IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 1 November 2024, accepted 17 November 2024, date of publication 25 November 2024, date of current version 4 December 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3506332

RESEARCH ARTICLE

Anomaly Detection in Logs Using Deep Learning

AYESHA AZIZ[1] **AND KASHIF MUNIR**[2]
[1]FAST School of Computing, National University of Computer and Emerging Sciences, Islamabad 44000, Pakistan
[2]Faculty of Computer Studies, Arab Open University, Riyadh, Saudi Arabia

Corresponding author: Kashif Munir (k.munir@arabou.edu.sa)

This work was supported in part by Arab Open University, Saudi Arabia.

ABSTRACT Detection of abnormalities is important for the security and reliability of computer systems as they heavily rely on logs to detect anomalies. The logs provide general information, errors, warnings, and debugging information. Existing approaches for detecting anomalies are sometimes inaccurate due to their limitations related to log-processing leading to loss of semantic significance. Existing approaches, like Deeplog and LogAnomaly, have restrictions in detecting irregularities in log frameworks mainly in large dynamic systems. In this paper, we propose a hybrid anomaly detection technique that combines unsupervised approaches such as Self-Organizing Maps, Bert Encoders, and Autoencoders to handle these issues. The approach improves anomaly identification accuracy by employing semantic vectors obtained by the Bert Encoder to recognize patterns with autoencoders. The evaluation results show that the proposed strategy outperforms the existing methods for various types of data including system logs, network traffic, and financial transactions.

INDEX TERMS Anomaly detection, classification, unsupervised learning, deep learning.

I. INTRODUCTION

Anomaly detection is a significant challenge in a variety of industries, including manufacturing, medical imaging, and cybersecurity [1]. Anomaly detection is the process of identifying patterns in data that are contrary to expected behavior or results. Logs are the primary source of anomaly detection methods in almost every computer system. The anomaly detection is now also conducted at the production level [2], [3]. A variety of events that describe the status of the runtime system like general information, errors, warnings, debugging information, etc., are all contained in these logs.

Logs are important data sources for finding anomalies in computer systems. They document the communications and changes that take place related to data, files, services, or applications. They are usually employed by developers and data-driven processes in order to look at system behaviors and identify, localize, and resolve any abnormal issues. Written by developers in a loose text structure, log messages provide a specific runtime system event to describe the current system status [4]. In particular, a log message comprises a constant string template and variable values stemming from the logging instruction (e.g., print(“a total of *i* faults detected”, 5)) in the source code. Figure 1 presents an instance of a log.

Large systems generally track their performance by analyzing the status of log files, which contain specifics about every action taken within the system. The increased complexity in systems and applications creates more bugs and vulnerabilities. Ultimately, this puts the integrity of the system at higher risk. As a result, it has become harder to detect anomalies and existing techniques are no longer successful. To solve such issues, a system that can efficiently detect anomalies in order to work in a smooth environment is required.

There are many existing approaches for detecting anomalies such as log mining [4], [5], in which different patterns from logs are extracted and passed to the model to classify whether there is any abnormal behavior in it. These techniques are limited to the dataset and fail in real-time. Some techniques do not parse the logs for patterns [6]. They generate embedding vector sequences and store the keys in a list to determine whether the log is normal. However, these techniques fail when dealing with unknown data. Such

The associate editor coordinating the review of this manuscript and approving it for publication was Ines Domingues.

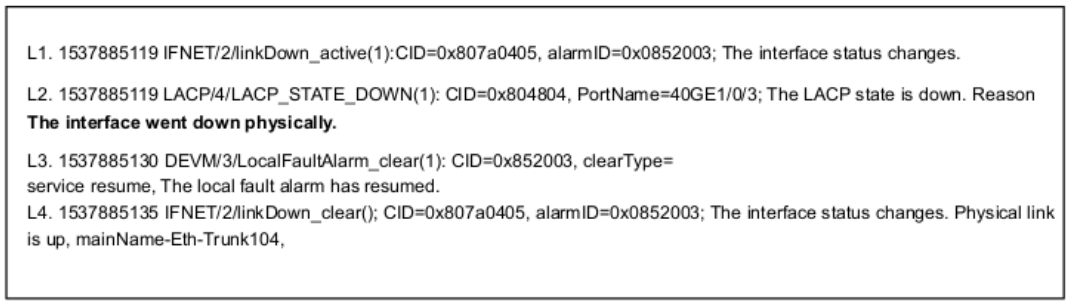

FIGURE 1. An example log [4].

models may produce good results if there is high similarity during testing. For confirming this, there is a need to provide low similarity data during testing to determine whether the model is effective at detecting anomalies.

Figure 2 illustrates the various patterns formed from the logs. We can only recognize these anomalies from their log sequences, which are distinguished by many logs that deviate from the expected patterns. Using keyword matching to detect anomalies in logs results in false alarms generated by the keyword 'down' in both L1 and L2. This is usual since the switch can recover itself as shown in L4. An automatic technique to anomaly identification based on log sequences is required. Log template indexes that do not display semantic links inside log items may result in overlooking of the essential information. Some templates may not be included in template indexing while having equivalent language features, which can lead to false warnings.

Existing techniques have drawbacks because services can generate new log templates between two periodic re-trainings. Providing manual labeling for a significant number of new log templates is not efficient. The proposed model is designed for real-time scalability by using BERT embeddings, SOM clustering, and autoencoders, which capture evolving log patterns without relying on fixed templates or exhaustive parsing. This hybrid approach balances computational efficiency by dividing tasks across each component, minimizing resource demands, and processing logs with low latency. BERT and SOM are optimized for incremental data processing, which enhances resource efficiency and enable high throughput anomaly detection even in dynamic environments. This ensures the method's suitability for real-time log monitoring with high accuracy and adaptability to new log patterns.

Anomalies are inevitable as complexity and scale of different systems have increased. A minor flaw in the system can cause poor performance, corruption of data, and degradation of performance. Complex and critical systems need anomaly detection for their quality assurance. The existing methods for preprocessing semi-structured log data rely on their ability to parse logs. To procure log events, log parsers disengage the variable part of log messages while preserving the stable part. Furthermore, because some techniques rely on data semantics and similarity, they do not properly work on unseen data. To address such issues, we examine different methods in hybrid machine learning to see which models are best for detecting anomalies in real-time. When given unseen data, they should properly function. Following are our contributions:

- We use random sampling to ensure diverse and representative data. The proposed approach utilizes data similarity for the effective identification and detection of anomalies with unique characteristics.
- We employ autoencoders and self-organizing maps (SOMs) to enhance the performance of deep learning for detecting abnormalities for the variations in the complicated and evolving structures.

Section II provides a thorough review of the existing techniques describing their strengths, major weaknesses, and relevance to the proposed work. The review helps in identifying the gaps in the existing research and sets the foundation for the proposed approach. In section III, the proposed methodology is introduced by describing the used techniques, algorithms, and models. Section IV focuses on the evaluation of the proposed methodology. An examination and analysis of the findings of these experiments is provided. In Section V, we conclude by summarizing the key findings and discussing the potential future research avenues.

