


Improved Data-Driven Root Cause Analysis in a Fog Computing Environment

Chetan M. Bulla, KLECET, Chikodi, India*

 <https://orcid.org/0000-0002-6463-0509>

Mahantesh N. Birje, Visvesvaraya Technological University, Belagavi, India

ABSTRACT

Internet of things (IoT) and cloud computing are used in many real-time smart applications such as smart healthcare, smart traffic, smart city, and smart industries. Fog computing has been introduced as an intermediate layer to reduce communication delay between cloud and IoT devices. To improve the performance of these smart applications, a predictive maintenance system needs to adopt an anomaly detection and root cause analysis model that helps to resolve anomalies and avoid such anomalies in the future. The state-of-the-art work on data-driven root cause analysis suffers from scalability, accuracy, and interpretability. In this paper, a multi-agent-based improved data-driven root cause analysis technique is introduced to identify anomalies and their root causes. The deep learning model LSTM autoencoder is used to find the anomalies, and a game theory approach called SHAP algorithm is used to find the root cause of the anomaly. The evaluation result shows the improvement in accuracy and interpretability as compared to state-of-the-art works.

KEYWORDS

Anomaly Detection, Autoencoder, Fog Computing, LSTM, Root Cause Analysis, SHAP

INTRODUCTION

Internet of Things (IoT) and cloud computing are used in many real-time smart applications such as smart health-care, smart traffic, smart industries and smart city. To reduce communication delay between cloud and IoT Devices, Cisco introduced fog computing (Yousefpour et al., 2019) as intermediate computing infrastructure. To improve performance of fog computing, a monitoring system is required that keep track of all the activities and behavior of fog infrastructure with associated IoT Devices (Birje & Bulla, 2019). Monitoring system is also used in predictive maintenance system to detect and predict faulty or deviating behavior of fog nodes and IoT devices (Birje & Manvi, 2011). The anomaly detection and root cause analysis models play a vital role in improving performance of smart applications such as smart industries and smart healthcare system.

The anomaly detection techniques find unknown pattern or outliers in unlabeled data when something unusual occurs or when condition deviates from normal behavior. Root cause Analysis (RCA) is a systematic process to understand reason for anomalies or faults that helps

DOI: 10.4018/IJIT.296238

*Corresponding Author

This article published as an Open Access Article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

operator to diagnose the problem and solve the issue within short period of time (Singh, 2020). The root cause analysis allows end users to accurately identify anomalies and its root cause to avoid failures that may occur in future. The following requirements have to be satisfied for effective root cause analysis (Steenwinckel et al., 2021) i) Accuracy: the RCA should give accurate cause of the problem and reduce false positives, ii) Minimal human effort: the RCA work automatically without human involvement, iii) Context aware: the RCA should provide context to increase the performance, iv) Adaptive: the RCA system should be capable of adapting detection behavior of changing conditions, v) Interpretable: The user of system must easily understand the failures and its cause to plan appropriate action. vi) Scalable: The RCA must work efficiently even with huge data.

There are three main techniques (Steenwinckel et al., 2021, Solé et al., 2017) to find root cause analysis in fog computing infrastructure: i) data-driven model: it identifies anomaly and its root cause based on the unusual pattern using machine learning or deep learning approach ii) Knowledge-driven techniques: it works on expert knowledge iii) Hybrid: it combines the both data-driven and knowledge-driven technique to meet the requirement of RCA. The above techniques suffer from few of the critical issues such as data-driven technique suffers from interpretability and accuracy, knowledge-driven technique is unable to find new types of faults and its causes and hybrid model consumes more computational resources.

In the new era of smart industries, Anomaly detection and root cause analysis play a vital role in improve the performance of machine and reduce maintainance cost. Anomaly detection identify abnormal behavior of production machine and outliers /quality deviation in the production line. The root cause analysis of anomaly may help to resolve tor fix the issue. The existing the anomaly detection and root cause analysis models does not meet all the requirement and are computationally expansive. Therefore, there is a need of an effective root cause analysis model which provides high interpretability and accuracy with minimum overhead. The existing works have focused on data-driven root cause analysis considering the above mentioned techniques, but failed to meet requirements such as accuracy, scalability and interpretability. Also, no work has been carried out highlighting the importance of views of multi-agent in predictive maintainance system. Hence, this paper proposes a multi-agent based data-driven root cause analysis model using SHAP algorithm. The main objectives of proposed root cause analysis model are: first, increase the accuracy and reduce the false positives in detecting anomalies in fog computing environment. second, 2) To develop a light weight root cause analysis that fulfill all requirments of root cause analysis with reduced the overhead.

Multi-Agents System (MAS) perform well in dynamic and complex infrastructures like grid, fog and cloud computing. So MAS is used to perform various operatios such as data collection, anomaly detection and root cuase anlaysis. The proposed system has two main folds: anomaly detection and root cause analysis. The anomaly detection method uses autoencoder and Long Short Term Memory model (LSTM) to detect and predict the anomalies. To improve the accuracy of anomaly detection, the autoencoder with one class support vector machine model (OC-SVM) is used for better classification. The LSTM model is used to predict the future anomalies based on historical data. Once the anomalies are identified, its information send to root cause analyzer and dashboard agent. The root cause analyzer uses SHAP Algorithm to identify root cause of the anomaly and display it on dashboard.

Contribution of the proposed work is as follows:

- Anomaly detection model is proposed using deep learning model called Autoencoder LSTM with one class support vector to increase accuracy and reduce the false positives.
- The root cause analysis model is introduced that uses a light weight game theory approach called SHAP (SHapley Additive exPlanation) to improve interpretability and accuracy with reduced complexity.

The remainder of paper is organized as follows: Section 2 gives overview of state of art root cause techniques. The proposed model and its methodology is discusses in Section 3. The experiment evaluation is presented in Section 4. Section 5 summarized proposed approach.

RELATED WORK

The anomaly detection and root cause analysis helps in predictive maintenance to improve performance and availability of fog computing services. Three types root cause analysis found in literature. Knowledge based RCA, Data-driven RCA and Hybrid RCA. In this section, comparison of these techniques are presented.

Data Driven RCA

The Data driven approach works on analysis of existing data and its patterns. Two types of data driven techniques are found literature, these are tree-based RCA and log-based RCA. The tree based RCA works in three phases first, construct tree for anomalous service/event and second construct tree for current running service and lastly compare anomalous tree with current running service. If both tree matches then the root cause of existing matched service is considered as root cause of currently identified anomaly. To find root cause of performance anomalies, a application specific system called MicroRCA is introduced (Wu et al., 2020). The proposed model has three phases in root cause analysis: attributed graph construction, anomaly sub-graph construction and fault service identification. The initially the attributed graph is constructed for all types of Microservice. Then sub graph for various performance anomalies are constructed. These subgraph are matched to present microservices to identify anomalies. If both graphs matched then cause of existing anomaly is considered as cause of presently identified anomaly. The model suffers from computational overhead especially in data collection and graph construction. Further, model does not support for fine grained monitoring interval.

A data driven Bayesian Belief Network (BBN) with fuzzy cognitive map based root-cause analysis model is proposed in (Wee et al., 2015). The main objective of proposed model is to increase the accuracy of root cause of anomaly. It also perform other operations such as casual rezoning, classification and feature selection. The proposed model predict the future outcome and diagnose the root cause of an event. The proposed model static in nature, consume more computational resources and suffers from scalability issue. A log based root cause analysis model is proposed to find root cause of anomaly by extracting features of anomalous event from Spark logs (Lu et al., 2019). The proposed model uses General Regression Neural Network (GRNN) with weight assignment approach to find abnormal tasks and find its root causes. The experimental results show that proposed model locate and identify root cause of the problem/anomalies accurately. GRNN is complex and computationally expensive. The model works well with Spark application and may not find faults and cause accurately in other applications.

Knowledge Driven Model

The knowledge driven model works in two phases: 1) Knowledge acquisition, and 2) Knowledge transformation. The knowledge acquisition phases capture domain knowledge from experts such as normal behaviors of system, different types of faults occurs and its root causes. Next in knowledge transformation phase, transfer these information into rules. These rules gives the description of fault detected and its causes. Two popular techniques for knowledge acquisition are Failure Mode and Effect Analysis (FMEA) (Stamatis, 2003) and Fault Tree Analysis (FTA) (Lee et al., 1985). FMEA captures possible failures with its root cause and FTA applies Boolean login to analysis the abnormal states of system. The FMEA and FTA consumes more time and it suffers from computational overhead and misinterpretation or complicit with different experts opinions. To resolve these issues, ontology based risk analysis methods have been designed. Ontology based solutions provide high level classes with its interactions (Steenwinckel et al. 2016). The experts are not familiar with ontology design,

additional tools are required to understand ontology. Further, it requires more computational resources for knowledge extraction and streaming real time data using ontology. To reduce the computational resources, semantic Complex Event Processing (SCEP) is introduced. The SCEP defines various rules to identify faults and unwanted behaviors. These rules are easy to interpret the ontology but it works well for small applications.

A fuzzy cognitive maps (FCM) and particle swarm optimization (PSO) (Yue et al., 2018) technique are used to extract the knowledge from system experts to identify root cause of anomalies. The FCM is used to extract the knowledge from experts and transform it into appropriate format. The existing FCM techniques use fixed incidence matrix and proposed model uses dynamic incidence matrix. The PSO is used to calculate incidence matrix and characterize various features. Three types of RCA model are used to predict future event, identify root cause of anomaly and presenting measure for abnormal event. To proposed model consumes more computation overhead to calculate dynamic incidence matrix. A knowledge graph and causal mining based root cause analysis approach is defined for Micro service applications in cloud (Qiu et al., 2020). The proposed model works in three phases, anomaly detection, root cause analysis and root cause diagnosis. The monitoring system continuously track the various Key performance indicators and identify anomalies based on pattern matching. The root cause analysis build knowledge graph from operational and maintenance experts for normal and abnormal behavior of various components in system. Next, causal graph is constructed using knowledge graph and casualty pattern. The current MSA are matched with casualty graph and identify root cause of anomalies. The root cause diagnosis list of causal path list and assign ranking. Based on these ranking and severity, the anomalies are resolved. The proposed model work high good accuracy but computationally expensive. The existing knowledge-driven approach gives good accuracy and interpretability, but not supports adaptively and scalability.

