NETWORK ANOMALY DETECTION BASED ON KEYWORD EMBEDDING LOG

Yong Song[1], Zhiwei Yan[2], Yukun Qin[2], Yuchen Xie[1], Xiaozhou Ye[1], Ye Ouyang[3]
[1] Telco Artificial Intelligence Labs, AsiaInfo Technologies (China), Beijing, China,
[2] Telco Artificial Intelligence Labs, AsiaInfo Technologies (Nanjing), Nanjing, China,
[3] AsiaInfo Technologies (China, Guangzhou), Guangzhou, China

NOTE: Corresponding author: Zhiwei Yan, yanzw@asiainfo.com

Abstract *–Log anomaly detection is an important and challenging task in the field of Artificial Intelligence for IT Operations (AIOps). Logs that record important runtime information are widely used for troubleshooting purposes. There have been many studies that use log data to construct deep learning methods for detecting system anomalies, which are usually based on log parsing. However, they ignore the effect of keywords that are promising for system status analysis. Here, we propose KELog (Keyword Embedding Log), a novel log anomaly detection approach that utilizes keyword information. We build a keyword library by keyword information extraction and fuse them into log representations. In this way, KELog can raise the reliability of anomaly detection. The experimental results on a real-world log dataset of a communications operator show that the F1 score of our proposed KELog method achieves a maximum increase of 0.341 compared with the commonly used machine learning algorithms (PCA, SVM, Invaiant Mining) and a maximum increase of 0.039 compared with deep learning algorithms (DeepLog, LogBERT) respectively. In 2021, ITU launched the second ITU AI/ML in 5G Challenge. We used KELog to participate in the thematic track of the Artificial Intelligence Innovation and Application Competition in the China Division, and won first place with a full F1 score.*

Keywords – Artificial intelligence for IT operations (AIOps), deep learning, keyword information, log anomaly detection, transformer

1. INTRODUCTION

High available and reliable systems are essential for sustainable development of communications operators. Inevitable anomalies accompany the increasing complexity and scale of systems. A small problem in the system may cause performance degradation, data corruption, and even significant losses of customers and revenue. Thus, anomaly detection is necessary to maintain the stability of the communications operator's system.

Large-scale systems generate a large amount of log data every day to record important events during the operation of the system, and to track and monitor the running status of the computer. The operations and maintenance engineers can utilize the log data to understand the system status, detect abnormalities and locate the root causes. Due to the large amount of logs, log anomaly detection based on manual analysis is usually time-consuming and error-prone. Thus, there is an urgent need to introduce artificial intelligence algorithms to improve the detection of log anomalies and reduce network operation and maintenance costs, which is also the problem statement of 5G+AI network application competition, a thematic track of the AI innovation and application competition of China. At the same time, ITU launched the second ITU AI/ML in 5G Challenge, and this question belonged to one of the 16 challenging issues [1].

Over the years, researchers have proposed many data-driven methods to automatically detect anomalies [2, 3]. For instance, machine learning-based methods (e.g. Logistic Regression (LR) [4], Support Vector Machine (SVM) [4], Invariant Mining (IM) [5]) extract log events and employ supervised or unsupervised learning to detect system anomalies. Deep learning-based methods, such as LogRobust [6] and LogAnomaly [7], utilize Word2vec [8] to get the embedding vectors of log events and then apply the Long Short-Term Memory (LSTM) model to detect anomalies.

However, existing methods rely on log parsing to preprocess semi-structured log data. A log parser removes the variable part from the log message and keeps the constant part for log events. Log parsing is often uncertain, and once a parsing error occurs, it will directly reduce the performance of anomaly detection. For example, generation of new log events that do not appear in the training data due to the process of system update and evolution, or the wrong identification of log parameters and log keys during log parsing could lead to parsing errors.

Here, we propose a novel anomaly detection method, Keyword Embedding Log (KELog), to overcome the above-mentioned disadvantages of existing methods with a keyword embedding strategy. Unlike existing methods, KELog does not rely on any log parsing, thereby preventing the impact of information loss caused by log parsing errors on anomaly detection. Log messages are directly converted into semantic vectors and fused with keyword information, which can capture the semantic information and identify the abnormal key information. Then, the fused semantic vectors are inputted into a classification model to detect anomalies. KELog achieves effective and efficient anomaly detection on the real-world log datasets from Chinese communications operators.

