Pipeline for Real-time Anomaly Detection in Log Data Streams using Apache Kafka and Apache Spark

Poojitha G
Department of CSE
R V College of Engineering
Bengaluru, India

Sowmyarani C. N, PhD
Associate Professor
Department of CSE
R V College of Engineering
Bengaluru, India

ABSTRACT
Anomaly detection is a standout amongst the most critical assignments so as to construct a system that is trustworthy and secure. The aim of anomaly detection is to detect significant deviation of the system behavior from that of the normal behavior. This approach is broadly used on static data, for instance on dumps of log data. Most systems require a real-time detection of anomalies with a specific end goal to lessen the harm that can be caused by the ignorance of an anomaly or detection at a later time. The recent implementations of the anomaly detection are mostly based on self-learning methods. Machine learning has brought about a significant transformation in the field of anomaly detection. One of the methodologies for anomaly detection depends on clustering algorithms. The implementation discussed in this paper utilizes a time-series evaluation approach for anomaly detection. The paper explains the pipeline built for anomaly detection and the visualization of the results.

General Terms
Pattern recognition, Artificial intelligence, Time series prediction, Big data, Apache Kafka, Apache Spark

Keywords
Anomaly detection, real-time, elastic stack, Long Short-Term Memory, Apache Spark

1. INTRODUCTION
Log data is a valuable asset for understanding the status of a server. Server logs are a great source of information for anomaly detection and online monitoring as they record every event as and when they occur from active processes. The verbose log data is a valuable information source for anomaly detection as it is available in human-readable text format. Thus, the required information can be easily extracted from the logs for further analysis. This procedure involves minimal processing of the logs making it straightforward and quick. The quick log processing, being the first phase of the pipeline, helps make the pipeline operate in real-time.

A data-driven approach for anomaly detection that leverages the server logs is proposed. A key idea behind the concept is: log entries are viewed as elements of a sequence that follow certain patterns and grammar rules. Indeed, a server log is produced by a program that follows a rigorous set of logic and controls and is very much like a natural language (though more structured and restricted in vocabulary).
The implemented pipeline is an end-to-end set-up for log anomaly detection constituting three modules namely, log collector, prediction module and the elastic stack. The prediction module is a deep neural network that models the sequence of log entries using a Long Short-Term Memory (LSTM) and allows system to automatically learn a model of log patterns from normal execution and flag deviations from normal execution as anomalies. Furthermore, since it is a learning-driven approach, it is possible to incrementally update the model so that it can adapt to new log patterns that emerge over time.

2. BACKGROUND AND RELATED WORK
Anomaly detection plays a significant role in the field of Web security, and log messages recording detailed system runtime information has become an important data analysis object accordingly. Log analysis has been the crucial solution to detect the web attacks. A technique is proposed in [1] that employs both pattern matching and supervised machine learning methods to perform multi-stage log analysis.

Data mining techniques are used in [2] to analyze network traffic, based on firewall audit logs. It determines if statistical analysis of the logs can be used to identify anomalies.

Predicting the performance of web server using LSTM RNN units has been proposed in [3]. The paper also showed the analysis of Nginx web server logs which contain user's URL access sequence and predicted the performance of the server by using RNN- LSTM.

Big Data can bring vital information and value to the organizations, if data is well processed in real-time. [4] describes suggests ELK stack (Elasticsearch, Logstash and Kibana) to handle Big Data.

[5] describes the employment Big Data frameworks, such as Hadoop and Spark, which can handle analysis jobs even for large amount of network traffic. To cope with streaming data, various stream-processing-based frameworks have been proposed, such as Storm, Flink, and Spark Streaming. The typical experiments showed that such systems perform well for large Internet traffic measurement and monitoring.

3. OBJECTIVES
The major objective of the project is to develop a data analytics pipeline to understand the behavior of servers/applications using the logs generated by these servers and to detect the anomalies in the system. The objective of the project can be divided into three parts.

- To have a log collector to validate and publish the incoming logs in real-time to the next modules.

- To build a Machine Learning pipeline that processes the logs in real-time and makes predictive analysis.
- To have the Elasticsearch-Logstash-Kibana stack that is fed with the predictions and observation from the machine learning module and visualized on Kibana dashboard.