II. LITERATURE REVIEW

Previous research works concentrate on predefined log templates [8] and patterns that are unpredictable due to the vast variety of the log domains. Many tools such as deeplog [9], LogAnomaly [5], etc., are used to detect anomalies. The main limitation of the existing techniques is that they require log patterns to train their models, which are predictable in many conditions. There are many cluster-based techniques such as Isolation Forest [10], [11], two deep-encoders [12], K-Nearest Neighbors [7], [13], Density-based clustering [4], and Template-to-Vec [5]. The flaws in such techniques are related to accuracy issues, false alarms, and the cost of their techniques. Some log templates are statically defined and then compared to logs for determining which log belongs to the comparing template. However, this technique

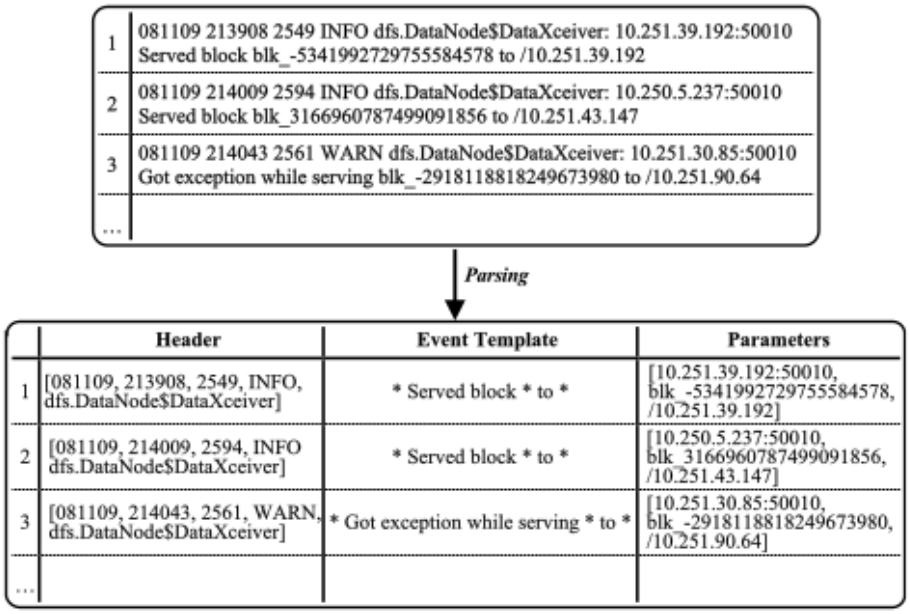

FIGURE 2. An example of HDFS logs and parsed results [7].

has two flaws. It incorrectly maps templates to logs and it cannot detect sequential and quantitative anomalies at the same time.

Du et al. [4] utilize a log parser with LSTM to automatically uncover log patterns during standard operations. A key strength of this method is its ability to be easily adapted and incorporated into different execution patterns. However, a limitation is that it cannot find anomalies when the log patterns vary from the model that it is trained on. Meng et al. [5] propose combining Template2vec and LSTM to detect sequential and quantitative log anomalies. The approach is useful at retrieving semantic information concealed inside log templates. It struggles to detect both sequential and quantitative irregularities at the same time making it less useful in certain situations.

Guo et al. [14] formulate BERT, a model that learns sequence patterns from normal log data using two innovative self-supervised techniques. This technology excels at detecting abnormalities when patterns depart from the expected log sequences. The main disadvantage is that it cannot detect anomalies in case of low resemblance with previously observed data and hence limiting its applicability on unseen data. Liu et al. [6] utilize a self-attention neural network for evaluating anomalies and decision making. An advantage of this strategy is data augmentation to establish decision limits based on available normal training data. The generated anomaly scores are not interpretable and the results for actual aberrant log messages can vary.

Nedelkoski et al. [15] propose a self-attention encoder network that distinguishes between normal and auxiliary easy-access log files. The framework obtains benefits by mastering brief log representations that embody the distinctive qualities of normal versus abnormal logs. However, a limitation is that it uses an incorrect distance from the centroid, which can result in inaccurate results. The study by [12] applies a two-step strategy for the clustering of log events. It uses threshold filtering to minimize the scope of event analysis and constructs the log according to the features of identifying anomalies. Some research studies use the log parsing technique [16] prior to feeding the log information into machine learning models. This method is hard to apply on large volumes of data in real-time. In real-world problems, millions of logs are ingested in real-time and log parsing for those logs is prohibitively expensive.

Log parsing is used to extract patterns from system logs by eliminating parameters and focusing on keywords. Various log parsing techniques such as pattern mining [17], language modeling [10], and heuristic-based approaches [7] are employed. Figure 2 illustrates an example of a log template. It shows the record of an HDFS system containing all its parameters and keywords. The extracted log templates are used to map and categorize incoming logs as normal or anomalous [18]. However, the accuracy of these techniques in template extraction is not always guaranteed especially when dealing with unseen data that significantly differ from the training data. To address this, researchers in [4] propose an approach that combines log parsing techniques with nested LSTM models to detect log anomalies. The technique assigns probabilities to new templates and sorts them into categories that reflect their high or low probabilities. However, it has limited abilities to identify delicate abnormalities that deviate slightly from typical patterns on account of its dependence on high probability thresholds [19].

An approach presented in [5] integrates Template2Vec to derive semantic information from log templates blending sequential and quantitative log patterns to bolster predictions. However, it focuses on either sequential or quantitative anomalies rather than simultaneously detecting both of them. Another method proposed in [16] and [20] involves creating pattern hierarchies using a fast clustering algorithm and a fast pattern recognition algorithm to detect anomalies. However, sometimes it cluster outliers into normal logs leading to an increased false positive rate. Log clustering is a problem identification technique [21] that aims at clustering similar log entries. It can be used as a diagnostic tool to find solutions to existing problems or to enhance system performance. However, the correctness of clustering essentially depends on the quality of the log data and the selected similarity measures. Interpretation of the clusters requires specialized knowledge and maintaining relevant log clusters in evolving systems can be problematic [22]. An unsupervised cluster evolution technique for identifying anomalies in dynamic log files is described in [23]. It is sensitive to fluctuations and noise in the data, which can disrupt the clustering mechanism and generate false detection or missed anomalies [24].

Another approach presented in [25] uses Sparse Canonical Correlation Analysis (SCCA) to find relationships among the network attributes and improve the anomaly detection by concentrating on those relationships. The method focuses on detecting anomalies based on attribute correlations. However, the effectiveness of the method for general anomaly types remains unexplored. For the random masking and padding, they address the problems related to the missing data. SCCA does not fully address the scalability issues of large networks or the anomalies related to evolving dynamic systems.

In [26], the authors introduce an ingenious method of anomaly detection based on residual learning. It enhances the performance of graph convolutional networks and maps the node embeddings to a hypersphere. It is an effective method for capturing structural and attribute-based anomalies between attributed networks. The method is mainly applicable on static network structures and does not fully address the scalability problem when used on large scale networks. Although, it is able to handle structural anomalies well, it lacks the ability to simultaneously handle complex attribute driven outliers in dynamic and evolving network structures. Similarly, in [27], the anomalous node detection in attributed social networks combines variational autoencoders with GANs to model network structure and distributions. However, the training stability of variational autoencoders and GANs remains a problem in large datasets. Additionally, while it mainly works on social networks, its use on other types of attributed networks and more complex evolving anomalies needs to be further explored. Both of these studies leave gaps in their ability to deal with dynamic networks and maintain scalability when dealing with high dimensional datasets.

