue)**

Abstract

A robust model of neural network was built to control sequence dependency difficulties which is known as recurrent neural networks RNN. The Long Short Term Memory LSTM network is a model of recurrent neural network RNN which is utilized in deep learning due to huge architectures or constructions could be well trained. LSTM is obviously established to prevent the long term dependency difficulty. Keeping information for long periods of time is basically its evade behaviour. Modified Long Short Term Memory MLSTM is an adaptation of LSTM model. This modification is conducted on one of the parameters of LSTM. Threshold value is essential to control the input value. This paper defines deep learning classification with modified long short term memory. Shapes and designs of Arrays will be specifically concentrated on due to it is one of the most frequent faced and misunderstood difficulty. As a consequence, this paper covers; one hot encoding. Besides, selecting input and output dimensions in the layers is illustrated. In addition, training the modified long short term memory is addressed. Before the last, assessment of the model in the perspective of training and testing is conducted. Lastly, utilizing MLSTM in making sequential model.

Keywords: *Modified LSTM, MLSTM, Long Short Term Memory, LSTM, RNN model.*

Introduction

Until now, different techniques were improved specifically in artificial intelligence AI. Deep Learning is a feature of artificial intelligence. Besides, it is an essential part of artificial intelligence. Long Short Term Memory LSTM is a method of deep learning. LSTM is a special kind of recurrent neural network RNN which has the ability to learn from long dependency [1]. LSTM was invented by Hochreiter & Schmidhuber in 1997. Researchers made it common and popular by improving it and utilizing it in different fields and to solve various and large difficulties. In addition, it is utilized broader now. Long term dependency in recurrent neural network is a problem which is solved by utilizing robust type of RNN which is LSTM [2]. In the standard RNN, neural network modules were repeated as a form of chain in the entire recurrent neural network. The repeating of the modules has a plain structure for example single layer. Figure 1 shows single layer in standard RNN repeating module:

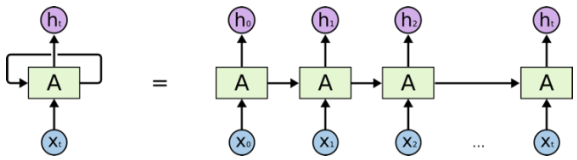

Figure 1: Single Layer repeating module in standard RNN [3]

*Corresponding authors' ORCID ID: 0000-0002-5074-7853
DOI: https://doi.org/10.14741/ijmcr.v.9.3.2

LSTM has this chain structure like RNN nevertheless its repeating module structure is different compared to RNN. The difference between RNN and LSTM repeating module structure is that RNN has a single neural network layer. In contrast, LSTM has four integrated neural network layers [4]. Figure 2 represents the LSTM four layers repeating module structure:

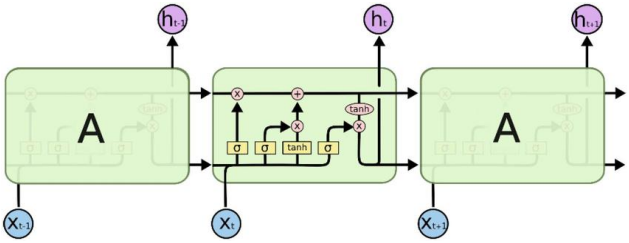

Figure 2: LSTM four layers repeating module structure [5]

The important part of LSTM is cell state. Cell state is like a belt goes through the LSTM chain and pass information to the output. So, LSTM is capable to prevent or pass information to the cell state by regulating structures via gates. Information is going through gates optionally. In this step sigmoid function which is multiplication is conducted in neural network layer. The outcome of sigmoid will be between (0, 1). This means that to what extent the information should be let goes through. As a result (0) means that let nothing to pass through. On the

other hand, (1) mean let the entire information to pass through [6]. To handle and protect the cell state, LSTM has three gates. Below, the phases of LSTM is illustrated:

