Unsupervised Cross-system Log Anomaly Detection via Domain Adaptation

Xiao Han
Xiao.Han@usu.edu
Utah State University
Logan, Utah, USA

Shuhan Yuan
Shuhan.Yuan@usu.edu
Utah State University
Logan, Utah, USA

ABSTRACT

Log anomaly detection, which focuses on detecting anomalous log records, becomes an active research problem because of its importance in developing stable and sustainable systems. Currently, many unsupervised log anomaly detection approaches are developed to address the challenge of limited anomalous samples. However, collecting enough data to train an unsupervised model is not practical when the system is newly deployed online. To tackle this challenge, we propose a transferable log anomaly detection (LogTAD) framework that leverages the adversarial domain adaptation technique to make log data from different systems have a similar distribution so that the detection model is able to detect anomalies from multiple systems. Experimental results show that LogTAD can achieve high accuracy on cross-system anomaly detection by using a small number of logs from the new system.

CCS CONCEPTS

• **Computing methodologies** → **Anomaly detection**.

KEYWORDS

sequential anomaly detection; transfer learning; domain adaptation

ACM Reference Format:
Xiao Han and Shuhan Yuan. 2021. Unsupervised Cross-system Log Anomaly Detection via Domain Adaptation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3459637.3482209

1 INTRODUCTION

Online services, e.g. cloud computing, web platforms, and remote databases, have become essential in our daily life. How to maintain reliability and stability is a major challenge of operating successful online services, since a small glitch would cause a crash of the whole system that affects users' experience or even leads to financial loss. Therefore, effectively detecting anomalous status is critical to building reliable online systems.

In practice, system logs, which are extensively adopted for recording states of online systems, take a critical part in detecting the anomalous status of online systems. In recent years, many machine learning-based log anomaly detection approaches are proposed to detect anomalous log sequences [2, 4, 5, 7–11, 13–15]. Especially, due to limited anomalies, many unsupervised machine learning approaches are proposed to identify the anomalous log sequences, such as Principal Component Analysis (PCA) [14], DeepLog [2], and LogBERT [3]. The key idea of these models is to capture normal patterns from a large number of normal logs. Then, the models can detect anomalous log sequences based on the obtained normal patterns. However, online systems are deployed constantly. Given a newly deployed system, it takes some time to collect enough logs to train an unsupervised anomaly detection model. On the other hand, a newly deployed system usually has an urgent requirement to automatically identify the abnormal status.

To tackle this cold-start problem, several transfer learning-based approaches are proposed to detect anomalies in different systems, such as LogTransfer [1]. The basic assumption of these approaches is that the anomalous log sequences in different online systems usually share similar patterns. For example, the words in log messages from different systems usually have some overlaps. Meanwhile, the normal workflows of different systems also have some similarities. The idea is to build a cross-system classifier to detect anomalies in both source and target systems. The major disadvantage of existing models is that they usually require both normal and abnormal log data from both source and target systems to build the classifier. However, in practice, it is often infeasible to collect sufficient anomalous data from the target system effectively.

To address the limitation of current transfer learning approaches for cross-system log anomaly detection, in this work, we propose a transferable log anomaly detection approach, called LogTAD, which does not require the labeled anomalous data from both source and target systems. Inspired by the DeepSVDD approach [12], which makes the normal data close to a center of a hypersphere, the core idea of our approach is to derive a system invariant center representation based on the normal data in both source and target systems. Specifically, we leverage the domain adversarial technique to map both source and target log sequences into the same hypersphere with a similar distribution and derive a shared center for both systems. Then, the anomalous sequences from both systems can be detected by having large distances to the center.

The main contributions of this work are as follows. First, we develop a cross-system anomaly detection model that only using the normal samples. Second, the proposed model only requires a small number of samples from the target system so that it can address the cold-start problem of anomaly detection. Third, the experimental results on two log datasets show that LogTAD can achieve good performance on both source and target systems.