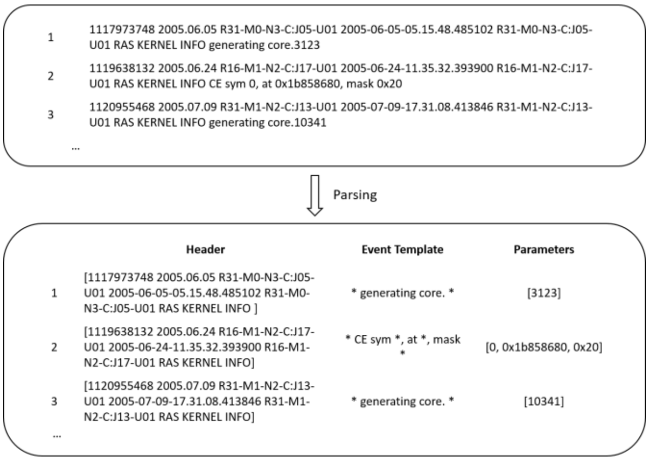
Main contributions of this paper are as follows:

- A novel model named KELog is proposed, including a keyword information extraction module, keyword embedding module and classification module. It detects log anomalies through deep understanding of keyword information with the aid of a masked keyword information task.
- The proposed KELog model integrates keyword information into log messages, which is able to deeply understand the semantics of log data, and capture anomalous information.
- We evaluated the proposed KELog model on a communications operator's log dataset. Experimental results show that KELog achieves an F1 score of 0.834, improved by 0.01 (12.1%).
- We also adopted KELog in the AI innovation and application competition of China and won the championship, and also participated in the competition of the second ITU AI/ML in 5G Challenge.

2. BACKGROUND

2.1 Log data

Log data records some important events that occur when the system is running, and plays a vital role in judging the state of the system and diagnosing system problems. Fig. 1 shows several raw logs generated by Blue Gene/L (BGL)[9]. The raw log messages are semi-structured texts, which contain header and content. The header is determined by the logging framework and includes information such as timestamp, verbosity level, and component. The log content consists of constant parts that reveal the event template and variable parameters that carry dynamic runtime information.



```
1    1117973748 2005.06.05 R31-M0-N3-C:J05-U01 2005-06-05-05.15.48.485102 R31-M0-N3-C:J05-
     U01 RAS KERNEL INFO generating core.3123
2    1119638132 2005.06.24 R16-M1-N2-C:J17-U01 2005-06-24-11.35.32.393900 R16-M1-N2-C:J17-
     U01 RAS KERNEL INFO CE sym 0, at 0x1b858680, mask 0x20
3    1120955468 2005.07.09 R31-M1-N2-C:J13-U01 2005-07-09-17.31.08.413846 R31-M1-N2-C:J13-
     U01 RAS KERNEL INFO generating core.10341
  ...
```

	Header	Event Template	Parameters
1	[1117973748 2005.06.05 R31-M0-N3-C:J05-U01 2005-06-05-05.15.48.485102 R31-M0-N3-C:J05-U01 RAS KERNEL INFO]	* generating core. *	[3123]
2	[1119638132 2005.06.24 R16-M1-N2-C:J17-U01 2005-06-24-11.35.32.393900 R16-M1-N2-C:J17-U01 RAS KERNEL INFO]	* CE sym *, at *, mask *	[0, 0x1b858680, 0x20]
3	[1120955468 2005.07.09 R31-M1-N2-C:J13-U01 2005-07-09-17.31.08.413846 R31-M1-N2-C:J13-U01 RAS KERNEL INFO]	* generating core. *	[10341]
...			

Fig. 1- Examples of BGL logs and parsed results

2.2 Log parsing

Log parsing automatically converts each log message into a specific event template by removing parameters and keeping the constant parts. For example, the log template "∗ generating core. ∗" can be extracted from the first log message in Fig. 1, where "∗" denotes the parameter.

Log parsing methods include clustering [10], frequent pattern mining [11], language modeling [12], heuristics [13], etc. Heuristics-based methods exploit the properties of logs and have been found to perform better than other techniques in terms of accuracy and time efficiency [14]. For example, Drain applies a fixed-depth tree structure to represent log messages and efficiently extract common templates [15]. Spell parses logs in a streaming manner using the longest common subsequence algorithm [16]. IPLoM adopts an iterative partitioning strategy to divide log messages into groups based on message length, token position, and mapping relationship [17].