Hybrid Model

The hybrid root cause analysis combines data driven and Knowledge driven RCA to improve the performance and accuracy. A hybrid model called FLAG (Steenwinckel et al., 2021) is introduced to find anomalies and root cause analysis in sensor network to satisfy all the requirements of RCA that are discussed in previous section. Both data-driven and knowledge driven approaches are combined and executed parallelly to get benefits. The knowledge-driven approach extract the knowledge from experts and stored it the form of ontology. The data-driven approach called matrix profiling is used to find anomalies. To find root cause analysis, association rule mining technique are used. The complexity of proposed approach is higher because the adoption of two model requires more computation capabilities. To improve accuracy in detecting anomalies and identify its root cause (Abele et al., 2013), a Bayesian network based root-cause analysis model is developed. The proposed RCA model combines the experts' knowledge and machine learning approach in detecting anomaly and its root causes. The knowledge based system collects information from experts and put in appropriate format. These formatted data is used to model the RCA system. Further, machine learning approach is used to find anomalies and its causes based on historical data and its pattern. The proposed model reduces false positives but consume more computational capacity for knowledge modeling. A hybrid root cause analysis model called interpretable logic tree analysis (ILTA) is proposed to improve interpretability and reduce human involvement (Waghen & Ouali, 2021). The proposed RCA model uses knowledge discovery and fault tree analysis to extract useful information about root cause and construct a logic tree. A burn and build algorithm is used to remove redundant patterns from logic tree. The model accuracy is completely depends on data about system state. If the system state information is not available then its constructs partial interpretable logic tree. In such a situation, expert has to involve to provide necessary information to find root cause of anomaly. The hybrid RCA approach meet of the requirements

but suffers from high complexity in construction of partial logic tree. The main objectives of proposed root cause analysis model is to fulfill all the requirements with less complexity.

PROPOSED MODEL

The automation in predictive maintenance of smart industries becomes essential to improve the performance of the services, specifically in troubleshooting activities. The anomaly detection and root cause analysis are the initial phases of troubleshooting activities. The anomaly detection identifies abnormal behaviors and outliers in the smart machineries. The root cause analysis model finds the root cause of anomaly using SHAP algorithm. The anomaly detection and root cause analysis models are integrated in fog monitoring system. The fog monitoring system collects, filters, and analyses the data to find anomalies and root causes. Multiple cooperative agents are used to perform these tasks. In this section, the working of root cause analysis model is discussed. Figure 1 shows the architecture of root cause analysis in smart industrial applications.

There are three layers: cloud layer, fog layer and IoT layer. The IoT layer contains various smart machineries with limited amount of processing capabilities. In smart machines, various sensors are installed in different part of machines to keep track of the activities and act accordingly when something unusual happens. Most of anomalies in smart industry are due to outliers and sensor malfunction. So when unusual events happens, it is necessary to check which sensors data leads to critical condition. This information will help in troubleshooting the anomalies. The fog layer consist of fog nodes that are geographical distributed and these contain virtual machines with limited computational resources. The fog monitoring system is installed in these fog nodes. Multiple agents such as data collection agent, anomaly detection agent, root cause analyzer agent and dashboard agent are used to perform various activities of monitoring system. Data collection agent collect and preprocess the sensor data and put into aggregated form. The anomaly detection agent collects the data from data collection agent and identify the anomaly based on data pattern. Further, root cause analysis agent collects information about anomalies and apply SHAP algorithm to find root cause of anomaly. Finally, the dashboard agent display various performance metrics, anomalies detection and its root causes. Figure 2 shows the interaction between these agents.

The domain experts can see the information about anomalies and its causes in the dashboard and provide possible feedback regarding correctness of detected anomalies and its root causes. Based on domain expert feedback, the anomaly detection and root cause analysis agents updates its model to

Figure 1. Fog computing environment

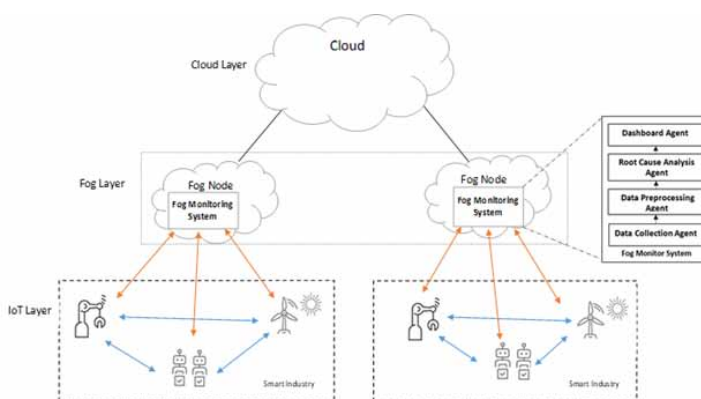
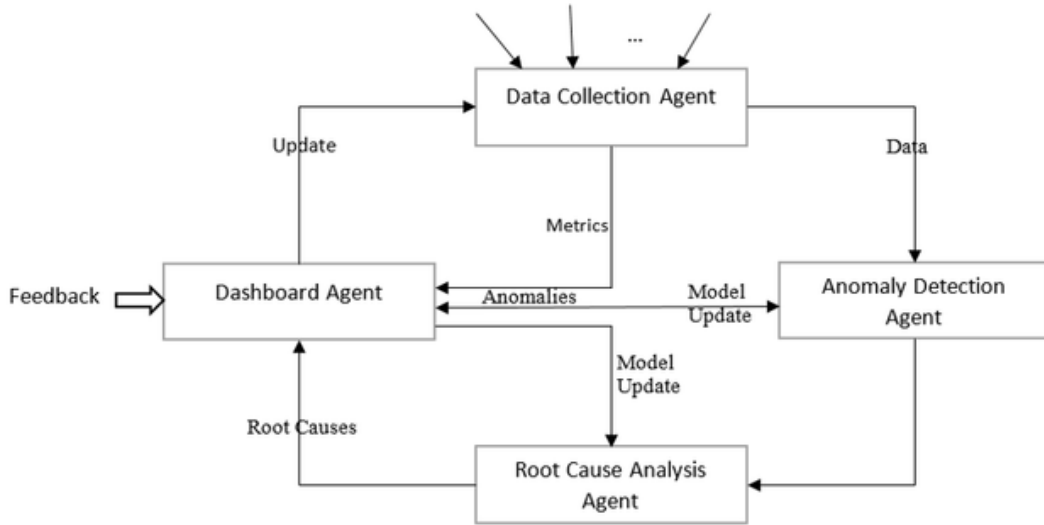


Figure 2. Multiple agents and its interaction to find root cause of anomaly



improve the accuracy. The following sub sections discuss about working of these agents. Figure 3 shows the process of detecting anomalies and its root cause analysis model.

Data Collection Agent

The collection agent collects all the sensors data and preprocess to improve the quality of data. The sensors in IoT Devices generates large amount of redundant data frequently that consumes more communication and computation capacity (Bulla & Birje, 2021). To reduce the overhead a dynamic interval based data collection model is introduced. The proposed data collection model collects the data based on User Threshold Degree (UTD) and Change Degree (CD). To avoid redundant data, the data collected based on Change Degree (CD) parameter. The Change Degree is defined as the difference between previous update value, and the new value read from sensors In IoT nodes. It is defined as:

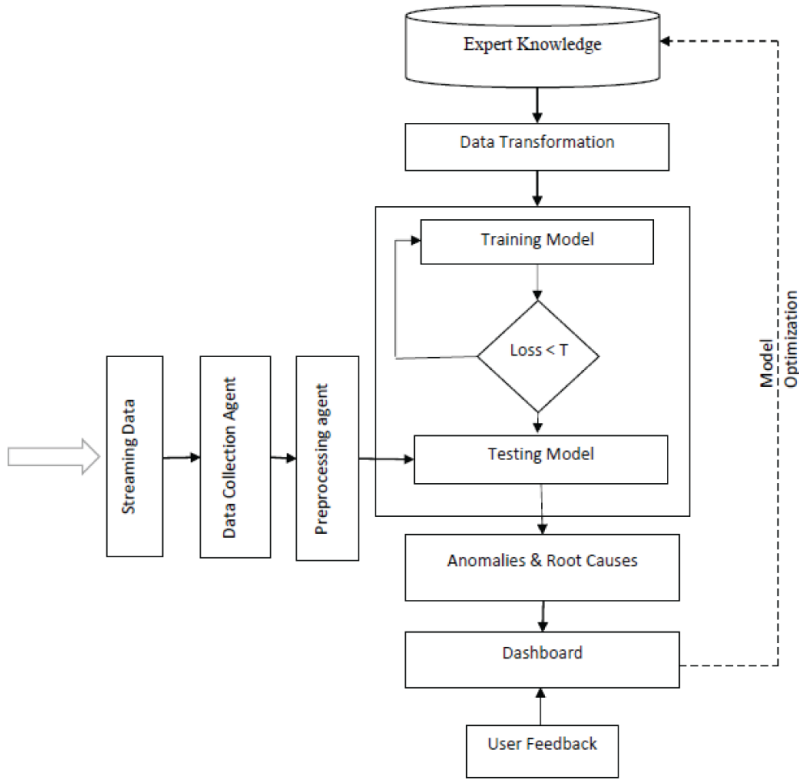
$$CD_t = \frac{DP(t) - DC(t)}{MAX} \times 100 \quad (1)$$

where DP (t) is new value and DC (t) old value at Fog node. The User Tolerant Degree (UTD) is Degree of User (application) Tolerance for particular application. The collection is smaller if UTD is large otherwise collection interval is larger side. A sliding window mechanism is used to capture recent values of sensors. Let $w_0, w_1, \dots, w_j, j \leq M$, is set of recent sensor values in windows of size N. The average amount of changes is used to calculate update rate and it is defined as:

$$\text{Avg_stat_window} = \frac{\sum_{i=1}^N (w_i)}{N} \quad (2)$$

where w_i is the status value in the window, and N is the number of status values in a window. The dynamic UTD is derived from the current UTD and the average amount of variation for each resource/

Figure 3. The root cause analysis model



sensor status. Dyn _ Update _ interval is calculated using DynUTD and critical values of resources or IoT nodes as given below:

$$DUI = CI \times \left[1 - \frac{|MAX_CriticalVal - Curr_Val|}{MAX} \right] UTD \left[1 - \frac{Avg_stat_window}{MAX} \right] + \Delta \quad (3)$$

if current MAXCrit < Avg_stat_window

$$DUI = CI \times \left[1 - \frac{|MIN_CriticalVal - Curr_Val|}{MAX} \right] UTD \left[1 - \frac{Avg_stat_window}{MAX} \right] + \Delta \quad (4)$$

if current MINCrit > Avg_stat_window

$$DUI = CI \times \left[1 - \frac{UTD \times \left[1 - \frac{Avg_stat_window}{MAX} \right] + \Delta}{100} \right] \quad (5)$$

if current MINCrit <= currentVal <= MAXCrit

where UTD is the current user threshold degree, Avg_stat_window is mean of window values, and MAX is the resource's maximum value. As the Avg_stat_window value increases, the Dyn_UTD value decreases. Δ is a constant used to prevent the value of Dyn_UTD from approaching zero. DUI stands for Dynamic Update Interval, and CI stands for Current Interval. The critical values ($Curr_val$) for each resource and IoT node are used to dynamically adjust the update period. If a specific resource or IoT node value is in a sensitive range, the data is periodically changed.