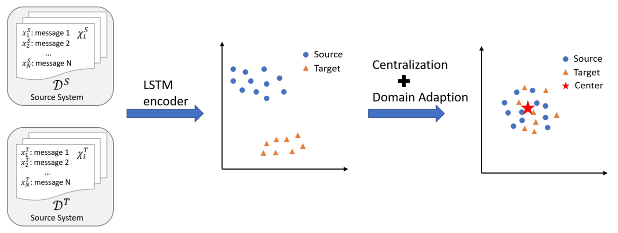


Figure 1: Framework of LogTAD

2 FRAMEWORK

In this work, we assume the existence of one source system S and one target system T. A log sequence generated by a system is denoted as $\mathcal{X} = \{x_n\}_{n=1}^{N}$, where x_n indicates the n-th log message. Given a dataset $\mathcal{D}$ consisting of normal log sequences from the source system $\mathcal{D}^S = \{\mathcal{X}_i^S\}_{i=1}^{M_S}$ and a small number of normal log sequences from the target system $\mathcal{D}^T = \{\mathcal{X}_i^T\}_{i=1}^{M_T}$ ($M_T << M_S$), we aim to build an unsupervised and transferable log anomaly detection model (LogTAD), which is able to detect the anomalous log sequences from both source and target systems. Our framework consists of two parts. One is an encoder to map log sequences from both systems into a low dimensional space, while another part is to map the representations of log sequences from both systems into one hypersphere with a shared center. Specifically, we leverage the idea of DeepSVDD [12] to map normal sequences close to a center of a hypersphere. Meanwhile, a domain adversarial framework is proposed to ensure the representations of sequences from both systems are close to one center. Then, we can detect the anomalies in both systems that have large distances to the center. Figure 1 shows the framework of LogTAD.

2.1 Log Sequence Representation

Log messages generated by the online systems are descriptive texts. Many existing log anomaly approaches adopt tools to generate templates from log messages and transfer raw log sequences into sequences of templates. However, under the setting of cross-system anomaly detection, the templates from source and target systems can be totally different. Then, it is infeasible to use the knowledge from the source system to improve the performance of anomaly detection in the target system. Hence, in this work, we propose to use the information from the raw messages. The intuition is that the descriptive words from different systems should share some similarities, i.e., there are some overlaps in terms of words from source and target systems. Then, we are able to transfer the knowledge from the source system to the target system. Specifically, we first use word2vec to represent words in each log message as embedding vectors and then adopt the mean operation over the word embeddings in a log message to derive the representation of one message. Formally, we now represent a log sequence as $\mathcal{X} = \{\mathbf{x}_n\}_{n=1}^{N}$ where $\mathbf{x}_n \in \mathbb{R}^d$ is the feature vector of n-th message.

Then, an encoder encodes the log messages in a sequence to a sequence representation with an LSTM model:

$$\mathbf{h}_n = LSTM(\mathbf{x}_n, \mathbf{h}_{n-1}), \tag{1}$$

where $\mathbf{x}_n$ is the feature vector of the n-th message; $\mathbf{h}_n$ indicates the n-th hidden vector of LSTM. The last hidden vector $\mathbf{h}_N$ captures the information of an entire sequence and is considered as the log sequence representation $\mathbf{v} = \mathbf{h}_N$. Here, we use a shared LSTM model to get the sequence representations from both source and target systems. Specifically, we denote the representation of the i-th sequence from the source system as $\mathbf{v}_i^S$, while the representation of i-th sequence from the target system as $\mathbf{v}_i^T$. One advantage of using one shared LSTM model is that using one LSTM to model the sequences from both systems can reduce the requirement on the samples from the target system that has limited samples available.

2.2 Log Sequence Centralization

Inspired by the DeepSVDD that the normal log sequences should be in a hypersphere and close to the center in the embedding space, we first derive the center of all the log sequences from both source and target systems as the mean over all log sequences in both datasets, i.e., $\mathbf{c} = Mean(\mathbf{v}_i^\epsilon)$, where $\epsilon \in \{S, T\}$ indicates the source or target dataset. To make the representation of normal log sequences close to the center $\mathbf{c}$, we develop the objective function as

$$\mathcal{L}_{en} = \sum_{\epsilon \in \{S,T\}} \sum_{i=1}^{M_\epsilon} \|\mathbf{v}_i^\epsilon - \mathbf{c}\|^2. \tag{2}$$

By minimizing the Equation 2, we expect all normal log sequences are close to the center.