4. THEORY AND FUNDAMENTALS OF LSTM CELL

Long Short-Term Memory networks – LSTM's are a special kind of RNN. These are capable of learning long-term dependencies. LSTMs are explicitly designed to avoid the long- term dependency problem. In this chapter, the architecture of LSTM Network and the functioning of LSTM gates is explained in detail along with the pictorial representation.

4.1. Architecture of LSTM Networks

The key technique behind LSTMs is the cell state; the cell state is maintained and persisted from end to end. The LSTM cell has the flexibility to get rid of or add data to the cell state. This task of manipulating the cell state is accomplished through the regulated structures referred to as gates. They are shaped out of sigmoid neural net layer and a pointwise multiplication operation. The sigmoid layer generates an output numbers between zero and one that indicates to what extent the part of knowledge needs to be preserved. An LSTM has 3 of such gates, to protect and control the cell state as shown by [12].

In the initial stage, LSTM makes the choice on what information must be maintained. This call is made by a sigmoid layer known as the forget gate layer. It looks at h_{t-1} and x_t and outputs - a number between "0" and "1" - for every number within the cell state C_{t-1}.

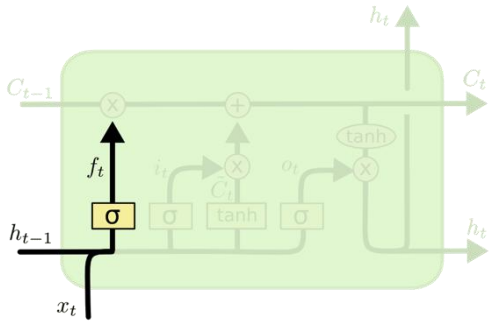

Figure 1: Forget gate layer in LSTM cell.

If the output is "1", it signifies to retain the knowledge utterly whereas output "0" signifies to get rid of the knowledge utterly. An intermediate value between "0" and "1" resembles the partial retention and rejection of information. The schematic representation of the forget gate layer is shown in the Figure. 1 by Chris Olah [12].

In the scenario of the current implementation, model trying to predict the number of errors at next time step based on all the previous ones within the window. In such a problem, the cell should retain the number of errors at previous time step, number of errors on previous day at corresponding time step and even number of errors on previous weekday at corresponding time step.

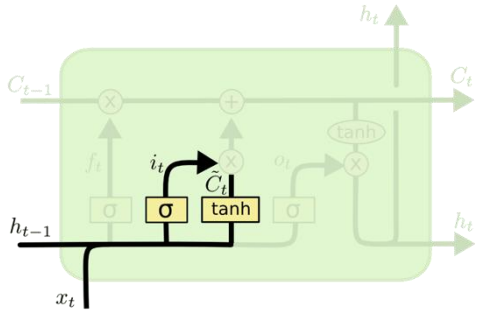

Figure 2: Input gate layer and tanh layer in LSTM cell

In the next phase, the LSTM decides on what new information has to be added to the cell state. This decision involves two parts. The graphical representation of the input gate layer and tanh layer in LSTM cell is shown in the Figure 2. First, an input gate layer i.e. sigmoid layer decides which values to be updated. Next, a tanh layer creates a vector of new candidate values, Ct that could be added to the state of cell. In the next phase, cell combines these to create an update to the state.

In the project's scenario, the model can drop the information regarding accepted requests while predicting the number of errors and model needs to add the new observed number of errors to the cell state. In next phase, the cell has to update the old cell state, C_{t-1} into the new cell state C_t. In this phase, the old information is dropped and add the new information, as decided in the previous phases.

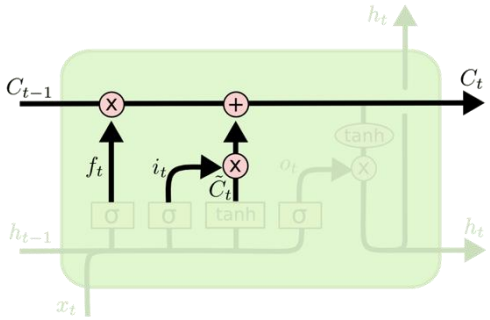

Figure 3: Update of old cell state to new cell state in LSTM cell

Figure 3 by Chris Olah [12] shows the procedure to update the old cell state to new cell state. In the final phase, the cell makes decision about the output. The resulted output depends on the cell state but will be filtered version of it. First, the sigmoid layer decides on what parts of the cell state to output. Then, the cell state is passed through a tanh to push the values between "-1" and "1" and then multiply it by the output of the sigmoid gate, so that it only outputs the parts decided to.

