Sprelog: Log-Based Anomaly Detection with Self-matching Networks and Pre-trained Models

Haitian Yang[1,2](✉), Xuan Zhao[3], Degang Sun[2](✉), Yan Wang[1], and Weiqing Huang[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{yanghaitian,sundegang,wangyan,huangweiqing}@iie.ac.cn

[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

[3] York University, Ontario, Canada
xuanzhao@eecs.yorku.ca

Abstract. With the development of software systems, log has become more and more important in system maintenance. During the past few years, log-based anomaly detection has attracted much attention. We propose a novel log-based anomaly detection model, called Sprelog, which captures "inconsistent" information during the evolution of log messages by exploring word-word interactions features. Firstly, we compute the interactive information of each word-word pair in the input log sequence, constructing self-matching attention vectors. Next, we use these self-matching attention vectors to manage the log sequence and construct the representation vectors. Hence, the log sequence can be matched word-by-word, adapting to the evolution of log messages. In addition, we combine pre-trained models in our proposed network to generate the higher-level semantic component information. More importantly, we use a low-rank bi-linear pooling approach to connect inconsistent and compositional information, thus our model can reduce potential information redundancy without weakening the discriminative ability. Experiment results on publicly available datasets demonstrate that our model significantly outperforms extant baselines on standard evaluation metrics, including precision, recall, F1 score and accuracy.

Keywords: Log analysis · Anomaly detection · Self-matching networks · Pre-trained models

1 Introduction

With the continuous development of software systems, the scale of systems becomes larger and larger, hence it is almost impossible to detect system anomalies manually. During the past decade, we witness the introduction of many automated log-based approaches [1,2]. These methods often apply useful information from logs to detect system anomalies. We observe that some methods adopt data mining and machine learning techniques to analyze log data and detect the occurrence of system anomalies. For example, Xu, et al. [2] treated the log-based anomaly detection task as an unsupervised learning problem and utilized Principal Component Analysis (PCA) to detect anomalies.

However, most of these log-based anomaly detection approaches are not sufficiently robust in the real-world implementation. Therefore, in this paper, we propose a method named Sprelog - a novel log-based anomaly detection approach, which can achieve accurate and robust anomaly detection on real-world, ever-changing, and noisy log data. More importantly, we evaluate the proposed approach using the public log data collected from Hadoop. Specially, we reorganize the injection ratios of the Hadoop log data to evaluate the effectiveness of the proposed approach. Our experimental results demonstrate that when we increase the injection rate from 5% to 20%, the F1-score merely decreases from 0.97 to 0.94. Hence, the experiment not only shows that our approach can effectively detect anomalies of the online service system with the ever-changing and noisy log data, but also, more importatnly, very robust.

We summarize the main contributions of this paper as follows:

(1) We aim to solve the task of unstable log anomaly detection. Specifically, we adopt a self-matching network that captures "inconsistent" information during the evolution of log messages by exploring word-word interaction features. This self-matching network assists our method to manage both the instability during the evolution of log messages and the noise in the log data.
(2) We adopt two semantic representations, the local static-based word embedding (word2vec [3] or glove [4]) and the global dynamic word embedding (ReBERTa [5]). Also, we apply the low-rank bi-linear pooling approach to integrate these two semantic representations effectively.
(3) We have evaluated Sprelog using two public datasets. The results confirmed the effectiveness of our approach.

2 The Proposed Model

2.1 Task Description

In this research, the log-based anomaly detection task can be described as a tuple of three elements (S, I, y), where $S = [s^1, s^2, \ldots, s^g]$ represents the log event whose length is g. I denotes the current log message task ID and $y \in Y$ conveys the anomaly detection status of the log. More detailed, Y={Yes, No} which Yes represents that log is normal, and No means that the log is abnormal. Generally, the function of our model Sprelog is to assign a label to each task ID based on the conditional probability $Pr(y|S, I)$ according to the given set of $\{S, I\}$ to solve the log-based anomaly detection task.