2.3 Log anomaly detection

Over the years, researchers have proposed many machine learning algorithms to detect anomaly detection. For unsupervised methods, Xu et al. [18] employed Principal Component Analysis (PCA) to generate normal space and abnormal space of log count vectors. If the log count vector of a log sequence is far from the normal space, it is considered an anomaly. IM [5] finds linear relationships among log events from log count vectors, and log sequences that violate the relationship are considered anomalies. There are also many supervised anomaly detection methods. For example, after a log sequence is represented as a log count vector, SVM, LR, and decision tree algorithms are applied to detect anomalies, respectively. However, these methods often suffer from poor performance.

In recent years, researchers have proposed many deep learning-based models to analyze log data and detect anomalies. For example, DeepLog [19] first applies the Spell parser to extract log templates, then utilizes the indexes of log templates and feeds them to the LSTM model to predict the next log templates, and finally detects anomalies by comparing whether the incoming log template matches. LogRobust [6] combines a pretrained Word2vec model, FastText, with TF-IDF weights to learn the representation vectors of log templates generated by Drain, and then input an attention-based Bi-LSTM model to detect anomalies. LogBERT [20] learns the patterns of normal log sequences by two novel self-supervised training tasks and is able to detect anomalies where the underlying patterns deviate from normal log sequences. However, these methods may lose the semantics of log messages due to the imperfection of log parsing, resulting in inaccurate detection results.

Here, we propose a novel anomaly detection method, Keyword Embedding Log (KELog), to overcome the above-mentioned disadvantages. We notice that the traditional approaches often explicitly use keywords (e.g., “fail") or regular expressions to detect anomalous logs. However, they have poor accuracy. On the one hand, the keywords related to an anomaly mostly rely on experience and need to be updated. On the other hand, some logs with the defined keywords may not be abnormal. That is, understanding of the log message is important. So, we propose KELog to capture the semantic information and identify the abnormal key information, which does not rely on any log parsing, thereby preventing the impact of information loss caused by log parsing errors on anomaly detection.

3. MODEL ARCHITECTURE

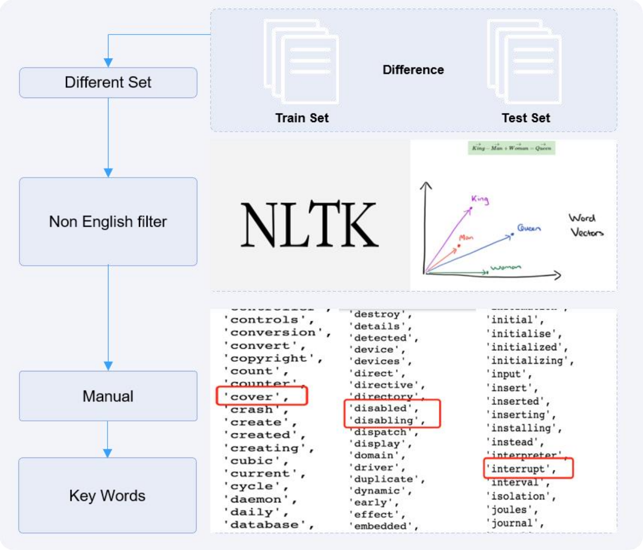
In this section, we present the overall framework of KELog and its detailed implementation, including the keyword information extraction in Section 3.2, the encode model in Section 3.3, the masked keyword information model in Section 3.4, and the classification in Section 3.5.

3.1 Notations

We denote a token sequence as $\{t_1, \ldots, t_n\}$, where n is the length of the token sequence. Meanwhile, we denote the keyword aligning to the given tokens as $\{k_1, \ldots, k_m\}$, where m is the length of the keyword sequence. Note that m is not equal to n in most cases, as not every token can be aligned to a keyword. Furthermore, we denote the whole vocabulary containing all tokens as V_L , and the whole vocabulary containing all keywords as V_K. V_K is a library consisting of keywords/key phrases manually screened out after automatic extraction that plays a key role in detecting whether a log is abnormal.If a token t has a corresponding keyword k, their alignment is defined as $f(t)=k$.