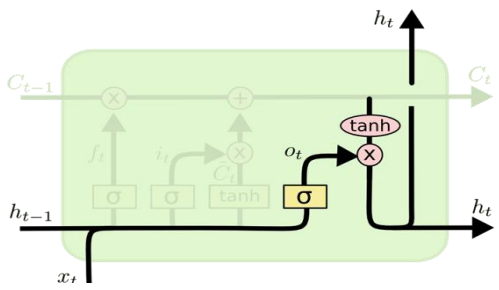

Figure 4: Output gate layer in LSTM cell

Finally, the decision about the output is made. This output will be based on new cell state but will be a filtered version. Figure 4 represents the output gate layer in LSTM cell. First, a sigmoid layer decides what parts of the cell state to output. Then, the cell state is passed through tanh layer to push the values to be between "-1" and "1" and multiplied to the output of the sigmoid gate, so that it only outputs the part that it is supposed to. In the case of this project, the model needs to output the number of error and information level logs for the next time step based on the current cell state, current input vector and current weight matrix.

The four fundamental gate layers of LSTM cell are unique in function. The presence of these gate layers in each cell allows the LSTM network to learn the long-term dependencies. Log data contains the time stamped information which makes it a time series data. This time series data has inherent long-term dependencies. Hence, in such cases LSTM network is a best choice to learn the pattern.

5. METHODOLOGY

The project methodology is divided into following parts:

5.1. Log collection

The very first task in the log data analytics pipeline is the collection of the logs for analysis. This is a continuous process and has to be performed in real-time. This is achieved using Filebeat and Logstash. The logs from the server are pushed into the pipeline by having a Filebeat instance running on the server configured to tail the logs into the Logstash. The Logstash at beginning of the pipeline accepts these logs. The logs collected by the Logstash are parsed, matched with the pre-defined format to check the validity. The logs are aggregated to obtain the number of logs with information and error log level in a window of one minute. These aggregations are passed on to the machine learning module for further analytics.

5.2. Predicting the behavior and computing anomaly score

The preprocessed data is analyzed by the machine learning module. An advanced deep neural network with LSTM architecture is used to learn the behavioral pattern of the logs. The model generated by training the neural network predicts the next value by taking the current value of the pre-processed log data. The network is trained with the new data at every pre-defined retrain interval so that it can adapt to new log patterns that emerge over time. An anomaly score is calculated at every minute which gives the measure of deviation of the observed value from that of the predicted. The observation along with the predicted values are then sent to the elastic stack. All these processes are run as a Spark process in order to handle the massive amount of data. The transfer of data between every sub-modules is performed over Kafka to achieve minimal lag in streaming of the data.

5.3.Visualization of results

The predictions made by the machine learning module along with current observed values and the anomaly score is indexed into Elasticsearch. The data stored in Elasticsearch is queried every five seconds and visualized as graphs on Kibana dashboard in real-time. The URL of the dashboard can be shared with the user of the pipeline for monitoring purpose. An alert message sent to the user when the anomaly score exceeds the defined threshold value. The alerting can either be a mail sent to user's email address or a message to the user's Slack account. Figure 5 shows the architecture of the pipeline.

