Long Short-Term Memory based Operation Log Anomaly Detection

Vinayakumar R*, Soman KP* and Prabaharan Poornachandran†
*Centre for Computational Engineering and Networking (CEN), Amrita School of Engineering, Coimbatore
†Center for Cyber Security Systems and Networks, Amrita School of Engineering, Amritapuri
Amrita Vishwa Vidyapeetham, Amrita University, India
Email: vinayakumarr77@gmail.com

***Abstract*—Long short-term memory (LSTM) architecture is an important approach for capturing long-range temporal dependencies in sequences of arbitrary length. Moreover, stacked-LSTM (S-LSTM: formed by adding recurrent LSTM layer to the existing LSTM network in hidden layer) has capability to learn temporal behaviors quickly with sparse representations. To apply this to anomaly detection, we model the operation log samples of normal and anomalous events occurred in 1 minute time interval as time-series with the aim to detect and classify the events as either normal or anomalous. To select an appropriate LSTM network, experiments are conducted for various network parameters and network structures with the dataset provided by Cyber Security Data Mining Competition (CDMC2016). The experiments are run up to 1000 epochs with learning rate in the range [0.01-05]. S-LSTM network architecture has showed its strength by achieving the highest accuracy 0.996 with false positive rate 0.02 on the provided real-world test data by CDMC2016.**

***Index Terms*—Anomaly detection, Operation logs, deep learning: Recurrent neural network (RNN), Long short-term memory (LSTM) and Stacked-long short-term memory (S-LSTM)**

I. Introduction

A cloud infrastructure is of present interest, mostly adopted by many of the information technology (IT) firms instead of constructing their own monolithic or distributed IT infrastructures. The main reason behind this may be computing resources are more resilient, reliable, convenience and charge less cost to its users. It is adopted and largely benefitted by the top most IT companies such as Amazon, Microsoft, Google and many others. Despite of these immense advantageous, predicting the system failures is of main important task, otherwise it might cause performance degradation of resources or entire downtime of resources inside cloud infrastructure. The system failures can arise from various sources such as software defects, hardware faults, attacks and misconfigurations. This necessiate a novel mechanism to enhance the reliability of components in cloud infrastructures.

Each system generates an event sequence. An event sequence has characterized the nature of a particular system. These are either normal or anomalous, occurs at every second, minute, hour etc. Anomalous are events occurs frequently due to deliberate or non-deliberate faults. Moreover, anomalous events don't follow the regular behavioral patterns. The anomalous events are inspected as critical events. These critical events represent the system characteristics.

Virtualization is used as a technology to adopt more than one operating system (OS) in a single machine. This has been largely used in big data centers and cluster computing infrastructures mainly to create and run multiple services with the aim to amplify the utilization of hardware resources. Each system generates their own system events and collectively works together to infect the system resource or the whole system. The hardware resources are limited and this cannot allocate the requested resource all the times to virtualization. The events generated by various virtualizations are different and diverse. The analysis of these events to detect the system failures is of significant to maintain the system stability. Additionally, this enables to prevent anomalous events and recover the resources after an anomalous event has occurred by raising a signal to a network admin.

System logs or log files houses the system events. Log files are rich source of information for knowing the condition and characteristics of the system in run-time. They are the only way to get information for system failures [1]. Commonly used technique to identify and classify the anomalous events from normal events in system logs is anomaly detection. This has been widely studied by research communities due to it acts as a foundational approach in various fields, most importantly in Cyber security applications mainly in intrusion detection, fraud detection, fault isolation and many other applications [2].

Anomaly detection solutions existing in commercial systems are merely based on threshold based or statistical measures mainly use cumulative sum (CUSUM) and weighted moving average (EWMA) over time-interval to identify the hidden distribution of abnormal event patterns [3]. Both methods are parameterized and require an appropriate time-interval value to achieve a significant performance. Additionally, these methods can successfully detect only simple anomalous patterns by expressing them as counts and distributions. On the other direction, the existing anomaly detection methods have largely focused on single point based [4]. They haven't really taken the benefit of the past information in identifying the future event as either normal or anomalous. To overcome from these problems, we use long short-term memory (LSTM). LSTM is most prominent method for time-series data modeling that captures long-range temporal dependencies across time-steps. It has performed well on various long-standing artificial intelligence (AI) tasks [5]. This paper aims at transferring these performances towards the CDMC2016[1] task, real-world operation log anomaly detection.