3.2 Keyword information extraction

In our model, we define the keyword information as those words appearing in the abnormal logs but not in the normal logs, which have an important role in detecting anomalies. We adopted a taking-the-difference strategy to preliminary screen keyword information, as shown in Fig. 2.



```
'controls',        'destroy',        'initial',
'conversion',      'details',        'initialise',
'convert',         'detected',       'initialized',
'copyright',       'device',         'initializing',
'count',           'devices',        'input',
'counter',         'direct',         'insert',
'cover',           'directive',      'inserted',
'crash',           'directory',      'inserting',
'create',          'disabled',       'installing',
'created',         'disabling',      'instead',
'creating',        'dispatch',       'interpreter',
'cubic',           'display',        'interrupt',
'current',         'domain',         'interval',
'cycle',           'driver',         'isolation',
'daemon',          'duplicate',      'joules',
'daily',           'dynamic',        'journal',
'database',        'early',
                   'effect',
                   'embedded',
```

Fig. 2 – Flow diagram of keyword information extraction

First, word segmentation was performed on normal and abnormal log data to get the vocabulary V_{normal} and $V_{abnormal}$ respectively. Then, V_{diff} was obtained by taking the difference $V_{abnormal} - V_{normal}$. Next, we used natural language processing tools such as nltk, jieba, etc. to filter non-English strings from V_{diff} , followed by word clustering via word2vec. In this way, the candidate keywords with similar semantics are grouped together for the convenience of manual screening. After removing invalid keywords, the final library V_K consisting of keywords/key phrases was obtained, which will be used in the information fusion.

3.3 Keyword embedding architecture

As shown in Fig. 3, the keyword embedding architecture of KELog consists of three parts: (1) bottom encoder layer, which is responsible for capturing basic lexical or syntactic information from log text and keywords in advance; (2) information fusion layer, which is responsible for integrating keyword information into log text information; (3) top encoder layer, which continues to encode the fused semantic vectors. Here, the bottom coding layer is stacked with 3 layers, the information fusion layer has 1 layer, and the upper coding layer is stacked with 9 layers.

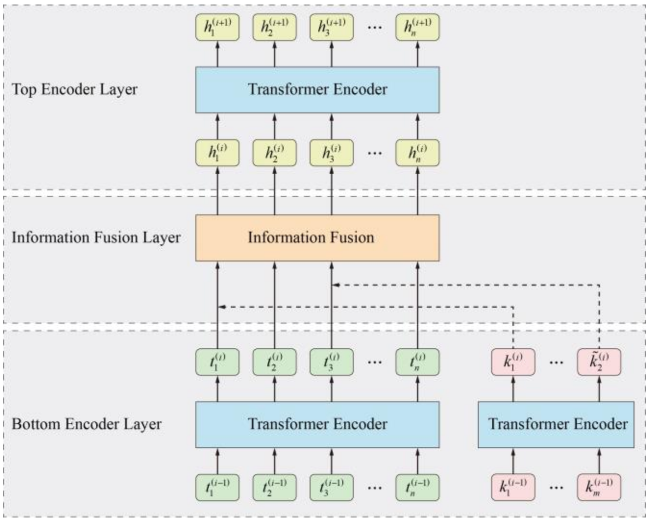


Fig. 3 – Keyword embedding structure

3.3.1 Bottom encoder layer

Given a token sequence and its corresponding keyword sequence, the bottom encoder layer firstly sums the token embedding, segment embedding, positional embedding for each token to compute its input embedding, and then computes lexical and syntactic features as follows,

$$t, k = Multihead(t_{input}, k_{input}) \tag{1}$$

where *Multihead*(·) is a multilayer bidirectional transformer encoder identical to its implementation in BERT [21].

3.3.2 Information fusion layer

The information fusion layer is designed for fusing heterogeneous features of tokens and keywords, where the mutual integration of the token and keyword sequence is conducted to obtain the output embedding for each token and keyword. For a log text token t_p and its aligned keyword $k_q = f(t_p)$, the information fusion process is as follows,

$$h_p = \sigma(W_t^{(i)} t_p^{(i)} + W_k^{(i)} k_q^{(i)} + b^{(i)}) \tag{2}$$

where h_p is the inner hidden state integrating the information of both the log token and the keyword information. σ(·) is the non-linear activation function, which usually is the GELU function. *W and b* are the weight and bias parameters, respectively. For the tokens without corresponding keywords, the information fusion layer computes the output embeddings without integration as follows:

$$h_p = \sigma(W_t^{(i)} t_p^{(i)} + b^{(i)}) \tag{3}$$

These output embeddings computed by the information fusion layer will be used as the inputs of the top encoder layer.