2.3 System-agnostic Representation via Domain Adversarial Training

Although we adopt one shared LSTM model to map log sequences into a hypersphere, the representations of log sequences from different systems can be still located in different regions of the hypersphere instead of clustering around the center $\mathbf{c}$. Hence, we propose an adversarial training method for cross-system (domain adaptation) data mapping, which encourages the representations of log sequences from different systems close to each other.

We formulate the adversarial training as follows. A discriminator D is used to distinguish whether the representations of log sequences are from the source or target system. We adopt the logistic function to make the prediction, i.e., $D(\mathbf{v}^\epsilon) = \sigma(\mathbf{w}^T \mathbf{v}^\epsilon + b)$, where $\sigma(\cdot)$ indicates the logistic function, $\mathbf{w}$ and b are the trainable parameters. The shared LSTM as a generator, i.e., $\mathbf{v}^\epsilon = G(\mathcal{X}^\epsilon)$, is trained to make representations of log sequences from different systems have similar distributions so that the discriminator is unable to distinguish the source samples from the target samples. Specifically, the objective function is

$$\begin{aligned}\mathcal{L}_{adv} = \min_G \max_D \big(&\mathbb{E}_{\mathcal{X}^S \sim P_{\text{source}}}[\log D(G(\mathcal{X}^S))] + \\ &\mathbb{E}_{\mathcal{X}^T \sim P_{\text{target}}}[\log(1 - D(G(\mathcal{X}^T)))]\big),\end{aligned} \tag{3}$$

where $\mathcal{X}^S$ and $\mathcal{X}^T$ indicate the source and target log sequences. By training through the minmax optimization, the representations of log sequences generated by the shared LSTM model are able to mislead the discriminator, which means the distributions of source and target log sequences are mixed. Finally, we train our framework by the following objective function:

$$\mathcal{L} = \mathcal{L}_{en} + \lambda \mathcal{L}_{adv}, \quad (4)$$

where λ is a hyperparameter to balance the two parts. After the training phase, the encoder gains the ability to embed both source and target samples close to the center **c** so that the abnormal samples from both systems should have large distances to the center.

Log Anomaly detection. To detect the anomalies, we need to set a radius γ as a decision boundary that separates normal and anomalous sequences. The samples with distances to the center greater than γ will be labeled as anomalous sequences. We adopt a validation set in the source and target systems respectively to find the best radius γ^S and γ^T. In our experiments, we randomly select one-day log sequences from each system as the validation set.

3 EXPERIMENTS

3.1 Experimental Setup

Table 1: Statistics of the Datasets

Dataset	# of Logs	# of Log Sequences	
		Normal	Anomalous
BGL	1,212,150	265,583	37,450
TB	3,737,209	565,817	368,481

Datasets. We adopt the following two datasets: (1) BlueGene/L supercomputer system (**BGL**) dataset [11], which is collected from a BlueGene/L supercomputer system at Lawrence Livermore National Labs; (2) Thunderbird (**TB**) dataset [11], which is collected from a Thunderbird supercomputer system at Sandia National Labs. The statistics of the two datasets are shown in Table 1. Table 2 further shows the numbers of shared words across different datasets. The normal log sequences in the BGL and Thunderbird datasets have 664 and 1753 unique words, respectively, while the number of shared words is 254. The anomalous log sequences in the BGL and Thunderbird datasets have 195 and 54 unique words, respectively, while the number of shared words is 16.

Table 2: Statistics of Shared Words

	BGL Normal	BGL Anomalous	TB Normal	TB Anomalous
BGL Normal	664	133	254	25
BGL Anomalous	133	195	99	16
TB Normal	254	99	1753	49
TB Anomalous	25	16	49	54

Baselines. We compare our framework with the following anomaly detection methods to evaluate the performance of LogTAD: (1) Principal Component Analysis (*PCA*), which is a dimensional reduction based anomaly detection approach; (2) *LogCluster* [13], which is a clustering-based log anomaly detection approach; (3) *DeepLog* [2], which is a deep learning based log anomaly detection method; (4) Deep Support Vector Data Description (*DeepSVDD*) [12], which is a deep learning based one-class classification model; (5) *LogTransfer* [1], which can achieve cross-system anomaly detection but requires labeled data from both systems to train a classifier.