The rest of the paper is organized in the following manner. Section II discusses the selected related work on anomaly detection, section III gives mathematical information of recurrent neural network (RNN) and long short-term memory, how they are trained and how they are adopted to predict the system failures based on the model created using the LSTM networks with the given operation log anomaly data. Section IV presents the details of CDMC 2016 task, parameter tuning in LSTM network including network structures, the proposed architecture for anomaly detection and evaluation results and finally we conclude a paper with conclusion and future work directions in section V.

II. Related Work

Anomaly detection has applicable in various tasks related to the diverse area. Thus, it has become a vivid are of research. Researchers have introduced many concepts and methods to deal with anomaly detection. The purpose of this section is to discuss the previously introduced selected approaches for anomaly detection briefly.

Analysis of system logs to find out anomalous event has been extensively used in recent days. [6] surveyed the anomaly detection methods in statistical and machine learning methods. The methods followed by researchers in current days to detect the anomalous events are briefly outlined in [7]. [8] author proposed a sequential pattern mining problem and further it has been studied by [9], [10]. [11] proposed Apriori-based method specifically generalized sequential pattern algorithm to find the length-1 candidates. They ignored candidates based on the support and count. For length-n candidates, an algorithm has to scan the data base n times. The performance of this approach is very limited due to the high computational cost and longer time for larger sequential patterns. [12] proposed a fuzzy set based solution that learns time varying patterns in sequential information. [13] introduced temporal pattern search that searches for time-varying patterns in historical data.

[14] used hidden markov model (HMM) to model the sequences of events from syslog to detect the critical events. Additionally, they showed their efficacy of model with real log system data. They also made a statement that HMM based mixture log anomaly detection model appear as a baseline system for event log files, including system calls, command lines, Web access logs. [15] proposed a new HMM based anomaly detection on latent data. They claimed that their approach outperformed the one-class SVM on real sequence data. [16] conducted experiments to analyze the comparable log analysis platform using the Naive Bayes and recurrent neural network (RNN) for software log analysis. They concluded RNN architecture was outperformed naive Bayes in all cases of experiments and the proposed system efficient to predict the next units in sequences based on the past sequences information.

The discussed methods have largely focused on single point anomalies. When dealing with the collective anomaly detection, it is necessary to use the past information in evaluating the current time-step value. The group of associated anomalous events concerning in the whole data set is mentioned as collective anomaly [2]. A single data point in collective anomaly is not generally considered as anomaly. A series of single points occurrence in collective anomaly can be considered as an anomalous event. LSTM has known for modeling time-series data by facilitating to associate the past information in current event time-step value. This mechanism has followed by [17], [18], [19]. [17] discussed the effectiveness of long short-term memory to anomaly or fault detection by modeling task relevant data as time-series. They trained LSTM network on normal data and used them as a predictor. The predicted errors across time-steps were modeled as multivariate Gaussian distribution. This newly formed distribution was used by them to estimate anomalous behaviors. The performance of the proposed approach was evaluated on ECG, space shuttle, power demand, and multi-sensor engine dataset. [19] discussed the various RNN mechanisms with deep neural network for log file anomaly detection. They used real-time logs of Very-large-scale integration (VLSI) design process with length of the sequence in logs followed varying length from 500 to 20000. [18] used LSTM encoder-decoder approach for multi-sensor anomaly Detection. They discussed the efficacy of the proposed approach on power demand, space shuttle, and ECG, and two real world engine datasets including the predictive and unpredictable characteristics

III. Background

This section provides the necessary background details of recurrent neural network and its variants, how they are trained and idea behind to map on the task of operation log anomaly detection.

A. Recurrent neural network (RNN)

Recurrent neural network (RNN) was introduced in initial time for time-series data modeling [20]. They are same as feed-forward network (FFN) with an additional cyclic loop, shown in Fig 1. This cyclic loop carries out information from one time-step to another. As a result, RNN are able to learn the temporal patterns, value at current time-step is estimated based on the past and present states. RNN has achieved a significant performance in long-standing AI tasks; machine translation, language modeling and speech recognition [5].

