

A Hydraulic Condition Monitoring System Based on Convolutional BiLSTM Model

Kyutae Kim and Jongpil Jeong(✉)

Department of Smart Factory Convergence, Sungkyunkwan University, Suwon, Gyeonggi-do 16419, Republic of Korea
{ryukkt62,jpjeong}@skku.edu

Abstract. In this paper, to monitor the conditions of hydraulic system, a real-time monitoring method based on convergence of convolutional neural networks (CNN) and a bidirectional long short-term memory networks (BiLSTM) is proposed. This method uses CNN and BiLSTM. In the CNN, the feature is extracted from the time-series data entered as an input, and in the BiLSTM, information from the feature is learned. Then, the learned information is sent to the Sigmoid classifier and it classified whether the system is stable or unstable. The experimental results show that compared to other deep learning models, this model can more accurately predict the conditions of the hydraulic system with the data collected by the sensors.

Keywords: Long short term memory · Deep learning · Classification · Convolutional neural networks · Hydraulic system · Bidirectional LSTM

1 Introduction

Mechanical fault diagnosis is an important part of the smart factory and is the key in the 4th industry [1]. Numerous sensors are installed in the hydraulic system to detect failures and the data collected from these sensors can be used to monitor whether the equipment is working properly [2,3].

Recent years has seen a growing popularity of performing fault detection with data collected from these sensors [6]. As data volumes increase and computing power advances, there are increasing attempts to take advantage of deep learning. In particular, convolutional neural networks (CNN) have been widely used in classification problems and it have shown high-performance [4,5]. However, the CNN relies too much on extracting high-dimensional features [7]. Too many convolutional layers cause vanishing gradient, and too few convolutional layers can't find global features [8]. Therefore, this paper proposes a method to monitor the condition of the hydraulic system using a model that combines a CNN and a bidirectional long short-term memory (BiLSTM) networks [9].

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12250, pp. 588–603, 2020.
https://doi.org/10.1007/978-3-030-58802-1_42

The data collected from the sensor first enters the CNN for feature extraction, and then into the BiLSTM networks for feature extraction of the long-distance dependence information. Finally, learned features are sent to a sigmoid classifier that identifies the conditions of the hydraulic system [10]. This method can be useful for diagnosing the conditions of a hydraulic system in an actual industrial sites.

The remainder of this paper is organized as follow. Section 2 presents the CBLSTM model proposed in this paper. Section 3 introduces the data used in the experiment, presents the equipment, and the experimental process. Section 4 analyzes the experimental results on the conditions of the hydraulic system. Section 5 concludes with a summary and presents a future study plan.

2 Related Work

In this paper, a model combining a CNN and a recurrent neural networks (RNN) was applied to monitor the condition of a hydraulic system. The fusion model, convolutional bidirectional long short-term memory, reveals the correlation between time-series data that was ignored when using CNN alone, and avoids the gradient vanishing and gradient explosion problems frequently encountered in RNN [11].

2.1 CNN

The CNN is a model inspired by the way the brain's visual cortex works when it recognizes objects [12]. It has received great attention because of its excellent performance in image classification. This has made great strides in machine learning and computer vision applications. One-dimensional CNN can be used to analyze time series data collected from sensors. Several filters in the convolution layer filter the input data and superimpose the obtained results to get feature maps [13]. The feature map of the convolutional layer can be expressed as follows:

$$y_j = F(\sum_{i=0}^{n} x_i \times w_{ij} b_j) \tag{1}$$

y_j is the j-th output data, and x_i is the feature map of the previous layer. w_{ij} is the j-th kernel, b_j is the bias of the j-th kernel.