Table 3: Anomaly Detection on Source and Target Systems

	Method	Source		Target	
		F1	AUC	F1	AUC
BGL → TB	PCA w/ TB	0.322	0.587	0.731	0.776
	PCA w/o TB	0.642	0.816	0.558	0.504
	LogCluster w/ TB	0.530	0.746	0.677	0.716
	LogCluster w/o TB	0.713	0.829	0.559	0.504
	DeepLog w/ TB	0.662	0.854	0.590	0.619
	DeepLog w/o TB	0.678	0.867	0.556	0.500
	DeepSVDD w/ TB	0.499	0.725	0.567	0.616
	DeepSVDD w/o TB	0.566	0.789	0.577	0.646
	LogTransfer	0.971	0.972	0.792	0.828
	LogTAD	0.926	0.964	0.758	0.804
TB → BGL	PCA w/ BGL	0.789	0.798	0.577	0.773
	PCA w/o BGL	0.760	0.779	0.229	0.658
	LogCluster w/ BGL	0.708	0.688	0.697	0.886
	LogCluster w/o BGL	0.724	0.716	0.223	0.500
	DeepLog w/ BGL	0.687	0.701	0.527	0.843
	DeepLog w/o BGL	0.660	0.677	0.223	0.500
	DeepSVDD w/ BGL	0.660	0.699	0.196	0.537
	DeepSVDD w/o BGL	0.794	0.808	0.195	0.497
	LogTransfer	0.995	0.995	0.788	0.833
	LogTAD	0.788	0.797	0.845	0.909

Implementation Details. We adopt a sliding window with size 20 to cut log files into short sequences and leverage the Drain package [6] to extract templates from raw log messages. The LSTM encoder consists of two hidden layers with 128 dimensions. For the domain discriminator, we use a two-layer fully connected neural network with 64 hidden dimensions. We use 100,000 normal sequences from the source system and 1,000 normal sequences from the target system in the training stage. The λ defined in Equation 4 is 0.1. Our code is available online [1].

3.2 Experimental Results

We evaluate the cross-system log anomaly detection on two scenarios, i.e., BGL → TB, where BGL is the source system while Thunderbird is the target system, and TB → BGL, where Thunderbird is the source system while BGL is the target system. We evaluate the performance in terms of F1 score and AUC. Table 3 shows the performance of LogTAD as well as baselines.

LogTAD v.s. Unsupervised Anomaly Detection Approaches. As PCA, LogCluster, and DeepLog are unsupervised models and not designed for domain adaptation, we evaluate these models in two scenarios, i.e., training with or without using the log sequences from the target system. First, for both BGL → TB and TB → BGL scenarios, PCA, LogCluster, and DeepLog can achieve reasonable F1 scores and AUC values in the source system with no training samples from the target system. However, when given log sequences from the target system in the training dataset, PCA, LogCluster, and DeepLog gain better F1 scores and AUC values in the target system with worse results in the source system. It indicates that the existing state-of-the-art unsupervised log anomaly detection models do not have the capability for cross-system adaptation. Using the log sequences from both source and target systems can only

[1]https://github.com/hanxiao0607/LogTAD

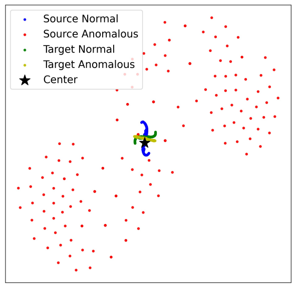
make the detection models confused about the distribution of normal samples, which leads to poor performance. On the other hand, LogTAD achieves better performance on both source and target systems under BGL ⟶ TB and TB ⟶ BGL scenarios compared with PCA, LogCluster, and DeepLog.

LogTAD v.s. DeepSVDD. DeepSVDD gets a lower F1 score in both source and target systems when given target sequences in the training dataset. This is because mixed training data lead to diverse data distribution. The center derived from such distribution is not useful for detecting anomalies in the target system since we only use a small number of samples from the target system. On the other hand, without domain adaptation, the samples from the target systems are more like outliers to the source system. As a result, the performance of DeepSVDD is also not good on the source system. Hence, using adversarial domain adaptation to make samples have similar distribution is key to achieve cross-system anomaly detection. Another observation is that when only using data from the source system, the performance of DeepSVDD is still not as good as LogTAD. This could be because LogTAD derives the center by using more samples with the same distribution.

