Investigating optimal features in log files for anomaly detection using optimization approach

Shivaprakash Ranga, Nageswara Guptha Mohankumar
Department of Computer Science, Sri Venkateshwara College of Engineering, Bengaluru, India

ABSTRACT
Logs have been frequently utilised in different software system administration activities. The number of logs has risen dramatically due to the vast scope and complexity of current software systems. A lot of research has been done on log-based anomaly identification using machine learning approach. In this paper, we proposed an optimization approach to select the optimal features from the logs. This will provide the higher classification accuracy on reduced log files. In order to predict the anomalies three phases are used: i) log representation ii) feature selection and iii) Performance evaluation. The efficacy of the proposed model is evaluated using benchmark datasets such as BlueGene/L (BGL), Thunderbird, spirit and hadoop distributed file system (HDFS) in terms of accuracy, converging ability, train and test accuracy, receiver operating characteristic (ROC) measures, precision, recall and F1-score. The results shows that the feature selection on log files outperforms in terms all the evaluation measures.

1. INTRODUCTION
During the functioning of contemporary systems, a large number of log files are frequently generated. It represents the system’s current status and capture the relevant data of certain system occurrences. They are quite useful in determining the current condition of the system [1]. As a result, system logs have become a key data resource for performance analysis and anomaly identification, as well as a prominent topic of researches in the area. Identifying anomalies is a significant problem that has been explored for decades [2]. To identify abnormalities in various applications such as intrusion detection, system log analysis, Realtime processing of bigdata, fraud detection, medical monitoring application, outlier detection, aviation safety study and several diverse approaches have been developed and employed.

Historic logs allow for investigation process of prior occurrences and provide system administrators with a way to track out the source of issues [3]. Furthermore, logs may aid in the recovery of a non-faulty state, the resetting of wrong transactions, the restoration of data, and the replication of circumstances that result in inaccurate conditions. Because log files may be properly compressed, storing them is often affordable. Furthermore, one of the fundamental drawbacks of forensic log analysis is that vulnerabilities are only discovered after the fact; as a result, contemporary cybersecurity methodologies have shifted from strictly forensic to proactive research [4]. This allows for quicker reactions and, as a result, lowers incident and cyberattack expenses. When it comes to major corporate systems, the amount of daily created log lines can easily reach the millions. Some of the earliest algorithms for detecting abnormalities are statistical anomaly detection approaches. Statistical techniques create a statistical model of the data’s normal behaviour [5]. After
that, a statistical inference test may be used to determine whether or not a given occurrence corresponds to this model. Statistical anomaly detection is carried out using a variety of ways. This comprises strategies that are based on proximity, as well as parametric, non-parametric, and semi-parametric approaches. Machine learning (ML) techniques are becoming more popular as a method of detecting abnormalities [6]. There have been hundreds of presentations of machine learning algorithms during the last several decades. The technique uses a system that can differentiate between typical and aberrant groups. Figure 1 illustrates a classification scheme for anomaly detection based on the training data function used to build the model.

Figure 1. Machine learning approaches for anomaly detection

Several ML-tech based anomaly detection methods have been explored in the literature. In terms of precision, recall, and F1- measure, supervised learning strategies have been shown to perform better than their unsupervised counterparts in experimental settings. As a result, supervised learning is an excellent option. All too frequently, log lines are used to capture system behaviour, including both regular and problematic behaviour. The logs that capture normal conduct and those that record aberrant behaviour differ in some ways [7]. Despite the fact that these technologies have improved performance significantly, they have the following drawbacks in terms of practical application [8]: i) Insufficient interpretability- To be effective, automated log anomaly detection must provide interpretable results, such as which logs are important or which system components are causing problems. Yet, many traditional systems just offer a basic prediction for input and no more information; ii) Weak adaptability-These approaches frequently need the set of different log events to be known ahead of time during feature extraction. However, as contemporary systems continue to add features and enhance their systems, previously undiscovered log events may appear.

In order to overcome the drawbacks, we used the feature selection (FS) concept instead of feature extraction to find the optimal features which affects the log files. FS is widely used as classification task to find the relevant subsets from the datasets [9]. In the real time scenario, the raw log files may contain the irrelevant and irredundant values. It affects the performance of the predictive model. FS is classified into three types: Filter, Wrapper and Embedded [10]. Filter method does not have any learning algorithms associated it the predictive model [11]. The prominent methods are correlation [11], Chi-square test [12], analysis of variance (ANOVA) [13] and many more. The wrapper methods can interact among the features in the dataset [14]. There is a learning algorithm used to find the optimal results. The well-known methods are particle swarm optimization [15], genetic algorithm (GA) [16] and whale optimization [17]. The embedded methods are a hybridization of both filter and wrapper [18]. The least absolute shrinkage and selection operator (LASSO) [19] and ridge regression [20] are the popular embedded methods in the literature. However, FS is used to find the optimal features we have used the metaheuristic (MH) optimization technique for proposed approach.