The pooling layer is usually located immediately after the convolution layer. In the pooling layer, the input data is divided into multiple zones and the maximum values are extracted from the zones to form new data. This is called max pooling. There is another pooling method, which is called average pooling as a method of extracting the average value in a zone. Data that passes through the convolution layer and pooling layer is called a feature map. After the input data is subjected to the convolution calculation, the activation function is applied and then the feature map is created as it passes through the pooling layer [7]. The

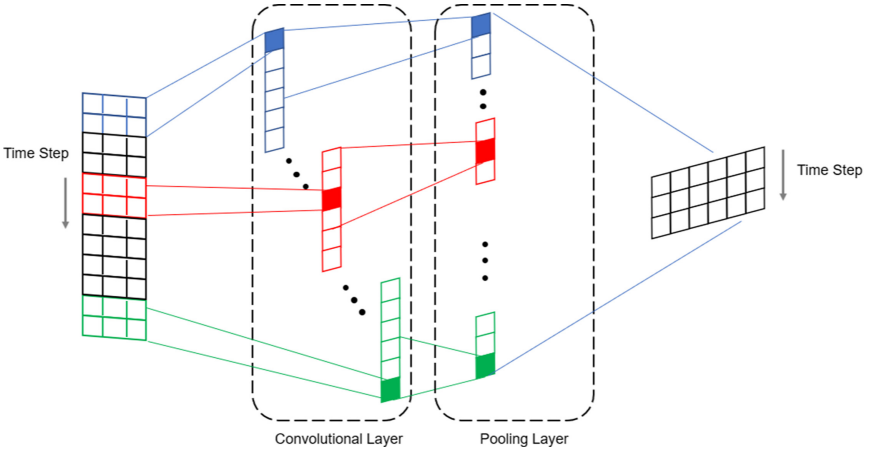

Fig. 1. The structure of the one-dimensional convolutional neural network

set of convolutional layers and pooling layers is a basic structure of CNN (see Fig. 1).

In order for elements in the original data array to participate equally in the operation, virtual elements must be added to both ends of the original array. In this case, using 0 as a virtual element is called zero padding. You can make all elements of the original array participate in the operation equally by adding the appropriate number of zero paddings [14]. The padding method in which all elements of the original array participate in the operation equally is called full padding (see Fig. 2).

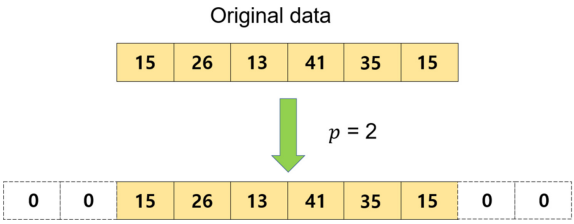

Fig. 2. One-dimensional padding

2.2 LSTM

LSTM was first introduced to overcome the gradient vanishing and gradient exploding problems encountered in RNN. The basic building block of LSTM is a memory cell, which means a hidden layer [15,16]. To solve gradient vanishing and gradient exploding problems, each memory cell has a circular edge that

maintains the proper weight $w = 1$. The output of this circular edge is called the cell state. The structure of the LSTM is shown in Fig. 3.

The cell state $C^{(t-1)}$ of the previous time step is changed without being multiplied by any weight directly to obtain the cell state $C^{(t)}$ of the current time step. The flow of information in a memory cell is controlled by several operations described below. In Fig. 2, $\odot$ means element-wise multiplication and $\oplus$ means element-wise addition. Also, $x^{(t)}$ is the input data at time step t, and $h^{(t-1)}$ is the output of the concealed unit at time step $t-1$. The four boxes are represented by a sigmoid function (σ) or tanh activation function and a series of weights. This box is linearly combined after matrix-vector multiplication on the input. The unit calculated by the sigmoid function is called a gate and is output through $\odot$. There are three types of gates in LSTM cells. Forget gate, input gate, output gate.