In general, given a sample of 1-minute time-interval data in the form of $x = (x_1, x_{2,}...., x_{T-1}, x_T)$ as input to RNN, it computes the hidden vector sequence $h = (h_1, h_2, ..., h_{T-1}, h_T)$ and output vector sequence $y = (y_1, y_2, ..., y_{T-1}, y_T)$ from $t = 1$ to T iteratively by using the below equations.

$$H(x, h) = F(w_{xh}x + w_{hh}h + b) \qquad (1)$$

[1]http://www.csmining.org/cdmc2016/index.php?id=5

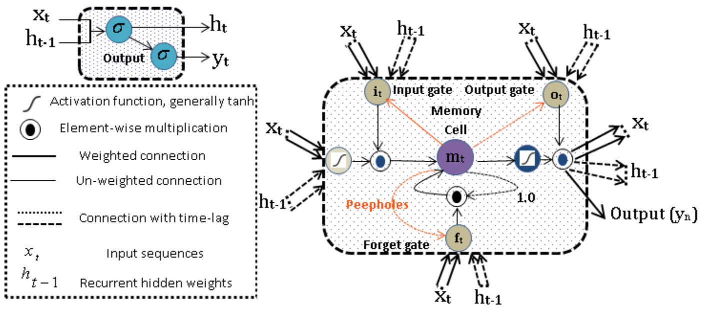

Fig. 1: Architecture of traditional RNN unit and LSTM memory block as adopted from [21]

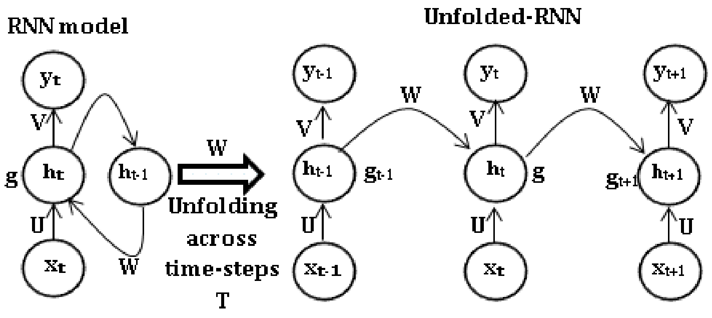

Fig. 2: Unfolding RNN as adopted from [21]

$H : R^m \times R^n \to R^n, w_{hx} \in R^{n\times m}, w_{hh} \in R^{n\times n}, b \in R^k$ where, F is a hidden layer activation function as $sigmoid$, b is a bias vector, w denotes the weight matrices (w_{xh} is input-to-hidden weight matrix, w_{hh} is hidden-to-hidden weight matrix and w_{hy} is hidden-to-output weight matrix) Here, RNN passes the hidden layer value h_0 to the next layer $h_1 = f(x_1, h_0)$ This can be generally defined as $h_T = F(x_T, h_{T-1})$. Further, h_T is passed to output layer. It uses non-linear activation function such as $sigmoid$ activation function to apply linear transformation.

$$y_t = \sigma(w_{yh}h_t + b_y) \tag{2}$$

A unit in RNN has cycles, removing them enables to understand the dynamics of each time-step. To achieve this, we use unfolding as represented in Fig 2. It represents the network structures typically looks same as FFN for whole sequence of given input. For eaxmple, if an input sequence of lentgh l, then the unfolding RNN generates l FFNs

In Fig 2 U, V and W is used to denote the weight parameters related to input-to-hidden, hidden-to-output and hidden-to-hidden layers, g is a batch of artificial neurons with values g_t at time step t. Let's define unfolded RNN as UF_t and a transition function for unfolded graph UF_t is defined as,

$$h_t = UF_t\,(x_t, x_{t-1}, \cdots x_1) \tag{3}$$

Total loss function for RNN is estimated by calculating the loss of all time-steps.

$$L = d(ep, pr) = \sum\nolimits_{i=1}^{T} d(ep, pr) \tag{4}$$

Loss function minimization is done by finding optimal parameters for U, V and W. To do this, we used stochastic gradient descent (SGD) mechanism. Applying SGD across time-steps through chain rule is typically called as back propagation through time (BPTT). BPTT has a vanishing and exploding gradient issue in backpropogating error gradient across many time-steps. This was discussed in a detailed way and introduced long short-term memory (LSTM) network [22]. They claimed that their method has the capability to store and update its values across time-steps.