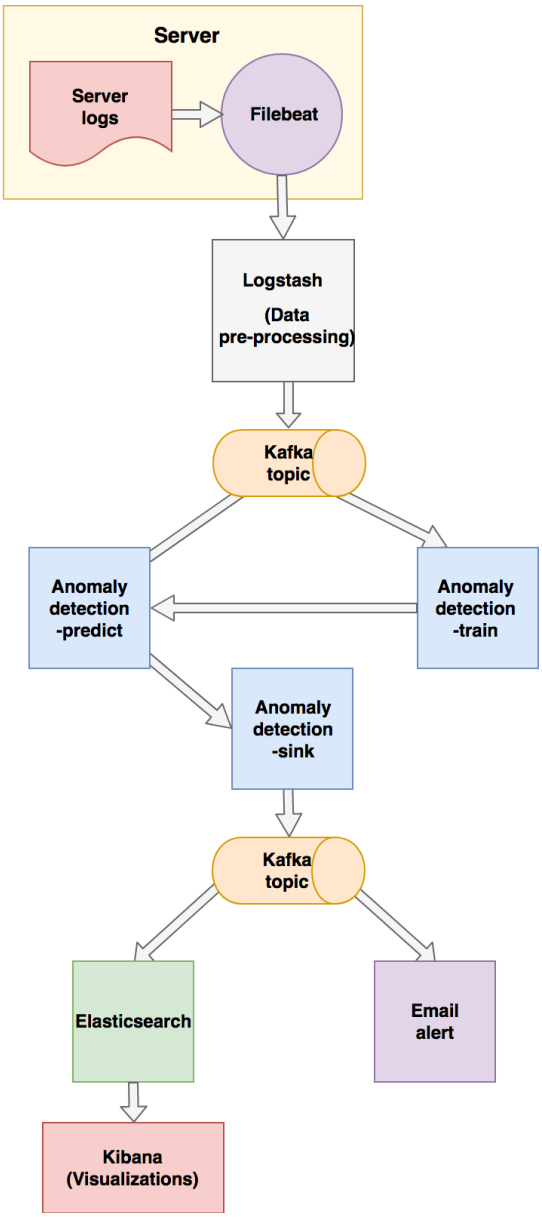

Figure 5: The architecture of the pipeline

6. EXPERIMENTAL DATASET

The dataset used for the testing of the pipeline is the log files of the one of the servers. The tomcat server has logs in the Catalina format which is parsed, and the required fields are extracted. The pipeline is run with dynamic data generated in the log files. The deep learning model is trained continuously at a specific interval. Around 2GB of data is generated in a log file over a day. The pipeline is run with a retrain interval of four days in order for the model to learn the pattern of the behavior of the system.

7. LIMITATIONS OF THE PROJECT

The system proposed and implemented in this project works efficiently for most of the test cases. However, the 'cold start' still poses a major problem. This results in no predictions until the first retrain interval is complete. The pipeline can handle only a limited number of log formats.

The project has implemented anomaly detection. Further implementation improvement would be to predict anomalies beforehand and act accordingly.

8. FUTURE WORK

There can always be a better way of doing something. Similarly, every project can be enhanced to provide more features and better results. A few of the possible enhancement that can be made in the project are discussed below.

8.1. Extend the formats of logs supported:

The system currently can analyze the logs of Catalina and Apache log formats. The future enhancement would be to add compatibility to different formats of logs and build parser for these logs. This can be done by adding the formats of the log to be supported to the Grok filter patterns directory. The prominent fields of the logs should be identified and extracted accordingly during the pre-processing of the logs.

8.2. Improve the algorithm and attempt new algorithms:

In the future, we could try some new algorithms or new features to improve the current algorithm to yield a better result than the one obtained now. The algorithm that is currently used is LSTM (Long Short-Term Memory). Experimentation can be performed with different configurations of the network layers in order to obtain better results.

8.3. Different modes to forward the logs:

A user has to run an instance of Filebeat on the server in order to forward the logs to the pipeline. Client can be authenticated with secret key to load the logs into to Amazon S3 storage with the help of interface that sits in-between client application and Amazon S3. Then the logs could be pulled by Amazon S3 input plugin for Logstash and streamed into the pipeline by a Logstash instance for further processing. Other log collectors such as Fluentd and Splunk can also be used.

9. RESULTS AND ANALYSIS

Logs are aggregated to obtain number of logs with information and error log levels over one-minute window, which forms the current timestamp observed values. The current timestamp observed value serves as an input to the prediction module. The prediction module with the help of trained model predicts the number of logs with information and error log levels for the next timestamp which is one minute ahead of current timestamp.