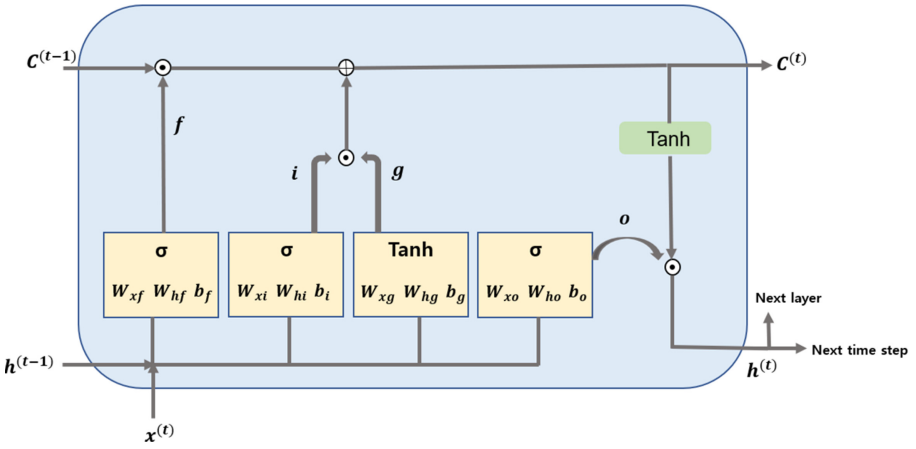

Fig. 3. The structure of the LSTM neurons

The forget gate f_t resets the cell state so that the memory cell does not grow indefinitely. In fact, the forget gate determines what information to pass and what to suppress. It is calculated as:

$$f_t = \sigma(W_{xf}x^{(t)} + W_{hf}h^{(t-1)} + b_f) \tag{2}$$

The input gate (i_t) and input node (g_t) are responsible for updating the cell status, and are calculated as follows.

$$i_t = \sigma(W_{xi}x^{(t)} + W_{hi}h^{(t-1)} + b_i) \tag{3}$$

$$g_t = tanh(W_{xg}x^{(t)} + W_{hg}h^{(t-1)} + b_g) \tag{4}$$

At time step t, the cell state is calculated as follows.

$$C^{(t)} = (C^{(t-1)} \odot f_t) \oplus (i_t \odot g_t) \tag{5}$$

The output gate (o_t) updates the output value of the hidden unit.

$$o_t = \sigma(W_{xo}x^{(t)} + W_{ho}h^{(t-1)} + b_o) \tag{6}$$

Invented in 1997 by Schuster and Faliwal, bidirectional LSTM was introduced to enable networks to utilize larger amounts of information. BiLSTM connects two hidden layers in opposite directions to the same output. With this type of structure, information can be simultaneously obtained from the previous sequences and the subsequent sequences [7]. BiLSTM does not need to modify the input data and their future input information can be reached in the current state [17]. The structure of the BiLSTM network is shown in Fig. 4.

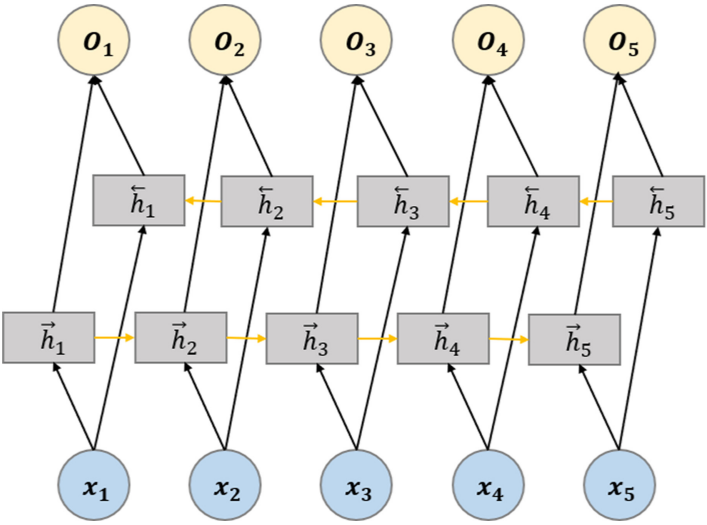

Fig. 4. The structure of the BiLSTM network model

The features of input data are generated as $\overrightarrow{h_t}$ by forward LSTM network, and $\overleftarrow{h_t}$ is generated by inverse LSTM network. And a vector P_t at time step t is generated by the BiLSTM network. The formula is as follows:

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(h_{t-1}, x_t, C_{t-1}) \tag{7}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(h_{t+1}, x_t, C_{t+1}) \tag{8}$$

$$P_t = [\overrightarrow{h_t}, \overleftarrow{h_t}] \tag{9}$$

The model used in this paper was applied with a dropout of 0.2 probability, which can prevent overfitting. In addition, the relu function was used for the activation function, and the sigmoid function was used for the last fully connected layer. Binary-crossentropy was used as a loss function because it is a binary classification model [18]. The formula for sigmoid and cross-entropy error are as follow:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{10}$$

$$E = -\sum_{n=1}^{N} (t_n log(y_n) + (1 - t_n) log(1 - y_n)) \tag{11}$$

In the sigmoid equation, if the value of x moves away from 0 in the negative direction, y is close to 0 because the denominator increases. Also, if the value of x is away from 0 in the positive direction, y is close to 1 because e^{-x} approaches 0.