B. Long-short term memory (LSTM)

Long short-term memory (LSTM) is an improved method of traditional RNN network that solves the vanishing and exploding gradient issue by forcing the constant error flow. LSTM introduced a memory block instead of a simple RNN unit, as shown in Fig 1. A memory block contains a memory cell with adaptive multiplicative gates; input, output, forget and a self- recurrent connection with a fixed weight 1.0. This value will be activated when in the absence of value from the outside signal. A memory cell acts as a container for storing values across time-steps. These values are past information that is used in the estimation of current time step value to capture the long-term temporal dependencies. These values are controlled by the multiplicative gates, specifically input gate allows or denies a value to a memory cell. Likewise, forget gate forgets or reset the past information that exist in self-connection, once a value becomes unusable and output gate controls the output flow of a memory cell. LSTM uses the same discussed training procedure such as BPTT. LSTM has performed well in learning long-range temporal dependencies in various artificial intelligence (AI) tasks [5]

In general, given a sample of 1-minute time-interval data in the form of $x = (x_1, x_{2,}...., x_{T-1}, x_T)$ as input to LSTM, It estimates output sequence by continuously updating the values of 3 multiplicative units (input (i) , output (y) and forget gate (f)) on a memory cell (m) in an iterate manner from $t = 1$ to T in the recurrent hidden layer of LSTM architecture. At each time-step T , the LSTM recurrent hidden layer function is generally formulated as follows:

$$x_t, h_{t-1}, c_{t-1} \to h_t, c_t$$

$$i_t = \sigma(w_{xi}x_t + w_{hi}h_{t-1} + w_{ci}m_{t-1} + b_i) \tag{5}$$

$$f_t = \sigma(w_{xf}x_t + w_{hf}h_{t-1} + w_{mf}m_{t-1} + b_f) \tag{6}$$

$$m_t = f_t \odot m_{t-1} + i_t \odot \tanh(w_{xm}x_t + w_{hm}h_{t-1} + b_m) \tag{7}$$

$$y_t = \sigma(w_{xy}x_t + w_{hy}h_{t-1} + w_{my}m_t + b_y) \tag{8}$$

$$h_t = y_t \odot \tanh(m_t) \tag{9}$$

Where i,f,y are the input, forget and output gate respectively, b_i, b_f, b_m, b_y denotes bias units of input gate, forget gate, memory cell and output gate respectively, w is the weight matrices, m is the state of a memory cell, h is the output of

TABLE I: DESCRIPTION OF CDMC2016 UNITECLOUD LOG FOR ANOMALY DETECTION

Total Samples	Total Features	Total Classes	Training	Testing
82,363	243	8	57,654	24,709

hidden layer and $sigm$ and $tanh$ are $sigmoid$ and $tanh$ non-linear activation function, values in the range [0,1] and [-1,1] respectively.

IV. Methodology, Result and Discussions

LSTM takes certain set of parameters with the given input set, so the performance of anomaly detection is directly rely on the optimal parameters. These parameters are selected through hyper parameter tuning approach. We trained an LSTM network in all experiments using the backpropogartion through time (BPTT) with ADAM update rule. The input and output activation function of a LSTM memory cell used hyperbolic tangent which in range [-1, 1]. Logistic sigmoid is adopted as non-linear activation function by gating units and other neurons in an LSTM memory cell.

A. Description of Data set

A resilient cloud infrastructure includes more than one server, which is formed by private firm UniteCloud in New Zealand Institute of Technology. It includes OpenNebula for cloud orchestration and KVM for virtualization. They collected log data from 243 sensors that are attached inside real-time running UniteCloud servers in the span of 60 seconds time-interval. After, these logs are preprocessed and labeled as either normal or anomalous event where anomalous event itself has 7 various anomalous classes. This data set is released by Cyber Security Data Mining Competition (CDMC 2016) for the task of operation log anomaly detection [23]. CDMC 2016 is an associated event of 9^{th} international workshop on data mining and cyber security (DMC2016) which is held in conjunction with 23^{rd} international conference on neural information processing (ICONIP2016). The main aim of the competition is to classify the log files as either normal or an anomaly and to correctly identify the abnormal events from various sensor log files. A data set consists of 82,363 samples in which train has 57,654 samples and test has 24,709 samples. Each sample is defined as a vector V that contains 243 features and 8 classes. A vector is defined mathematically as