Data Preprocessing

The quality of data is very crucial in training the deep learning. The data preprocessing is the process of removing unwanted data and making it suitable for deep learning models. The proposed system utilizes the fog monitoring system to collect, filter, and normalize the data coming from IoT devices. In the proposed model, two data processing strategies: i) handling missing data ii) data normalization are present. To handle the missing data, the mean, median, or mode of a particular feature is calculated and added to missing values. The mean and median values of a particular feature is used to fill up the missing values in training data:

$$\bar{x} = \frac{\sum_{i=1}^N x}{N} \quad (6)$$

$$Median = \left(\frac{n+1}{2} \right)^{th} \text{ term (if n is even)} \quad (7)$$

$$Median = \frac{\left(\frac{n}{2} \right)^{th} \text{ term} + \left(\frac{n}{2} + 1 \right)^{th} \text{ term}}{2} \text{ (if n is odd)} \quad (8)$$

The multivariate data contain different types of data with various ranges. It is essential to scale these data to trained machine learning / deep learning model. The data is scaled using following equation:

$$x_{scale}^i = \frac{x^i - x_{min}^i}{x_{max}^i - x_{min}^i}, i = 1, \dots, k \quad (9)$$

where, $x_{max}^i - x_{min}^i$ are the maximum and minimum values of x_i in the data set, respectively. To reduce the storage complexity, the aggregation model is used based on requirements of application. The tree based data aggregation model, aggregate the data based on timestamp using standard aggregation functions. The aggregated sensor data is sent to cloud server through fog nodes.

Anomaly Detection Agent

In this section, the proposed data-driven root cause analysis model is introduced. The proposed model works in two phases: i) Anomaly detection ii) Root cause analysis. The anomaly detection model is designed using Autoencoder model with Long Term Short Memory. To classify the anomalous and

normal values, One Class Support Vector model is used. Once the anomalous data is found, it is fed to root cause analysis model to identify root cause for the anomaly. Figure 4 shows the flow of root cause analysis model.

Long Short Term Memory Model

The Long Short Term Memory (LSTM) (Tian et al., 2018) is a special type of Recurrent Neural Network (RNN) that allows the network to learn temporal interference like time series data by retaining long-term interdependencies between data. It is stronger than RNN, as it resolves the problem vanishing gradient, learns from inputs that are independent of each other and can store longer memories. LSTM networks are well suited for analyzing, classifying and forecasting time series data. The LSTM network is used to build autoencoder model to find the anomalies and perform multi-step prediction. Figure 5 shows the LSTM unit with its components.

The LSTM has three gates: input gate, forget gate, and output gate. These three gates regulate the information/data flow inside the LSTM cell. An input gate allows only necessary input values in the LSTM cell. Once the data comes inside the cell, the forget gate removes unimportant data. The output gate decides which values to give as the output of the LSTM cell. The input x_t is fed to the network at time t . The forget gate identifies the output of h_{t-1} is relevant and irrelevant information w.r.t current data x_t and forwards relevant information to cell state. The activation function is applied to $(h_{t-1} + x_t)$ to identify the previous value is relevant or not. If the output of activation function is closer to zero then it is considered as irrelevant otherwise it is considered as relevant. The forget Gate is determined as:

Figure 4. Root cause analysis framework

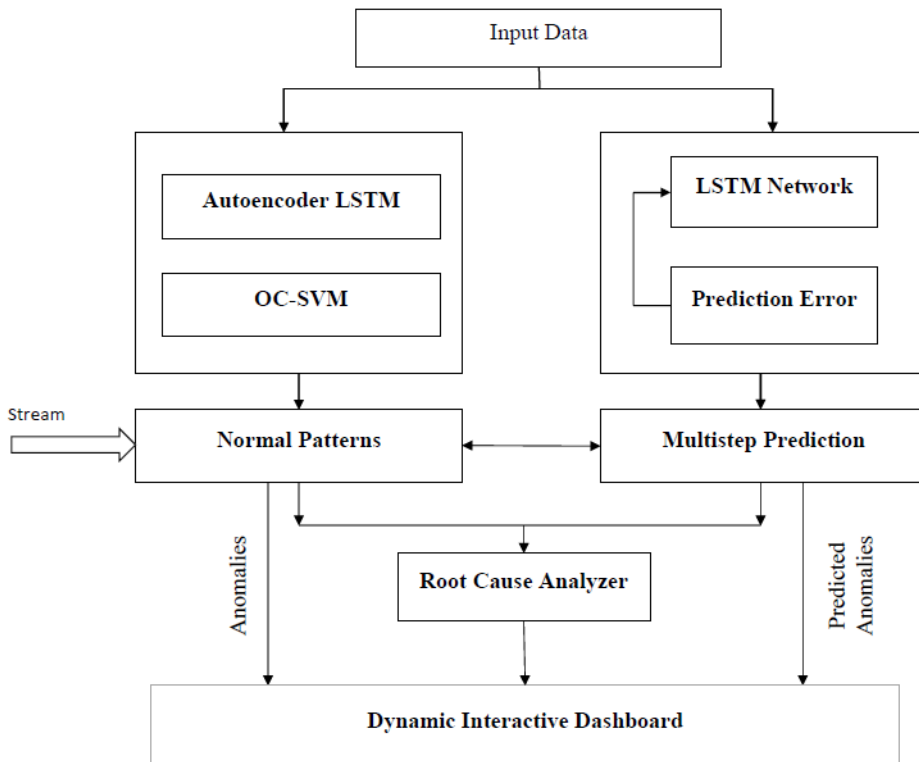
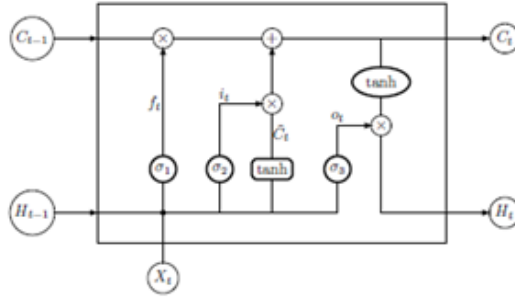


Figure 5. LSTM Cell Structure



$$f_t = \sigma(Wt_f \cdot [h_{t-1}, X_t] + bs_f) \quad (10)$$

where (Wt_f, bs_f) are the weights and the bias of the forget gate, respectively, σ is a sigmoid curve (activation function), and “.” implies multiplication of the matrix. The next step is to decide which new knowledge is contained in the cell state using two steps. First, the input gate checks which states are updated; then the \tanh activation function is applied to produce a vector of new values that could be applied:

$$i_t = \sigma(Wt_i \cdot [h_{t-1}, X_t] + bs_i) \quad (11)$$

$$\sim C_t = \tanh(Wt_c \cdot [h_{t-1}, X_t] + bs_c) \quad (12)$$

where (Wt_i, bs_i) and (Wt_c, bs_c) are the weights and biases of the input gate and the cell state layer, respectively. Outputs obtained from the forget gate, the input gate, and the \tanh function are then used to change the current cell state C_t :

$$C_t = f_t * C_{t-1} + i_t * \sim C_t \quad (13)$$

Finally, the output of network h_t is calculated by the output gate and a \tanh function, as:

$$O_t = \sigma(Wt_o \cdot [h_{t-1}, x_t] + bs_o) \quad (14)$$

$$h_t = O_t * \tanh(C_t) \quad (15)$$

where (Wt_o, bs_o) is the input weight and the bias of the output gate, respectively. Dropout regularization is used to remove some neurons to avoid over-fitting problems. It is intended to train each secret unit in a neural network with a randomly selected sample of other units.

Anomaly Detection Using Autoencoder and OCSVM

The autoencoder (Sakurada & Yairi, 2014) is unsupervised neural network model that works on the concept of encoding-decoding scheme. The encoder compress input data into latent-space,

whereas decoder decompresses latent-space to output data. The output of encoded-decoded data is compared with the initial data and the error is back-propagated to update the weights of the network. The difference between normal input and decoded output is called Reconstruction Error (RE). The autoencoder is trained with normal values and its reconstruction error is always smaller. When the anomalous value is fed to autoencoder, it may not reconstruct the value, so its RE is at higher side. If reconstruction error crosses predefined threshold, then it is considered as anomalous value. Figure 4 presents anomaly detection and root cause analysis model using autoencoder LSTM model. The left side of figure shows identification of anomaly detection using autoencoder model with OCSVM. The autoencoder model is trained by minimizing reconstruction error:

$$Loss = \frac{1}{2} \sum x - \bar{x} \quad (16)$$

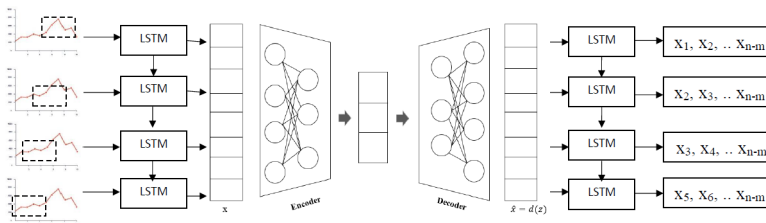
The main purpose of the autoencoder is to learn pattern of normal data and its silent features. There are two parts: i) encoder and ii) decoder. The encoder reduces the dimension of data keeping important information and store it in latent space in encoded form. The decoder reconstruct the data by decoding the data it and try to approximate the pattern.

The autoencoder uses LSTM model to draw temporal inferences. The model trained with normal pattern say $\{x_1, x_2, \dots, x_n\}$, where n is number of samples and $x_t = \{x_t^1, x_t^2, \dots, x_t^k\}$, $t = 1, 2, \dots$ is the multivariate time series value at time t with k number of variables. The sliding window mechanism is used to feed input values to the autoencoder model. The autoencoder model converts input data to encoded form and recreate it in the output form $\hat{x}_t = \{\hat{x}_t^1, \hat{x}_t^2, \dots, \hat{x}_t^k\}$ with $i=m+1 \dots N$. The autoencoder learn the model by approximation and reduce the reconstruction error. The new streaming data are fed to the autoencoder and compare the reconstruction error (RE). The RE is smaller for normal values and the larger for anomalous values. The anomaly is identified based on these estimation error vectors.