2.2 Overview of the Proposed Model

In this section, we describe our proposed model in detail. Model architecture is depicted in Fig. 1. First, we process the unstructured original log data into the structured log events by log parsing, and then convert each word of log events into a vector by word embedding. Then, to transform log events into fixed-dimensional log sequence semantic vectors, we combine word-to-word interactions vectorization with Reberta-based semantic feature vectorization. Finally, to further synthesize the vectorized log event sequences, we use Low-rank Bi-linear Pooling to integrate the log sequence information for the log-based anomaly detection task.

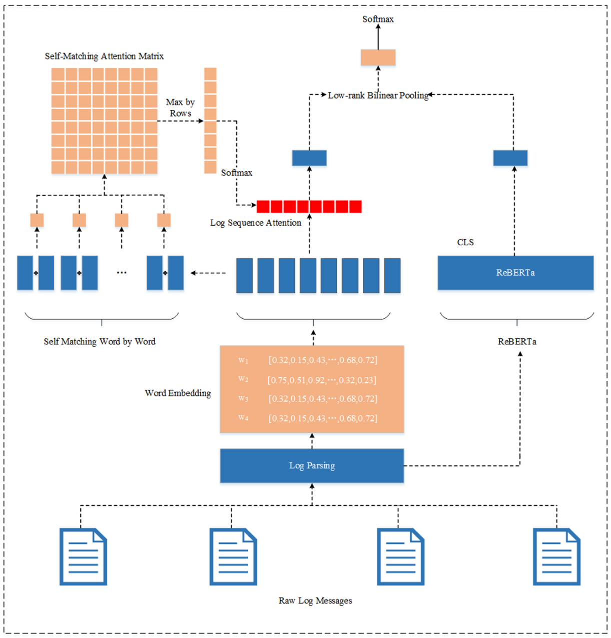
Softmax
Self-Matching Attention Matrix
Max by Rows
Low-rank Bilinear Pooling
Softmax
Log Sequence Attention
CLS
ReBERTa
Self Matching Word by Word
ReBERTa
Word Embedding
w1 [0.32,0.15,0.43,⋯,0.68,0.72]
w2 [0.75,0.51,0.92,⋯,0.32,0.23]
w3 [0.32,0.15,0.43,⋯,0.68,0.72]
w4 [0.32,0.15,0.43,⋯,0.68,0.72]
Log Parsing
Raw Log Messages

Fig. 1. Overview of our proposed Sprelog model.

2.3 The Representation of the Local Semantic Log Sequence

In this section, we acquire the representation of the local semantic information. Firstly, we construct word-to-word interactions features from parsed logs. Secondly, we combine the word information with the log sequence local semantic information via our self-matching networks [6]. After the above-mentioned steps, we fix the number of the word-to-word interactions vector of the whole log sequence.

It is worth noting that the self-matching network can generate a attended feature vector for the input log sequence: $f_a = S \cdot a$, where $a \in R^n$ is the self-matched attention vector. Literature [7] provides study to demonstrate the effectiveness of semantic incongruity as a predictor for log-based anomaly detection. Hence, attention vector a can be designed to capture log sequence incongruity.

In this paper inspired came from "co-attention" network proposed by Lu et al. to address the Visual Question Answering (VQA) task [8]. They introduce an affinity matrix C to attend input picture feature map V and text question representation Q. C is calculated by:

$$\mathrm{C} = \tanh\left(\mathrm{Q} \cdot \mathrm{W}_a \cdot \mathrm{V}\right) \tag{1}$$

where W_a contains attention weights.

A joint activation approach (e.g. maximize by rows and columns) is adopted to adjust attention weights for V and Q simultaneously. We modify this approach by introducing a weight matrix between word-to-word pair to improve the ability of capturing joint information of words.

Given a word pair (e_i, e_j), the the joint feature vector $w_{i,j}$ is computed:

$$w_{i,j} = \tanh\left(\mathbf{e}_i \cdot \mathbf{M}_{i,j} \cdot \mathbf{e}_j^{\mathrm{T}}\right) \tag{2}$$

where e_i and e_j are word embeddings for i and j [9], $w_{i,j} \in R$ measuring the joint information between word i and word j , and $M_{i,j} \in R^{k \times k}$ is a parameter matrix.