The authors in [28] attempt to survey the threats, which are posed by the dark web, illicit trade, and cyber terrorism. The existing detection techniques such as dark web monitoring and TOR analysis are reviewed. They lack in terms of scalability and effectiveness to counter the recent threats such as deepfakes, evolving encryption methods, etc. The review concludes that more adaptive and sophisticated detection mechanisms are needed to keep pace with the continuously evolving dark web conditions.

The authors in [6] present an approach to anomaly detection where the use of BERT embeddings is made to extract the semantics of messages within the log files alleviating the need for log parsing. It uses BERT embeddings for log messages and perform clustering to obtain similar log messages. Anomalies are identified as outliers in the clustering results demonstrating the effectiveness of the approach in detecting anomalies [29]. Similarly, Guo et al. [14] propose the BERT model in the encoder-decoder framework and apply multi-head attentions to forecast log templates while [14], [30] introduces the log-based anomaly detection without the log parsing with the sequence-to-sequence models. Logsy, presented by Nedelkoski et al. [15], [31], calculates the log embeddings via attention mechanisms and an encoder architecture. It identifies anomalous log entries based on distances with a centroid. Furthermore, there is an approach of Word2Vec of a log data set for performing unsupervised anomaly detection described in [2] and [32]. These methods have limitations such as the need for manual decision boundaries, dependence on labeled training data, or reliance on the quality of embeddings.

The authors in [33] present a robust architecture called Swisslog in which CNN [22] and LSTM Network [34] architectures are incorporated to capture the spatial and temporal effects in the logs. It is shown that Swisslog is capable of identifying various types of faults using system log data. However, it requires labeled training data for supervised learning [33]. Another approach, A2Log [6] employs self-attention neural network to assign serving score on log messages and decides the decision boundary by utilizing data augmentation. Since A2Log is trained only on normal log messages, scoring of anomaly for the abnormal ones becomes challenging [6].

In the next section, we explain the proposed hybrid approach for the anomaly detection. It uses machine learning algorithm, deep learning models, statistics, and analytical techniques to capture the pattern, perform data classification exercises, and distinguish between normality and abnormality. We discuss each component of the proposed hybrid approach to demonstrate how they jointly contribute to the improved accuracy and efficiency of the process of anomaly detection. By exploring the intricacies of the proposed hybrid approach, we aim to demonstrate its potential in addressing the challenges associated with anomaly detection in complex systems.

III. PROPOSED MODEL "ANYLOG"

Deep anomaly detection leverages deep learning techniques to learn feature representations or anomaly scores through neural networks. This approach has shown significant improvements over conventional methods addressing complex detection problems in real-world applications. However, existing categorizations and reviews of anomaly detection techniques primarily focus on traditional methods and overlook the unique challenges associated with deep anomaly detection such as effectively handling of the unseen data. To enhance the effectiveness and applicability of deep anomaly detection, it is crucial to address these gaps and develop better strategies for handling such complexities.

In log analysis, a common pipeline involves preprocessing the logs using tools like Filebeat and cleaning them for downstream models. The logs are then transformed into numerical embeddings using a BERT encoder capturing their semantic meaning. These embeddings are fed into a Self-Organizing Map (SOM), which is an unsupervised learning algorithm that clusters similar log entries into a 2D grid. By clustering the logs, patterns and anomalies can be identified aiding in troubleshooting and system optimization. However, there are challenges related to the labeled training data, model interpretability, and adaptability to different log formats and domains. Addressing these challenges further enhances the potential of deep anomaly detection in log analysis tasks leading to more effective insights extraction and system optimization capabilities. Overall, the pipeline of the proposed methodology is shown in Figure 3. The pipeline is used to process logs and extract meaningful insights from them. By clustering similar log entries, it is possible to identify patterns and anomalies in the logs, which can help with troubleshooting and system optimization.

A. PREPROCESSING OF LOGS

The log message shown in Figure 4 originates from the Hadoop Distributed File System's DataBlockScanner component and confirms the successful verification of a specific data block ensuring its integrity and absence of corruption. The removal of the newline character during preprocessing is the only modification made to the log message. Although, this preprocessed log message is deemed normal behavior, a comprehensive assessment of contextual information and analysis of other log messages and system behavior are required for accurate anomaly detection.

B. WORDPIECE TOKENIZATION

To perform WordPiece tokenization on a preprocessed log message, a pre-trained tokenizer model, such as those available in the Hugging Face Transformers library, is utilized. This model takes the preprocessed log message as an input and produces a sequence of WordPiece tokens. The subwords within larger words are indicated by the prefix ##. For instance, "block" and "scan" are represented as separate subwords. The following is an example of the WordPiece tokenization process utilizing the Hugging Face Transformers library's pre-trained WordPiece tokenizer:

Input: **info dfs datablockscanner verification succeeded for**

Output: **['info', 'dfs', 'data', '##block', '##scan', '##ner, 'ver', '##infica', '##tion', 'suc', '##ceed','##ed','for']**

C. BERT ENCODER

BERT Encoder is a classical pretrained neural network, which is used for text data to build semantic word vector. Semantic vectors of analyzing log messages are compared to reveal the patterns and similarities that define anomalies. Implementation of BERT Encoder leads to better results in anomaly detection by extracting the contextual information of the log messages against basic feature-based anomaly detection. The method enhances credibility without compromise of the system failures and cybercriminal activities. It helps in obtaining the important embeddings and makes the analysis of logs easier due to the ability to compare with the help of semantic data.

D. HYBRID MACHINE LEARNING

To improve the results and minimize false alarms, anomaly detection models use a combination of several unsupervised methodologies. Based on the integration of features of several algorithms including BERT, autoencoders, and SOM, the hybrid approach acquires semantic embeddings, pattern mining, and clustering of log messages for anomaly detection. This extensive strategy enhances the ability of detecting anomalies and the need to understand the fundamental causes of the anomalies.

E. AUTOENCODERS IN ANOMALY DETECTION

Autoencoders are models made up of neurons that encode input information into a compact point, which is then decoded to match the original state. Autoencoders support anomaly detection by learning a system's expected behavior and noting any differences from it. We create semantic vectors for log messages through the integration of autoencoders with BERT Encoder and SOM, which still preserves the context and meaning of the data. Then, these vectors are provided to the autoencoder, which is trained to reconstruct the input with minimal disparities. It is feasible to observe real-time abnormalities by comparing the reconstruction error with a predetermined threshold.