We highlight a unique optimization technique called the African vulture optimization algorithm (AVOA) created by Benyamin Abdollahzadeh et al. to replicate the eating and movement behaviour of actual African vultures in this work [21]. Beginning with the initialization of the number of vultures, which is specified by the problem space, AVOA is separated into four phases that must be executed in order. The optimal
solution is first sorted into two groups, each of which is characterised by the fitness of all possibilities. The pace at which the vultures get pleased or hungry is calculated in the second stage, and the exploration and exploitation capacities of the selected solution set are evaluated in the third and fourth phases, respectively. Two important aspects influenced our recent work: i) According to the no free lunch principle, no optimization method can solve all optimization problems [22]; ii) To the authors’ knowledge, AVOA has never addressed the FS problem in anomaly detection.

The following are the study’s main inferences: i) In this study, the huge system logs are analysed using the proposed model, which is based on a system analysis of the full log processing procedure (logging, feature selection, anomaly detection, prediction, and evaluation); ii) We put our method to the test on four different kinds of log datasets using the K-Nearest Neighbor classifier, and we utilised the AVOA metaheuristic technique to get the best possible outcomes; iii) The proposed model was tested for its precision, recall, F1-score, accuracy of classification, and capacity to converge. The remaining article is structured as follows: Section 2 deals with the related work with respect to log file anomaly detection. Section 3 presents a background of the AVOA. In section 4 deals with the detailed description of the datasets and the proposed model. Section 5 represents the discussion of the results. Finally, the conclusion and future scope is discussed in section 6.

2. RELATED WORK

Long short-term memory (LSTM) depth model was used by Du et al. [23] to detect anomalies in system logs. Anomalies are uncovered by DeepLog when usual execution log patterns deviate from the model learned from abnormal behaviour. DeepLog also creates model from the fundamental network log, allowing users to quickly diagnose and undertake root cause investigation whenever an anomaly is discovered. DeepLog offers online update/training of its LSTM models by incorporating user feedback, allowing it to adopt and adapt to new execution patterns. DeepLog has surpassed other current log-based anomaly detection algorithms based on standard data mining procedures in comprehensive empirical assessments across big log sets [24].

To identify network intrusion, Timenko et al. employed machine learning’s ensemble learning approach [25]. The different ensemble classifiers are analysed using the university of new south wales-network base 15 (UNSW-NB15) dataset. According to the findings, Bagged tree and GentleBoost have the greatest accuracy and receiver operating characteristic (ROC) values in the given environment and under evaluated settings, whereas RUSBoost has the lowest. To detect the abnormality in system tracking data, Nedelkoski et al. employed multimodal deep learning. Detecting an abnormality in the execution of system components using bimodal distributed tracing data from big cloud infrastructures [26]. It offers an anomaly detection system that combines information about the trace structure with a single modality of data. Leveraging the model’s ability to recreate the path to discover reliant and simultaneous occurrences. The authors conducted extensive comparative study in this area, with an emphasis on traditional ML-based strategies [27]. The following are some of the benefits that deep learning (DL)-based methods offer: improved generalisation to unseen logs, which is typical in modern software systems, and more readily interpretable results, which are crucial for engineers and analysts to take remedial action.

For log-based anomaly identification, Farzad et al. [28] were the first to use autoencoder and isolation forest. The autoencoder is used to extract features, while the isolation forest is used to identify anomalies based on the extracted features. The trained method is able of correctly encoding ordinary log patterns. To address this issue, they created LogRobust, which uses off-the-shelf word vectors to extract the semantic information of log events, which is one of the early research to examine log semantics [29]. Lu et al. were the first to investigate if convolutional neural network (CNN) might be used for log-based anomaly detection [30]. The authors started by creating log occurrences using identifier-based partitioning, which uses buffering or termination to achieve uniform sequence lengths. The authors then presented a logkey2vec embedding method to execute convolution calculations, which need a two-dimensional feature input.