$t_n \in \{0,1\}$ and $y_n \in [0,1]$, $y_n = \text{Sigmoid(net)}$. The optimization algorithm is a method of finding the point at which the value of the loss function is minimal. Optimization algorithms are very important to efficiently and reliably reach the global optimal solution. In this model, RMSProp was used, which overcomes the drawback of stagnating learning by reducing the amount of correction in Adagrad [19].

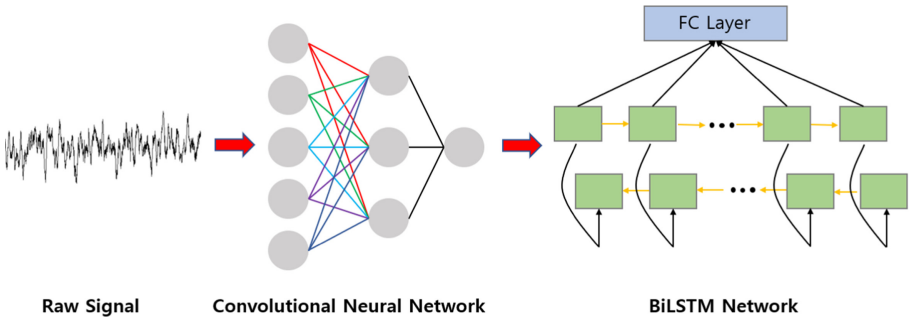

Fig. 5. Framework of CBLSTM

3 CBLSTM

This paper applies CNN and RNN convergence to monitor the condition of hydraulic system and constructs two networks models: CNN and BiLSTM. CBLSTM effectively resolves the problem that mutual relation between input data is ignored on a single CNN and prevents gradient vanishing and explosion [20,21].

A CBLSTM model is proposed to further improve the accuracy of model predictions [22]. The CBLSTM model consists of three parts: First, the input

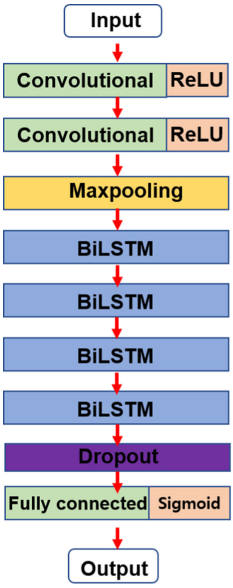

Fig. 6. Diagram of CBLSTM

data is fed into the model and the model extracts local feature from a time series data, primarily using one-dimensional CNN for filtering, In the second part, the BiLSTM network is used for information of long-distance dependence. Finally, the things that learned in the previous layers are transferred to classifier which is used to classify the condition of hydraulic system using sigmoid function. Figure 5 and Fig. 6 show model framework and diagram of CBLSTM.

4 Experiments and Discussion

In this experiment, time series data collected by sensors installed in the hydraulic system test rig were used. This test rig consists of working circuit, cooling circuit, and filtration circuit and the sensors installed in it measure various physical values such as pressure, vibration and temperature. Each sensor has a different sampling rate, which varies 1 Hz 100 Hz. The hardware platform used in this experiment: Intel R CoreTM i7-8700K CPU @ 3.70 GHz, and 32 GB of RAM. The software that was used to test was Python 3.7 and Tensorflow. It was shown in Table 1.

The model used in this paper is CBLSTM, which is a convergence of CNN and BiLSTM. It is consist of 2 one-dimentional convolution layers followed 1 max pooling layer, 2 bidirectional layers and fully connected layer. The parameter settings and diagram for this model are shown in Table 2.