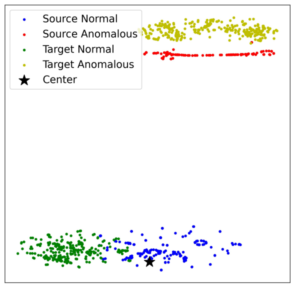
LogTAD v.s. LogTransfer. LogTransfer is a supervised transfer learning method that adopts normal and anomalous data from the source and target systems to train a cross-system anomaly detection classifier. LogTransfer can achieve promising performance when sufficient labeled data are available. In this experiment, we aim to identify how many labeled anomalous samples are required to train the LogTransfer so that it can have a similar performance as LogTAD. Especially, we use 100 anomalous sequences from the source system and 10 anomalous sequences from the target system to train LogTransfer. LogTransfer can achieve the best performance in the source system. For the scenario BGL ⟶ TB, LogTransfer also achieves slightly better performance than LogTAD, while for the scenario TB ⟶ BGL, using 10 anomalous sequences from the target system is not sufficient to outperform LogTAD. Therefore, in the cases where labeled anomalous samples are hard to obtain, LogTAD can still provide good performance by only using normal data.

Visualization. We show the effectiveness of our cross-system adaptation approach by visualizing sequence representations $\mathbf{v}$. We adopt the t-SNE algorithm to map the sequence representations into a two-dimensional space. For each dataset, we randomly select 300 normal and anomalous sequences, respectively, and consider the scenario BGL ⟶ TB. Figure 2 compares the sequence representations derived with and without domain adversarial training in the latent space. As shown in Figure 2a, for the model without domain adversarial training, normal sequences from the source and target systems and anomalous sequences from the target system are highly mixed. Meanwhile, these three types of sequences keep a small distance from the center in the latent space. Only anomalous sequences from the source system keep away from the center region. On the other hand, as shown in Figure 2b, with domain adversarial training, normal sequences from both source and target systems are close to the center, while anomalous sequences from both systems are out of the center region. Hence, LogTAD can detect anomalous sequences from both source and target systems.

Sample Size. We further evaluate the performance of LogTAD with various amounts of sequences from the target system. Figure 3 shows the experimental results. In general, we can notice that

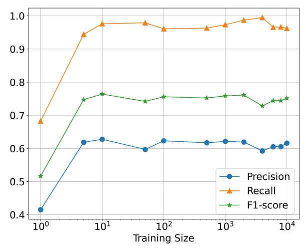


(a) Without domain adaptation

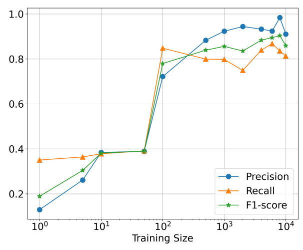


(b) With domain adaptation

Figure 2: Log Sequence Visualization (BGL ⟶ Thunderbird)



(a) BGL ⟶ Thunderbird



(b) Thunderbird ⟶ BGL

Figure 3: LogTAD performance with different training sizes from the target system. (x-axis is in log-scale)

LogTAD can achieve a relatively high anomaly detection performance on the target system by using a small number of normal sequences from the target system. Especially, for the scenario BGL ⟶ TB, only using 10 normal log sequences from the Thunderbird dataset is sufficient to achieve good performance. When considering the scenario TB ⟶ BGL, using around 100 samples can achieve reasonable performance, and the performance can keep improving when more samples are available. In general, even the novel online system is deployed for a short term, a handful of sequences from the target system can be easily obtained, so LogTAD has strong feasibility and accuracy in detecting anomalies in the novel system.

4 CONCLUSION

In this work, we present a cross-system log anomaly detection framework, LogTAD. It makes the normal log data in a hypersphere close to a center and utilizes the domain adversarial adaptation to make the log data from different systems follow similar distributions. Then, we can detect anomalies in different systems with large distances to the center. The experiment results show the effectiveness of our framework. One limitation of this work is that we assume the source and target systems should share similar patterns in terms of descriptive words. In the future, we plan to alleviate this assumption to improve the generalization of the approach.