3.3.3 Top encoder layer

Given the integrated embeddings of log tokens and keywords, the top encoder layer performs further semantic learning via the multilayer bidirectional transformer encoder as follows:

$$h = Multihead\left(h_p\right) \tag{4}$$

where *Multihead*(·) is a multilayer bidirectional transformer encoder identical to its implementation in BERT [21].

3.4 Masked keyword information task

KELog first performs a Masked Language Model (MLM) pretraining task of BERT [21], enabling the model to capture basic lexical and syntactic information from log texts. Next, to deeply incorporate key information into the linguistic representation of log text, we propose an auxiliary task that randomly masks some keywords and then requires the model to predict all corresponding keywords based on the fused representations. Similar to the MLM task of BERT, this strategy aims to help the model better understand the semantic information carried by the keywords. We perform the following operations: (1) In 5% of the time, for a given keyword, we replace it with another random keyword, aiming to train the model to correct the errors that the log token is aligned with a wrong keyword. (2) In 15% of the time, we mask the right corresponding keywords to train the model to correct the error that the model does not extract all corresponding keywords. (3) In the remaining time, keep the corresponding keywords unchanged, aiming to encourage the model to integrate key information into log representations for better understanding of log messages. The model predicts the masked keywords with a cross-entropy loss.

3.5 Classification

The log semantic vectors after the masked key information auxiliary task will be used for the next binary classification task of judging whether the log is abnormal. We use a linear classifier with the sigmoid activate function, where the binary cross entropy loss is used.

Owing to that the key information has been embedded in the representations, the model can learn the hidden relation of the key word information and the abnormal logs.

4. EXPERIMENT

4.1 Dataset and evaluation metrics

We used two datasets in this paper. The log type of these two datasets are similar, both from Linux operating systems. Specifically, dataset A comes from a Chinese communications operator[1]. This dataset contains the operation and maintenance log messages that have been manually annotated by experts. The logs are in a format of “Timestamp Logtype Logtext”. An example of a log message is given in Table 1. There were a total of 1000000 log messages with 950000 normal logs and 50000 abnormal logs in the dataset. Due to the imbalance of the data distribution, for comprehensive evaluation of the classification effect of the model, we have used three common evaluation metrics for the original labels, i.e. Precision (P, $P = TP/(TP+FP)$), Recall (R, $R = TP/(TP+FN)$), and F1 score ($F1 = 2PR/(P+R)$), where TP, FP, and FN represent true positive, false positive, and false negative values, respectively.

Table 1 - An example of a log message

2021-05-01T00:00:01.127409+08:00 INFO: Started Session 1308850 of user cps.		
Timestamp	**Logtype**	**Logtext**
2021-05-01 T00:00:01.127409+08 :00	INFO	Started Session 1308850 of user cps.

Dataset B comes from the 5G+AI network application competition, a thematic track of the AI innovation and application competition of China, which was provided by another Chinese communications operator[1]. It was required to detect abnormal network conditions by analyzing the log data of communication network equipment in a specific period of time. The dataset was divided into a training set and a test set. The training set contains 1419918 logs without any anomalies, that is, they are all normal logs. The test set contains 47870 logs that may be abnormal or normal. They were divided into about 4700 time slices every 5 minutes. If any log within a time slice was anomalous, this time slice would be labeled as an anomaly. The F1 score was calculated based on predictions of all time slices on the test set.