1.1 First Phase of LSTM

In the first phase, the decision is made to which information should be removed from cell state. The decision is made based on sigmoid layer output which is known as forget gate. It depends on three components which are; h_{t-1},x_t and c_{t-1}. The outcome is a value between (0, 1) for every value in the cell state. As a result, (0) means to entirely omit this information. In contrast, (1) means to keep the entire of this information [8]. Figure 3 shows the first phase of LSTM:

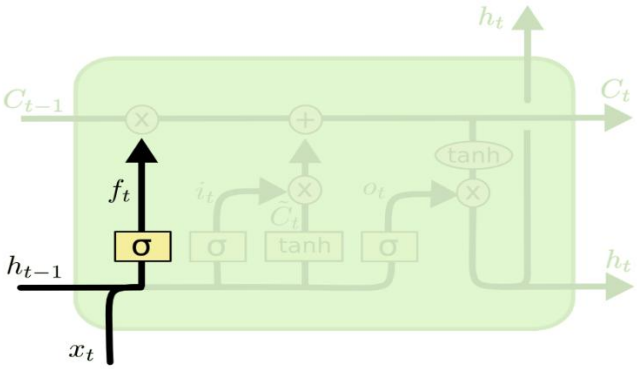

Figure 3: represents First Phase of LSTM [7]

1.2 Second Phase of LSTM

In the second phase, the decision is made on which information should be kept in the cell state. This is conducted based on two steps; firstly, input gate which is sigmoid layer make decision on which information or value should be updated. Secondly, a vector of new values is made by tanh layer that is capable to be added to the state which is represented by $\check{c}_t$ [8]. Figure 4 shows the second phase of LSTM:

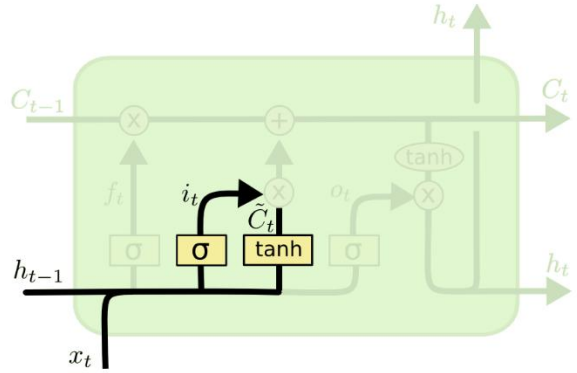

Figure 4: Represents the Second Phase of LSTM [7]

1.3 Third Phase of LSTM

In the second phase, the decision is made on what should be conducted. In the third phase, the old cell state is updated or renewed to a new cell state which are expressed as C_{t-1}, and C_t, respectively. In this phase, the multiplication between old state and the information that decided to be forgotten previously is conducted which is expressed by f_t. Next, the new value or candidate is added and the decision is made to update every state value as represented by $i_t * \check{c}_t$ [9]. Figure 5 shows the third phase of LSTM:

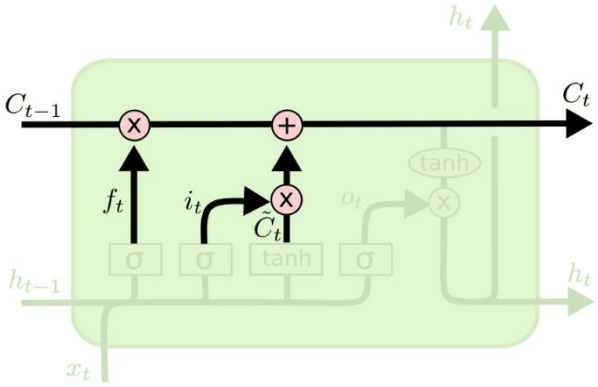

Figure 5: Shows the Third Phase of LSTM [7]

1.4 Fourth Phase of LSTM

In the last phase, the decision is made of what should be the outcome. The outcome is selected relied on the cell state. This goes through two steps; first, the decision is made on which feature of cell state is going to be the outcome by applying the sigmoid layer. Second, input cell state to the tanh layer namely to make the values between (-1, 1). Then, this outcome is multiplied by the value of sigmoid gate outcome. As a result, these features will be the output that decided on [9]. Figure 6 shows the fourth phase of LSTM:

