Using Language Models to Pre-train Features for Optimizing Information Technology Operations Management Tasks

Xiaotong Liu(✉), Yingbei Tong(✉), Anbang Xu(✉), and Rama Akkiraju(✉)

IBM Research Almaden, San Jose, USA
{xiaotong.liu,yingbei.tong}@ibm.com, {anbangxu,akkiraju}@us.ibm.com

Abstract. Information Technology (IT) Operations management is a vexing problem for most companies that rely on IT systems for mission-critical business applications. While IT operators are increasingly leveraging analytical tools powered by artificial intelligence (AI), the volume, the variety and the complexity of data generated in the IT Operations domain poses significant challenges in managing the applications. In this work, we present an approach to leveraging language models to pre-train features for optimizing IT Operations management tasks such as anomaly prediction from logs. Specifically, using log-based anomaly prediction as the task, we show that the machine learning models built using language models (embeddings) trained with IT Operations domain data as features outperform those AI models built using language models with general-purpose data as features. Furthermore, we present our empirical results outlining the influence of factors such as the type of language models, the type of input data, and the diversity of input data, on the prediction accuracy of our log anomaly prediction model when language models trained from IT Operations domain data are used as features. We also present the run-time inference performance of log anomaly prediction models built using language models as features in an IT Operations production environment.

Keywords: AI for IT operations · Language modeling · Anomaly detection

1 Introduction

Information Technology (IT) Operations management is a vexing problem for most companies that rely on IT systems for mission critical business applications. Despite best intentions, designs, and development practices followed, software and hardware systems are susceptible to outages, resulting in millions of dollars in labor, revenue loss, and customer satisfaction issues. IT downtime costs an estimated $26.5 billion in lost revenue each year based on a survey of

H. Hacid et al. (Eds.): ICSOC 2020 Workshops, LNCS 12632, pp. 150–161, 2021.
https://doi.org/10.1007/978-3-030-76352-7_18

200 companies across North America and Europe [4]. The best of the analytical tools fall short of detecting incidents early, predicting when incidents may occur, offering timely and relevant guidance on how to resolve incidents quickly and efficiently and helping avoid them from recurring. This can be attributed to the complexity of the problem at hand. IT applications, the infrastructure that they run on and the networking systems that support that infrastructure, all produce large amounts of structured and unstructured data in the form of logs and metrics. The volume and the variety of data generated in real-time poses significant challenges for analytical tools in processing them for detecting genuine anomalies, correlating disparate signals from multiple sources, and raising only those alerts that need IT Operations management teams' attention. To add to this, data volumes continue to grow rapidly as companies move to modular microservices-based architectures, further compounding the problem. Furthermore, the heterogeneous nature of environments, where companies' IT applications can run on a mix of traditional bare metal, virtual machines, and public or private clouds operated by different parties, adds to the diversity of formats, platforms and scale that IT Operations management solutions must deal with. These complex and dynamic environments demand a new approach to IT Operations management that is fast, real-time, adaptive, customizable, and scalable.

The rise of Artificial Intelligence (AI) powered by the advancements in hardware architectures, cloud computing, natural language processing (NLP), and machine learning (ML), has opened up new opportunities for optimizing various industries and business processes. Operations management of IT systems is one such an area that is prime for optimization. AI can help IT Operations management personnel in detecting issues early, predicting them before they occur, locating the specific application or infrastructure component that is the source of the issue, and recommending relevant and timely recommendation actions based on mining prior issue records. All these analytics help reduce the mean time to resolve (MTTR) an incident, which in turn, saves millions of dollars by preventing direct costs (lost revenue, penalties, opportunity costs, etc.) and indirect costs (customer dissatisfaction, lost customers, lost references, etc.). Many IT Operations management vendors are starting to embed AI capabilities into their products. An advanced IT Operations management system needs to take all kinds of data as inputs, detect anomalies early, predict when incidents may occur, offer timely and relevant guidance on how to resolve incidents quickly and efficiently, automatically apply resolutions when applicable, and proactively avoid them from recurring by enforcing the required feedback loops into the various software development lifecycles. This can increase the productivity of IT Operations personnel or Site Reliability Engineers (SREs) and thereby improve the mean times to detect, identify and resolve incidents.