Figure 6 shows the sliding window mechanism used to find anomalies. Initially the input data (expert knowledge) is fed to autoencoder LSTM and LSTM Network. The autoencoder LSTM is used to find anomalies and LSTM Network is used for multi-step prediction. Any distance metric is used to classify normal and anomalous data such as Euclidean, Manhattan distance, Minkowski Distance and Mahalanobis distance (Gjorgiev & Gievska, 2020) can be used. These metrics fit well for sequential data patterns.

To improve the accuracy on non-linear and complex data patterns, one class Support vector machine model is used as classification problem to differentiate normal and anomalous values. OCSVM is machine learning model that classify certain values by defining a hyperplane between two classes of data (normal and anomalous).

Figure 6. Sliding window based anomaly detection model



The LSTM autoencoder produced the reconstruction error for each feature of training data. Given a reconstruction error vector without any class information $x_i \in \mathbb{R}^n, i = 1, 2, \dots, l$ where, i is the number of data points in the error vector, \mathbb{R}^n is a set of input values with n dimensions. $\Phi(x)$ is a map function used to transform x to the feature vector space F . A hyper-plane or linear decision function $f(x)$ in the feature space F is defined as:

$$f(x) = (w \cdot \Phi(x)) - \rho \quad (17)$$

where, w is the norm perpendicular to the hyperplane and ρ is the bias of the hyperplane. The following optimization problem is helpful to solve w and ρ :

$$\min \frac{1}{2} W^T w + \frac{1}{vl} \sum_{i=1}^l \xi_i - \rho \quad (18)$$

subject to:

$$W^T \Phi(x) \geq -\rho - \xi_i, \xi_i \geq 0, i = 1, 2, l$$

where ξ_i are slack variables are used to optimization of function $f(x)$, which is the parameter that controls a tradeoff between maximizing distance of the hyperplane from origin(normal values) and number of data points contained by hyperplane.

It is essential to tune the parameter v , if it small the error points fall on same side and difficult to class normal and anomalous values. To tune this parameter, Lagrangian multiplier α_i is defined for each X_i and the dual problem of the optimization problem can be obtained. Solving the dual problem leads to:

$$w = \sum_{i=1}^l \alpha_i \Phi(x_i) \quad (19)$$

where, $0 \leq \alpha_i \leq \frac{1}{vl}$ accordingly the decision function $f(x)$ becomes a nonlinear function as:

$$f(x) = \sum_{i=1}^L \alpha_i K(x_i, x) - \rho \quad (20)$$

$$K(x_i, x) = e^{\left(\frac{-x_1 - x_2^2}{2\sigma^2} \right)} \quad (21)$$

where K is a kernel function in the input space. The different types of functions are available for kernel function. In this paper Radial Basic Function (RBF), is used as it is best match for SVM. If the difference between normal and anomalous is high then $f(x)$ returns negative otherwise it returns

positive. For any x , if $f(x)$ returns negative then it is considered as anomaly. The anomaly score can be calculated using the following expression:

$$S_x = \sqrt{(\tilde{x} - x)^T \Sigma^{-1} (\tilde{x} - x)} \quad (22)$$

where Σ is the covariance matrix of the training input, x is the input values and \tilde{x} is the reconstructed value. Since the covariance matrix of the input features is unknown, we use a robust covariance estimator. Once the model is trained with normal values, the stream values are fed to the LSTM network to identify anomalies in the data. In this next section, the data collection model and root cause analysis model is discussed.

Multi-Variate Time Series Prediction

The multivariate time series prediction mechanism is developed to predict anomalies and its root cause based on historical data. Let $x_t = \{x_t^1, x_t^2, \dots, x_t^k\}$, where t denotes timestamp and k represents the number of variables. Here, x_t is a sensor value such as temperature, humidity and acceleration sensors etc. The LSTM network is trained based on a sequence of observed data and its interdependencies. At first, data is preprocessed using the equation (6) (7) (8) (9).

The Sliding window mechanism is used to capture recent values and prediction value. The window slides to the next level after predicting its first value. This process is continuous for n steps. The size of the sliding window is n , $n < M$, where M is the size of data. The $m \times k$ inputs are passed to the LSTM network (Vega García & Aznarte, 2020, Davis et al., 2019) to predict the next value, say $x(1)$. Initially, the window contains $\{x_1, x_2, \dots, x_m\}$ inputs are fed to the LSTM network and it predicts x_{m+1} , at the second step $\{x_2, x_3, \dots, x_{m+1}\}$ are fed to the network to get x_{m+2} as the prediction value. This process continues until the window reaches the end of data. The weights of the LSTM network are updated at each epoch to improve the accuracy (Gjorgiev & Gievska, 2020). The weights are updated until the number of epochs reached or the loss function reaches an optimal value. The loss function is defined as follows:

$$f_{loss} = \sum_{i=m+1}^N error_i \quad (23)$$

where error is the difference between actual values and predicted values. The performance of the LSTM network is evaluated using the error metric:

$$RMSE = \sqrt{\frac{1}{N - M - 1} \sum_{i=m+1}^N (\hat{x}_i^1 - x_i^1)^2} \quad (24)$$

Once the network is trained with higher accuracy, it can be used for predicting other multivariate data. The performance of the LSTM network can be improved by adopting various optimization techniques like batch normalization, unit dropout, and tuning learning parameters etc. The tuning sliding window size consumes more time and computation capacity. So it is essential to keep standard window size based on application requirements as preconfiguration settings. For example, the smart sugarcane industry has different types of sensors at different machinery. These sensors have their UTD value and need to set window size as sensor UTD value.

Root Cause Analysis Agent

The root cause analysis agent reads the anomalous values from anomaly detection agent to identify the root cause of anomalous values. The root cause for anomaly may be either single value or multiple values. It is difficult to identify anomaly when anomaly is generated due to the multiple values. For example, two features values temperature T and vibration sensor value M are the two variable leads to anomaly. So, it is important to inspect both features values and find the correlation between them.

A game theory approach called SHAP (Shapley Addition Explanation) algorithm (Vega García & Aznarte, 2020, Kaur et al., 2020) is used to find the correlation between different features. These correlation values are further used to find most contributing feature for learning model. The SHAP algorithm calculates the marginal contribution (shap value) of each features in a given instance. It is the average of the marginal contributions across all permutations of features in single instance. The highest shap value is considered as root cause of an anomaly.

Shapley Additive exPlication Approach

Let $x = [x_1, x_2, \dots, x_m]$ be a feature vector of analogous instance. An equivalent binary representation of x is $x' ([1, 0, \dots, 1])$ and h_x as a mapping function which maps $x = h_x(x')$ that convert decimal data to binary. Now let z be another observation that is nearer to x . The idea of an additive feature attribution method is to try to ensure that $g(z') \approx f(h_x(z'))$ as $z' \approx x'$. We can define the model $g(\cdot)$ as:

$$g(x') = w_0 + \sum_{i=1}^m w_i z'_i \quad (25)$$

where m is the number of features and $w_i \in R$. Hence, given the single anomalous value x , the SHAP value w_i is contribution of the i^{th} feature on the anomalous instant of $f(x)$. To approximate the anomalous instance the sum of all features attributions w_i are used. To compute SHAP values, the following equation is used:

$$\phi_i = \frac{1}{n} \sum_{S \subseteq N \setminus \{i\}} |S|! (|N| - |S| - 1)! [f(S \cup \{i\}) - f(S)] \quad (26)$$

where $f(s)$ is autoencoder model with set of feature S of an instance, N is number of features (ϕ_i) i^{th} feature shap value. Shap values are calculated for each features. The main idea behind shap algorithm is to find out feature importance of anomalous instance. It trains the autoencoder model with permutations of subset and total set of features to understand importance of each features. The shap algorithm produce a list contribution value (shap values) for each feature of an instance. If the attribute having highest variation in the instance, then its values is larger otherwise it smaller. The larger shap values indicate, maximum contribution in anomalous instance. To find expected outcome, the following function is used:

$$f_x(s) = f(h_x(z')) = E[f(x) | x_s] \quad (27)$$

Root Cause Analysis Using SHAP

Given input instance X with a set of features $\{x_1, x_2, \dots, x_n\}$ and its corresponding output X_0 and reconstructed values $\{x_0, x_1, \dots, x_n\}$ using autoencoder model f , the reconstruction of the instance is

the sum of errors of each feature $L(X, \bar{X}) = \sum_{i=1}^n (x_i - \bar{x}_i)^2$. Let x_1, x_2, \dots, x_n be ordering of the features in ErrorList, such as $|x_1 - \bar{x}_1| \geq \dots \geq |x_n - \bar{x}_n|$, the topMfeatures = x_1, x_2, \dots, x_m is a set of features for which the total corresponding errors topMerrors: $|x_1 - \bar{x}_1| \geq \dots \geq |x_m - \bar{x}_m|$ represents an adjustable percent of $L(X, \bar{X})$. Figure 7 shows root cause analysis model using SHAP. Initially, select the features with maximum deviations using following equation:

$$X_{MaxDev} = \{x_i - (x_{max} - x_{min})\} > Threshold \quad (28)$$

where x_{max} is maximum values of the feature, x_{min} is minimum value of feature. Threshold value can be set based on type of feature (sensor value). X_{maxdev} gives maximum deviation values in anomalous instances. The equation (26) is used to calculate the shap values of all the features of an instant. There are two major operations in calculating shap values: i) Kernel explainer and ii) shap calculation. The kernel explainer convert the data into binary form and shap calculates the shapvalues of selected features. During shap calculation, various combination of features are trained to find the importance of feature. The higher shap values are the root cause of anomaly.

The *shapkernel* convert the feature data into appropriate form to calculate shap value and the *explainer* calculates the shap values. The SHAP produces the shapvalues for all the features of an instant. The highest shapvalues indicates more contribution towards model result. The root cause analysis algorithm produces a root cause feature for each anomalous event identified. These root causes are displayed in dynamic interactive dashboard to give the feedback about identified anomalies and its root causes.

