Chemical Substance Classification using Long Short-Term Memory Recurrent Neural Network

Jinlei Zhang¹, Junxiu Liu^{1*}, Yuling Luo¹, Qiang Fu¹, Jinjie Bi¹, Senhui Qiu^{1,2}, Yi Cao³, Xuemei Ding^{4,5}

¹ Guangxi Key Lab of Multi-Source Information Mining & Security,
Faculty of Electronic Engineering, Guangxi Normal University, Guilin, China, 541004

² Guangxi Experiment Centre of Information Science, Guilin, China

³ Department of Business Transformation and Sustainable Enterprise, Surrey Business School,
University of Surrey, Surrey, UK

⁴ Faculty of Software, Fujian Normal University, Fuzhou, China, 350108

⁵ School of Computing and Intelligent Systems, Faculty of Computing and Engineering, Ulster University, UK, BT48 7JL

*Email: liujunxiu@mailbox.gxnu.edu.cn

Abstract—This paper proposed a chemical substance detection method using the Long Short-Term Memory of Recurrent Neural Networks (LSTM-RNN). The chemical substance data was collected using a mass spectrometer which is a time-series data. The classification accuracy using the LSTM-RNN classifier is 96.84%, which is higher than 75.07% of the ordinary feed forward neural networks. The experimental results show that the LSTM-RNN can learn the properties of the chemical substance dataset and achieve a high detection accuracy.

Keywords- recurrent neural networks; chemical substances; long short-term memory; feed forward neural networks

I. INTRODUCTION

Artificial Neural Network (ANN) is an important machine learning technique. Currently, one of the most popular machine learning technique is Deep learning [1], [2]. Deep learning can achieve complex function approximation by studying a deep nonlinear network structure. It also shows a strong ability to learn essential features from a few samples. Deep learning methods, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Deep Belief Networks (DBNs), have become popular cognitive approach and have been applied in many fields [3]–[6].

Recently, RNNs [7], especially Long Short-Term Memory (LSTM) [8] model, is being studied due to the computational capabilities for solving many challenging problems, such as intrusion detection [9], action recognition [10], [11], multilingual machine translation [12], multimodal translation between videos and robot control [13]. In these applications, learning the correlations between different time steps is an important step in achieving artificial intelligence.

Multi-layer neural network has excellent performance. Even today, these networks can solve the classification tasks better than other methods [14]. These tasks include handwriting recognition, speech recognition, and image recognition. Pattern recognition is a most common artificial intelligence application. Among multi-layered neural networks, the typical network is the Back-Propagation (BP) neural network which is currently one of the most widely used neural network. The BP neural network is trained using the back-propagation algorithm [15]. It can learn and store a large number of input and output patterns, and does not need the

mathematical descriptions of input and output. The invention of this network has brought great hope for machine learning, especially the statistical-based learning. The BP algorithm allows an artificial neural network to learn statistical rules from a large number of samples to predict unknown events.

The functions which can be realized by BP network architecture is limited. The reason is that the human brain has amazing computing power. Actually the classification task is only a small part of the computing functions of the brain. We can not only identify individual cases, but also analyze the logical relationship of different inputs. Amount of content is included in these input information and logical relationships. The relevance between time and spatial is still complex and the content of the information is also varied. The input information are multi-dimensions and need to be processed simultaneously. Because they are changing in real time, the data or samples that need to be analyzed and processed are very complex and diverse. As a better neural network structure than BP, the RNN can learn the logical relationships between the information.

This work emphasis on the usage and implementation of LSTM-RNN for low-levels and mixed substances based target classification. The remainder of this work is structured as follows. Section II reviews RNNs and LSTM concepts and Section III introduces the chemical substance dataset. Section IV reports the experimental results and performance analysis. Section V provides a summary.