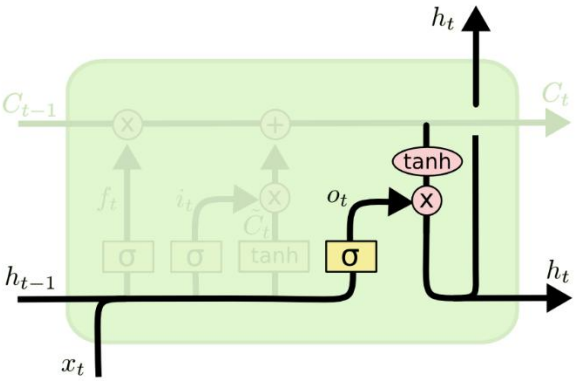

Figure 6: Represents the Fourth Phase of LSTM [7]

1.5 LSTM Equations

In this subsection the entire equations of LSTM phases are addressed:

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f \quad (1)$$

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i \quad (2)$$

$$\check{c}_t = tanh(w_c[h_{t-1}, x_t] + b_c \quad (3)$$

$$c_t = f_t * c_{t-1} + i_t * \check{c}_t \quad (4)$$

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

$$h_t = o_t * \tanh(c^t) \quad (6)$$

This research paper addresses various features in the following sections. The rest of the paper is; methodology in section 2. Section 3 represents the assessment of the model. Last, conclusion is addressed.

2. Methodology

In this section the methodology of the research paper is addressed. Thus, to conduct the methodology steps, various tasks have to be achieved. Dataset is crucial part as a first task. So, it is vitally important to have dataset to work on and to obtain results due to without data it is not possible to test the model. A parameter added to modify the model. One hot encoding to determine dimension of the vectors. Then, the sequential model should be made. After that the model should be validated. Later, the model should be assessed with training and testing data. Figure 7 represents the research paper methodology:

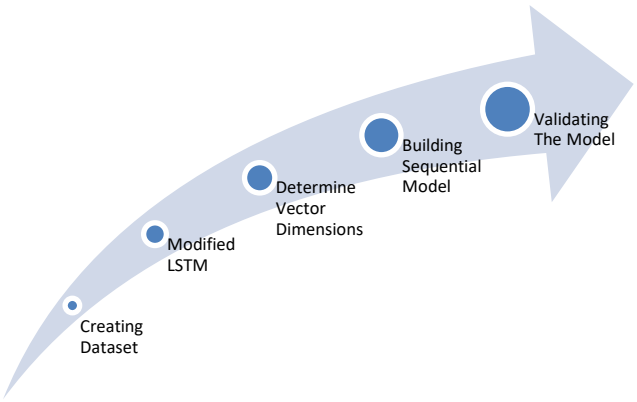

Figure 7: Represents the Methodology

2.1 Creating Dataset

A random dataset is created the data contains 3 columns and 10000 rows which is binary data. So, the dataset can be generated by utilizing the following code snippet:

Using R Programming
 # creating dataset which contains 3 columns and 10000 rows of uniform distribution.
dataset=matrix(data=runif(30000), nrow=10000, ncol=3)

2.2 Modified LSTM

A parameter is added to the model and a value is assigned to it. This parameter handles the output value of the model. This is conducted by calculating the sum of the columns value then if the result of the summation is above the parameter value it returns 1. On the other hand, if the summation value is smaller than the parameter value then it returns 0. The following code snippet shows the conduction of this task:

 # creating dataset which contains 3 columns and 10000 rows of uniform distribution.
 # Outcome
AddParameter <- 1.0
OverValue <- 1
UnderValue <- 0
outcome_data=ifelse(rowSums(dataset) > AddParameter, OverValue, UnderValue)

Following is the outcome of the generated dataset by using this code snippet:

head (dataset)

The following is the outcome of the above code snippet:

```
#        (,1)   (,2)   (,3)
# (1,)   0.49   0.60    0.78
# (2,)   0.56   0.87   0.34
# (3,)   0.08   0.75   0.69
# (4,)   0.79   0.63   0.73
# (5,)   0.65   0.90   0.13
# (6,)   0.23   0.26   0.27
```

head (outcome_data)

And the result is shown below:

```
# (1)
1
1
1
1
1
0
```

From the above results it can be clearly seen that the outcome of the summation of the columns for each row that only the last row summation is under the value of added parameter, thus it returns 0. On the other hand, the other cases returns 1 due to the calculation of the summation of other rows are above the value of added parameter.