$$V = (sf1, sf2, \cdots, sf243, cl) \tag{10}$$

Where $sf = sf1, sf2, \cdots sf243 \in R$ denotes a set of features of length 243 and cl denotes class label. The detailed description is placed in Table I.

B. Software frameworks

The choice of software frameworks for constructing deep learning algorithms is many. Most of the deep learning frameworks are available as open source. The comprehensive evaluation of deep learning frameworks is evaluated by [24] considering the extensibility, hardware utilization, and speed. We consider TensorFlow [25] and Theano [26] and to increase the speed of gradient computations of deep learning architectures, we use GPU enabled TensorFlow and Theano in conjunction with Keras[3] in single NVidia GK110BGL Tesla k40. We did not intent to compare the performance of TensorFlow and Theano computational frameworks as that is not our aim. But, in our experiments we found that TensorFlow performed relatively better in terms of speed. All deep learning architectures are trained using the back propagation through time (BPTT) technique.

C. Parameter tuning in LSTM network

The number of memory blocks, learning rate and number of hidden recurrent layers in LSTM network are primary hyper parameters that highly impact the anomaly detection rate. To find out optimal parameters of them, initially we used moderately sized LSTM network. This LSTM network has an input layer, hidden recurrent LSTM layer and an output layer. An input layer has 243 neurons, hidden recurrent layer contains 8, 16 and 32 memory blocks and output layer consist of 8 neurons. All the connections between the input to hidden layer and hidden to the output layer are fully connected. Experiments are run for each parameter associated with the number of memory block. In all these 3 trails of experiment, batch size is set to 32, ADAM as an optimizer, categorical_ crossentropy as loss function, and softmax as non-linear activation function in an output layer.

Intuitively, the train data of task named as UniteCloud Operation Log for Anomaly Detection in CDMC 2016 is randomly divided in to 77% for training and 23% for development. All the instances in both the train and development data are preprocessed and normalized in to the range [0-1] using scikit-learn[2]. To know the performance of each LSTM model, confusion-matrix is used. This has provided the class-specific metrics such as accuracy, recall, precision, and f1-measure. The training performance of LSTM model with 32 memory blocks is shown in Fig 5. Ideally, instead of increasing the memory blocks in a recurren LSTM layer, adopting recurren LSTM layer is most suitable. This maps the data to high dimensional space by passing through non-linear recurrent layers. The trained model is evaluated on the development data set and the performance across each epoch is displayed in Fig 4. This infers that the LSTM model has required more number of epochs specifically in the range [700-800] to effectively learn the patterns that distinguish the behaviors of log events as either normal or anomalous. Moreover, the model has completely learned the normal event patterns within the epoch ranged [0-100].

To effectively find out the learning rate parameter, the best performed model of the last experiment is used. The model is run for 800 epochs with learning rate parameter in the range [0.01-0.5] during training without using weight decay. As further, the trained model is evaluated on the validation data

[3]https://keras.io/

[2]http://scikit-learn.org/stable/

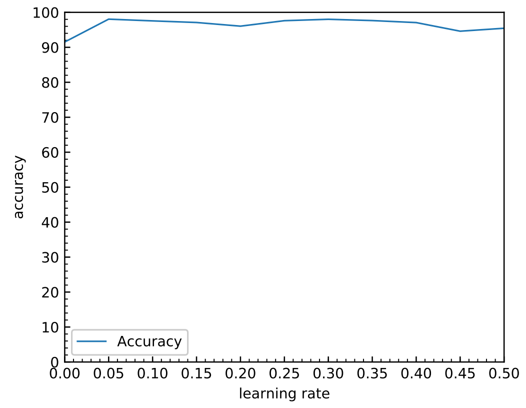

Fig. 3: Anomaly detection rate over learning rate in the range [0.01-0.5] using well-performed LSTM network, housing 32 memory blocks with one memory cell each.