In this work, we present an approach to leveraging language models to pretrain features for optimizing IT Operations management tasks such as anomaly prediction from logs. Specifically, using log-based anomaly prediction as the task, we show that the machine learning models built using language models

(embeddings) trained with the IT Operations domain data as features outperform those AI models built using language models with general-purpose data as features. Furthermore, we present our empirical results outlining the influence of factors such as the type of language models, the type of input data, and the diversity of input data, on the prediction accuracy of our log anomaly prediction model when language models trained from IT Operations domain data are used as features. We also present the run-time inference performance of log anomaly prediction models built using language models as features in an IT Operations production environment.

The structure of the rest of this paper is as follows: Sect. 2 discusses the related works and techniques. Section 3 presents our method to pre-train features using language models for IT Operations, followed by our case study on log anomaly prediction in Sect. 4. We conclude the paper in Sect. 5.

2 Related Works

The notion of application of AI to optimize IT is often referred to as AI Operations (or AIOps in short) in the industry. Coined by Gartner [6], the field of AIOps is a specific space, stretching across several markets including Application Performance Management (APM), IT Operations Management (ITOM), IT Automation and Configuration Management (ITACM), and IT Service Management (ITSM) with a specific focus on AI-infusion. Research problems in this field include anomaly detection and prediction [10], incident management [16], fault localization [21], root cause analysis [20], and so on. Our work uses log-based anomaly prediction as the task to study the effects of pre-trained features from language modeling. Prior work in this space mainly relies on parsing stable logs, where the set of distinct log events is known and will not change over time [22]. However, in practice log data often contains previously unseen log events or log sequences. Furthermore, it can be challenging for conventional log parsers to adapt to different microservices since logs in each microservice may have their own context information. In this work, we leverage language models to pre-train features from IT Operations log data, which makes it easier to parse dynamically evolving logs in a cloud environment for anomaly detection and prediction.

Language modeling and embedding representation learning has been an active area of research in NLP, starting from word embeddings that map words in a vocabulary to vectors of real numbers so that words of similar semantic meaning are close to each other in the embedding space. Two language models are typically used to learn the word embedding representations: Continuous Bag of Words Model (CBOW) and Skip-gram. In the CBOW model, the distributed representations of context are combined to predict the word in the middle, while in the Skip-gram model, the distributed representation of the input word is used to predict the context. Word2vec [12] was the first popular embedding method for NLP tasks. The embeddings were derived from a Skip-gram model represented as a neural network with a single hidden layer. GloVe [13] learned the embeddings through dimensionality reduction on the co-occurrence counts

matrix. FastText [2] introduced the concept of subword-level embeddings, based on the Skip-gram model. Each word is represented as a bag of character n-grams, and their embeddings are the sum of vector representations associated with each character n-gram. Recent research in language modeling and deep learning has advanced contextualized embedding learning to address the issue of polysemous and the context-dependent nature of words. ELMo [14] extracts context-sensitive features from a bidirectional LSTM language model and provides additional features for a task-specific architecture. ULMFiT [7] advocates discriminative fine-tuning and slanted triangular learning rates to stabilize the fine-tuning process with respect to end tasks. OpenAI GPT [15] builds on multi-layer transformer [17] decoders instead of LSTM to achieve effective transfer while requiring minimal changes to the model architecture. BERT [5] uses bidirectional transformer encoders to pre-train a large corpus, and fine-tunes the pre-trained model that requires almost no specific architecture for each end task. In this work, we empirically investigate the impact of embeddings and language models pre-trained using IT Operations domain data for optimizing the domain-specific tasks.

3 Approach

In this section, we describe our approach to pre-training features using language models for optimizing different IT Operations management tasks.