II. LONG SHORT-TERM MEMORY

The most general architecture of globally RNN was proposed by Williams and Zipser [16]. In the traditional neural network model, the flow of information is from the input layer to the hidden layer and then to the output layer. The pre and post-layer are fully connected, there is no connection between the neuron nodes in each layer. This neural network architecture cannot achieve a good performance for some applications, especially for the time series data. For example, in the speech recognition, the next word a person would like to say is based on the history of the previous words, as the words in a sentence are not isolated. However in the RNN, the current output of a sequence is related to the previous output, i.e. the RNN networks memorize previous information and

apply it to the calculation of the current output. Different to the conventional neural network, the neuron nodes in the hidden layer are connected together. And the input of the hidden layer includes not only the output of the input layer, but also the previous output of the hidden layer.

One problem of the RNN is the vanishing (or exploding) gradient problem. In this section, the architecture of RNNs and the gradient vanishing problem are briefly discussed. Then Long Short-Term Memory to address this problem is introduced.

A. Recurrent neural networks

The RNN is the extension of the conventional Feed Forward Neural Networks (FFNNs) in the time scale. Assuming that an input sequence, the hidden state sequence, and output vector sequence denoted by x , h and y , respectively. RNNs combine the input vector with the previous state vector to produce a new state vector. Hidden state h_t is described as

$$h_t = f(Ux_t + Wh_{t-1} + b_1), \quad (1)$$

where U , W are the weights for the connections from the input layer to the hidden layer, hidden layer to the hidden layer, respectively. Hidden state h_t captures information which happened in all the previous time steps. Output vector y_t is described as

$$y_t = g(Vh_t + b_2), \quad (2)$$

where V is the weights for the connections from the hidden layer to the output layer. In (1) and (2), the f and g are activation functions that squash the dot products to a specific range. The function f is usually $\tanh$ or ReLU . The g can also be a softmax . The b_1 and b_2 are biases that help offset the outputs from the origin. Fig. 1 shows the common topologies of RNNs.

Figure 1. The architecture of RNNs [17].

The conventional RNNs use Back Propagation Training Time (BPTT) to handle a variable-length sequence input [18]. RNNs have memory ability, but long-term memory is limited. It makes the gradient explode or vanish while training with BPTT algorithm, which is addressed in the approach of [19].

B. Long Short-Term Memory

The LSTM is an architecture which is first proposed by Hochreiter and Schmidhuber [8] and refined by many other researchers. Fig. 2 shows a single LSTM cell.

Figure 2. The architecture of LSTM [17].

For each time step t , x_t is the input to the memory cell layer, σ is the logistic sigmoid function, i_t , f_t and o_t are values of the input gate, forget gate, output gate respectively. They are described by

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i), \quad (3)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f), \quad (4)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o). \quad (5)$$

In LSTM, three gates control the information flow. The input gate decides which values will be update. The forget gate defines how much of the previous state h_{t-1} are allowed to pass through, and the output gate defines how much of the internal state are exposed to the next layer [20]. The candidate value g_t is computed by the current input x_t and the previous hidden state h_{t-1} . The key to LSTM is the cell state c_t , as the i_t , f_t and g_t interact with c_t . The g_t and c_t are described by

$$g_t = \tanh(W_{xg}x_t + W_{hg}h_{t-1} + b_g), \quad (6)$$

$$c_t = f_t c_{t-1} + i_t g_t, \quad (7)$$

where W_{xi} , W_{xf} , W_{xo} and W_{xg} are the weights for the connections. These weights propagates from the input data to the input gate, forget gate, output gate and candidate value, respectively. Similarly, this also applies to W_{hi} , W_{hf} , W_{ho} , W_{hg} , which are the weights for the connections from the hidden layer (at previous time-step) to the input gate, forget gate, output gate and candidate value (at the current time-step) respectively, and b_i , b_f , b_o and b_g are the corresponding bias.

Finally, the hidden state h_t at time t is computed by multiplying the $\tanh(c_t)$ with the output gate. This can be described by

$$h_t = o_t \tanh(c_t). \quad (8)$$

In the LSTM, the three gates (input gate, forget gate, output gate) are used to solve the vanishing and exploding gradient problems. In LSTM-RNN architecture, the recurrent hidden layer is replaced by LSTM cell.