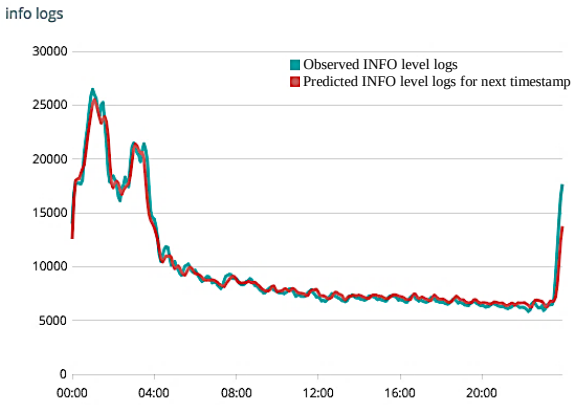

Figure 6. Observation and prediction graphs of information level logs (obtained over one day)

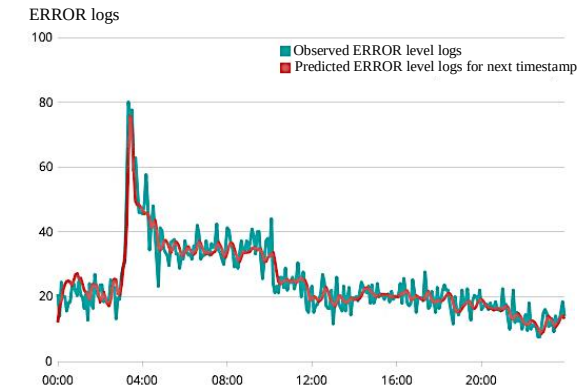

Figure 7. Observation and prediction graphs of error level logs (obtained over one day)

Figures 6 and 7 shows the performance of the model in predicting the number of information and error level logs respectively. The graph shows the data plotted over a day. The blue color curve corresponds to observed values and the red color curve corresponds to predicted values that are predicted for the following timestamp.

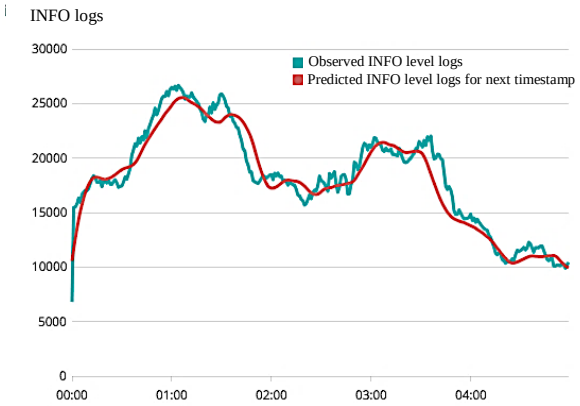

Figure 8. Observation and prediction graphs of information level logs (obtained over five hours)

The zoomed in view of the information level log analysis over a window frame on five hours is given in figure 8.

A common way to summarize how well a regression model fits the data is via the coefficient of determination or R^2. This can be calculated as the square of the correlation

between the observed y values and the predicted y^P values. If the predictions are close to the actual values, R^2 would expected to be close to 1 and if the predictions are unrelated to the actual values, then R^2=0. In all cases, R^2 lies between 0 and 1. The model's coefficient of determination or R^2 is found to be 0.84 on the number of accepted requests dataset. This signifies that the model closely fits to the dataset and the predictions are fairly close and accurate.

10. CONCLUSION

The project presents a framework for real-time log anomaly detection using a deep neural network-based approach. The model learns the pattern of the log message considering timestamp and log response level and performs anomaly detection at per log entry level, rather than at per session level as many previous methods are limited to. The anomaly-detection train module ensures that the LSTM model is trained after every preconfigured retrain interval hence is able to incorporate and adapt to new execution patterns. Future work includes incorporating other types of RNNs (recurrent neural networks) into the framework to test their efficiency and integrating log data from different applications and systems to perform more comprehensive system diagnosis.

11. REFERENCES

[1] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, Jürgen Schmidhuber, 'A Search Space Odyssey', IEEE Transactions on Neural Networks and Learning Systems, Volume: 28, Issue: 10, Oct. 2017

[2] Andrei Talaş, Florin Pop, Gabriel Neagu, 'Elastic stack in action for smart cities: Making sense of big data', 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 7-9 Sept. 2017, pp. 469-476.

[3] Melody Moh, Santhosh Pininti, Sindhusha Doddapaneni, 'Detecting Web Attacks Using Multi-Stage Log Analysis', Advanced Computing (IACC), 2016 IEEE 6th international Conference, 27-28 Feb 2016

[4] Tarun Prakash, Misha Kakkar, Kritika Patel, 'Geo-identification of web users through logs using ELK stack', in Cloud System and Big Data Engineering, Noida, India, 2016, pp. 606-610.