Dynamic Dashboard Agent

To visualize the health and corresponding machine' sensors value, a dynamic dashboard is created. The user can select certain component displayed on dashboard (Lempinen, 2012) to visualize the sensor

Figure 7. Root cause analysis for identified anomalies

Algorithm: Root cause analysis
Input: A- Anomalous instances X – Instance for which root cause need to find X _{ij} - Instances for kernel SHAP, ErrorList - An ordered list of error per feature, f - autoencoder model
Output: Root cause feature for anomaly FeatureSet S
Step 1: for each instance in anomalous instance Select Feature _{max} = Maximum values from FeatureErr
Step 2: Calculate SHAP Values for each feature for each $i \in \text{MAX}_{\text{features}}$ SHAP _{features} = shap.kernelExplainer(f,X _{1,j}) ShaptopMfeatures[i] = explainer. shapvalues(X, i)
Step 3: print Shap_Values
Step 4: Train the Autoencoder for identified Shap values X _i = f.estimate(X)[i]
Step 5: Identify most contributed feature for each i in shaptopMfeatures do if $x_i > \bar{x}_i$ then shapContribute[i] = shaptopMfeatures[i] < 0 else shapContribute[i] = shaptopMfeatures[i] > 0
return shapContribute
Print the shapContribute as root cause for the anomaly

values and give feedback for anomalies and its causes. Based on the experience, user can provide the feedback to improve the root cause analysis model. The feedback of user is used to optimize the model and update knowledge to improve the accuracy of anomaly detection and root cause analysis. Figure 8 shows the dynamic interactive dashboard architecture. It contains three layers: data layer, processing layer and presentation layer. The data layer collects the sensors data, anomalies and its root causes. The processing layer analyze the data that is gathered from data layer and aggregate data. The visualization layer display the sensors data, anomalies and its causes on the dashboard. The domain expert can visualize these data and provide the feedback on displayed data. The similar types of anomalies or faults can aggregated together to help RCA model to find similar events when same pattern can be found. The functions of dynamic dashboard are:

- To display concrete information without overlapping of information;
- To give the user feedback to improve the accuracy;
- Aggregate similar types of anomalies;
- Display predicted anomalies on dashboard to alert user to avoid future failures.

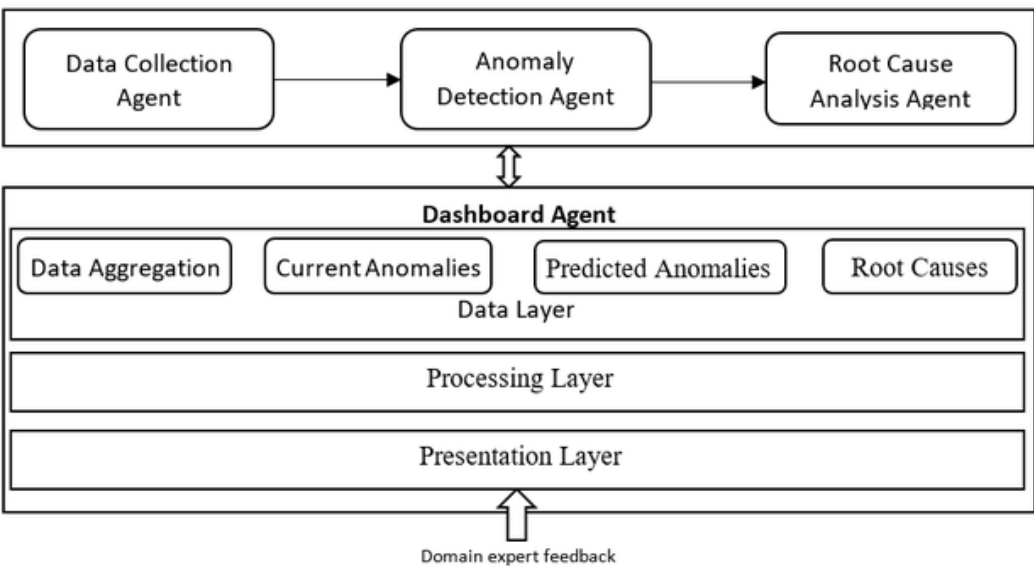
EXPERIMENT EVALUATION

This section discusses experiment setup and performance evaluation of the proposed anomaly detection and root cause analysis model. The proposed model is evaluated by comparing with state of the art anomaly detection and root cause analysis models.

Experiment Setup

The experiment was evaluated on Google Colab with Intel(R) Xeon(R) CPU (2.30GHz), 2 GB GPU, 16GB RAM, and 160 GB Hard disk space. The proposed model is implemented using Python programming language with TensorFlow Keras library.

Figure 8. Dynamic interactive Dashboard



Dataset

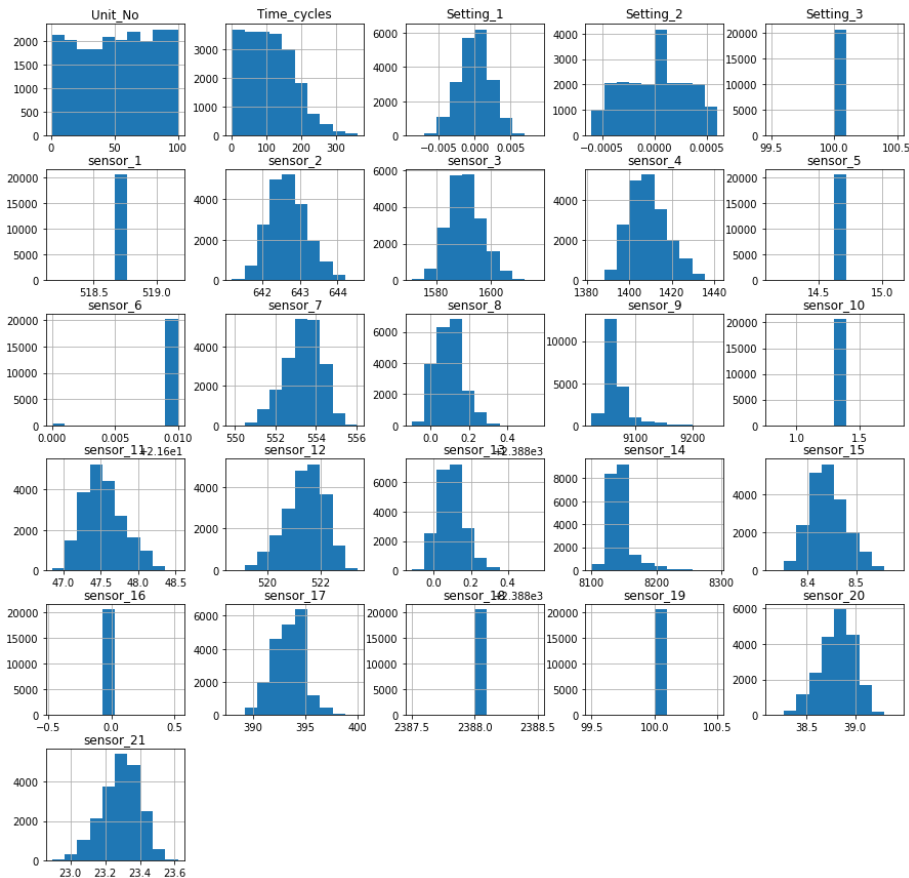
The NASA Turbofan Jet Engine Data Set (NASA Dataset,2019) is used for experiment evaluation. The dataset consists of multiple multivariate temporal data. Each dataset is divided into train and test dataset. The dataset contains engine numbers, three operational setting that have substantial effect on engine performance and 21 sensors values. The setting in the dataset are setting setting 1, setting 2 and setting 3. The sensor noise is added to all 21 sensors. The sensor type is not mentioned in the dataset. The figure 9 shows histogram values of dataset.

The sensors values in the dataset have different numerical ranges. In order to eliminate the influence of ranges, a standard python library ‘sklearn’ is used. The MinMaxScaler is used to scale down all the sensor values to a same range. The average, mean, and median techniques is used to fill the missing values in the dataset.

The Performance Metrics

To evaluate the proposed model, the performance metrics: precision, recall, and F1-score are used. These metrics represent the accuracy and efficiency of the anomaly detection process. To calculate these metrics it is essential to understand statistical measures of the performance of binary classification: sensitivity and specificity. The sensitivity measures the percentage of correctly identified positive categories and specificity measures the percentage of correctly identified negative categories. The following terms are used to identify these measures: True Positive (TP) and True Negative (TN)

Figure 9. Density graph for Jet Engine Sensors Dataset



indicates correctly predicted positive and negative samples. False Positive (FP) and False Negative (FN) represents incorrectly predicted positive and negative samples. The precision, recall and f1-score are calculated using sensitivity and specificity measures as follows:

1. Precision, P, is the percentage of correctly identified values. In other words, it is a measure of correctly identified anomalies:

$$precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (29)$$

2. Recall, R, which is the number of positive class prediction out of all positive examples:

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (30)$$

and (iii) F1-score, F1, which is the harmonic mean of Precision and Recall:

$$F1Score = \frac{2 * (Precision * Recall)}{Precision + Recall} \quad (31)$$

The F1-score represents the balance between precision and recall and it summarizes both. The detection rule with the highest F1-score indicates superior anomaly detection technique. The accuracy is another important key performance parameters to evaluate the anomaly detection model. The accuracy is defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (32)$$

The various loss functions are defined for machine learning models such as Root Mean Squared Error (RMSE), Mean Squared Logarithmic Error Loss (MSLEL), and Mean Absolute Error Loss (MAE), etc. The RMSE is the right choice to evaluate time series prediction model, and it is defined as:

$$Root\ Mean\ Squared\ Error = \sqrt{\frac{1}{n} \sum_{i=1}^n (y^i - y_p^i)^2} \quad (33)$$

The larger RMSE indicates less accuracy and least RMSE indicates highest accuracy. The various existing works are compared to the proposed model to evaluate the performance of the model. Machine learning models for multi-step time series prediction includes LSTM model [21], LSTM autoencoder model (Davis et al., 2019). Further, to compare the proposed model with the existing model, the parameters computational power, resource consumption, the delay, CPU, and energy consumption are used.

Results

This section consists of two subsections i) results of the proposed model, and ii) comparison of the proposed model with few existing models. The accuracy of multistep prediction, anomaly detection and Root cause analysis are discussed.

The proposed Data Collection Agent (DCA) model updates the sensor data based dynamic update interval to reduce computational and communicational capabilities. The more frequent updates consume more processing and storage capabilities. The update rate of a data collection model increases when the IoT device sensor data are in the critical range, and reduces in normal range. The proposed data collection model is compared with existing model. The existing models are MPP (Lempinen, 2012), Push and Pull model (PAP) (Jiang et al., 2015), Extended Push Pull Model (PAPX) [(Jiang et al., 2015). The proposed model regulates the data collection based on critical and normal condition. The update rate is lesser in normal condition as compared other model. Because the data updated based on change degree and user threshold degree. The figure 10 shows the comparison of update rate of various data collection model.