3.1 Motivation

IT Operations environment generates many kinds of data. These include metrics, alerts, events, logs, tickets, application and infrastructure topology, deployment configurations, and chat conversations. Of these, metrics tend to be structured in nature while logs, alerts, and events are semi-structured, and tickets are unstructured data types. Also, among all the data types, logs and metrics sometimes can be leading indicators of problems, while alerts, tickets and chat conversations tend to be lagging indicators. The volume, the variety and the complexity of these data offers both challenges and opportunities in developing AI-infused analytical tools to optimize IT Operations management tasks. Leveraging language models to pre-train features from IT Operations domain data, we present a transfer learning approach that serves as strong a foundation to build AI models for various different IT Operations management tasks, such as log anomaly detection, fault localization, named entity extraction, similar incident analysis, and event grouping (as illustrated in Fig. 1).

3.2 Embedding and Language Modeling

Embeddings, also known as distributed vector representations, have been widely used in NLP and natural language understanding. In a pretrained embedding

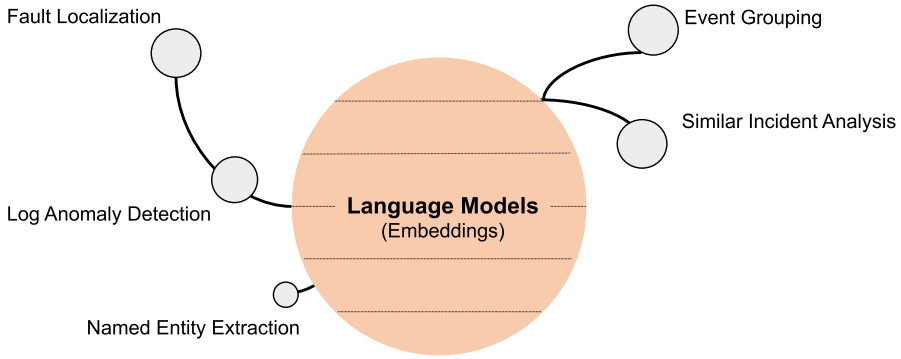

Fig. 1. An illustration of language models for different IT Operations management tasks.

space, words or phrases from the vocabulary are mapped to vectors of real numbers, and each is associated with a feature vector of a fixed dimension. Embeddings are pre-trained on large text corpus using a language modeling task, which assigns a probability distribution over sequences of words that matches the distribution of a language. After that, embeddings can be extracted from the pre-trained language model.

Typically, we can categorize embeddings into two types based on the language modeling approach in use: context-free embeddings and contextualized embeddings. Static word embeddings (e.g., Word2Vec, Glove, fastText) are context-free as the language models generate the same embedding for the same word even in different context. Deep pre-trained language models (e.g., ELMo, ULMFiT, BERT) can generate contextualized embeddings where the representation of each word also depends on the other words in a sentence (i.e., the context of the word). For the scope of this work, we select two representative embeddings from each type: fastText for context-free embedding and BERT for contextualized embedding.

3.3 Pre-training Language Models for IT Operations Management

Most existing embeddings available in the literature were created from text corpus in natural language such as Wikipedia pages and news articles. However, the text data generated in the IT Operations domain are different from natural language texts, as the vocabulary of the IT Operations domain is quite unique. For example, logs can contain a mix of the date, the time, the pod id, the level of logging, the component where the system runs, and the content of log message.

To pre-train language models in the IT Operations domain, we first process the input text data into a normalized format using predefined rules, extracting the most informative texts such as log messages, ticket descriptions and so on. We also remove duplicates of texts, which may be auto-generated multiple times by the system for the same event. Next, we randomly sample data

from each data source, and use the data samples to learn the vocabulary of the whole IT Operations domain. For fastText the vocabulary of words are learned when pre-training the language model. For BERT since the vocabulary has to be predetermined prior to the neural model training, we use sentencepiece [9] to learn the vocabulary of subwords. After that, we pre-train the language model using the sampled data, and tune the parameters based on model evaluation. An overview of the pre-training pipeline is shown in Fig. 2.