F. SELF ORGANIZING MAP (SOM)

It is possible to employ the autoencoder's encoding of log messages. The obtained embeddings enable the use of the subsequent SOM method [35]. SOM is a type of unsupervised machine learning in which it is possible to plot the data and reduce its dimensions [36]. In the domain of anomaly detection, an SOM is used to cluster log messages based on embedding space. Once the SOM is trained, more log

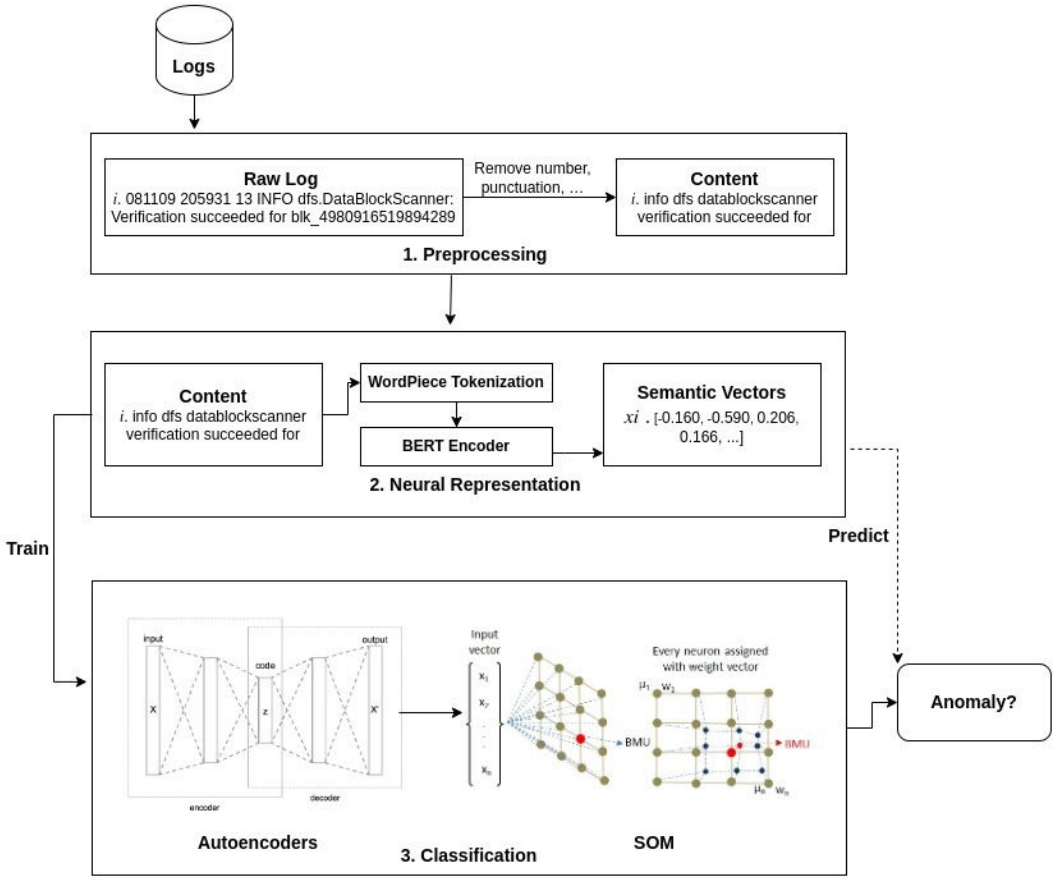

FIGURE 3. Proposed methodology.

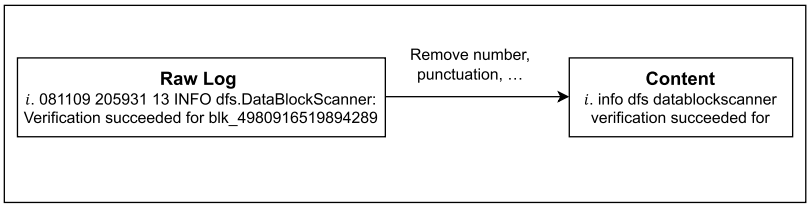

FIGURE 4. Preprocessing example.

message embeddings can be added and every neuron in the grid receives a score indicating how well the input is matched. The neuron that scores highest is characterized as the 'Best Matching Unit' for the input. With analysis of the generated map, we are able to identify the groups of similar log messages [37] created from anomalies. There are some circumstances in which an input embedding does not have a nearby 'Best Matching Unit' on the map. This is an indication of an abnormal condition.

G. SILHOUETTE ANALYSIS

The clusters are validated by Silhouette analysis [38]. It uses cohesion and separation as the two main principles. Cohesion of a cluster is the closeness of the points of a certain cluster. In case of K-Means algorithm, it calculates the distance from a data point to all other data points belonging to the same cluster [39]. In the proposed study, this is used to categorize anomalous situations as either critical or non-critical. It therefore means that where there is an abnormality, changes can be triggered in line with type of the abnormality. Silhouette coefficient values are between -1 and 1. In unsupervised learning, silhouette analysis measures the quality of cluster solutions based on clustering. They assist in deciding the right number of clusters and enhances the clustering outcomes. It determines the extent of clustering by evaluating the cohesiveness or compactness within a cluster and the separation between different clusters thereby proving easy to interpret clusters that are well separated and dense. In anomaly detection, this technique helps to group the abnormal points and hence increasing the chances

of noticing a threat or anomaly. In image segmentation, customer segmentation, and outlier detection, silhouette analysis enhances the reliability of clustering algorithms.

In silhouette analysis, k-means clustering is used to evaluate cohesion and separation of clusters. In this study, the employed non-primary clustering technique is Self Organized Maps (SOM) but k-means is incorporated as an additional case of gauging how well the clustering performs. The result of k-means clustering is clear and interpretable, which suits the silhouette analysis that measures the closeness of data points within a cluster (cohesion) and how they separate from other clusters (separation). The study intends to assure that the clusters found through SOM-based embedding have a proper structure by applying k-means.

In the next section, we provide a detailed investigation of the findings obtained from experimentation. We demonstrate the effectiveness of the proposed method and also provide the analysis of the outcomes.

IV. EXPERIMENTS AND DISCUSSION OF RESULTS

A number of steps are conducted to determine the validity and efficiency of the anomaly detection model for server logs. HDFS and OpenStack logs are collected via the loghub repository and a diverse set of data is picked at random to trace distinct trends and issues. This removes bias and helps in understanding of the data distribution. The BERT model is utilized to process the chosen data and generate trained embeddings, which are then fed into the SOM for anomaly classification. Figure 5 illustrates the use of silhouette analysis to distinguish between important and inconsequential irregularities. To validate test findings, log messages are extracted from Elasticsearch and categorized as ordinary or unusual by the model. This inclusive approach emphasizes the model's accuracy in detecting server log anomalies as well as the evidence of its practicality in real-world scenarios.

A. BERT RESULTS

A BERT model with 12 encoders and 768 hidden layers is used. Various experiments are performed to assess the impact of hyperparameters such as learning rate, number of epochs, batch size, size of vocabulary, and maximum length of the sequences. In the proposed model, other hyperparameters are set to the best values. For example, the learning rate is set to $2e-5$, the number of epochs of the model is set to 200 and the batch size for each node is set to 32. These parameters have a high Masked Language Model (MLM) accuracy of 0.9165 and a low MLM loss of 1.0047, which prove good feature learning from the log data. BERT's trained embeddings are subsequently integrated with the SOM for further investigation and anomaly detection.

In order to get the trained embeddings from BERT, the embeddings are then sent to the SOM for certain predictions. In order to evaluate the performance of the model, the class distribution needs to be taken into account and the performances have to be compared over different class distributions. Selecting an approach right for handling different class distributions can help in improving anomaly detection model to better detect such anomalies. We then assess its performance on the end-labeled HDFS and OpenStack data.