Three models were additionally tested to compare the performance of CBLSTM with other models. The basic LSTM model, bidirectional LSTM, and

Table 1. System specification for experiments

Item	Details
CPU	Intel(R) Core(TM) i7-8700K
CPU Frequency	3.70 GHz
Memory	32 GB
Language	Python 3.7

Table 2. The parameters of CBLSTM network

Layer name	Output feature size	Network
Input layer	(54, 1)	–
Convolution layer1	(52, 62)	Conv1D, kernel size = 3, param = 256
Convolution layer2	(50, 64)	Conv1D, kernel size = 3, param = 12352
Pooling layer	(16, 64)	Maxpooling1D
Bidirectional laer1	(16, 40)	param = 13600
Bidirectional laer2	(16, 40)	param = 9760
Bidirectional laer3	(16, 40)	param = 9760
Bidirectional laer4	40	param = 9760
Dropout	40	–
Fully connected layer	2	Dense, Sigmoid, param = 82
Output layer	2	Binary Crossentropy

CLSTM model combining CNN and LSTM were used. First, in the case of the LSTM model, the loss rapidly decreased as epoch progressed in both training and test. At first, the loss was greater than 0.6, but as learning progressed, it decreased to 0.5. On the other hand, the accuracy did not improve significantly, and it was 0.6 at the beginning of the training and stayed at 0.7 even after the training was completed. Finally, the result of the test was loss of 0.512 and accuracy of 0.680. The test results were not satisfactory and there was a possibility of further development as the learning progressed further (see Fig. 7 and 8).

The second is the result of experiments with bidirectional LSTM. In this case, as with LSTM, the loss was sharply reduced in both training and test. At first it was 0.65, but as learning progressed, it rapidly decreased to 0.45. In the test set, the loss increased sharply when the epoch was 12, but it did not significantly affect the performance. In the case of accuracy, it was about 0.6 at first, and as the epoch increased, the accuracy also increased, finally reaching 0.708 in the test set (see Fig. 9 and 10).

In the case of a combination model of CNN and LSTM, it was very unstable. In the case of loss, the test set showed a significant change, especially when epoch = 3 and 6, the loss increased significantly. Finally, the test results showed that the loss was only reduced to 0.52. Accuracy was seen to improve as learning progressed. There was no significant difference between the training set and the test set, and the accuracy was finally recorded as 0.742, which is better than the LSTM and BiLSTM alone (see Fig. 11 and 12).

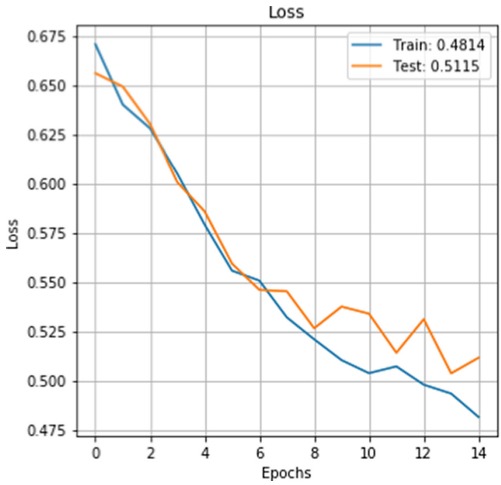

Fig. 7. Loss and accuracy of LSTM in training and test set

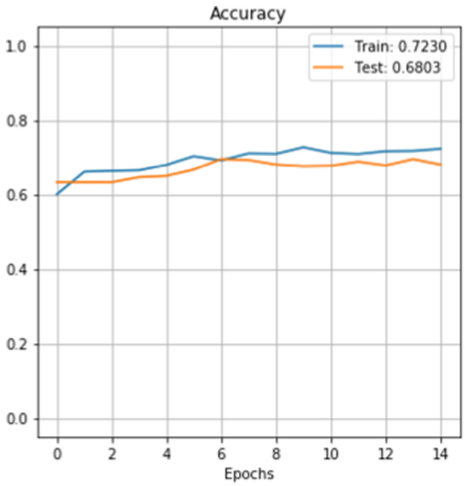

Fig. 8. Loss and accuracy of LSTM in training and test set

For the training parameters, 0.2 for the dropout, 15 for the epoch, and 16 for the batch size, 20% of the total dataset for the training set, and 25% for the validation set. Graphs were shown during the experiment and during the training.

