Abstract

Enterprise systems often produce a large volume of logs to record runtime status and events. Anomaly detection from system logs is crucial for service management and system maintenance. Most existing log-based anomaly detection methods use log event *indexes* parsed from log data to detect anomalies. Those methods cannot handle unseen log templates and lead to inaccurate anomaly detection. Some recent studies focused on the *semantics* of log templates but ignored the information of *parameter values*. Therefore, their approaches failed to address the abnormal logs caused by parameter values.

In this project, we propose a comparative study different ML and DL approaches based on different set of features utilizing different pre-processing approaches like Pre-Trained BERT extracted Features and Sentenced Tokenized Features. We are also using SMOTE oversampling method to tackle the problem of high data imbalance of classes

The ML algorithms that are mainly focused here are One-Class SVM (OC-SVM), Isolation Forrest (IF), Local Outlier Factor (LOF), Decision Tree (DT), Naïve Bayes (NB), Quadratic Discriminative Analysis (QDA) and so on. The purpose of these ML algorithms is to show the difference in Supervised Learning and Unsupervised Learning, as any IDS is always exposed to uncertainly of new attacks every day. But as the size of dataset increases, we suffer from improper classification and higher time complexity in case of ML algorithms.

Thus, we are extending our approached to DL approaches as well using ANN, CNN, RNN, Bi-RNN, LSTM, Bi-LSTM, GRU and Bi-GRU, where we see significant improvements compared to ML models without the help of SMOTE itself. Which shows that these DL models are able to capture the semantic information in both log template sequence and parameter values and handle various types of anomalies.

We evaluated our proposed methods on the dataset - HDFS (Hadoop distributed file system log). Our experimental results demonstrate a deep comparative case study analysis where each model outperforms the other and their significance in general concept of Anomaly Detection and specific to single dataset. We also assess the robustness of our proposed model on unstable log data.

***Index Terms*—Log data analysis, Anomaly Detection, Intrusion Detection System, Machine Learning, Deep Learning, Transformers.**

Introduction

Modern computer systems have become increasingly complex as systems grow in both size and functionality [1]. Anomaly detection has become an essential task to build a trustworthy computer system. A single anomaly issue can impact millions of users’ experience and service [2]. Accurate and effective anomaly detection model can reduce the damage from anomalies [3], which is crucial for service management and system maintenance. Logs are widely used to record significant events and system status in an operating system or other software systems. Since system logs contain noteworthy events and runtime status, they are one of the most important data sources for anomaly detection and system monitoring [4].

Existing log-based anomaly detection methods can be broadly classified into two categories: log event indexes-based approaches. Log event indexes-based approaches first extract log events from log messages and then convert log events into indexes feature spaces. These methods do not attempt to utilize semantic information in log messages. Thus, they cannot handle unseen log templates and suffer from inaccurate log parsing. Log template semantics-based approaches model a log stream as a natural language sequence. They convert log templates into vectors using word vectors and then train their model based on those vectors. Existing approaches are limited to the semantics of log templates but may miss the key values that can be used for anomaly detection. We observed that some anomalies are not shown as a deviation from a normal log template sequence but as an abnormal parameter value. Therefore, the values of some specific parameters can be essential factors to be considered in log-based anomaly detection models.

In this project, we propose various ML approaches that are based on Anomaly Detection in Intrusion Detection Systems, a comparative study analysis of these models based on their performance respectively. For feature engineering the text log data we are using the Sentence Tokenizer (which uses the concept of TF-IDF) and Pretrained-Bert feature extraction (to reduce the sparse dimensionality of the data sample space as well as to capture relevant word-level information as a generalized embedding feature out of it). Initially, to overcome to problem of huge data class imbalance, we are using oversampling methods like SMOTE, later we are able prove the un-necessity of it with DL models as they are able to recognize the patterns of detecting anomalies in dataset based on aspect of component mappings.

We evaluated our proposed methods only on one log dataset - HDFS [5], as we want device models to learn on highly class-imbalanced dataset as in real world the ratio of number of attack events with respect number of normal events is very low in most of the cases. Experimental results show gradual improvements in different approaches that outperforms other existing log-based anomaly detection methods on stable log data sets. We also investigate the impacts of hyperparameter tuning in ML models to optimize to best outcome possible and in case of DL models, we explore the impacts of different window size and number of layer size. In order to evaluate the robustness of our model, we also present the performance of these models on unstable log data from two aspects: unseen log events and unstable log sequences.

The key contributions of this article can be summarized as follows:

* **MS-1: Bert Model Trainer**

Features are extracted using self defined preprocess layer with Bert-base-uncased transformer and trained with distilbert-base-uncased Bert model.

**Result:** Due to highly imbalance nature of the dataset, the model cannot classify any anomalies. All the test long sequences are classified as “normal”.

* **MS-2: Bert Feature Extraction**

In this approach, Bert - hugging face (small\_bert\_uncased) is used for feature extraction and an ANN (dense layer/neural networks) is trained with those features.