4.2 Results and analysis

On dataset A, to evaluate the effect of anomaly detection of the log messages of the communications operator, we compared our KELog model with five other widely used models (See Section 2.3) as follows: PCA [18], SVM [4], IM [5], DeepLog [19], LogBERT[20]. The former three are based on machine learning, while the latter two are based on deep learning. We used the recommended hyper-parameters from the corresponding papers. Specifically, DeepLog was obtained from Github (https://github.com/Thijsvanede/DeepLog) and run in its original setting (learning_rate = 0.01, batch_size = 128, input_size = 300, hidden_size = 64 , output_size = 300, etc.). LogBERT was also obtained from Github (https://github.com/HelenGuohx/logbert) and run in its original setting (window_size = 128, seq_len = 512, hidden = 256, batch_size = 32, lr = 1e-3, etc.) As for our KELog, we first obtained the keyword information library based on the steps in Section 3.2 and used it to mask keywords in the log messages. Followed by the masked keyword information task and the final binary classification

[1] The datasets used to support the findings of this study have not been made available because of commercial confidentiality.

task of judging whether a log is abnormal, where settings (maxlen = 128, batch_size = 32, lr = 2e-5, epochs = 10) were applied.We applied a 5-fold cross-validation scheme to train and test the models in experiments.

We used the evaluation metrics discussed above in the anomaly prediction results. Table 2 shows the Precision (P), Recall (R) and F1 scores of the six models after comparison experiments. In general, our KELog model achieves the best performance. We can notice that PCA, SVM, and IM have relatively poor performance on log anomaly detection since they did not employ deep learning. Although these methods could achieve high precision or recall values, they cannot balance the performance on both precision and recall. This could be because using the counting vector to represent logs leads to the loss of semantic information. Compared with these methods, the F1 score of KELog achieves an increase of 0.341, 0.179, 0.147, respectively. DeepLog and LogBERT outperform the traditional approaches and achieve reasonable F1 scores, which show the advantage of deep learning models. Compared with them, the F1 score of KELog achieves an increase of 0.039, 0.01, respectively. Moreover, our proposed KELog achieves the highest recall and F1 scores compared with the five other methods. It indicates that by integrating keywords into log texts, KELog has a good representation ability to effectively detect candidate anomalous logs and further identify anomalies with high accuracy.

Table 2 - Comparison of precision, recall and F1 scores of different models

Method	**P**	**R**	**F1**
PCA	0.356	0.801	0.493
SVM	0.739	0.588	0.655
IM	0.588	0.825	0.687
DeepLog	0.820	0.771	0.795
LogBERT	0.803	0.847	0.824
KELog	0.755	**0.932**	**0.834**

On dataset B, due to the similarities in the log type to dataset A, we could apply our KELog model to dataset B. Specifically, we added the training set of dataset B to dataset A to further train KELog. Then we used KELog to predict the logs in the test set of dataset B. Finally, we won the first place with a full F1 score, 0.021 higher than the second place that used DeepLog (F1 scores of the top 5 rankings of 5G+AI network application competition were 1.0(ours), 0.9787, 0.9545, 0.9130, 0.8205 [2]). DeepLog counts on the quality of log parsing and may miss some anomalies caused by parsing errors, which reduces the performance of anomaly detection. While for our KELog, such a high score was owing to two main reasons. One is that the inputs of KELog integrated keyword information into log representations. The keyword information covers all anomalies, leading to significant improvement of recall scores. The other is that the strong representation ability of the transformer contributed to accurate extraction of anomalies. Due to the above advantages of KELog, we obtained the full F1 score compared with other methods that used DeepLog. It reflects the generalization ability of KELog.

5. CONCLUSION

In this paper, we propose KELog to incorporate keyword information into log representations. Accordingly, we propose the information fusion layer and the masked key information auxiliary task for better understanding log messages and detecting anomalies. The experimental results on real-world log datasets of Chinese communications operators show that KELog outperforms the state-of-the-art approaches for log anomaly detection.

This model has been launched in the of Artificial Intelligence for IT Operations (AIOps) system of a communications operator and improved the performance of anomaly detection. Moreover, we adopted KELog in the AI innovation and application competition of China, achieved a high F1 score of 1.0 and won the championship, which also participated in the competition of the second ITU AI/ML in 5G Challenge. In the future, we will consider integrating log parsing or log sequences into the model to investigate the possibility of effect improvement.

ACKNOWLEDGEMENT

Support Scheme of Guangzhou for Leading Talents in Innovation and Entrepreneurship (No: 2020010).

[2] http://www.aiinnovation.com.cn/#/AIcaict/trackDetail?id=a01bbc0d-f76b-11eb-b8ef-5254d03664f5

REFERENCES

[1] "Special issue on AI/ML solutions in 5G and future networks." https://www.itu.int/en/journal/j-fet/2022/004/Pages/default.aspx.