ACKNOWLEDGMENTS

This work was supported in part by NSF 2103829.

REFERENCES

[1] Rui Chen, Shenglin Zhang, Dongwen Li, Yuzhe Zhang, Fangrui Guo, Weibin Meng, Dan Pei, Yuzhi Zhang, Xu Chen, and Yuqing Liu. 2020. LogTransfer: Cross-System Log Anomaly Detection for Software Systems with Transfer Learning. In *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*.

[2] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*.

[3] Shuhan Yuan Haixuan Guo and Xintao Wu. 2021. LogBERT: Log Anomaly Detection via BERT. In *Proceedings of the International Conference on Neural Networks (IJCNN)*.

[4] Pinjia He, Jieming Zhu, Shilin He, Jian Li, and Michael R. Lyu. [n.d.]. An Evaluation Study on Log Parsing and Its Use in Log Mining. In *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)* (Toulouse, France, 2016-06). IEEE, 654–661. https://doi.org/10.1109/DSN.2016.66

[5] P. He, J. Zhu, S. He, J. Li, and M. R. Lyu. [n.d.]. Towards Automated Log Parsing for Large-Scale Log Data Analysis. 15, 6 ([n. d.]), 931–944. https://doi.org/10.1109/TDSC.2017.2762673 Conference Name: IEEE Transactions on Dependable and Secure Computing.

[6] Pinjia He, Jieming Zhu, Zibin Zheng, and Michael R. Lyu. [n.d.]. Drain: An Online Log Parsing Approach with Fixed Depth Tree. In *2017 IEEE International Conference on Web Services (ICWS)* (Honolulu, HI, USA, 2017-06). IEEE, 33–40. https://doi.org/10.1109/ICWS.2017.13

[7] S. He, J. Zhu, P. He, and M. R. Lyu. [n.d.]. Experience Report: System Log Analysis for Anomaly Detection. In *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)* (2016-10). 207–218. https://doi.org/10.1109/ISSRE.2016.21 ISSN: 2332-6549.

[8] Yinglung Liang, Yanyong Zhang, Hui Xiong, and Ramendra Sahoo. [n.d.]. Failure Prediction in IBM BlueGene/L Event Logs. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)* (Omaha, NE, USA, 2007-10). IEEE, 583–588. https://doi.org/10.1109/ICDM.2007.46

[9] S. Lu, X. Wei, Y. Li, and L. Wang. [n.d.]. Detecting Anomaly in Big Data System Logs Using Convolutional Neural Network. In *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)* (2018-08). 151–158. https://doi.org/10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00037

[10] Weibin Meng, Ying Liu, Yichen Zhu, Shenglin Zhang, Dan Pei, Yuqing Liu, Yihao Chen, Ruizhi Zhang, Shimin Tao, Pei Sun, and Rong Zhou. [n.d.]. *LogAnomaly: Unsupervised Detection of Sequential and Quantitative Anomalies in Unstructured Logs.* https://doi.org/10.24963/ijcai.2019/658 Pages: 4745.

[11] Adam Oliner and Jon Stearley. 2007. What supercomputers say: A study of five system logs. In *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)*. IEEE, 575–584.

[12] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep one-class classification. In *International conference on machine learning*.

[13] Risto Vaarandi and Mauno Pihelgas. 2015. Logcluster-a data clustering and pattern mining algorithm for event logs. In *2015 11th International conference on network and service management (CNSM)*. IEEE, 1–7.

[14] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I. Jordan. [n.d.]. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles - SOSP '09* (Big Sky, Montana, USA, 2009). ACM Press, 117. https://doi.org/10.1145/1629575.1629587

[15] Ting-Fang Yen, Alina Oprea, Kaan Onarlioglu, Todd Leetham, William Robertson, Ari Juels, and Engin Kirda. [n.d.]. Beehive: large-scale log analysis for detecting suspicious activity in enterprise networks. In *Proceedings of the 29th Annual Computer Security Applications Conference* (New York, NY, USA, 2013-12-09) *(ACSAC '13)*. Association for Computing Machinery, 199–208. https://doi.org/10.1145/2523649.2523670