The anomaly detection model uses Autoencoder with OCSVM algorithm to identify anomalous data. The autoencoder try to reduce reconstruction error to fit the model accurately. The anomalous values are injected in the dataset and these dataset are used to test the model accuracy. The figure 11 shows the reconstruction error for different epochs. The proposed anomaly detection model is

Figure 10. Comparison of existing and proposed data collection models

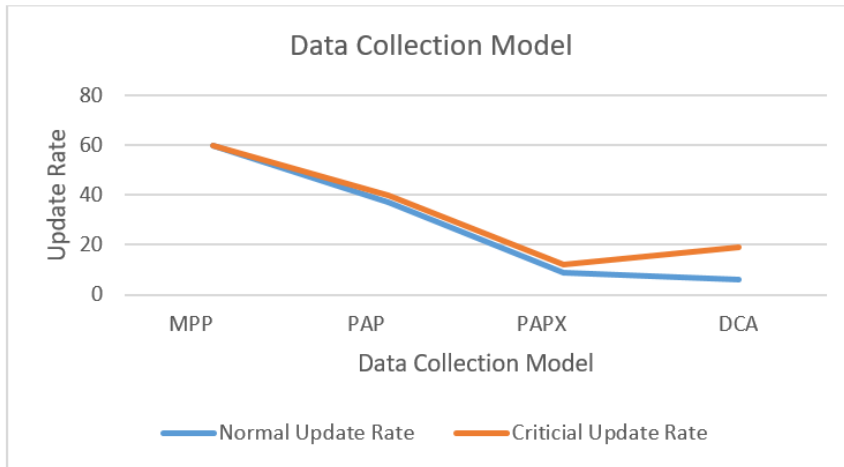
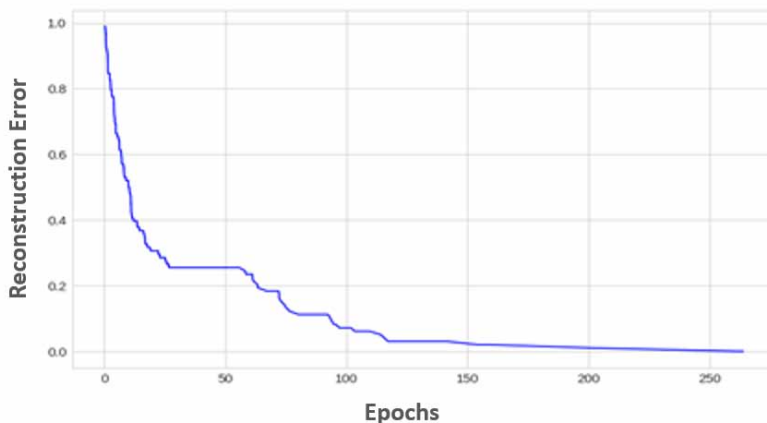


Figure 11. LSTM Autoencoder reconstruction error for different epochs



compared with existing models such as LSTM and autoencoder. The proposed model combines these two models to find anomalies. The proposed model has recorded reduced RMSE that indicates good accuracy compared to existing techniques. Figure 12 and Figure 13 shows comparison of various performance metrics such as RMSE (accuracy), recall, precision and F1-Score of anomaly detection. The existing works are based on LSTM (Zhang & Zou, 2018) and Autoencoder (Russo et al., 2020). The proposed model has highest accuracy in identify the anomalous data.

The anomaly detection model using deep learning techniques are divided into two catagories: reconstruction based methods and prodiction based methods. The proposed model uses reconstruction error based method to identify anomalies. To calculate computation time python library autotime is used. Two computation times are used for analysis, building time and inference time. The building time is the time taken to build deep leanring model using train dataset. The inference time is time taken to identify anomaly using test dataset. The reconstruction model consumes less time to detect anomalies compared prediction error based anomaly detection. The prediction error based model first predict the multistep ahead timesteps and compare current timestamp values. The difference between current and predicted values is greater than predefined throeshold value is considered as anomaly. The accuracy of anomaly detection is completely depends on the prediction accuracy. Where as reconstruction error based model is more accurate and consume less time. Table 1 illustrate the comparison of detection time for various anomaly detection model.

Building time for most of anomaly detection model consume more time as it requires multiple iteration. LSTM Model consume more processing time to perform back propogation on dataset. The

Figure 12. Comparison of RMSE of anomaly detection techniques

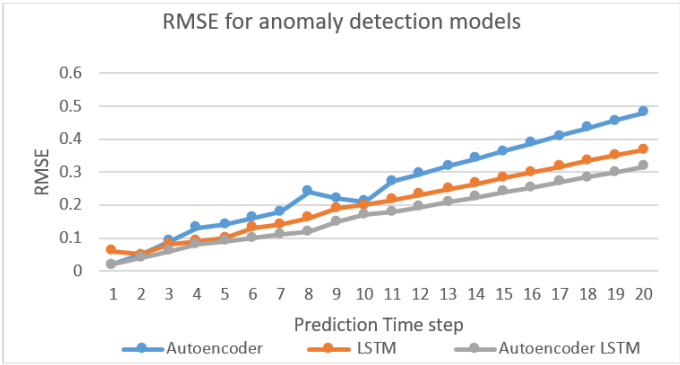


Figure 13. Precision, Recall, and F-Score comparison of the existing and proposed system

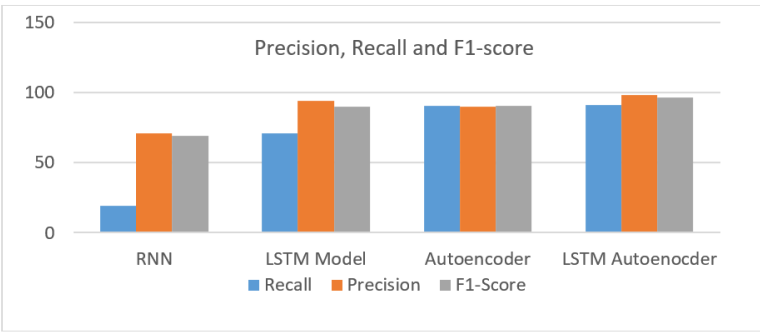


Table 1. Comparison of detection time of existing and proposed models

	LSTM Model (Davis et al., 2019)	Autoencoder (Gjorgiev, 2020)	dLSTM (Maya et al., 2019)	Proposed Model
Building Time	320 ms	170 ms	235 ms	167 ms
Inference Time	1.10ms	0.238	0.130 ms	0.203 ms

Autoencoder model takes lesser time to build model but the accuracy is compromised. dLSTM model consume less time in detecting anomaly. The proposed model consumes same time as autoencoder model but improves the accuracy. The top priority of proposed anomaly detection model is accuracy and processing time compromised for some times.

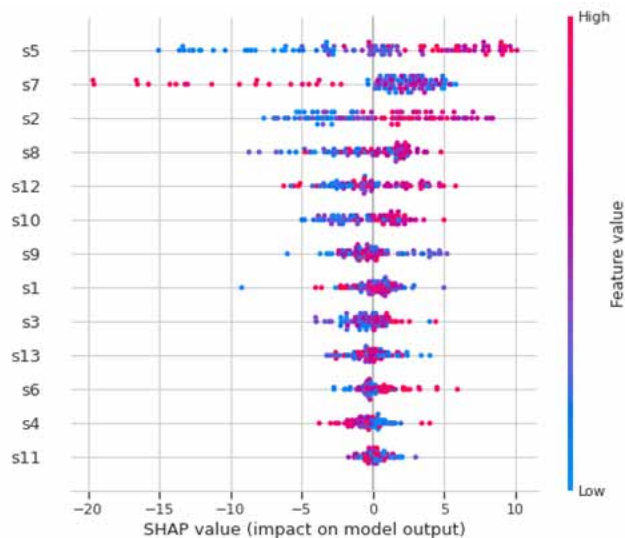
The shap values of all features are calculated for identified anomaly instant. To calculate the shap values, shap python library (SHAP library,2020) is used. Figure 14 shows the force plot where shap values are displayed for identified anomaly. The graph is generated using python shap library. The sensor 5 is high marginal contributing factor, so it is considered as root cause of anomaly. The operator can check the this sensor and its sensing environment to troubleshoot the anomaly.

The summary plot shown in figure 15 combines feature importance with feature effects. The x-axis represent shap values and y-axis represent features of anomalous instances. Two color are used to indicate contribution of feature, red color indicates high contribution and blue color represent

Figure 14. Shap values of all features of anomalous event



Figure 15. Summary Plot for SHAP Impact values



low contribution. The points (overlapping) in the graph represents the distribution of shap values per features. The features are ordered based on their importance. The sensor 5 (pressure sensor) has high marginal value and it is considered as root cause of anomaly.

Figure 16 shows mean of shap values of all the features that contributes for anomalous instance. The x axis represents the average values of shap and it describes average impact on model output magnitude. The Sensor 5 has highest shap values and it is considered as root cause of anomaly.

The anomaly and root cause analyze detection time is given in figure 17. The detection time is defined as time taken to detect anomaly and identify root cause analysis. For each dataset two anomalies are injected to understand the complexity of proposed model.

Figure 16. Summary plot for mean of shap values for anomalous instance

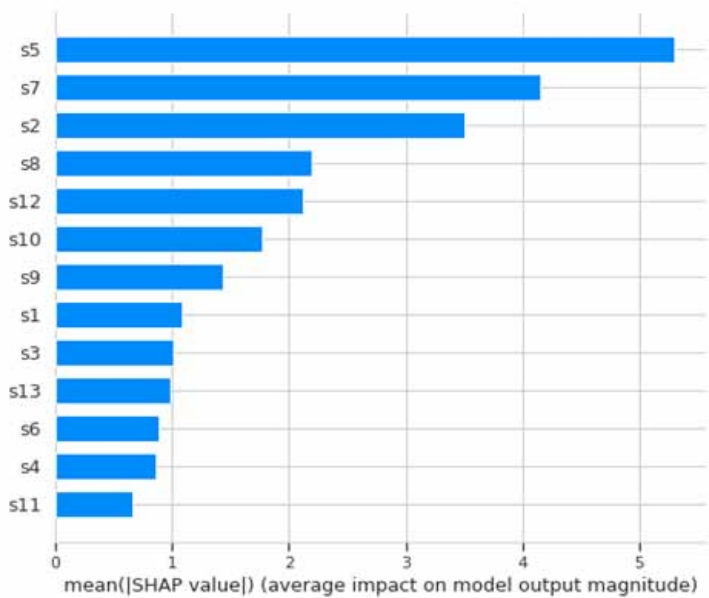
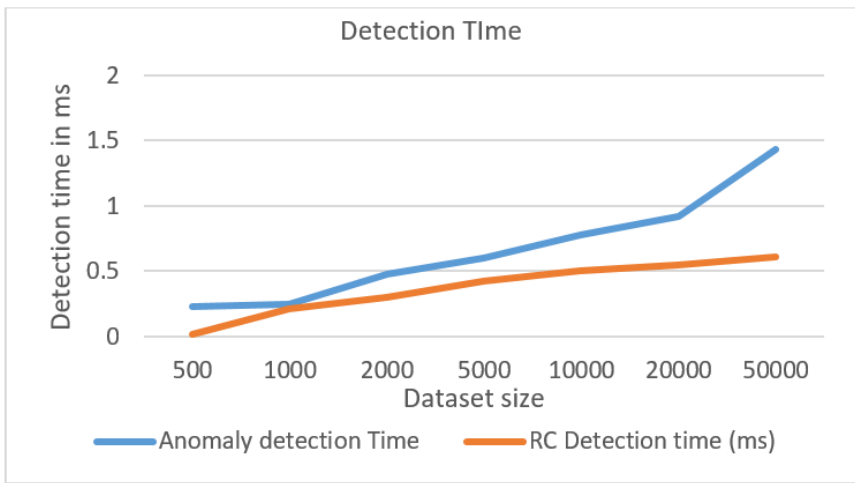


Figure 17. Time taken to detect anomaly and root cause



The existing tree based root cause analysis approach consume more time in casual tree construction and parsing the tree with streamed data. The shap algorithm uses less time to identify feature importance of an instance. So proposed model consume less time in anomaly and root cause detection.