We develop a confusion matrix, which results from the comparison of actual and predicted labels. Based on the confusion matrix, the sensitivity of the model is calculated as $\left(\frac{2737}{2858}\right) \times 100 \approx 95.7\%$, specificity as $\left(\frac{307}{351}\right) \times 100 \approx 87.4\%$, and recall is 96%.

For visualization purpose, we merge the HDFS and BGL datasets into a single dataset. The merged dataset is then split into 80% training and 20% test datasets. We provide the performance metrics for both the training and testing datasets in terms of accuracy, precision, recall, F1 score, and specificity as listed in Table 1. With the accuracy of 95% and high recall of 96%, the model successfully identifies most of the true positives for the training dataset. As shown in Table 1, the closely matching corresponding results of the test dataset show the good performance of the proposed model.

The confusion matrices of the training and test datasets are presented in Tables 2 and 3. These matrices show the number of true positives, true negatives, false positives, and false negatives predicted by the model. In the confusion matrix, it appears that there is a balance in the correct predictions with 300 true negatives and 2740 true positives for the training dataset. Similarly, the confusion matrix of the test dataset also has a high percentage of correctly identified values revealing the model's ability to generalize well on the unseen data.

The scatter plot is used to display the results in a graphical format so that one can easily view the results as shown in Figure 6. It is essential to visually represent the clustering and anomaly detection outcomes. Each colored dot in the scatter plot represents a specific cluster. Each cluster highlights different types of system activity. For example, one cluster can be based on system logins and a red dot within that cluster indicates an unusual or abnormal login behavior. Another cluster can be related to system interruptions and a red dot represents abnormal behavior in the interruptions. Each color-coded cluster shows a different type of system behavior with red dots highlighting anomalies in that particular category. The red dots indicate anomaly data points predicted by the model. The interpretability of the results is enhanced offering a clear understanding of how anomalies are detected in a dataset. This step ensures that Figures 6 and 7 accurately reflect the clustering and anomaly detection.

As shown in Figure 7, we create 3 random cluster categories. We apply this to the dataset, which has multiple server logs. We employ a data grouping method that aims to identify low-similarity data points. By grouping the data based on their similarity, we can distinguish between instances that exhibit similar patterns and those that significantly deviate. This grouping strategy allows us to focus on identifying and detecting anomalies that exhibit unique characteristics

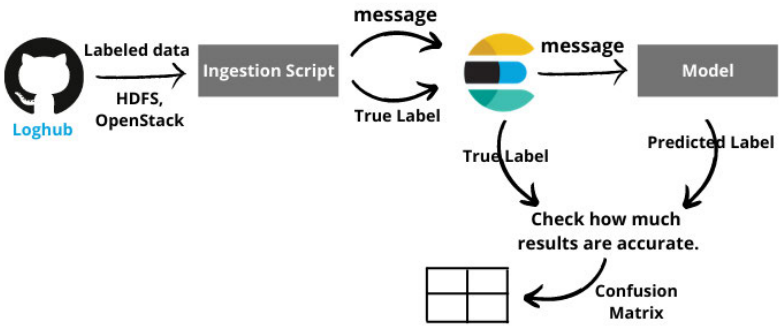

FIGURE 5. Evaluation methodology.

TABLE 1. Performance metrics for the training and testing datasets.

Dataset	Accuracy	Precision	Recall	F1 Score	Specificity
Training Set	95%	93%	96%	94.5%	88%
Testing Set	93%	90%	92%	91%	87%

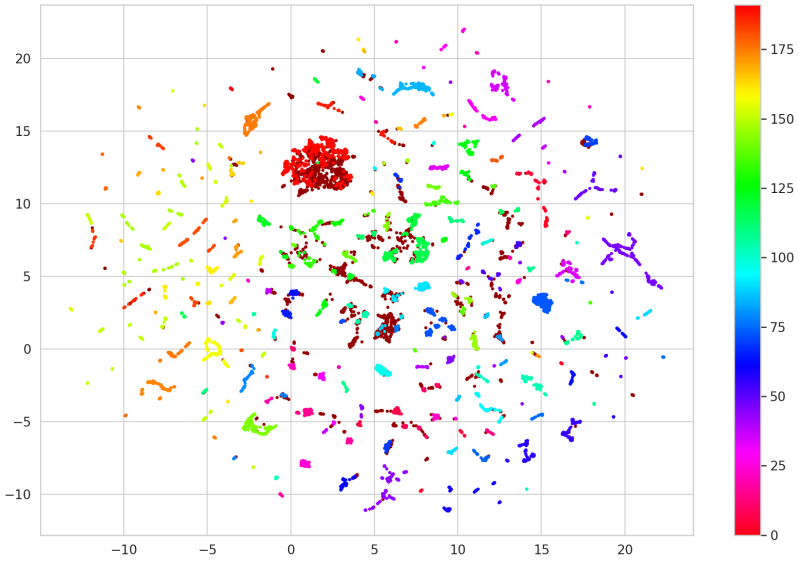

FIGURE 6. Scatter plot for predicted clusters.

TABLE 2. Confusion matrix for the training set.

Training Set	Predicted Positive	Predicted Negative
Actual Positive	2740	60
Actual Negative	100	300

TABLE 3. Confusion matrix for the testing set.

Testing Set	Predicted Positive	Predicted Negative
Actual Positive	2737	44
Actual Negative	121	307

compared to the majority of data points. Then, we perform the Silhouette analysis to classify the anomalies.

B. EXPERIMENTS

As mentioned before, we use two datasets, BGL and HDFS. The information from both datasets is used to check for anomalies in the system logs. The HDFS dataset consists of Hadoop Extended Record files from the Hadoop Distributed File System cluster. The logs are accumulated over a period of three weeks and contain information regarding file system usage, performance, and issues. Writing the logs to a file requires a large number of lines, roughly equal to 1, 200, 000. However, only around 12, 000 messages are considered unusual. We use these datasets to evaluate the efficacy of the proposed approach.

- BGL: Table 4 lists the total number of logs in the BGL dataset. Each log in the collection is labeled as either

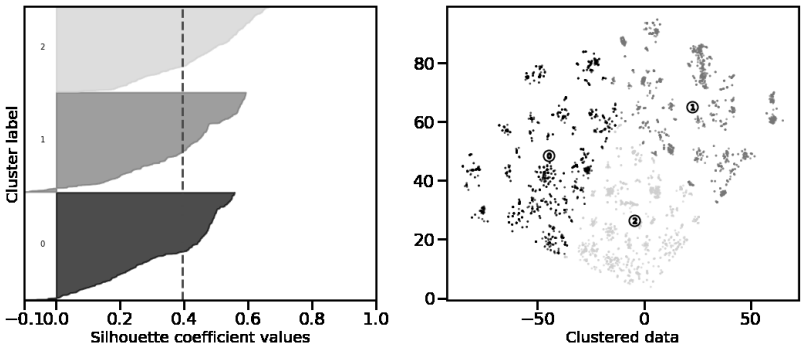

FIGURE 7. Silhouette analysis for k-means clustering with 3 clusters.

TABLE 4. Dataset details with duration, number of logs, and anomalies.