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

[3] B. Zhang, H. Zhang, P. Moscato, and A. Zhang, "Anomaly detection via mining numerical workflow relations from logs," in *2020 International Symposium on Reliable Distributed Systems (SRDS)*, 2020: IEEE, pp. 195-204.

[4] P. Bodik, M. Goldszmidt, A. Fox, D. B. Woodard, and H. Andersen, "Fingerprinting the datacenter: automated classification of performance crises," in *Proceedings of the 5th European conference on Computer systems*, 2010, pp. 111-124.

[5] J.-G. Lou, Q. Fu, S. Yang, Y. Xu, and J. Li, "Mining invariants from console logs for system problem detection," in *2010 USENIX Annual Technical Conference (USENIX ATC 10)*, 2010.

[6] X. Zhang *et al.*, "Robust log-based anomaly detection on unstable log data," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019, pp. 807-817.

[7] W. Meng *et al.*, "LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs," in *IJCAI*, 2019, vol. 19, no. 7, pp. 4739-4745.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781,* 2013.

[9] A. Oliner and J. Stearley, "What supercomputers say: A study of five system logs," in *37th annual IEEE/IFIP international conference on dependable systems and networks (DSN'07)*, 2007: IEEE, pp. 575-584.

[10] L. Tang, T. Li, and C.-S. Perng, "LogSig: Generating system events from raw textual logs," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, 2011, pp. 785-794.

[11] M. Nagappan and M. A. Vouk, "Abstracting log lines to log event types for mining software system logs," in *2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010)*, 2010: IEEE, pp. 114-117.

[12] S. Thaler, V. Menkonvski, and M. Petkovic, "Towards a neural language model for signature extraction from forensic logs," in *2017 5th International Symposium on Digital Forensic and Security (ISDFS)*, 2017: IEEE, pp. 1-6.

[13] Z. M. Jiang, A. E. Hassan, P. Flora, and G. Hamann, "Abstracting execution logs to execution events for enterprise applications (short paper)," in *2008 The Eighth International Conference on Quality Software*, 2008: IEEE, pp. 181-186.

[14] J. Zhu *et al.*, "Tools and benchmarks for automated log parsing," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2019: IEEE, pp. 121-130.

[15] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *2017 IEEE international conference on web services (ICWS)*, 2017: IEEE, pp. 33-40.

[16] M. Du and F. Li, "Spell: Streaming parsing of system event logs," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, 2016: IEEE, pp. 859-864.

[17] A. A. Makanju, A. N. Zincir-Heywood, and E. E. Milios, "Clustering event logs using iterative partitioning," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 1255-1264.

[18] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, 2009, pp. 117-132.

[19] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 1285-1298.

[20] H. Guo, S. Yuan, and X. Wu, "Logbert: Log anomaly detection via bert," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021: IEEE, pp. 1-8.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805,* 2018.

AUTHORS

Yong Song is the head of the communication business and application algorithm research department in Telco Artificial Intelligence Labs of AsiaInfo Technologies (China) Co., Ltd. His main research interests include AIOps, NLP, knowledge graphs, recommendations, etc.

Zhiwei Yan is an algorithm engineer at Telco Artificial Intelligence Labs of AsiaInfo Technologies (Nanjing) Co., Ltd. His main research interests include AIOps, NLP, knowledge graphs, etc.

Yukun Qin is an algorithm engineer at Telco Artificial Intelligence Labs of AsiaInfo Technologies (Nanjing) Co., Ltd. His main research interests include AIOps, NLP, knowledge graphs, etc.

Yuchen Xie is an algorithm engineer at Telco Artificial Intelligence Labs of AsiaInfo Technologies (Beijing) Co., Ltd. His main research interests include AIOps, NLP, knowledge graphs, etc.

Xiaozhou Ye is the senior director/chief scientist of Telco Artificial Intelligence Labs of AsiaInfo Technologies (China) Co., Ltd. His main research direction is communication networks and artificial intelligence. His main research focuses on communication networks and artificial intelligence.

Ye Ouyang is the chief technology officer and the senior vice president of AsiaInfo Technologies (China) Co., Ltd. and a national distinguished expert, the distinguished expert of Beijing Municipal Government. His main research interests include mobile communication, artificial intelligence, data science, technology R&D innovation and management.