The proposed model is evaluated in two different configuration of the fog nodes, considered as minimum and maximum capacities. The Google Colab comes with two system configuration i) Colab and ii) Colab Pro. The Colab has 12GB RAM, 107.12 GB Disk Space with CPU 2.3 GHz. The Colab pro has 24GB RAM, 215 GB Disk space with CPU 2.3 GHz. The Colab pro is used for complex and heavy applications. These two system configurations are used as minimum and maximum capacities of fog node. To analyze the computation time for both configurations, the proposed model executed in Colab and Colab pro systems. The comparison of computation time are shown in table 2.

Two computation times are used for analysis, building time and inference time. Fog node 1 takes more time in detecting anomaly and its root cause compared fog node 2.

The accuracy is one of important factor in assessing data driven/model driven techniques. The state of art techniques are FLAG (Steenwinckel et al., 2021), Graph-Based RCA (Brandón. et.al, 2020) and Fuzzy congitive map with particle swarm optimization algorithm (Wee. et.al, 2015) considered for comparison, The FLAG is hybrid model that achive 75% accuracy from data driven technique and improve the accuracy knowledge driven model is used. It requires more computational resources to execute hybrid model. The proposed model achieve 97% accuracy and it is highest as compared to state of art models. The graph based RCA and FCM achives 94% and 96% accuracy respectively. But these model consumes more time in finding root cause analysis. Figure 18 shows the comparison of accuracy of existing and proposed model.

The detection time of root cause is different for data-driven, knowledge driven and hybrid model. The data driven techniques uses deep learning approach to identify root cause of anomaly. The knowledge driven techniques has two important factor knowledge acquisition and representation. These techniques need generation of knowledge graph that represents the interaction among difference nodes of graph. Figure 19 shows comparosion of detection time of existing model and proposed model. The knowledge drivn model consume more time in knowledge contruction and it need human involvement. Hence it takes more to detect root cause analysis. The graph based model also takes more time in construction of error graph. FCM with PSO model is very complex and consume more computation power to identify root cause of anomaly. Hnec proposed effective interms of detection time compared to state of art models.

Discussion

The anomaly detection model identifies certain data points as anomaly based on the variation in sensor values. The autoencoder model is unsupervised algorithm and it is well suited for anomaly detection. To improve the model approximation and accuracy, LSTM Network is proposed. The reconstruction error is used to classify anomalous and normal values. The root cause analyzer takes anomalous

Table 2. Comparison of computation time in different configuration of fog nodes

Configuration	Computation Time		
	Time	AD Time	RCA Time
Fog Node 1 CPU:2.3 GHz, RAM: 12GB, Disk Space: 107.12 GB	Building Time	167 ms	2 ms
	Inference Time	0.203 ms	0.045
Fog Node 2 CPU: 2.3 GHz, RAM:24 GB, Disk Space: 215 GB	Building Time	123 ms	1.32 ms
	Inference Time	0.102 ms	0.028 ms

Figure 18. Comparison of accuracy of existing and proposed model

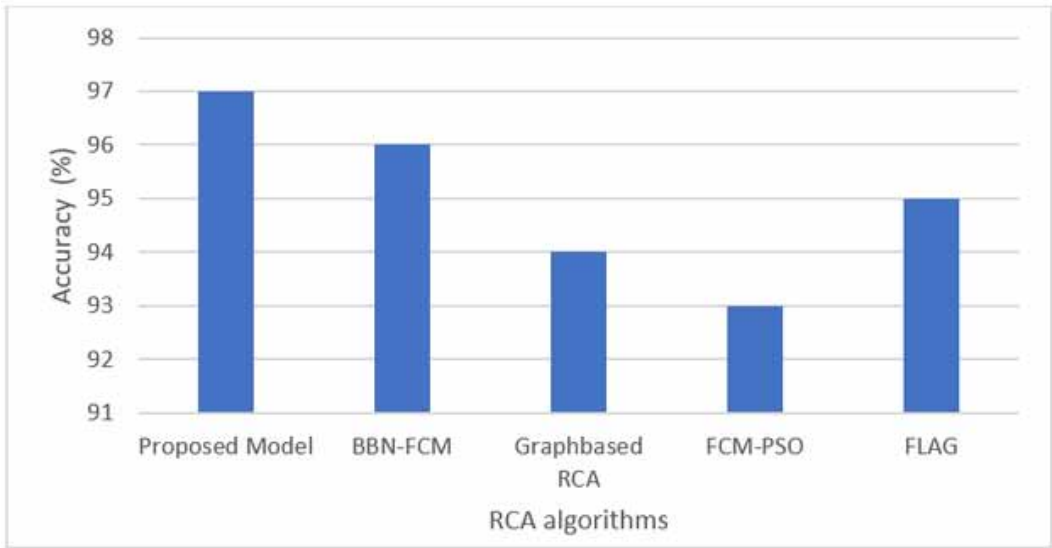
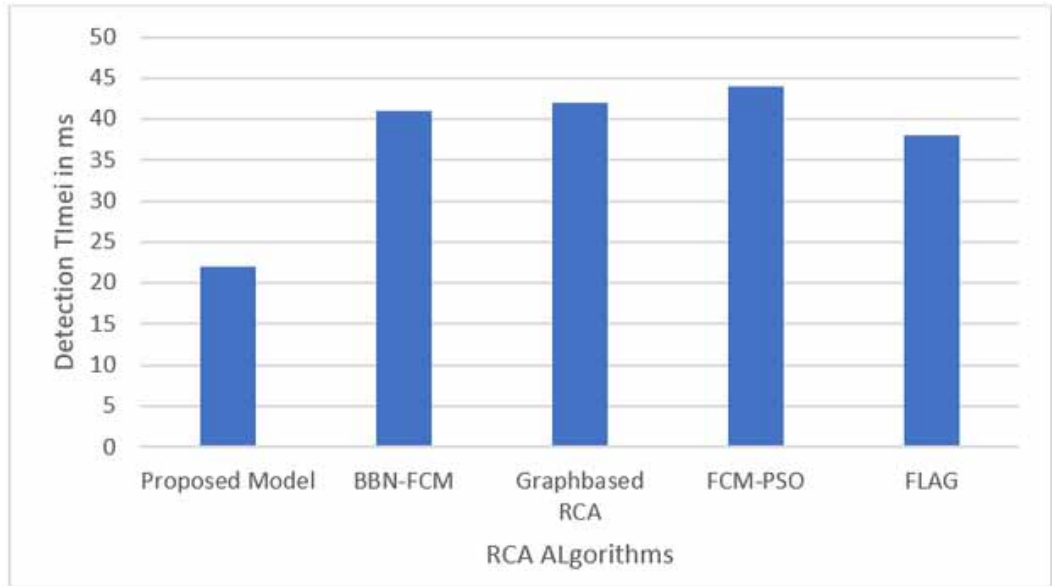


Figure 19. Comparison of detection time of existing model and proposed model



instances and identify the feature of an instance that is contributing more for anomalous values. The SHAP is a lightweight algorithm used to find marginal contributing feature in an anomalous instance. The proposed model meets all the requirements of RCA that are mentioned in above sections. The comparison of proposed approach and state of art approach are given in table 3.

The main objectives of proposed model is to reduce the complexity and improves the accuracy of RCA model. Additional objective is to satisfy all the requirement of RCA model. The proposed model improves the anomaly detection and RCA accuracy compared to the existing state of art models. The

Table 3. Comparison of proposed model and state of art RCA works

RCA Requirement	Data-driven	Knowledge-driven	Hybrid	Proposed Model
Accuracy	No	Yes	Yes	Yes
Human involvement	No	Yes	Partially yes	No*
Scalability	No	No	No	Yes
Context Aware	Yes	Yes	Yes	Yes
Adaptive	Yes	No	Yes	Yes
Interoperability	No	Yes	Yes	Yes

*only for feedback

precision, recall and F1-score of indicates higher accuracy. 3% false positives are recorded in our experiment results that is lesser compared to existing approaches. The SHAP algorithm algorithm identify most contributing features that leads to anomaly. The RCA model improves its accuracy based on domain expert feedback.

The existing knowledge based methods and hybrid methods requires 100% human involvement to provide domain knowledge. The data driven RCA model may not give 100% accuracy in dynamic and complex system such as cloud computing and fog computing application. The human involvement is second requirement of RCA and it is essential to improve the performance of the system. The proposed model uses expert feedback to improve the accuracy of RCA Model. The domain expert/user check the dashboard and confirm the root cause results and give feedback about correctness.

The proposed model also meet the scalability requirement, as it uses the light-weight data collection and root cause analysis approaches. The proposed model works efficiently when size of data increases. The data collection model collects the data based on change degree and dynamic UTD of data which avoids redundant data collection. The RCA model uses a light weight SHAP algorithm to find most contributing factors in the list of features and it consume less computational resources (CPU and memory) compared to graph based RCA (Brandón et al., 2020) and Knowledge driven RCA (Liu et al., 2021). From this perspective the proposed model satisfy scalability requirement. The context aware expert knowledge is used to train the deep learning model and it can be modified based on user feedback to improve the performance of anomaly and root cause detection.