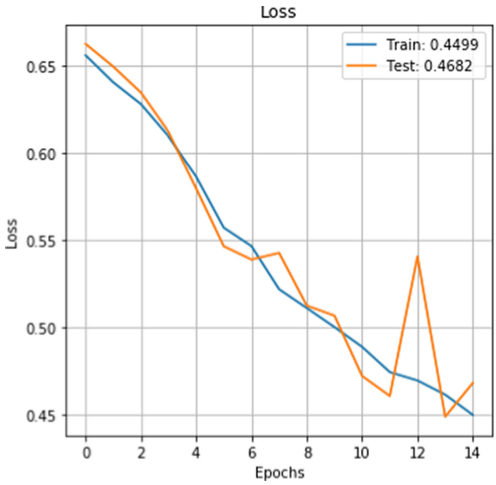

Fig. 9. Loss and accuracy of BiLSTM in training and test set

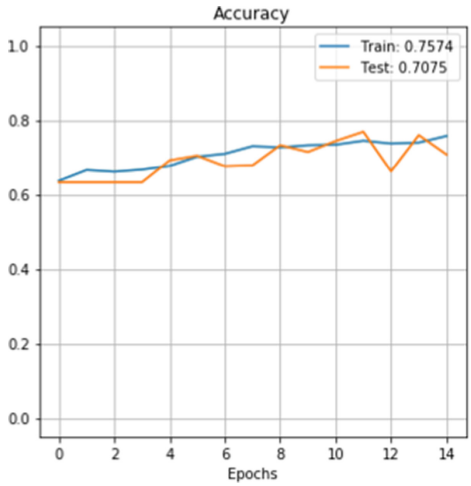

Fig. 10. Loss and accuracy of BiLSTM in training and test set

Looking at the experimental results of CBLSTM, loss decreased stably in the training set, but in the test set, the epcoch increased significantly at 8. The accuracy was lower than that of the training set, but we got decent accuracy (see Fig. 13 and 14). To compare with other models, three more models were tested in addition to CBLSTM. The experimental result of simple LSTM model,

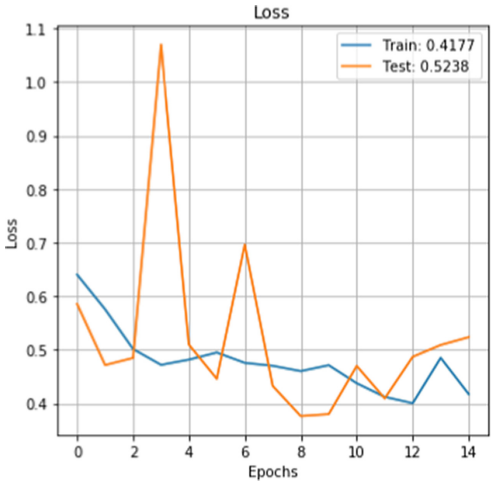

Fig. 11. Loss and accuracy of CLSTM in training and test set

a BiLSTM model, a model combining CNN and LSTM and a model combining CNN and BiLSTM are shown in Fig. 15 and 16. Looking at the loss first, as the learning progressed, all four models were lowered. LSTM and BiLSTM showed similar patterns, and CLSTM and CNNBiLSTM showed similar patterns. The simple LSTM model only reduced the loss to around 0.5, the BiLSTM to 0.45, the CLSTM to 0.43, and the CBLSTM to 0.35. When looking at the accuracy, LSTM was the lowest, and BiLSTM, CLSTM, and CBLSTM were the highest. Finally, the accuracy of CBLSTM increased to 0.83, that of CLSTM increased to 0.8, and BiLSTM and LSTM of 0.75. In particular, CLSTM showed unstable appearance, such as when the epoch was between 12 and 14, the accuracy was greatly reduced (see Fig. 15 and 16). Table 3 shows the loss and accuracy of the train and test sets of the four models.