The self-matching information matrix W based on all joint information $w_{i,j}$, $i.j \in (1, 2, \ldots, n)$ is computed:

$$\mathbf{W} = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1} & w_{2,2} & \cdots & w_{n,n} \end{pmatrix} \tag{3}$$

A maximization activation approach is applied to calculate the self-matched attention vector a. We first calculate an intermediate vector $m \in R^n$, by maximizing elements in W by rows.

$$m_i = \max\left(w_{i,1}, w_{i,2}, \ldots, w_{i,n}\right), \forall i \in (1, 2, \cdots, n) \tag{4}$$

Then, we input m into a standard softmax function to calculate a: $a = softmax(m)$. Softmax function is adopted for the purpose of normalization.

2.4 The Representation of the Global Semantic Log Sequence

In this section, we use Reberta to obtain global semantic features in log sequence. Specifically, the Reberta applied contains token embeddings, segment embeddings, position embeddings, and Transformer with multi-head attention [5] as the encoder layer to obtain H-dimensional encoded log sequence containing contextual information. To achieve these representations, we first use $S = [s^1, s^2, \ldots, S^g]$ to represent the log event whose length is g. In the pre-trained method Reberta, we obtain global contextual log sequence representation f_l by applying the fixed-dimensional output vector as the semantic representation of the entire log event. The output vector is symbolized as the [cls]. The detailed equation is demonstrated as follows:

$$f_l = Reberta_{cls}(S) \tag{5}$$

2.5 Low-Rank Bilinear Pooling

After the previous steps, two log sequences feature vectors are acquired: f_a generated by the self-matching network and f_l generated by the Reberta encoder. Here, we concatenate these two feature vectors for the final prediction. We employ a Low-rank Bilinear Pooling (LRBP) method based on Hadamard product to reduce the dimension of the final input vector to control the potential information redundancy without reducing feature vector's discriminative power [10].

In this work, we follow the concept of LRBP to pool information from two input feature vectors: $f_a \in R^k$ and $f_l \in R^d$. The final projection feature vector for the input log sequence is calculated:

$$f = U^T \cdot f_a \circ V^T \cdot f_l + g \tag{6}$$

where $\circ$ represents the Hadamard product, $f \in R^c$, $U \in R^{k \times c}$, and $V \in R^{d \times c}$ are parameters that need to be learned. $g \in R^c$ is bias, c, k and d are hyperparameters.

f is the final feature vector for the inputted log sequence. We input f into a standard softmax classification layer to make the log-based anomaly detection prediction:

$$p_i = softmax(W_f \cdot f + b) \tag{7}$$

where $p_i \in R^2$ represents whether the input log event sequence is normal or not, and $W_f \in R^{2 \times c}$, $b \in R^2$ are parameters to be learned.

2.6 Training Objective

The lost function of this Log-Based Anomaly Detection classification task is a standard cross-entropy:

$$J(\theta) = -\sum_{i=1}^{N} [y_i \cdot \log p_i + (1 - y_i) \cdot \log (1 - p_i)] + \lambda \cdot R \tag{8}$$

where N is the size of training dataset, y_i is the true label for log sequence i. $\theta = \{M_{i,j}, U, V, g, W_f, b\}$ are model parameters. $R = \|\theta\|_{L2}$ regularization term, λ is a hyperparameter measuring the weight of regularization term.

3 Experimental Setup

3.1 Dataset and Hyper Parameters

We evaluate our proposed Sprelog on two datasets, including the original HDFS datasets [11] and the synthetic unstable HDFS datasets. To prepare for the synthetic datasets, we randomly collect 51,000 log sequences from the original HDFS datasets consisting of 50,000 normal and 1,000 anomaly sequences. We inject the unstable log data into it and create two testing sets: NewTesting 1 and 2, which contain injected unstable log events and unstable log sequences, respectively. The details of the two datasets are show Table 1:

Table 1. The synthetic HDFS datasets

Set	Unstable event	Unstable seq.	Normal	Anomaly	Total
Training	No	No	6,000	6,000	12,000
NewTesting1	Yes	No	50,000	1,000	51,000
NewTesting2	No	Yes	50,000	1,000	51,000

We fix all the hyper-parameters applied to our model. Specifically, We train our networks by stochastic gradient descent with the learning rate of 0.1, the momentum of 0.9, the weight decay of 0.0005, the dropout ratio of 0.5, and the gradient clipping of 0.1. The training batch size for all datasets is tuned amongst 64, 128, 256. The L2 regularization is set to 10^{-5} for the original HDFS datasets, and 10^{-3} for synthetic unstable HDFS datasets.