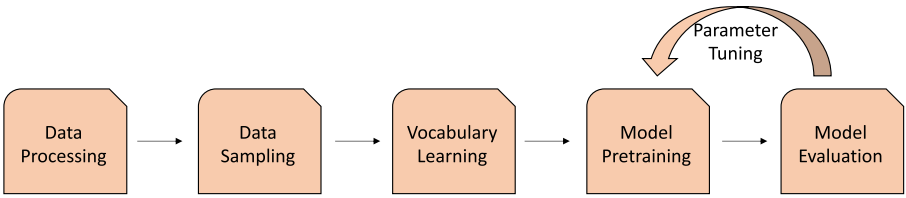

Fig. 2. The pipeline of pre-training language models using IT Operations domain data.

4 Case Study: Log Anomaly Detection

To demonstrate the feasibility of our approach, we describe a case study of using language models to pre-train features for building log anomaly detection and prediction models in IT Operations management.

4.1 Problem Statement

Anomaly detection from logs is one fundamental IT Operations management task, which aims to detect anomalous system behaviors and find signals that can provide clues to the reasons and the anatomy of a system's failure. Log messages are inherently unstructured, since system events can be recorded by developers using any text for the purpose of convenience and flexibility. Traditionally, log parsing is usually applied as a first step towards down-stream log analysis tasks to convert unstructured textual log messages into a structured format that enables efficient searching, filtering, grouping, counting, and sophisticated mining of logs. In particular, log templates are extracted from logs to represent an abstraction of log messages by masking system parameters recorded in logs. However, existing log parsing approaches are unable to adapt to evolving logs, making it challenging for continuous model improvement and customization. Furthermore, it is difficult to capture the semantic information in log messages with log templates.

4.2 System Overview

To tackle these challenges, we develop a system to perform anomaly detection and prediction based on pre-trained features from language models. An overview of our system is shown in Fig. 3. Our system consists of two subsystems: Off-line Training and Runtime Inference. The Off-line Training subsystem focuses on log parsing, embedding extraction and anomaly detector training. The input to this subsystem are randomly sampled log data and a pre-trained language model (fastText or BERT). We generate a vector representation for each log message: from fastText embeddings, we aggregate the embeddings from the words in a sentence using tf-idf weighting; from the pre-trained BERT model, we take its final hidden state of the token [CLS] as the aggregate sequence representation. To model the system behavior over time, we group the log messages of every 10 s time window based on the logging timestamp, and average the vectors of logs within each time window to form the feature vector. We learn a Principal Component Analysis (PCA) [18] transformation matrix from feature vectors in training data collected when the system was running in normal condition. The output of this subsystem are trained log anomaly detection models, which are saved to the storage repository Data Lake. The Runtime Inference subsystem checks if an anomaly occurs for a given time window during runtime. The input to this subsystem are the trained models from the Off-line Training subsystem in the Data Lake, as well as new logs in a streaming fashion through Kafka [8], a unified, high-throughput, low-latency platform for handling real-time data streams. Our system will predict anomaly if the feature vector of a new time window is sufficiently different from the normal space constructed by PCA. The output of this subsystem are anomalies detected from the logs.

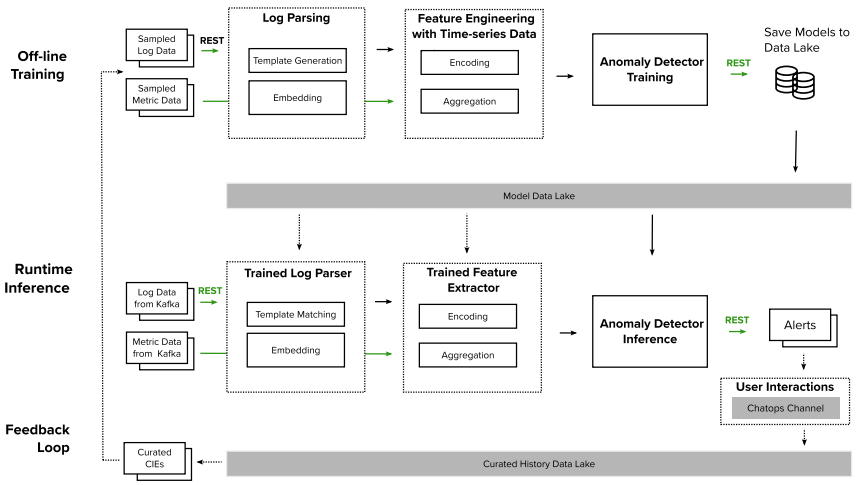

Fig. 3. An overview of our anomaly detection system built with embeddings from pre-trained language models.