Table 3. Comparison of accuracy and loss

	Train loss	Train acc	Test loss	Test acc
LSTM	0.481	0.723	0.511	0.680
BiLSTM	0.449	0.757	0.468	0.707
CLSTM	0.417	0.804	0.523	0.742
CBLSTM	0.350	0.833	0.341	0.810

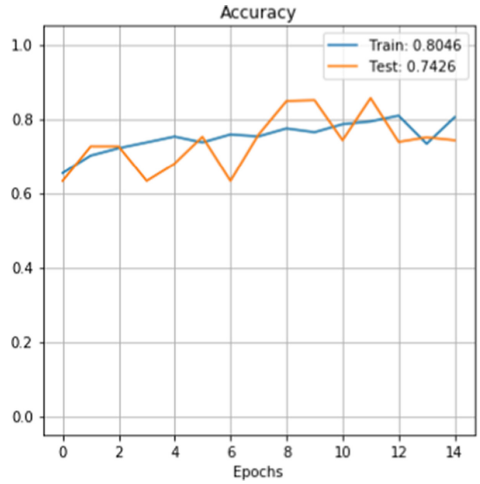

Fig. 12. Loss and accuracy of CLSTM in training and test set

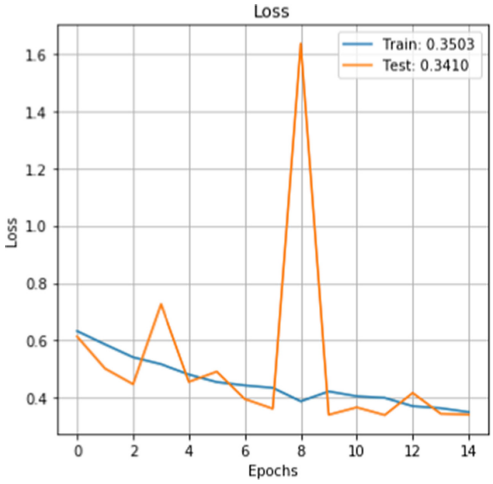

Fig. 13. Loss and accuracy of CBLSTM in training and test set

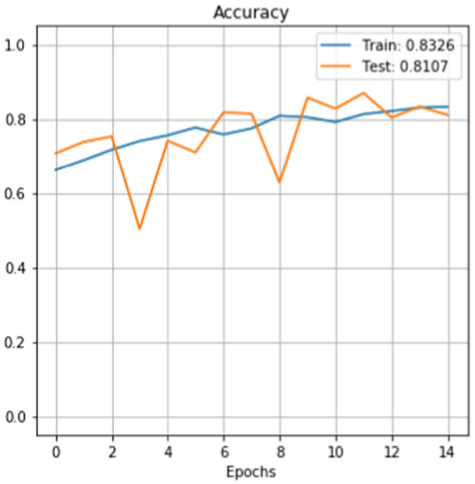

Fig. 14. Loss and accuracy of CBLSTM in training and test set

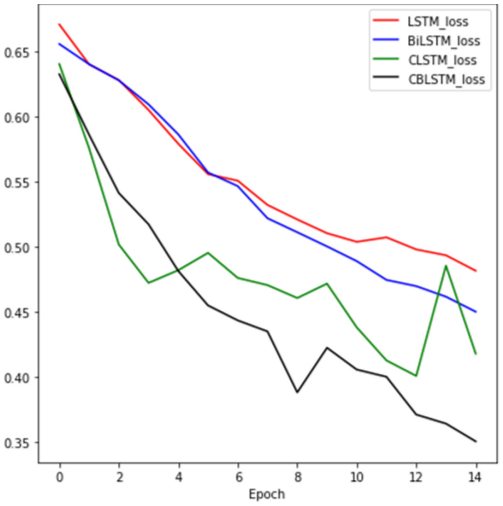

Fig. 15. Loss and accuracy of the training set

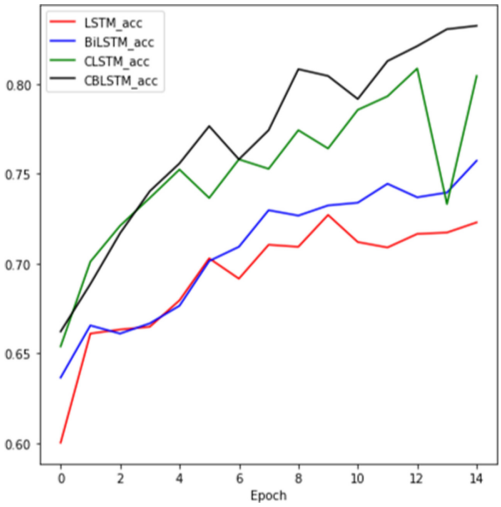

Fig. 16. Loss and accuracy of the training set

5 Conclusion

In this paper, we proposed a method to classify the condition of a system as stable or unstable using data collected from sensors in a hydraulic system using a model combining CNN and BiLSTM. The accuracy of the CBLSTM reached 81%, recording higher accuracy compared to the other three models tested with the same dataset and showing superior performance with lower loss. In a future study, we will add attention modules to the CBLSTM network to develop the model and apply it to other manufacturing domains.