III. DATASET

In this paper, the dataset used for the experiments are based on the Mass Spectrometry data from the ICARIS 2009 & 2010 competition [21]. There are 58,499 records in the dataset and each record has 270 features. The dataset consists of four parts and each part contains a pair of files –the data file with comma-separated values of signal strength for each time-step, and the text file containing information about the substances present in each set. Seven different substances are introduced in this dataset.

IV. EXPERIMENT

In this section, two experiments are carried out to test the performance. The first experiment is to measure the performance of ordinary Feed Forward Neural Networks (FFNNs). The second experiment aims to measure the performance of LSTM-RNN.

A. The model set up

A three-layer model is built up in this experiment. The input layer includes 270 nodes, and the hidden layer has 500 nodes. The output layer is 8 dimensions where 7 of them are contained in the data set. There is also an unknown substance. Therefore a total of 8 dimensions are needed. The Softmax function is used as the activation for output layer and RMSprop is used instead SGD for an optimizer. And the loss function is mean squared error and the learning rate is a fixed value of 0.001. Our model is based on the Python Deep Learning library Keras [20].

B. Experimental results using FFNNs

Table I show the modules of Keras (dense, activation). 32 refers to mini-batch size. A good estimate of the true gradient can be got quickly and help speed up gradient descent by using mini-batch size [15].

TABLE I. FFNN MODEL SUMMARY

Layer (type)	Output Shape	Parameter
dense_1 (Dense)	(32, 500)	135,500
activation_1 (Activation)	(32, 500)	0
dense_2 (Dense)	(32, 8)	4008
activation_2 (Activation)	(32, 8)	0

In this experiment, 20,000 records are used for training and 8,500 records are used for testing. The model summary is shown in Table I, 139,508 parameters need to be trained in FFNNs model.

C. Experimental results using LSTM-RNN

Table II show the module descriptions of Keras (lstm, activation), 32 refers to mini-batch size. This experiment uses the same training set and testing set as the first experiment.

The testing result of FFNNs is show in the top part of Table III. The average percentages of test loss and accuracy of LSTM-RNN in a scheduled size training epoch are given in the bottom of Table III. By comparing the results in Table III, the performance of the LSTM-RNN is stable and reliable, which is better than the FFNNs. 1,546,008 parameters need to be trained in LSTM-RNN model.

TABLE II. LSTM-RNN MODEL SUMMARY

Layer (type)	Output Shape	Parameter
lstm_1 (LSTM)	(32, 500)	1,542,000
dense_1 (Dense)	(32, 8)	4,008
activation_1 (Activation)	(32, 8)	0

TABLE III. PERFORMANCE COMPARISON USING DIFFERENT NETWORKS

Types	Epoch	Loss	Accuracy
FFNNs	100	0.0676	67.93%
	200	0.0445	79.22%
	300	0.0486	78.05%
LSTM-RNN	100	0.0075	96.86%
	200	0.0076	96.76%
	300	0.0074	96.89%

Traditional multilayer FFNN makes the assumption that all inputs are independent of each other. Certain types of input create certain types of hidden layers which create certain types of output layers. Unlike FFNNs, RNNs incorporate this dependence by having a hidden state, or memory, which holds the essence of what has been seen so far [12]. In reality, the output of LSTM-RNN model is not a pure function of the input. The input is just changing the content in the memory (LSTM cell), and the output is exclusively based on the memory.

V. CONCLUSION

In this paper, the LSTM-RNN is employed to classify the chemical substances which were collected by using the mass spectrometry. The experimental results showed that the LSTM-RNN is suitable to process this type of time-series data. Compared with FFNN, the classification result of LSTM-RNN is 96.84% which is much higher than FFNN of 75.07%. Therefore, LSTM-RNN achieves a better performance for the chemical substance classifications.

ACKNOWLEDGMENT

This research was partially supported by the National Natural Science Foundation of China under Grants 61661008 and 61603104, the Guangxi Natural Science Foundation under Grants 2015GXNSFBA139256 and 2016GXNSFCA

380017, the funding of Overseas 100 Talents Program of Guangxi Higher Education, the Research Project of Guangxi University of China under Grant KY2016YB059, Guangxi Key Lab of Multi-source Information Mining & Security under Grant MIMS15-07, the Doctoral Research Foundation of Guangxi Normal University, the grant from Guangxi Experiment Centre of Information Science, the Scientific Research Funds for the Returned Overseas Chinese Scholars from State Education Ministry, the Funds for Young Key Program of Education Department from Fujian Province, China (Grant No. JZ160425), Program of Education Department of Fujian Province, China (Grant No. I201501005).