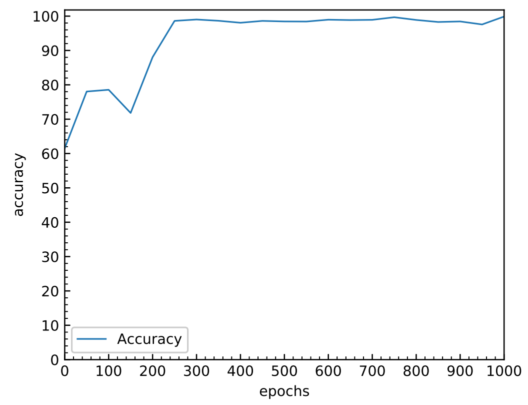

Fig. 4: Performance of LSTM network, housing 32 memory blocks with one memory cell each, across epochs in the range [0-1000].

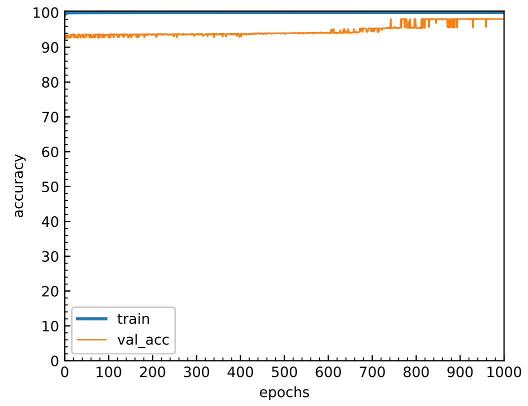

Fig. 5: LSTM 1 layer with 32 memory blocks train and validation accuracy.

TABLE II: 5-FOLD CROSS-VALIDATION RESULTS OF RNN AND LSTM NETWORKS

Algorithm	Accuracy	Precision	Recall	F-measure
RNN 1 layer	0.933	1.00	0.917	0.957
RNN 2 layer	0.949	1.00	0.937	0.968
RNN 3 layer	0.981	1.00	0.977	0.988
LSTM 1 layer	0.935	0.997	0.922	0.958
LSTM 2 layer	0.983	1.00	0.979	0.989
LSTM 3 layer	0.994	1.00	0.993	0.996

and the performance of them is noted across varied learning rate in the interval [0.01-0.5] is displayed in Fig 3. An LSTM network with the lower learning rate has reasonably learnt the patterns of anomalous event. Specifically, learning rate in the range [0.04-0.06] has showed better anomaly detection rate. After learning rate parameter value 0.06, the LSTM network has followed fluctuation in its anomalous event detection rate. By considering the training time and anomaly detection rate, the learning rate is set 0.05 for the rest of the experiments. However, running the same network with learning rate 0.05 till 1000 epochs may increase the anomalous detection rate. To show this, we run the same LSTM network till 1000 epochs. This has substantially increased the anomalous event detection rate. Thus, the rest of the experiments in this paper will be run till 1000 epochs.

D. Network topologies of RNN and LSTM network

In order to select the most suitable deep learning architecture, 3 trails of experiments are run for each of recurrent neural network (RNN) and long short-term memory (LSTM) network topologies with learning rate set 0.05 in 5-fold cross-validation settings. All experiments are run till 1000 epochs. The network topologies are

1) RNN 1 layer with 8 units
2) RNN 2 layer with 16 units
3) RNN 3 layer with 32 units
4) LSTM 1 layer with 8 units
5) LSTM 2 layer with 16 units
6) LSTM 3 layer with 32 units