Datasets	Duration	No. of logs	No. of anomalies
HDFS	38.7 hours	11,175,629	16,838 (blocks)
BGL	7 months	4,747,963	348,460 (logs)

anomalous or normal. The BGL dataset was developed on the Blue Gene/L supercomputer, which is a powerful system housed at Lawrence Livermore National Lab. This dataset provides critical information and insights into the system's behavior and features allowing us to create effective anomaly detection systems and accurately assess their efficacy. This dataset includes labeled anomalies, which primarily relate to system errors, including hardware failures, communication issues, and specific process failure. These anomalies often indicate system's instability or specific faults within the system's components.

  - There are 348, 460 labeled anomalous log entries out of nearly 4.7 million total entries in the BGL dataset. It is nearly 7% of the total log entries. This relatively high proportion of anomalies allows for a more granular exploration of various anomaly types and assists in testing a model's capability to distinguish between normal and abnormal system behaviors.

- HDFS: The HDFS dataset contains 11, 175, 629 logs gathered from more than 200 Amazon EC2 machines as listed in Table 4. As part of program executions, different actions and operations, including writing and shutting files, are logged in the HDFS. Typically, a program running in HDFS creates a block of logs. The labeled dataset gives significant insights into the HDFS system's behavior and trends allowing for the creation and testing of effective anomaly detection systems for preserving system integrity and dependability. The anomalies in the dataset are primarily related to issues like file corruption, loss of data blocks, and unexpected file system operations. Given HDFS's operational complexity, these anomalies challenge the model's detection accuracy especially in the cases where the anomalies are just subtle deviations from the expected patterns.
  - The HDFS dataset includes a significantly lower proportion of anomalies than those in BGL dataset. Only 16, 838 out of over 11 million logs are labeled as anomalies. This sparse anomaly distribution (less than 0.2% of the dataset) reflects the real-world conditions where anomalies are rare within predominantly normal data sequences.

Based on the timestamps of the logs, we divide the datasets into training and testing sets in our experiments. We use the first 80% of the logs, in chronological order, as training data and the remaining 20% as testing data. This guarantees that the model is trained on a representative subset of the data and tested on previously unknown samples. We utilize the labels assigned to both the BGL and HDFS datasets to assess the efficacy of our anomaly detection techniques. We compare the expected labels to the true labels that are given to determine the accuracy and effectiveness of the proposed algorithms for spotting irregularities in the data logs.

1) COMPARISON OF ANYLOG WITH EXISTING MODELS

We compare the proposed technique "Anylog" with the four unsupervised existing methods namely LogCluster [21], PCA [40], Invariant Mining (IM) [41], and Deeplog [4]. In anomaly detection, performance metrics like precision, recall, and F1 score are used to assess the performance of a classification method. These metrics indicate the ability of a model to correctly categorize the irregularities. The model provides the results in the form of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). The correct identification of anomalous logs (or blocks) in the HDFS dataset is referred to as TP. TN represents the number of times the model accurately identifies the normal logs. An FP is generated when the model predicts a normal log as an abnormal pattern. However, if an anomalous log is classified as normal, it is labeled as FN. Equations 1, 2, and 3 are used to calculate the precision, recall, and F1 score of the models.

$$\text{Precision} = \left(\frac{TP}{TP + FP}\right) \tag{1}$$

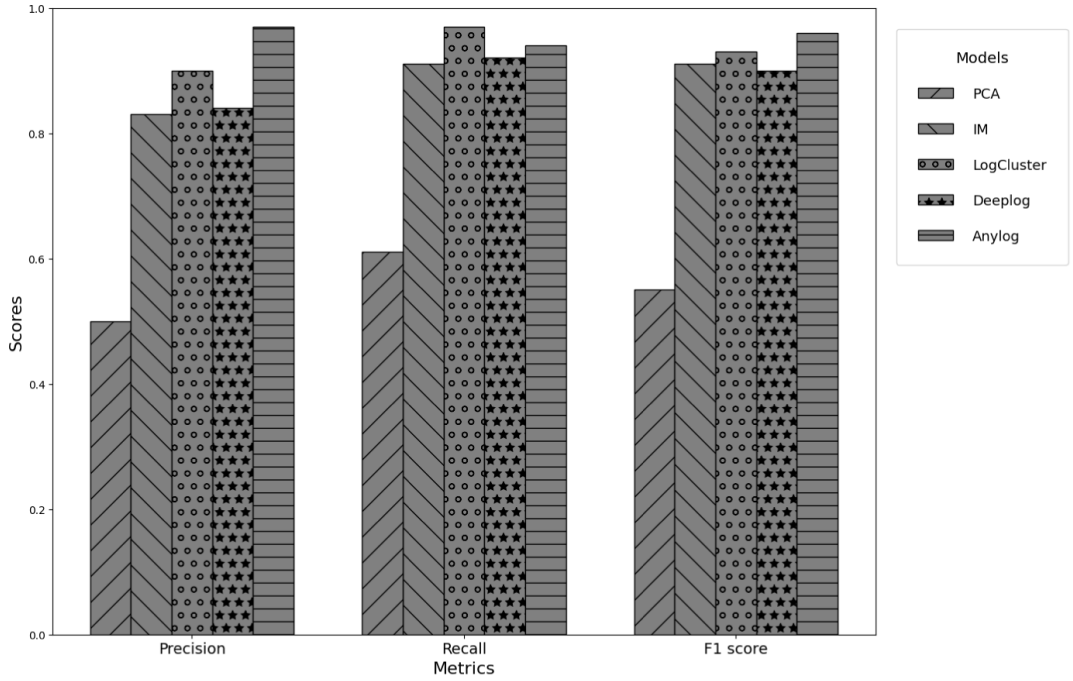

FIGURE 8. Comparison of the models using BGL dataset.

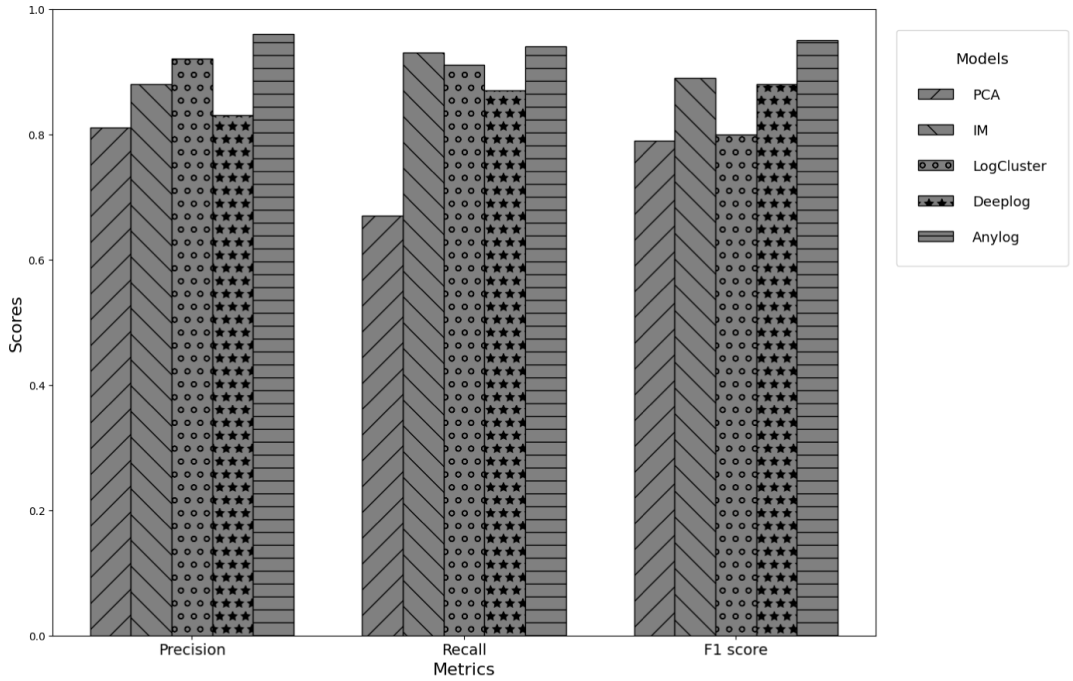

FIGURE 9. Comparison of the models using HDFS dataset.