3.2 Results and Analysis

To analyze the effectiveness of our model, we take some current competitive methods as baselines on the above two datasets to compare the performance of Sprelog with other models. The results are demonstrated as follows.

Table 2. Experiment results on synthetic HDFS dataset of untable log sequences (the NewTesting1 set)

Injection ratio	Metric	LR [10]	SVM [12]	IM [13]	PCA [11]	LogAnomaly [14]	PLELog [15]	LogRobust [7]	Sprelog
5%	Precision	0.25	0.36	0.78	0.90	0.97	0.91	1.00	0.99
	Recall	0.92	0.96	0.56	0.66	0.89	0.78	0.91	**0.95**
	F1-score	0.39	0.53	0.65	0.76	0.93	0.84	0.95	**0.97**
10%	Precision	0.18	0.11	0.88	0.90	0.86	0.82	0.89	**0.92**
	Recall	0.95	0.89	0.40	0.64	0.94	0.89	1.00	0.96
	F1-score	0.30	0.20	0.56	0.74	0.90	0.85	0.94	0.94
15%	Precision	0.08	0.11	0.84	0.82	0.82	0.78	0.86	**0.90**
	Recall	0.85	0.90	0.41	0.42	0.97	0.85	0.99	0.99
	F1-score	0.14	0.20	0.55	0.55	0.89	0.81	0.92	**0.94**
20%	Precision	0.06	0.09	0.82	0.82	0.92	0.78	0.99	0.96
	Recall	0.87	0.89	0.43	0.41	0.88	0.75	0.81	**0.92**
	F1-score	0.11	0.16	0.56	0.54	0.90	0.76	0.89	**0.94**

(1) Experiments on the Synthetic HDFS Dataset
Our work focuses on the anomaly detection problem in unstable logs. To further prove that the model proposed in this paper can effectively solve the anomaly detection issue in unstable logs, we conduct two groups of experiments. First, for the unstable log events, our model is trained on the original HDFS log datasets and tested on the synthetic unstable log event datasets (NewTesting1). The comparison results on the NewTesting1 set are shown in Table 2. We can note that as the proportion of unstable log injection increases, the performance of the five baselines continues to decline. The f1 value of our model Sprelog is around 0.94 based on different injection rates, which strongly proves that our method has high robustness and can effectively solve the anomaly detection issue in unstable logs. The main reason is that, during the log presentation process, our model applies the pre-trained model and semantic vectors, projecting the logs into higher dimensions, thus higher-level semantic information can be obtained.

Next, we conduct another group of experiments on the unstable log data. Specifically, our model is trained on the original HDFS log datasets and tested on the synthetic unstable log sequence datasets (NewTesting2). The experimental results are shown in Table 3.

Table 3. Experiment results on synthetic HDFS dataset of untable log sequences (the NewTesting2 set)

Injection ratio	Metric	LR [10]	SVM [12]	IM [13]	PCA [11]	LogAnomaly [14]	PLELog [15]	LogRobust [7]	Sprelog
5%	Precision	0.97	0.94	0.03	0.95	0.98	0.93	0.99	0.98
	Recall	0.85	0.98	0.84	0.65	0.92	0.81	0.93	**0.96**
	F1-score	0.96	0.96	0.06	0.77	0.95	0.87	0.96	**0.97**
10%	Precision	0.44	0.77	0.03	0.96	0.92	0.96	0.94	**0.96**
	Recall	0.93	0.97	0.97	0.63	0.96	0.76	0.99	0.97
	F1-score	0.61	0.86	0.06	0.76	0.92	0.82	0.96	0.96
15%	Precision	0.09	0.21	0.02	0.83	0.95	0.83	0.98	0.98
	Recall	0.88	0.93	0.97	0.39	0.92	0.74	0.91	**0.95**
	F1-score	0.17	0.33	0.04	0.53	0.93	0.78	0.94	**0.96**
20%	Precision	0.07	0.07	0.01	0.87	0.90	0.76	0.92	**0.95**
	Recall	0.82	0.86	0.98	0.37	0.96	0.68	0.97	**0.98**
	F1-score	0.12	0.14	0.03	0.52	0.93	0.72	0.95	**0.96**