As requirements of customer changes, the proposed model adopt new requirements. The proposed RCA uses expert feedback to confirm correct root cause of anomaly. Based on the expert feedback. The model itself tune for new changes. So it also fulfill context aware and and adaptability requirements. The operators in smart industry may not familiar with different types of anomalies and its cause. It is essential to give the root causes in simple and interpretable manner. Interpretability is most important requirement of RCA as it helps in understanding cause of errors and anomalies. The SHAP algorithm helps to understand and interpret it results very easily. So it fulfil the interpretability requirement of RCA.

CONCLUSION

The anomaly detection and root cause analysis models are helpful for predictive maintenance system to improve the performance and availability. The effective anomaly detection and root cause analysis model must be accurate, adaptive, scalable, interpretable, and should reduce human involvement. The existing data-driven root cause analysis are computationally expansive and failed to meet the requirements. In this paper, multi-agent based root cause analysis model is proposed that covers all the requirements. The multiple cooperative agents are integrated in fog monitoring

system to perform activities such as data collection, anomaly detection, root cause identification and managing dashboard. The proposed model uses LSTM autoencoder model with OCSVM model to find the anomalous instances. The root cause analysis model uses SHAP algorithm to find the root cause of an anomalous instance. The SHAP algorithm calculates marginal contribution of each feature and identify high marginal values as root cause of anomaly. The SHAP algorithm light weight and find the most contributing feature for analogous instance in short period of time. The experiment results shows that proposed model achieves high accuracy and interpretability with reduced complexity. The proposed model can be evaluated on real-time application with reduced complexity of agents in future work.

REFERENCES

- Andreadis, G., Klazoglou, P., Niotaki, K., & Bouzakis, K.-D. (2014). Classification and Review of Multi-agents Systems in the Manufacturing Section. *Procedia Engineering*, 69, 282–290. doi:10.1016/j.proeng.2014.02.233
- Birje, M. N., & Manvi, S. S. (2011). WiGriMMA: A Wireless Grid Monitoring Model Using Agents. *Journal of Grid Computing*, 9(4), 549–572. doi:10.1007/s10723-011-9181-4
- Brandón, Á., Solé, M., Huélamo, A., Solans, D., Pérez, M. S., & Muntés-Mulero, V. (2020). Graph-based root cause analysis for service-oriented and microservice architectures. *Journal of Systems and Software*, 159, 110432. doi:10.1016/j.jss.2019.110432
- Bulla, C. M., & Birje, M. N. (2021). A Multi-Agent-Based Data Collection and Aggregation Model for Fog-Enabled Cloud Monitoring. *International Journal of Cloud Applications and Computing*, 11(1), 73–92. doi:10.4018/IJCAC.2021010104
- Cloud Monitoring System. (2019). Cloud Monitoring System: Basics, Phases and Challenges. *International Journal of Recent Technology and Engineering*, 8(3), 4732–4746. doi:10.35940/ijrte.C6857.098319
- Davis, N., Raina, G., & Jagannathan, K. (2019). *LSTM-Based Anomaly Detection: Detection Rules from Extreme Value Theory*. 10.1007/978-3-030-30241-2_48
- Gjorgiev, L., & Gievska, S. (2020). Time Series Anomaly Detection with Variational Autoencoder Using Mahalanobis Distance. In V. Dimitrova & I. Dimitrovski (Eds.), *ICT Innovations 2020. Machine Learning and Applications* (Vol. 1316, pp. 42–55). Springer International Publishing. doi:10.1007/978-3-030-62098-1_4
- Huang, H., & Wang, L. (2010). P&P: A Combined Push-Pull Model for Resource Monitoring in Cloud Computing Environment. *2010 IEEE 3rd International Conference on Cloud Computing*, 260–267. doi:10.1109/CLOUD.2010.85
- Jiang, Y., Sun, H., Ding, J., & Liu, Y. (2015). A Data Transmission Method for Resource Monitoring under Cloud Computing Environment. *International Journal of Grid and Distributed Computing*, 8(2), 15–24. doi:10.14257/ijgcd.2015.8.2.03
- Kaur, H., Nori, H., Jenkins, S., Caruana, R., Wallach, H., & Wortman Vaughan, J. (2020). Interpreting Interpretability: Understanding Data Scientists' Use of Interpretability Tools for Machine Learning. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 1–14. doi:10.1145/3313831.3376219
- Lee, W. S., Grosh, D. L., Tillman, F. A., & Lie, C. H. (1985). Fault Tree Analysis, Methods, and Applications Ⓢ A Review. *IEEE Transactions on Reliability*, R-34(3), 194–203. doi:10.1109/TR.1985.5222114
- Lempinen, H. (2012). Constructing a Design Framework for Performance Dashboards. In C. Keller, M. Wiberg, P. J. Ågerfalk, & J. S. Z. Eriksson Lundström (Eds.), *Nordic Contributions in IS Research* (Vol. 124, pp. 109–130). Springer Berlin Heidelberg. doi:10.1007/978-3-642-32270-9_7
- Lin, F., Muzumdar, K., Laptev, N. P., Curelea, M.-V., Lee, S., & Sankar, S. (2020). Fast Dimensional Analysis for Root Cause Investigation in a Large-Scale Service Environment. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 4(2), 1–23. doi:10.1145/3393691.3394185
- Liu, C., Lore, K. G., Jiang, Z., & Sarkar, S. (2021). Root-cause analysis for time-series anomalies via spatiotemporal graphical modeling in distributed complex systems. *Knowledge-Based Systems*, 211, 106527. doi:10.1016/j.knosys.2020.106527
- Lu, S., Wei, X., Rao, B., Tak, B., Wang, L., & Wang, L. (2019). LADRA: Log-based abnormal task detection and root-cause analysis in big data processing with Spark. *Future Generation Computer Systems*, 95, 392–403. doi:10.1016/j.future.2018.12.002
- Lu, X., Yin, J., Xiong, N. N., Deng, S., He, G., & Yu, H. (2016). JTangCMS: An efficient monitoring system for cloud platforms. *Information Sciences*, 370–371, 402–423. doi:10.1016/j.ins.2016.06.009
- Lyu, L., Jin, J., Rajasegarar, S., He, X., & Palaniswami, M. (2017). Fog-Empowered Anomaly Detection in IoT Using Hyperellipsoidal Clustering. *IEEE Internet of Things Journal*, 4(5), 1174–1184. doi:10.1109/JIOT.2017.2709942
- NASA Turbofan Jet Engine Data Set: Run to Failure Degradation Simulation. (n.d.). <https://www.kaggle.com/behrad3d/nasa-cmaps>

- Rodríguez-Pérez, R., & Bajorath, J. (2020). Interpretation of machine learning models using shapley values: Application to compound potency and multi-target activity predictions. *Journal of Computer-Aided Molecular Design*, 34(10), 1013–1026. doi:10.1007/s10822-020-00314-0 PMID:32361862
- Russo, S., Disch, A., Blumensaat, F., & Villez, K. (2020). *Anomaly Detection using Deep Autoencoders for in-situ Wastewater Systems Monitoring Data*. <https://arxiv.org/abs/2002.03843>
- Sakurada, M., & Yairi, T. (2014). Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis - MLSDA'14*, 4–11. doi:10.1145/2689746.2689747
- Shap Documentation. (n.d.). <https://shap.readthedocs.io/en/latest/>
- Singh, H. (2020). Big data, industry 4.0 and cyber-physical systems integration: A smart industry context. *Materials Today: Proceedings*, S2214785320352639. doi:10.1016/j.matpr.2020.07.170
- Solé, M., Muntés-Mulero, V., Rana, A. I., & Estrada, G. (2017). *Survey on Models and Techniques for Root-Cause Analysis*. <https://arxiv.org/abs/1701.08546>
- Stamatis, D. H. (2003). Failure mode and effect analysis: FMEA from theory to execution (2nd ed.). ASQ Quality Press.
- Steenwinckel, B., De Paepe, D., Vanden Haute, S., Heyvaert, P., Bentefrit, M., Moens, P., Dimou, A., Van Den Bossche, B., De Turck, F., Van Hoecke, S., & Ongenae, F. (2021). FLAGS: A methodology for adaptive anomaly detection and root cause analysis on sensor data streams by fusing expert knowledge with machine learning. *Future Generation Computer Systems*, 116, 30–48. doi:10.1016/j.future.2020.10.015
- Thudumu, S., Branch, P., Jin, J., & Singh, J. (2020). A comprehensive survey of anomaly detection techniques for high dimensional big data. *Journal of Big Data*, 7(1), 42. doi:10.1186/s40537-020-00320-x
- Tian, C., Ma, J., Zhang, C., & Zhan, P. (2018). A Deep Neural Network Model for Short-Term Load Forecast Based on Long Short-Term Memory Network and Convolutional Neural Network. *Energies*, 11(12), 3493. doi:10.3390/en11123493
- Vega García, M., & Aznarte, J. L. (2020). Shapley additive explanations for NO2 forecasting. *Ecological Informatics*, 56, 101039. doi:10.1016/j.ecoinf.2019.101039
- Wee, Y. Y., Cheah, W. P., Tan, S. C., & Wee, K. (2015). A method for root cause analysis with a Bayesian belief network and fuzzy cognitive map. *Expert Systems with Applications*, 42(1), 468–487. doi:10.1016/j.eswa.2014.06.037
- Wu, L., Tordsson, J., Elmroth, E., & Kao, O. (2020). MicroRCA: Root Cause Localization of Performance Issues in Microservices. *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 1–9. doi:10.1109/NOMS47738.2020.9110353
- Yousefpour, A., Fung, C., Nguyen, T., Kadiyala, K., Jalali, F., Niakanlahiji, A., Kong, J., & Jue, J. P. (2019). All one needs to know about fog computing and related edge computing paradigms: A complete survey. *Journal of Systems Architecture*, 98, 289–330. doi:10.1016/j.sysarc.2019.02.009
- Zhang, R., & Zou, Q. (2018). Time Series Prediction and Anomaly Detection of Light Curve Using LSTM Neural Network. *Journal of Physics: Conference Series*, 1061, 012012. doi:10.1088/1742-6596/1061/1/012012

Chetan Bulla is working as an assistant professor at KLE College of Engineering and Technology, Chikodi. He is pursuing part-time Ph.D in Visvesvaraya Technological University Technology University, Belagavi. His research area are: cloud computing, fog computing, and cloud monitoring.

Mahantesh N. Birje is working as Professor, Department of Computer Science and Engineering, Visvesvaraya Technological University, Belagavi, Karnataka, India. His research areas are distributed computing, grid computing, cloud computing, and fog computing.