$$\text{Recall} = \left(\frac{TP}{TP + FN}\right) \tag{2}$$

$$\text{F1 score} = \left(\frac{2 \times Precision \times Recall}{Precision + Recall}\right) \tag{3}$$

2) EVALUATION

In a previous section, we presented the visualization of merged dataset through scatter plot and silhouette analysis. Now, we use individual dataset (BGL and HDFS) to compare the proposed model 'Anylog' with the four existing unsupervised baseline models, LogCluster, Principal Component Analysis (PCA), Invariant Mining (IM), and Deeplog.

Regarding precision and F1 score, Anylog outperforms all other models for both datasets as shown in Figures 8 and 9. In terms of recall, the proposed approach produces favorable results for both datasets. Considering recall, Anylog performs

better than PCA, IM and Deeplog models for the BGL dataset but LogCluster model performs slightly better than Anylog as shown in Figure 8. However, Anylog demonstrates better recall as compared to that of all other models for the HDFS dataset as shown in Figure 9.

The proposed method generates lower number of false alarm instances than those of Invariant Mining and Deeplog. Through integrating autoencoders and SOMs in deep learning, it is possible to detect relevant patterns and relationships in the data, which improves the identification of anomalies in numerous complicated and frequently evolving systems. In contrast, the autoencoder helps in the learning of the normal distribution of the data and the SOM assists in giving visualization of the data. It also helps in the form of clustering of the data in order to detect the abnormalities. This combination enhances the ability of the model to identify small deviations when operating in a dynamic environment leading to improved anomaly detection rate. In our case, where $FPR = 97\%$ and 76% for the BGL and HDFS datasets respectively, the anomaly detection capability is considered good. It indicates that the proposed model has a low FN rate, which means that this type of method can detect most true abnormalities in the data. The proposed method enhances understanding of log templates by considering semantically related data using autoencoders and SOM in order to accurately determine minor variations and discrepancies.

V. CONCLUSION AND FUTURE WORK

In this paper, a new anomaly detection approach that combines both unsupervised and supervised learning has been introduced. In order to improve the anomaly detection accuracy, the work employs Self-organising Maps (SOMs), Bert Encoders, and Autoencoders. Based on Bert Encoder for creating the semantic vectors and clustering features by SOMs, anomalies are detected in the log data. Autoencoders are employed for pattern recognition. The proposed methodology provides solution to the limitations found with classical and deep learning systems by integrating uncategorized learning systems and semantic sense making. The results confirm the ability of the proposed approach to analyze system logs, network traffic, and financial activity confirming the possibility of the approach usage.

In future, the real-time monitoring of logs may also be explored by combining several anomaly detection models like ensemble technique, stacking, boosting, cascade anomaly detection technique, etc. Moreover, the incorporation of domain specific knowledge and opinions can enhance the model's ability to identify meaningful patterns and deviations.

REFERENCES

[1] V.-H. Le and H. Zhang, "Log-based anomaly detection without log parsing," in *Proc. 36th IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE)*, Nov. 2021, pp. 492–504.

[2] M. Wang, L. Xu, and L. Guo, "Anomaly detection of system logs based on natural language processing and deep learning," in *Proc. 4th Int. Conf. Frontiers Signal Process. (ICFSP)*, Sep. 2018, pp. 140–144.

[3] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "An effective unsupervised network anomaly detection method," in *Proc. Int. Conf. Adv. Comput., Commun. Informat.*, Aug. 2012, pp. 533–539.

[4] M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: Anomaly detection and diagnosis from system logs through deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 1285–1298.

[5] W. Meng, Y. Liu, Y. Zhu, S. Zhang, D. Pei, Y. Liu, Y. Chen, R. Zhang, S. Tao, P. Sun, and R. Zhou, "LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 4739–4745.

[6] Z. Liu, T. Qin, X. Guan, H. Jiang, and C. Wang, "An integrated method for anomaly detection from massive system logs," *IEEE Access*, vol. 6, pp. 30602–30611, 2018.

[7] P. He, J. Zhu, S. He, J. Li, and M. R. Lyu, "An evaluation study on log parsing and its use in log mining," in *Proc. 46th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2016, pp. 654–661.

[8] A. J. Hoglund, K. Hatonen, and A. S. Sorvari, "A computer host-based user anomaly detection system using the self-organizing map," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Netw. IJCNN. Neural Comput., New Challenges Perspect. New Millennium*, vol. 5, May 2000, pp. 411–416.

[9] A. Farzad and T. A. Gulliver, "Unsupervised log message anomaly detection," *ICT Exp.*, vol. 6, no. 3, pp. 229–237, Sep. 2020.

[10] Z. Ding and M. Fei, "An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window," *IFAC Proc. Vol.*, vol. 46, no. 20, pp. 12–17, 2013.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inform. Process. Syst. (NIPS)*, 2017, pp. 5998–6008.

[12] A. I. Hajamydeen, N. I. Udzir, R. Mahmod, and A. A. A. Ghani, "An unsupervised heterogeneous log-based framework for anomaly detection," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 24, pp. 1117–1134, Aug. 2016.

[13] B. Wang, S. Ying, and Z. Yang, "A log-based anomaly detection method with efficient neighbor searching and automatic k neighbor selection," *Sci. Program.*, vol. 2020, pp. 1–17, Jun. 2020.

[14] B. Min, J. Yoo, S. Kim, D. Shin, and D. Shin, "Network anomaly detection using memory-augmented deep autoencoder," *IEEE Access*, vol. 9, pp. 104695–104706, 2021.

[15] S. Nedelkoski, J. Bogatinovski, A. Acker, J. Cardoso, and O. Kao, "Self-attentive classification-based anomaly detection in unstructured logs," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Jun. 2020, pp. 1196–1201.

[16] H. Hamooni, B. Debnath, J. Xu, H. Zhang, G. Jiang, and A. Mueen, "LogMine: Fast pattern recognition for log analytics," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2016, pp. 1573–1582.

[17] A. Raza, K. Munir, M. S. Almutairi, and R. Sehar, "Novel class probability features for optimizing network attack detection with machine learning," *IEEE Access*, vol. 11, pp. 98685–98694, 2023.

[18] S. He, J. Zhu, P. He, and M. R. Lyu, "Experience report: System log analysis for anomaly detection," in *Proc. IEEE 27th Int. Symp. Softw. Rel. Eng. (ISSRE)*, Oct. 2016, pp. 207–218.