Cao et al. employ machine learning as part of a system to identify anomalies in web application log data. Traditional log analysis, which relies on manual inspection and regular expression matching, has flaws, according to the report. The Hidden Markov Represent is used to model the data and the decision tree is utilised for categorization. It’s also noteworthy that a result is reported in terms of accuracy and false positive rate, allowing for comparisons [31].

Vaaandri et al. established a methodology for identifying abnormalities in system log files that is based on data mining. This approach has the benefit of being able to discover previously unknown error circumstances, as compared to other strategies that do not use pattern recognition. Many additional methods necessitate the use of human specialists to determine patterns for log messages that need to be investigated further [32].
3. AFRICAN VULTURE OPTIMIZATION APPROACH

The history of the AVOA Meat-heuristic (MH) algorithm is discussed here. Benyamin Abdollahzadeh introduced a novel MH strategy dubbed AVOA, which takes cues from mob mentality and binge eating. The four steps of this strategy are i) identifying two sets of ideal solutions (vultures) and updating them at each iteration; ii) forecasting how soon the vultures will be content and hungry; iii) exploring possible solutions; and iv) exploiting them. At the first stage, compute the fitting of all possible options to identify the best vulture among all classes; this optimal choice will be the first class, the second-best answer will be the second category, and all other answers will attempt to approach either the first or second class. In the second phase, the vultures' fullness or hunger rate is determined. The exploratory skills of the third-step solutions are evaluated using the AVOA approach, which mimics vultures' ability to locate food and predict sick or dead animals. As a result, we know there are vultures that have enhanced eyesight and can fly further distances in quest of food. The last stage is an exploit phase, and it comprises of two additional internal stages and two separate strategies that vary with the values of the parameters.

4. IMPLEMENTATION

Blue Gene/L (BGL) [33], Thunderbird (TB), Spirit (SPI), and Hadoop Distributed File System (HDFS) log data (described in Table 1) were used to evaluate our approach. It contains the various range of anomalies in the range between 362793–78360273. The log lines are huge in the spirit dataset whereas in the BGL dataset contains less log lines as compared to all the datasets.

<table>
<thead>
<tr>
<th>Systems</th>
<th>Size</th>
<th>Log lines</th>
<th>Number of anomalies</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGL</td>
<td>1.207 G</td>
<td>4747963</td>
<td>949024</td>
</tr>
<tr>
<td>Thunderbird</td>
<td>27.367 G</td>
<td>211212192</td>
<td>43,087,287</td>
</tr>
<tr>
<td>Spirit</td>
<td>30.28 G</td>
<td>272298969</td>
<td>78360273</td>
</tr>
<tr>
<td>HDFS</td>
<td>1.38 G</td>
<td>11175629</td>
<td>362793</td>
</tr>
</tbody>
</table>

4.1. Evaluation measures

In the field of machine learning, performance metrics like as precision, accuracy, f1-score, and recall have shown their worth for evaluating prediction models. The aforementioned performance metric is represented mathematically as,

\[
\text{Accuracy} = \frac{TN + TP}{TN + TP + FN + FP}\]

\[
\text{Precision} = \frac{TP}{TP + FP}\]

\[
\text{Recall} = \frac{TP}{TP + FN}\]

\[
f1 - score = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}\]

4.2. Proposed approach

This section deals with the detailed description about the workflow of the proposed model. This section contains three phases: i) Log representation ii) Feature selection iii) Performance evaluation. The detailed process of each stage is represented in the below subsection 4.2.1 to 4.2.3. Each stage contains the process flow which is essential for the suggested approach which is shown in the Figure 2.

4.2.1. Log representation

To begin, software systems often send logs to the system console or to preset log files to record operational status. A log is a line of semi-structured text that is generated by a logging statement in the program's source code. Large-scale, dispersed systems often collect these logs regularly. The availability of log data has allowed for a variety of log analysis operations, such as anomaly discovery and fault localisation. The vast amount of gathered logs, on the other hand, is overloading the present troubleshooting mechanism. The lack of labelled data also makes analysing logs challenging. In second step, raw logs are typically semi-organized after collecting and must be processed into a proper manner for subsequent analysis. This is known
as log parsing. Log parsing aims to distinguish between the constant/static and variable/dynamic parts of a raw log line. The constant component is also known as a log event, log template, or log key.

### 4.2.2. Feature selection

The parsed log file is used as input in this step. The standard Min-Max normalization is used for data normalization. The feature selection is performed using the recent MH algorithm AVOA. Since it is developed recently and outperforms in different benchmark functions, we used this for feature selection. The standard sigmoid function is used to classify the data sample and check for algorithmic success. The suggested model is evaluated by using the k-nearest neighbors (KNN), support vector machine (SVM) technique to find significant features. The data set is divided into training and testing halves so that the 10-fold cross validation (CV) method may be used. The risk of the prediction model over-fitting is reduced by ten times by using a 10-fold CV.