REFERENCES

- [1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, "Deep Learning," Book in preparation for MIT Press, 2016.
- [2] Y. LeCun, B. Yoshua, and H. Geoffrey, "Deep Learning," *Nature*, vol. 521, no. 7553, 2015, pp. 436–444.
- [3] Gihan J. Mendis, Jin Wei, and Arjuna Madanayake, "Deep Learning Cognitive Radar for Micro UAS Detection and Classification," *Cognitive Communications for Aerospace Applications Workshop (CCAA)*, 2017, pp. 1–5.
- [4] Meiyin Wu and Li Chen, "Image Recognition Based on Deep Learning," *Chinese Automation Congress (CAC)*, 2015, pp. 542–546.
- [5] A. El Housseini, A. Toumi, and A. Khenchaf, "Deep Learning for Target Recognition from SAR Images," *Seminar on Detection Systems Architectures and Technologies (DAT)*, 2017, pp. 1–5.
- [6] J. Lemley, S. Bazrafkan, and P. Corcoran, "Deep Learning for Consumer Devices and Services: Pushing the Limits for Machine Learning, Artificial Intelligence, and Computer Vision," *IEEE Consum. Electron. Mag.*, vol. 6, no. 2, 2017, pp. 48–56.
- [7] M. Schuster and K. K. Paliwal, "Bidirectional Recurrent Neural Networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, 1997, pp. 2673–2681.
- [8] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, 1997, pp. 1735–1780.
- [9] J. J. Kim, J. J. Kim, H. L. T. Thu, and H. Kim, "Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection," *International Conference on Platform Technology & Service (PlatCon)*, 2016, pp. 1–5.
- [10] J. Donahue, L. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-term Recurrent Convolution Networks for Visual Recognition and Description," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, 2014, pp. 677–691.
- [11] A. Grushin, D. D. Monner, J. A. Reggia, and A. Mishra, "Robust Human Action Recognition via Long Short-Term Memory," *International Joint Conference on Neural Networks (IJCNN)*, 2013, pp. 1–8.
- [12] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *International Conference on Learning Representations (ICLR)*, 2015, pp. 1–15.
- [13] H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, and J. Schmidhuber, "A System for Robotic Heart Surgery that Learns to Tie Knots Using Recurrent Neural Networks," *Adv. Robot.*, vol. 22, no. 13–14, 2008, pp. 1521–1537.
- [14] N. H. Phong, B. Ribeiro, and P. De Marrocos, "Offline and Online Deep Learning for Image Recognition," *4th Experiment@International Conference (exp.at'17)*, 2017, pp. 171–175.
- [15] Michael A. Nielsen, "Neural Networks and Deep Learning," *Determination Press*, 2015.
- [16] R. J. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Comput.*, vol. 1, no. 2, 1989, pp. 270–280.
- [17] V. Jithesh, J. M. K. Srinivasa, "LSTM Recurrent Neural Networks for High Resolution Range Profile Based Radar Target Classification," *International Conference on Computational Intelligence & Communication Technology (CICIT)*, 2017, pp. 1–6.
- [18] P. J. Werbos, "Backpropagation Through Time: What It Does and How to Do It," *Proc. IEEE*, vol. 78, no. 10, 1990, pp. 1550–1560.
- [19] Y. Bengio, P. Simard, and P. Frasconi, "Learning Long Term Dependencies with Gradient Descent is Difficult," *IEEE Trans. Neural Networks*, vol. 5, no. 2, 1994, pp. 157–166.
- [20] A. Gulli and S. Pal, "Deep Learning with Keras," *Packt Publishing Ltd*, 2017.
- [21] *International Conference on Artificial Immune Systems (ICARIS) Competitions*, 2010.