2.3 Determine Vector Dimensions

One hot encoding, determining the vector dimensions or matrices or arrays dimensions is an essential part of deep learning. This is needed for the model. Therefore, one hot encoding is made to conducted binary classification. This task can be done even for more than two classes. The following examples shows building class binary classification for more than two classes:

For the vector (0, 1) the value is 1
For the vector (1, 0) the value is 0

To conduct this task the function “to_categorical” is utilized as shown in the following code snippet:

```
one_hot_data=to_categorical(outcome_data,
num_classes = 2)

head(one_hot_data)
```

The result is shown below

```
#      (, 1) (, 2)
# (1,)      0     1
# (2,)      0     1
# (3,)      0     1
# (4,)      0     1
# (5,)      0     1
# (6,)      1     0
```

As a result, "to_categorical" function merely accepts numeric data as an input value. Thus, non-numerical data must be converted to numerical data so that it can be processed by utilizing to_categorical function. For instance, true and false data or A, B, C data and etc. must be converted to numerical data and it can be converted.

2.4 Building Sequential Model

Sequential model can be created by utilizing stacking layers in Keras. Dense layers is determined in stacking layers, for this purpose, three dense layers are used. Consequently, implicit input layer with provided data and output layer included. There are several parameters that are utilized:

- Model= Keras_sequential() which creates sequential model.
- Input_shape (ncol(dataset)) is the first layer with 3 columns value that produces a vector that the value of input variables are demonstrated. The most of the things in deep learning is vector.
- The input of the next layer is obtained from earlier layer or input_shape.
- Dense_layer which expresses the last outcome. It has 2 columns that demonstrates the two conceivable results ncol(one_hot_data).
- Loss function which is utilized to determine the loss of data by using categorical accuracy function. Besides, optimizer functions is utilized to provide the type of optimizer to be used in the model. Additionally, metrics especially accuracy is utilized for the outcome of classification to see to what extent of level the model is accurate.

2.5 Validating the Model

Validating is about testing the created model to see the outcomes whether the model is validate or not. To conduct this task, test data has to be created. For instance, 3000 random data with 1000 rows and 3 columns are created to test the proposed model.

```
test_dataset=matrix(data=runif(3000),
nrow=1000, ncol=3)

Outcome_data_prediction=predict_classes(model,
test_dataset)
```

Predict classes is utilized to conduct one hot encoding automatically. On the other hand, predict is used to return dimensionality of the taken data which return the same dimensionality of the training data i.e. rows and classes of prediction. In addition. In classification every class probability endorsed to be returned as it is conducted by using summation of each row. To obtain the probability, softmax function is used for the final layer. Consequently, the outcome will be between (0 and 1) for each of the class. Besides, the summation of every class is equal to 1 naturally.

3. Assessing the Model

Every model that created should be evaluated. It is crucial due to from assessing the model the outcome of model is represented either it is significant or not. To conduct assessment of the model the comparison between the main or real target and the predicted data is done. To do so, the training and test dataset are needed:

```
real_data=ifelse(rowSums(dataset_test) > 1.0, 1, 0)
outcome_real_data_onehot=to_categorical(real_data)
```

Then, the training data will be assessed:

```
# Trained data
evaluate(model, dataset, outcome_data_onehot, verbose
= 0)

# one hot data is necessary for Testing data assessment
evaluate(model, dataset, outcome_real_data_onehot,
verbose = 0)

# assessment results for training data
"loss of data" is  0.0117954
"accuracy" is  0.9982

# assessment results for testing data
"loss of data" is  0.01597541
"accuracy" is  0.998
```