We can observe that LogAnomaly [14], LogRobust [7], and our proposed Sprelog are semantic-based models. When the injection ratio of log sequences increases, these models performance reduces slower compared to other methods. In particular, our proposed method Sprelog still remain significant performance, when the injection ratio of log sequences increases, log sequences suffer from missed, duplicated, or shuffled problems. The f1 value of the model Sprelog is basically around 0.96. The reason is that our model uses a self-matching network in the log sequence representation. The self-matching network explores the contextual information embedded in the log sequence and learns the different importance of log events through the attention mechanism. Therefore, it makes our model robust to small changes in the sequence.

4 Conclusion

Engineers can use logs (for example, system log messages) to investigate the anomalies. However, due to the continuous evolution of log statements and the emergence of processing log noise, the current log anomaly analysis model are not robust enough. To overcome this issue, we propose a new log-based anomaly detection method - Sprelog. Which can capture the inconsistency in the evolution of log sequences and higher-level semantic component information. Experiment results on publicly available datasets, our proposed Sprelog model achieves state-out-of-art performance, outperforming the most advanced log-based anomaly detection models that exist. In the future, our research group would like to improve the computing speed of Sprelog to further level up the performance of our solution.

References

1. He, P., Zhu, J., He, S., Li, J., Lyu, M.R.: Towards automated log parsing for large-scale log data analysis. IEEE Trans. Dependable Secure Comput. **15**(6), 931–944 (2017)
2. Xu, W., Huang, L., Fox, A., Patterson, D., Jordan, M.I.: Detecting large-scale system problems by mining console logs. In: Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles, pp. 117–132 (2009)
3. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
4. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, pp. 1532–1543 (2014)
5. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., et al.: Roberta: a robustly optimized Bert pretraining approach. Corr abs/1907.11692 (2019)
6. Park, C., Song, H., Lee, C.: S3-net: SRU-based sentence and self-matching networks for machine reading comprehension. TALLIP **19**(3), 1–14 (2020)
7. Zhang, X., Li, Z., Chen, J., He, X., et al.: Robust log-based anomaly detection on unstable log data, pp. 807–817, August 2019
8. Lu, J., Yang, J., Batra, D., Parikh, D.: Hierarchical question-image co-attention for visual question answering. In NIPS, pp. 289–297 (2016)
9. Tay, Y., Tuan, L.A., Hui, S., Su, J.: Reasoning with sarcasm by reading in-between. pp. 1010–1020, January 2018
10. Kim, J.H., On, K., Kim, J., Ha, J.W., Zhang, B.T.: Hadamard product for low-rank bilinear pooling. arXiv:1610.04325 (2016)
11. Xu, W., Huang, L., Fox, A., Patterson, D., Jordan, M.: Detecting large-scale system problems by mining console logs, pp. 37–46, January 2010
12. Zhang, Y., Sivasubramaniam, A.: Failure prediction in ibm bluegene/l event logs. In: ISPA, pp. 1–5 (2008)
13. Lou, J.G., Fu, Q., Yang, S., Xu, Y., Li, J.: Mining invariants from console logs for system problem detection (2010)
14. Meng, W., Liu, Y., Zhu, Y., et al.: Loganomaly: unsupervised detection of sequential and quantitative anomalies in unstructured logs. In: IJCAI, vol. 19, pp. 4739–4745 (2019)
15. Yang, L., Chen, J., Wang, Z.: Semi-supervised log-based anomaly detection via probabilistic label estimation. In: ICSE, pp. 1448–1460. IEEE (2021)