Acknowledgment. This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2020-2018-0-01417) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation).

This work was supported by the Smart Factory Technological R&D Program S2727186 funded by Ministry of SMEs and Startups (MSS, Korea).

References

1. Mahmood, M.F., Hussin, N.: Information in conversion era: impact and influence from 4th industrial revolution. Int. J. Acad. Res. Bus. Soc. Sci. **8**(9), 320–328 (2018)
2. Helwig, N., Pignanelli, E., Schütze, A.: Condition monitoring of a complex hydraulic system using multivariate statistics. In: 2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings, pp. 210–215. IEEE (2015)

3. Helwig, N., Pignanelli, E., Schütze, A.: D8. 1-detecting and compensating sensor faults in a hydraulic condition monitoring system. In: Proceedings SENSOR 2015, pp. 641–646 (2015)
4. Lawrence, S., Giles, C.L., Tsoi, A.C., Back, A.D.: Face recognition: a convolutional neural-network approach. IEEE Trans. Neural Netw. **8**(1), 98–113 (1997)
5. Hershey, S., et al.: CNN architectures forlarge-scale audio classification. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 131–135. IEEE (2017)
6. Mehranbod, N., Soroush, M., Panjapornpon, C.: A method of sensor fault detection and identification. J. Process Control **15**(3), 321–339 (2005)
7. Chen, Q., Xie, Q., Yuan, Q., Huang, H., Li, Y.: Research on a real-time monitoring method for the wear state of a tool based on a convolutional bidirectional LSTM model. Symmetry **11**(10), 1233 (2019)
8. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
9. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)
10. Han, J., Moraga, C.: The influence of the sigmoid function parameters on the speed of backpropagation learning. In: Mira, J., Sandoval, F. (eds.) IWANN 1995. LNCS, vol. 930, pp. 195–201. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-59497-3_175
11. Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: Eleventh Annual Conference of the International Speech Communication Association (2010)
12. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
13. Yang, J., Nguyen, M.N., San, P.P., Li, X.L., Krishnaswamy, S.: Deep convolutional neural networks on multichannel time series for human activity recognition. In: Twenty-Fourth International Joint Conference on Artificial Intelligence, pp. 3995–4001 (2015)
14. Muquet, B., Wang, Z., Giannakis, G.B., De Courville, M., Duhamel, P.: Cyclic pre-fixing or zero padding for wireless multicarrier transmissions? IEEE Trans. Commun. **50**(12), 2136–2148 (2002)
15. Cowan, N.: What are the differences between long-term, short-term, and working memory? Prog. Brain Res. **169**, 323–338 (2008)
16. Cowan, N.: Activation, attention, and short-term memory. Memory Cogn. **21**(2), 162–167 (1993). https://doi.org/10.3758/BF03202728
17. Ullah, A., Ahmad, J., Muhammad, K., Sajjad, M., Baik, S.W.: Action recognition in video sequences using deep bi-directional LSTM with CNN features. IEEE Access **6**, 1155–1166 (2017)
18. Li, C.H., Lee, C.: Minimum cross entropy thresholding. Pattern Recogn. **26**(4), 617–625 (1993)
19. Mukkamala, M.C., Hein, M.: Variants of RMSProp and adagrad with logarithmic regret bounds. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 2545–2553. JMLR. org (2017)
20. Huang, Y., Wang, W., Wang, L.: Bidirectional recurrent convolutional networks for multi-frame super-resolution. In: Advances in Neural Information Processing Systems, pp. 235–243 (2015)

21. Cai, R., Zhang, X., Wang, H.: Bidirectional recurrent convolutional neural network for relation classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 756–765 (2016)
22. Liu, Q., Zhou, F., Hang, R., Yuan, X.: Bidirectional-convolutional LSTM based spectral-spatial feature learning for hyperspectral image classification. Remote Sensing **9**(12), 1330 (2017)