The following figure represents the results of assessing the model for training and testing data:

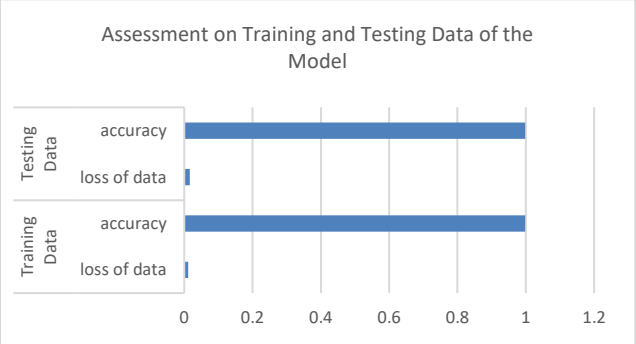

Figure 8: Shows the Final Result of Assessing the Model with Training and Testing Data

In figure 8 it can be clearly seen that the model is very significant and improved due to the results of assessing the model for training and testing data is very close to 1 which is vitally important. In addition, loss of the data which is more essential, is very close to 0 for the training and testing data. This shows that the model is completely significant.

Conclusion

In conclusion, this research paper modified long short term memory LSTM technique. A parameter was added to decide of the outcome of the values. The research goes through several tasks; creating dataset, modification of LSTM, determining vector dimensions, building or establishing sequential model, validating the built model, and assessing the model. Consequently, the results of assessing for training and testing data were vitally important and very significant due to the model accuracy picked the top and very close to 1. Additionally, loss of the data was almost or very close to 0 which is more crucial. As a result, the modification of LSTM technique and creating the model based on this modification was very significant and perfect.

References

[1] Ralf C., and Eric R. (2019). Understanding LSTM -a tutorial into Long Short-Term Memory Recurrent Neural Networks. Neural and Evolutionary Computing. arXiv:1909.09586

[2] Alex Sherstinsky, (2021), Fundamentals of Recurrent Neural Network (RN) and Long Short-Term Memory (LSTM) Network, Elsevier "Pysica D: Nonlinear Phenomena" journal, Volume 404, March 2020: Special Issue on Machine Learning and Dynamical Systems, DOI: 10.1016/j.physd.2019.132306

[3] Nimesh inha, (2018), Recurrent Neural Networks. Available at: https:/towardsdatascience.com/understanding-lstm-and-its-quick-implementation-in-keras-for-sentiment-analysis-af410fd85b47. Accessed on (13/1/2021)

[4] Xuan-Hien Le, Hung Viet Ho, Giha Lee, and Sungho Jung, (2019), Application of Long Short-Term Memory (LSTM) Neural Network forFlood Forecasting, Water Journal, MDPI.

[5] Sun, Jianlei (John) & Ma, X. & Kazi, M.. (2018). Comparison of Decline Curve Analysis DCA with Recursive Neural Networks RNN for Production Forecast of Multiple Wells. 10.2118/190104-MS.

[6] L. Rahman, N. Mohammed and A. K. Al Azad, (2016), A new LSTM model by introducing biological cell state, 2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT), Dhaka, Bangladesh, 2016, pp. 1-6, doi: 10.1109/CEEICT.2016.7873164.

[7] Manik Soni, (2018), Understanding architecture of LSTM cell from scratch with code. Available at: https://medium.com/hackernoon/understanding-architecture-of-lstm-cell-from-scratch-with-code-8da40f0b71f4. Accessed on 12/2/2021

[8] Van Houdt, G., Mosquera, C. and Nápoles G., (2020) A review on the long short-term memory model. Artif Intell Rev 53, 5929–5955 (2020). https://doi.org/10.1007/s10462-020-09838-1

[9] Pranjal Srivastava, (2017), Essentials of Deep Learning : Introduction to Long Short Term Memory. Available at: https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/. Accessed on (1/2/2021)