**Result**: Model cannot classify any anomalies due to highly imbalanced data

* **MS-3: Oversampling Features**

Feature extracted is done using transformer small\_bert\_uncased. Oversampling method - SMOTE is applied to features to tackle the imbalance problem in the dataset. Then, we train those SMOTE features into various ML algorithms.

* K-Nearest-Neighbors,
* Quadratic Discriminant Analysis,
* Support Vector Machine,
* One class SVM
* Local-Outlier-Factor,
* Isolation Forest,
* Linear Regression,
* Stochastic Gradient Descent,
* Gaussian Naïve Bayes,
* Decision Tree,
* Random Forest,
* Light-Grade-Boost-Machine

**Result**: Since the imbalance nature of the dataset has been treated with SMOTE, good accuracy is achieved.

* **MS-4: Extending ML Implementations on Entire data**

Features extracted for the entire dataset using bert\_cased in hugging face package. Applying ML algorithms did previous on these features also with SMOTE and without SMOTE.

**Results:** Some of the ML Algorithms were not trainable (One Class SVM, SVC, Local Outlier Factor, Gradient Boosting, Random Forest, Ada Boost, Gaussian Process) due to exponential time complexity. (One-Class and SVM has a training time that scales quadratically with the number of samples, or worse.) Apart from these, the rest all ML algorithms gave the similar results

* **MS-5: DL and ML approaches on Text Tokenized vectors**

Sentence to sequence Tokenizer is applied directly on the log data. Then this tokenized vectors and SMOTE (tokenized vectors) is trained into various DL methods (separate models) – RNN, Bi-RNN, LSTM, Bi-LSTM, GRU, Bi-GRU, CNN.

**Result:** Better promising results of without-smote fitted model than with-smote fitted model. This gives a better way to avoid oversampling, but most of the model is supervised learning which we are trying to avoid as real time labelling manually of log is impossible. (that’s why we started basic anomaly detection unsupervised ML approaches on a small portion of data – One-Class SVM and Isolation Forrest Especially)

* **MS-6: DL approaches on Bert Features**

Basically, we are implementing the same approach as above, except that we giving the Bert-feature extracted from Bert-cased (hugging face) previously as input into the same DL methods (separate models for with-smote fit and without-smote fit) for comparison purposes.

**Result**: The results of DL models on Sentence Tokenized input vectors done previously were significantly better than this approach.

The rest of this project is organized as follows. We introduce the background of our work in Section- II. The model methodologies involved are described in Section-III. Section-IV describes the experimental settings, and result comparison that we evaluated based on the performance of various models on direct features of log data and oversampled features of log data. Case studies of Related works is introduced in Section-V. Finally, in Section-VI, we conclude our work and state possible future work.

Respective references used:

[1] Y. Tan and X. Gu, “On predictability of system anomalies in real world,” in *Proc. IEEE Int. Symp. Model. Anal. Simulat. Comput. Telecommun. Syst.*. 2010, pp. 133–140.

@inproceedings{tan2010predictability,

title={On predictability of system anomalies in real world},

author={Tan, Yongmin and Gu, Xiaohui},

booktitle={2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems},

pages={133--140},

year={2010},

organization={IEEE}

}

[2] S. Huang, C. Fung, K. Wang, P. Pei, Z. Luan, and D. Qian, “Using recurrent neural networks toward black-box system anomaly prediction,” in *Proc. IEEE/ACM 24th Int. Symp. Qual. Service (IWQoS)*. 2016, pp. 1–10.2]

@inproceedings{huang2016using,

title={Using recurrent neural networks toward black-box system anomaly prediction},

author={Huang, Shaohan and Fung, Carol and Wang, Kui and Pei, Polo and Luan, Zhongzhi and Qian, Depei},

booktitle={2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)},

pages={1--10},

year={2016},

organization={IEEE}

}

[3] S. Huang *et al.*, “Arena: Adaptive real-time update anomaly prediction in cloud systems,” in *Proc. 13th Int. Conf. Netw. Service Manag. (CNSM)*. 2017, pp. 1–9.

@inproceedings{huang2017arena,

title={Arena: Adaptive real-time update anomaly prediction in cloud systems},

author={Huang, Shaohan and Fung, Carol and Liu, Chang and Zhang, Shupeng and Wei, Guang and Luan, Zhongzhi and Qian, Depei},

booktitle={2017 13th International Conference on Network and Service Management (CNSM)},

pages={1--9},

year={2017},

organization={IEEE}

}

[4] 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)*. 2016, pp. 102–111.

@inproceedings{lin2016log,

title={Log clustering based problem identification for online service systems},

author={Lin, Qingwei and Zhang, Hongyu and Lou, Jian-Guang and Zhang, Yu and Chen, Xuewei},

booktitle={2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C)},

pages={102--111},

year={2016},

organization={IEEE}

}

[5] 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. Oper. Syst. Principles*, 2009, pp. 117–132.

@inproceedings{xu2009detecting,

title={Detecting large-scale system problems by mining console logs},

author={Xu, Wei and Huang, Ling and Fox, Armando and Patterson, David and Jordan, Michael I},

booktitle={Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles},

pages={117--132},

year={2009}

}