[5] Robert Winding, Timothy Wright, Michael Chapple, 'System Anomaly Detection: Mining Firewall Logs', Securecomm and Workshops Conference, 2006

[6] Liu Yunpeng, Hou Di, Bao Junpeng, 'Multi-step Ahead Time Series Forecasting for Different Data Patterns Based on LSTM Recurrent Neural Network', Web Information Systems and Applications Conference (WISA), 11-12 Nov. 2017

[7] Yuriy Kochura, Sergii Stirenko, Oleg Alienin, Michail Novotarskiy, Yuri Gordienko, 'Comparative analysis of open source frameworks for machine learning with use case in single-threaded and multi-threaded modes', 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT), Lviv, Ukraine 2017, pp. 373-376

[8] Zheng Zhao, Weihai Chei, Xingming Wu, Peter C Y Chen, Jingmeng Liu, 'LSTM network: a deep learning approach for short-term traffic forecast', IET Intelligent Transport Systems (Volume: 11, Issue: 2), March 2017, pp. 68-75

[9] Baojun Zhou, Jie Li, Xiaoyan Wang, Yu Gu, Li Xu, Yongqiang Hu, Lihua Zhu, 'Online Internet traffic monitoring system using spark streaming', Big Data Mining and Analytics, Volume: 1, Issue: 1, March 2018, pp. 47-56

[10] Qinkun Xiao, Yang Si, 'Time series prediction using graph model', 3rd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China, Dec 2017, pp. 1358-1361

[11] Karl Eric Harper, Jiang Zheng, Sam Ade Jacobs, Aldo Dagnino, Anton Jansen, Thomas Goldschmidt, Adamantios Marinakis, 'Industrial Analytics Pipelines', First IEEE International Conference on Big Data Computing Service and Applications (BigDataService), 30 March - 2 April 2015, Redwood City, CA, USA, pp. 242-248

[12] Chris Olah, 'Understanding LSTM Networks', http://colah.github.io/posts/2015-08-Understanding-LSTMs/, 27 August 2015.

[13] Min Du, Feifei Li, Guineng Zheng, Vivek Srikumar, 'DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning', Computer and Communications Security Conference, ACM, New York, USA, 2017, pp. 1-14

[14] Sung Jun Son , Youngmi Kwon, 'Performance of ELK stack and commercial system in security log analysis', IEEE 13th Malaysia International Conference on Communications (MICC), Johor Bahru, Malaysia, 28-30 Nov, 2017, pp. 187-190

[15] Tian Guo , Zhao Xu , Xin Yao , 'Robust Online Time Series Prediction with Recurrent Neural Networks', IEEE International Conference on Data Science and Advanced Analytics (DSAA), Montreal, QC, Canada, 17-19 Oct, 2016, pp. 816-825

[16] Jiajun Peng, Zheng Huang , Jie Cheng, 'A Deep Recurrent Network for Web Server Performance Prediction', IEEE Second International Conference on Data Science in Cyberspace (DSC), Shenzhen, China, 26-29 June, 2017, pp. 500-504

[17] Angel Garcia-Pedrero, Pilar Gomez-Gil, 'Time series forecasting using recurrent neural networks and wavelet reconstructed signals', 20th International Conference on Electronics, Communications and Computer (CONIELECOMP), Cholula, Mexico, 22-24 Feb, 2010, pp. 169-173

[18] Ramanna Hanamanthrao, S Thejaswini, 'Real-time clickstream data analytics and visualization', 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 2017, pp. 2139-2144.