All network topologies in LSTM network have used forget gates, constant error carousel (CEC), triggered when it doesn't receive any signal from outside and peephole connections. In order to effectively find out the required epochs parameter, 3 trails of experiments run for each network topologies. In the case of identifying the event as normal or anomalous, most of the RNN and LSTM networks have resulted considerable accuracy at epochs in the range [400-450]. All network topologies have attained considerable lowest error rate at epochs in the range [650-700]. Most notably, the complex network topologies have involved in large iterations to attain the considerable accuracy. Though, at last the complex networks have attained highest accuracy too. The number of optimal epochs required for learning the various anomalous events for each network topologies is of different. Moreover, network topologies have started to overfitting, once it has learned the specific anomalous patterns. It specifically means that the network has started to memorize the train data samples and results in decreasing the generalization performance for that specific anomalous event. Among all RNN and LSTM network topologies, 3 layer LSTM network with 32 memory blocks with one memory cell each, learning rate as 0.05 showed highest cross_validation accuracy. This LSTM network structure is used in our major experiment. More importantly, the LSTM network topologies have outperformed RNN network topologies in all 3 types. This infers that the LSTM networks are more suitable mechanism towards identifying the anomalous event in operation log smaples. This may be due to the fact that they house complex memory cell in their architecture. This enables LSTM to store and memorize the past information in evaluating the future events and update them when it is necessary. The detailed 5-fold cross validation results of all network structures are displayed in Table II.

E. Architecture of proposed LSTM network

The past experiments stated that the 3 layer LSTM networks with 32 memory blocks, each memory block has a memory cell, learning rate 0.05, is considerably optimal architecture for operation log anomaly detection, as displayed in Fig 7. The detailed architecture of Fig 7 including the inner units is displayed in Fig 6. This has an input layer with 243 neurons, recurrent hidden layer with 2 memory blocks (contains 32 memory blocks, only 2 is shown), each memory block has a single memory cell, adaptive multiplicative gates such as input, output and forget gate and a self-recurrent connection. Additionally, a memory cell has a peephole connection to learn the precise timing of the inputs. Moreover, to avoid the state of difficulty in understanding the proposed LSTM architecture, only few connections are displayed. Moreover each LSTM layer interleaved dropout layer to avoid overfitting. dropout layer interleaves dense layer and activation layer. An activation layer has softmax as non-linear activation function which gives the class probability values. The trained model is used to classify the real-world test data set, provided by CDMC 2016 and the predicted labels are submitted to them.

Table III demonstrates that the log anomaly detection encompasses long-term dependencies along with short-term dependencies. Thus, LSTM is able to do significant enhancement in F-score in comparison to the RNN approach. By considering this factor, we have submitted one run based on 3 layer LSTM network for anomaly detection to CDMC 2016 task organizer. The statistical measures reported by them are really intriguing. More importantly, the proposed system showed FPR of 0.02. The empirical evaluation of test results is displayed in Table 3. Based on this, we claim LSTM is a suitable mechanism for modeling operation log anomaly detection to classify the events as either normal or anomalous and categorize anomalous events to its 7 different categories.

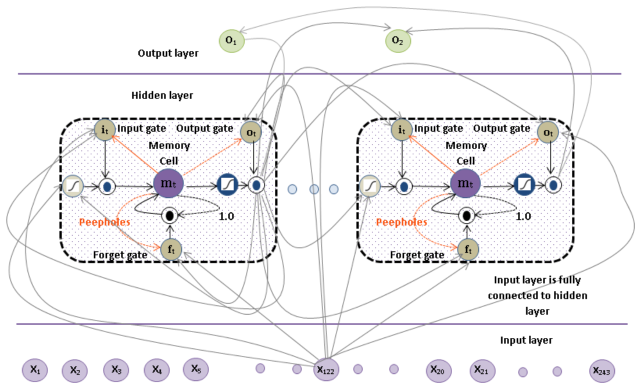

Fig. 6: A subnet of LSTM network architecture including two memory blocks, each memory block has a single memory cell. Only fewer connections are shown.

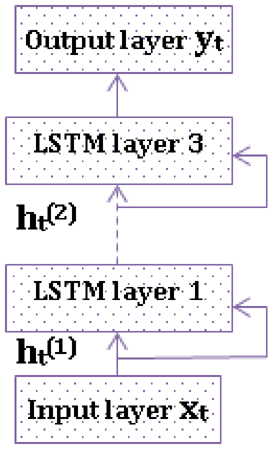

Fig. 7: Architecture of S-LSTM

V. Conclusion

This paper has reviewed the effectiveness of LSTM network to detect and classify the anomalous events accurately in sensor log files. The optimal results are obtained by evaluating the experiments for various network parameters and network structures. S-LSTM has itself showed its strength by achieving the highest accuracy in comparison to other used network structures with not only in the foreseen anomalous pattern additionally in unforeseen anomalous patterns too. This is due to the fact that the S-LSTM has capability to learn long-range temporal dependencies quickly with sparse representations in the absence of preliminary knowledge on time order information. Additionally, the standard recurrent neural network performance is much closer and comparable to the LSTM networks only in classifying the foreseen anomalous events. Based on our results, we suggest LSTM models are more robust to anomaly detection in comparison to the RNN prediction models. Additionally, we lack in explaining the internal dynamics of LSTM network, this remained as our future work. This can be showed by converting the state of network to linearized form and thereby estimate and analyze the structure of Eigen values and Eigen vectors from them