Feedback on the anomaly detection results will be used to continuously improve the Off-line Training subsystem.

4.3 Evaluation

In our experiments, we explore the effects of three factors that may affect language model pre-training for IT Operations management: the type of language models (context-free or contextualized embeddings), the type of pretrained data (domain-specific data or general-purpose data), and the diversity of pre-trained data.

Datasets and Benchmarks. For pre-training the language models, we take 3 million sampled logs from 64 different applications: 48 cloud microservices in IBM Watson Assistant (WA) [1] (1 million logs), and 15 applications in Loghub collection [22] (2 million logs), ranging from distributed systems, supercomputers, operating systems to server applications. To test the accuracy of our trained anomaly detector models, we collect logs with ground truth from two IBM Watson Assistant microservices (denoted as *WA-1 and WA-2*) when the system was running in normal or abnormal condition, along with the HDFS ground truth data from Loghub. We compare the predictions with ground truth and compute the per-class accuracy as the percentage of correct predictions in the normal or abnormal test data set, respectively.

Accuracy Testing. We trained the following variants of anomaly detection models using different pre-trained features in our experiments:

- **Baseline**. Our baseline uses count vectors of log templates as feature vectors to build the model [19].
- **fastText-origin**. The model was built using features from the original pre-trained fastText embedding using general-purpose data [11].
- **fastText-wa**. The model was built using features from our fastText embedding pre-trained using IBM Watson Assistant data.
- **fastText-wa-loghub**. The model was built using features from our fastText embedding pre-trained using IBM Watson Assistant and Loghub data.
- **BERT-origin**. The model was built using features from the original pre-trained BERT using general-purpose data [5].
- **BERT-wa**. The model was built using features from our pre-trained BERT using IBM Watson Assistant data.
- **BERT-wa-loghub**. The model was built using features from our pre-trained BERT using IBM Watson Assistant and Loghub data.

In Table 1, we report the accuracy results of anomaly prediction on the benchmark datasets across various models. Firstly, we can see that the models trained with context-free embeddings consistently outperform our baseline as well as the models trained with contextualized embeddings. While our pre-trained fastText

embedding using the WA data is comparable to the original general-purpose fastText embedding, we get a significant boost in accuracy of 8% with our pre-trained fastText embedding using both the WA data and the Loghub data, compared with the general-purpose fastText embeddings. This demonstrates the effectiveness of using diversified domain-specific data for pre-training embedding features to enhance the log anomaly detection task.

On the other hand, we can see that our pre-trained BERT embedding with the WA data is also comparable to the original general-purpose BERT embedding, though both are slightly worse than the baseline. Surprisingly, when we introduce more diverse data from both the WA data and the Loghub data to pre-train BERT, the accuracy drops considerably. One possible reason is that log data are much simpler than the general purpose text data such as Wikipedia articles or news articles, and the context in log data is not rich enough for BERT to learn during model pre-training. Besides, since log anomaly detection is essentially an unsupervised learning task, it is difficult to fine-tune pre-trained BERT models as in supervised learning tasks.

Overall, our experiments show that context-free embeddings are more robust and effective than contextualized embeddings to pre-train features for the log anomaly detection task, and it is even better when pre-training the embeddings with the IT Operations domain data of diverse applications.

Table 1. Accuracy results on log anomaly detection with pre-trained features from embeddings.

Model	HDFS		WA-1		WA-2	Average
	Normal	Abnormal	Normal	Abnormal	Normal	
Baseline	99.99%	44.9%	93.3%	66.7%	95%	80%
fastText-origin	98.8%	66%	93.3%	100%	98.3%	91%
fastText-wa	98.8%	59.7%	93.3%	100%	98.3%	90%
fastText-wa-loghub	99.6%	99.9%	96.7%	100%	98.3%	99%
BERT-origin	2%	99%	93.3%	100%	98.3%	79%
BERT-wa	97.1%	52.7%	93.3%	100%	95%	78%
BERT-wa-loghub	96.8%	47.5%	93.3%	100%	40%	67%

Performance Testing. We test the performance of our log anomaly prediction models built using language models as features in an IT Operations production environment. The production environment is a cluster with 2 CPUs and 4Gb memory. The test data consists of 10,000 randomly sampled logs from the WA data. We consider pre-trained BERT models of different numbers of layers, as well as the one-layer fastText model.