[19] Kasun Amarasinghe, Milos Manic, Ryan Hruska, 'Optimal Stop Word Selection for Text Mining in Critical Infrastructure Domain', IEEE International Conference on Data Mining, Philadelphia, PA, USA, 18-20 Aug 2015, pp. 1-6

[20] Haitao Zhao, Shaoyuan Sun, Bo Jin, 'Sequential Fault Diagnosis based on LSTM Neural Network', IEEE Access, 30 Jan 2018, pp. 12929-12939

[21] Nicolo Navarin, Beatrice Vincenzi, Mirko Polato, Alessandro Sperduti, 'LSTM networks for data-aware remaining time prediction of business process instances', IEEE Symposium Series on Computational Intelligence, Honolulu, HI, USA, 27 Nov-1 Dec 2017, pp. 1-7

[22] Qimin Cao, Yinrong Qiao, Zhong Lyu, 'Machine learning to detect anomalies in web log analysis', 3rd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China, Dec 2017, pp. 519-523

[23] Yangdong Liu, Yizhe Wang, Xiaoguang Yang, Linan Zhang, 'Short-term travel time prediction by deep learning: A comparison of different LSTM-DNN models', IEEE 20th International Conference on Intelligent Transport Systems (ITSC), Yokohama, Japan, Oct 2017, pp. 1-8

[24] Serkan Kiranyaz, Adel Gastli, Lazhar Ben-Brahim, Nasser Alemadi, Moncef Gabbouj, 'Real-Time Fault Detection and Identification for MMC using 1D Convolutional Neural Networks', IEEE Transactions on Industrial Electronics, 2018, pp. 1-1

[25] Daniel Schachinger, Jürgen Pannosch, Wolfgang Kastner, 'Adaptive learning-based time series prediction framework for building energy management', IEEE International Conference on Industrial Electronics for Sustainable Energy Systems (IESES), Hamilton, New Zealand, New Zealand, Feb 2018, pp. 453-458

[26] Rishika Shree, Tanupriya Choudhury, Subhash Chand Gupta, Praveen Kumar, 'KAFKA: The modern platform for data management and analysis in big data domain', 2nd International Conference on Telecommunication and Networks (TEL-NET), Noida, India, Aug 2017, pp. 1-5

[27] Paul Le Noac'h, Alexandru Costan, Luc Bougé, 'A performance evaluation of Apache Kafka in support of big data streaming applications', IEEE International Conference on Big Data (Big Data), Boston, MA, USA, Dec 2017, pp. 4803-4806

[28] Ayae Ichinose, Atsuko Takefusa, Hidemoto Nakada, Masato Oguchi, 'A study of a video analysis framework using Kafka and spark streaming', IEEE International Conference on Big Data (Big Data), Boston, MA, USA, Dec 2017, pp. 2396-2401

[29] Subhash Kumar, 'Evolution of Spark framework for simplifying big data analytics', 3rd International Conference on Computing for Sustainable Global Development, New Delhi, India, March 2016, pp. 3597-3602

[30] Marcin Bajer, 'Building an IoT Data Hub with Elasticsearch, Logstash and Kibana', Future Internet of Things and Cloud Workshops, November 2017, pp. 63-68

[31] Aniruddha Parvat, Jai Chavan, Siddhesh Kadam, Souradeep Dev, Vidhi Pathak, 'A survey of deep-learning frameworks', International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 19-20 Jan 2017, pp. 1-7

[32] Anush Sankaran, Rahul Aralikatte, Senthil Mani, Shreya Khare, Naveen Panwar, Neelamadhav Gantayat, 'DARVIZ: Deep Abstract Representation, Visualization, and Verification of Deep Learning Models', 39th IEEE/ACM International Conference on Software Engineering: New Ideas and Emerging Technologies Results Track (ICSE-NIER), Buenos Aires, Argentina, 20-28 May 2017, pp. 47-50

[33] Karl Eric Harper, Jiang Zheng, Sam Ade Jacobs, Aldo Dagnino, Anton Jansen, Thomas Goldschmidt, Adamantios Marinakis, 'Industrial Analytics Pipelines', First IEEE International Conference on Big Data Computing Service and Applications (BigDataService), Redwood City, CA, USA, 30 March - 2 April 2015, pp. 242-248

[34] John A. Miller, Casey Bowman, Vishnu Gowda Harish, Shannon Quinn, 'Open Source Big Data Analytics Frameworks Written in Scala', IEEE International Congress on Big Data (BigData Congress), San Francisco, CA, USA, 27 June-2 July 2016, pp. 389-393

[35] M S Bhat, D G Nair, D Bansal, J Vaishnavi, 'Data structure-based performance evaluation of emerging technologies - A comparison of Scala, Ruby, Groovy, and Python', Sixth CSI International Conference on Software Engineering (CONSEG), Indore, India, 5-7 Sept. 2012, pp. 1-5