TABLE III: SUMMARY OF TEST RESULTS USING 3 LAYER STACKED LSTM NETWORK WITH 32 MEMORY BLOCKS

Overall accuracy	Class wise accuracy	Precision	Recall	Specificity	G-mean	F1-measure	Relative sensitivity	FPR	AUC
0.996	0.999	0.423	0.646	0.975	0.716	0.489	0.675	0.02	0.81

over time-steps [27].

This paper has followed a very simple LSTM network due to the fact that complex architecture ingests high computational cost with our current hardware and monolithic training environment. Thus, we lack behind in representing the experimental results with more complex architecture settings. The reported results can be further enhanced by using more complex LSTM network and training on advanced hardware in a distributed environment.

REFERENCES

[1] J. Valdman, "Log file analysis," *Tech. Rep. DCSE/TR-2001-04*, 2001.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[3] M. Basseville, I. V. Nikiforov *et al.*, *Detection of abrupt changes: theory and application*. Prentice Hall Englewood Cliffs, 1993, vol. 104.

[4] S. Hawkins, H. He, G. Williams, and R. Baxter, "Outlier detection using replicator neural networks," in *International Conference on Data Warehousing and Knowledge Discovery*. Springer, 2002, pp. 170–180.

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[6] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial intelligence review*, vol. 22, no. 2, pp. 85–126, 2004.

[7] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[8] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Data Engineering, 1995. Proceedings of the Eleventh International Conference on*. IEEE, 1995, pp. 3–14.

[9] F. Masseglia, F. Cathala, and P. Poncelet, "The psp approach for mining sequential patterns," *Principles of Data Mining and Knowledge Discovery*, pp. 176–184, 1998.

[10] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu, "Freespan: frequent pattern-projected sequential pattern mining," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2000, pp. 355–359.

[11] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," *Advances in Database TechnologyEDBT'96*, pp. 1–17, 1996.

[12] Y.-L. Chen and T.-K. Huang, "Discovering fuzzy time-interval sequential patterns in sequence databases," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 5, pp. 959–972, 2005.

[13] T. D. Wang, A. Deshpande, and B. Shneiderman, "A temporal pattern search algorithm for personal history event visualization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 799–812, 2012.

[14] K. Yamanishi and Y. Maruyama, "Dynamic syslog mining for network failure monitoring," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 499–508.

[15] N. Görnitz, M. Braun, and M. Kloft, "Hidden markov anomaly detection," in *International Conference on Machine Learning*, 2015, pp. 1833–1842.

[16] C. Liu, "Data analysis of minimally-structured heterogeneous logs: An experimental study of log template extraction and anomaly detection based on recurrent neural network and naive bayes." 2016.

[17] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *Proceedings*. Presses universitaires de Louvain, 2015, p. 89.

[18] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[19] T. Yang and V. Agrawal, "Log file anomaly detection," *CS224d Fall*, vol. 2016, 2016.

[20] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[21] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with lstm recurrent networks," *Journal of machine learning research*, vol. 3, no. Aug, pp. 115–143, 2002.

[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] S. Pang and R. Zhang, "2016 cdmc task 2: Unitecloud operation anomaly detection," 2016.

[24] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, "Comparative study of deep learning software frameworks," *arXiv preprint arXiv:1511.06435*, 2015.

[25] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, Georgia, USA*, 2016.

[26] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: A cpu and gpu math compiler in python," in *Proc. 9th Python in Science Conf*, 2010, pp. 1–7.

[27] R. Moazzezi, "Change-based population coding," Ph.D. dissertation, UCL (University College London), 2011.