[19] X. Zhang, Y. Xu, Q. Lin, B. Qiao, H. Zhang, Y. Dang, C. Xie, X. Yang, Q. Cheng, Z. Li, J. Chen, X. He, R. Yao, J.-G. Lou, M. Chintalapati, F. Shen, and D. Zhang, "Robust log-based anomaly detection on unstable log data," in *Proc. 27th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, Aug. 2019, pp. 807–817.

[20] J. Breier and J. Branišová, "Anomaly detection from log files using data mining techniques," in *Information Science and Applications*. Cham, Switzerland: Springer, 2015, pp. 449–457.

[21] Q. Lin, H. Zhang, J.-G. Lou, Y. Zhang, and X. Chen, "Log clustering based problem identification for online service systems," in *Proc. IEEE/ACM 38th Int. Conf. Softw. Eng. Companion (ICSE-C)*, May 2016, pp. 102–111.

[22] M. Catillo, A. Pecchia, and U. Villano, "AutoLog: Anomaly detection by deep autoencoding of system logs," *Expert Syst. Appl.*, vol. 191, Apr. 2022, Art. no. 116263.

[23] M. Landauer, M. Wurzenberger, F. Skopik, G. Settanni, and P. Filzmoser, "Dynamic log file analysis: An unsupervised cluster evolution approach for anomaly detection," *Comput. Secur.*, vol. 79, pp. 94–116, Nov. 2018.

[24] C. Cavallaro and E. Ronchieri, "Identifying anomaly detection patterns from log files: A dynamic approach," in *Proc. 21st Int. Conf. Comput. Sci. Appl. (ICCSA)*, Cagliari, Italy. Cham, Switzerland: Springer, Sep. 2021, pp. 517–532.

[25] W. Khan, M. Ishrat, A. Neyaz Khan, M. Arif, A. A. Shaikh, M. M. Khubrani, S. Alam, M. Shuaib, and R. John, "Detecting anomalies in attributed networks through sparse canonical correlation analysis combined with random masking and padding," *IEEE Access*, vol. 12, pp. 65555–65569, 2024.
[26] W. Khan, A. Mohd, M. Suaib, M. Ishrat, A. A. Shaikh, and S. M. Faisal, "Residual-enhanced graph convolutional networks with hypersphere mapping for anomaly detection in attributed networks," *Data Sci. Manage.*, Sep. 2024, doi: 10.1016/j.dsm.2024.09.002.
[27] W. Khan, S. Abidin, M. Arif, M. Ishrat, M. Haleem, A. A. Shaikh, N. A. Farooqui, and S. M. Faisal, "Anomalous node detection in attributed social networks using dual variational autoencoder with generative adversarial networks," *Data Sci. Manage.*, vol. 7, no. 2, pp. 89–98, Jun. 2024.
[28] W. Khan, M. Ishrat, M. Haleem, A. N. Khan, M. K. Hasan, and N. A. Farooqui, "An extensive study and review on dark Web threats and detection techniques," in *Advances in Cyberology and the Advent of the Next-Gen Information Revolution*. Hershey, PA, USA: IGI Global, 2023, pp. 202–219.
[29] P. Ryciak, K. Wasielewska, and A. Janicki, "Anomaly detection in log files using selected natural language processing methods," *Appl. Sci.*, vol. 12, no. 10, p. 5089, May 2022.
[30] J. Kim, J.-H. Yun, and H. C. Kim, "Anomaly detection for industrial control systems using sequence-to-sequence neural networks," in *Comput. Security: ESORICS 2019 ESORICS 2019 International Workshops, Cyber-ICPS, SECPRE, SPOSE, and ADIoT*. Luxembourg City, Luxembourg: Springer, 2019, pp. 3–18.
[31] L. G. Mandagondi, "Anomaly detection in log files using machine learning techniques," M.S. thesis, Fac. Comput., Blekinge Inst. Technol., Karlskrona, Sweden, 2021.
[32] J. Wang, C. Zhao, S. He, Y. Gu, O. Alfarraj, and A. Abugabah, "LogUAD: Log unsupervised anomaly detection based on Word2 Vec," *Comput. Syst. Sci. Eng.*, vol. 41, no. 3, pp. 1207–1222, 2022.
[33] X. Li, P. Chen, L. Jing, Z. He, and G. Yu, "SwissLog: Robust and unified deep learning based log anomaly detection for diverse faults," in *Proc. IEEE 31st Int. Symp. Softw. Rel. Eng. (ISSRE)*, Oct. 2020, pp. 92–103.
[34] V.-H. Le and H. Zhang, "Log-based anomaly detection with deep learning: How far are we?" in *Proc. IEEE/ACM 44th Int. Conf. Softw. Eng. (ICSE)*, May 2022, pp. 1356–1367.
[35] X. Qu, L. Yang, K. Guo, L. Ma, M. Sun, M. Ke, and M. Li, "A survey on the development of self-organizing maps for unsupervised intrusion detection," *Mobile Netw. Appl.*, vol. 26, no. 2, pp. 808–829, Apr. 2021.
[36] S. Eltanbouly, M. Bashendy, N. AlNaimi, Z. Chkirbene, and A. Erbad, "Machine learning techniques for network anomaly detection: A survey," in *Proc. IEEE Int. Conf. Inf., IoT, Enabling Technol. (ICIoT)*, Feb. 2020, pp. 156–162.
[37] J. Tian, M. H. Azarian, and M. Pecht, "Anomaly detection using self-organizing maps-based k-nearest neighbor algorithm," in *Proc. PHM Soc. Eur. Conf.* Princeton, NY, USA: Citeseer, 2014, vol. 2, no. 1, pp. 1–9.
[38] R. C. Ripan, I. H. Sarker, S. M. M. Hossain, M. M. Anwar, R. Nowrozy, M. M. Hoque, and M. H. Furhad, "A data-driven heart disease prediction model through K-means clustering-based anomaly detection," *Social Netw. Comput. Sci.*, vol. 2, no. 2, pp. 1–12, Apr. 2021.
[39] A. Bigdeli, A. Maghsoudi, and R. Ghezelbash, "Application of self-organizing map (SOM) and K-means clustering algorithms for portraying geochemical anomaly patterns in moalleman district, NE Iran," *J. Geochem. Explor.*, vol. 233, Feb. 2022, Art. no. 106923.
[40] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in *Proc. ACM SIGOPS 22nd Symp. Operating Syst. Princ.*, Oct. 2009, pp. 117–132.
[41] J.-G. Lou, Q. Fu, S. Yang, Y. Xu, and J. Li, "Mining invariants from console logs for system problem detection," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC)*, 2010, pp. 1–14.

AYESHA AZIZ received the B.S. and M.S. degrees in computer science from the National University of Computer and Emerging Sciences, Islamabad. Her research interests include pattern recognition and deep learning, with a focus on natural language processing. Her job includes developing new algorithms to increase the machine's ability to understand human language, sentiment analysis, and text classification.

KASHIF MUNIR received the Ph.D. degree in computer science from the University of Innsbruck, Austria, in 2009. He conducted a post-doctoral study with IMT, France, from February 2011 to December 2012. He is the author of numerous peer-reviewed conferences and journal articles. His research interests include data analytics, artificial intelligence, cybersecurity, and performance evaluation of distributed systems. He serves as a reviewer for numerous prestigious journals.