### 4.2.3. Performance evaluation

Anomaly detection, or identifying aberrant log occurrences, may be done using the log characteristics created in the previous step. Many conventional ML-based anomaly detectors use the log feature count to forecast the entire log sequence is anomalous. Finally, the performance is evaluated based on the classical evaluation measures such as precision, recall, f1-score and accuracy.

![Figure 2. Proposed model](image)

5. **RESULTS AND DISCUSSION**

The proposed model is run using the python environment and 100 epochs for each dataset. Figure 3 shows the error rate of the prediction model throughout the operation. The decrease in error rate as the model progresses through each aeon reveals the suggested model's potential to converge to global minima. The ability of the developed approach to predict is assessed using a classifier such as KNN. The BGL dataset starts converging after 25 epochs whereas other three dataset starts converging after 40 epochs. So, the average model converging ability is 40 epochs. Furthermore, HDFS datasets is not shown the significant performance than the other datasets. It shows the model is not fit into overfitting. The suggested model's ROC curve is shown in Figure 4. The suggested technique may assign a greater probability to a randomly picked genuine positive sample than a negative sample on average, based on the higher area under the curve (AUC) value for datasets.

*Investigating optimal features in log files for anomaly detection using optimization ... (Shivaparaksh Ranga)*
The KNN classifier is used to verify the component selection. The suggested model's larger ROC curve demonstrates that the characteristics it chose can give substantial confidence in knowledge discovery and decision making. Furthermore, a smaller collection of chosen characteristics can produce more accurate results than the whole number of features in the input data. Figure 5 shows a prediction capability of KNN classifier. The suggested approach yields accuracy values between [0.66, 0.95] and [0.60,0.95] during training and testing. In the early phases, the suggested model has a higher propensity to enhance accuracy. The gap between the trading and testing accuracy should be less. In HDFS dataset, the gap is very high as compared to other datasets [range:0.675 to .825]. To validate the feature subsets, Table 2 summarise and compare the performance analyses of the predictive models for the selected feature subsets. From the table we can witness that proposed model scored the better classification accuracy in all the datasets.

![Figure 3. Converging ability of proposed model](image)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGL</td>
<td>0.82</td>
<td>0.94</td>
<td>0.92</td>
<td>0.96</td>
</tr>
<tr>
<td>Thunderbird</td>
<td>0.86</td>
<td>0.83</td>
<td>0.92</td>
<td>0.99</td>
</tr>
<tr>
<td>Spirit</td>
<td>0.89</td>
<td>0.92</td>
<td>0.96</td>
<td>0.94</td>
</tr>
<tr>
<td>HDFS</td>
<td>0.75</td>
<td>0.91</td>
<td>0.68</td>
<td>0.89</td>
</tr>
</tbody>
</table>

6. CONCLUSION

This article proposed a new model to detect the anomalies from the log files using features selection with recent AVOA optimization approach. Conventionally, feature extraction is majorly used in the literature with the standard ML classifiers. However, it shows the better performance, the proposed model is used select the optimal features using AVOA optimization approach to detect the anomalies in the log files. The performance of the suggested model is evaluated using four benchmark datasets in terms of converging ability, training and testing accuracy, precision, recall and F1-score. From the results, Except the HDFS dataset, the remaining datasets outperforms in all the evaluation measures. The suggested algorithm can be implemented using different ML classifiers and deep learning environment will be the part of the future work.
APPENDIX

Figure 4. AUC performance of proposed model

Figure 5. Training and testing accuracy of proposed model
REFERENCES


Int J Artif Intell
ISSN: 2252-8938


BIOGRAPHIES OF AUTHORS

Shivaprakash Ranga holds a Postgraduate in computer science and engineering at SIT, Tumkur. Currently he is pursuing Ph.D under VTU Belagavi in the domain of anomaly detection. His research interests are Optimization, Metaheuristic approaches and Networks. He can be contacted at email: shivaprakashranga@gmail.com.

Nageswara Guptha received the Ph.D from Anna University, Chennai. Currently he is working as a principal at SVCE Bengaluru. He is having more than 15 SCI research articles in reputed journals. His research interests are Human machine interaction, Artificial intelligence, Robotic process automation. He can be contacted at email: mnguptha@yahoo.com.