In Table 2, we report the total time (in seconds) and the average speed (the number of log lines per second). We observed that as the BERT model gets more

complex with more layers, the average speed of embedding inference from the pre-trained model drops significantly. The fastText model is over 40 times faster than the one-layer BERT model since it is basically perform a lookup once the pre-trained embeddings are loaded into memory.

Table 2. Performance testing results on log anomaly detection with pre-trained features from embeddings.

	fastText	BERT		
	1 layer	1 layer	3 layers	6 layers
Total time	1.4 s	60 s	450 s	1700 s
Average speed	7000 lines/s	166 lines/s	22 lines/s	6 lines/s

5 Conclusion and Future Work

As IT complexity grows and the use of AI technologies expands, enterprises are looking to bring in the power of AI to transform how they develop, deploy and operate their IT. Pre-training a language model can accelerate the development of text-based AI models for optimizing IT Operations management tasks at a large scale. We investigate the effects of this language model pre-training approach for IT Operations management through a series of experiments on different language model types, data domains and data diversities. We present the empirical results on the prediction accuracy of our log anomaly prediction model and its run-time inference performance using language models as features in an IT Operations production environment. We show that the machine learning models built using context-free embeddings trained with diverse IT Operations domain data as features outperform those AI models built using language models with general-purpose data. Our pre-trained language models for IT Operations will be released soon. We hope that the insights gained from these experiments will help researchers and practitioners develop solutions and tools that enable better scalability, integration and management in the IT Operations domain. In the future, we will continue to explore the effects of pre-trained features using language models on different IT Operations management tasks such as fault localization and similar incident analysis. Besides, we plan to extend the studies to use more advanced language models such as GPT-3 [3].

References

1. I.W. Assistant (2020). https://www.ibm.com/cloud/watson-assistant/
2. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Trans. Assoc. Comput. Linguist. **5**, 135–146 (2017)

3. Brown, T.B., et al.: Language models are few-shot learners. arXiv preprint arXiv:2005.14165 (2020)
4. CA-Technologies: The avoidable cost of downtime (2010)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186 (2019)
6. Gartner (2017). https://www.gartner.com/en/newsroom/press-releases/2017-04-11-gartner-says-algorithmic-it-operations-drives-digital-business
7. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification. arXiv preprint arXiv:1801.06146 (2018)
8. Kafka (2020): https://kafka.apache.org/
9. Kudo, T., Richardson, J.: Sentencepiece: a simple and language independent subword tokenizer and detokenizer for neural text processing. arXiv preprint arXiv:1808.06226 (2018)
10. Meng, W., et al.: Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. In: IJCAI, pp. 4739–4745 (2019)
11. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., Joulin, A.: Advances in pre-training distributed word representations. arXiv preprint arXiv:1712.09405 (2017)
12. 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)
13. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
14. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)
15. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018). https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/languageunderstandingpaper.pdf
16. Sarnovsky, M., Surma, J.: Predictive models for support of incident management process in it service management. Acta Electrotechnica et Informatica **18**(1), 57–62 (2018)
17. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
18. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemom. Intell. Lab. Syst. **2**(1–3), 37–52 (1987)
19. 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)
20. Zhang, Y., Rodrigues, K., Luo, Y., Stumm, M., Yuan, D.: The inflection point hypothesis: a principled debugging approach for locating the root cause of a failure. In: Proceedings of the 27th ACM Symposium on Operating Systems Principles, pp. 131–146 (2019)

21. Zhou, X., et al.: Latent error prediction and fault localization for microservice applications by learning from system trace logs. In: Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 683–694 (2019)
22. Zhu, J., 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), pp. 121–130. IEEE (2019)