Reconstruction-based anomaly detection for the cloud - A comparison on the Yahoo! Webscope S5 dataset

Tanja Hagemann

Technische Universität Berlin
Telekom Innovation Laboratories
Service-centric Networking
Berlin, Germany
tanja.hagemann@tu-berlin.de

Katerina Katsarou

Technische Universität Berlin
Telekom Innovation Laboratories
Service-centric Networking
Berlin, Germany
a.katsarou@tu-berlin.de

ABSTRACT

The detection of anomalies in cloud metrics is an important way to identify suspicious data instances that indicate a system problem such as hardware failures, performance bottlenecks or intrusions. Yet, especially in a cloud computing infrastructure where the amount and variety of services is constantly increasing, it is getting more and more challenging to monitor and maintain the system manually. Thus, it is beneficial to use machine learning to detect anomalies at least partially in an automated way. The contribution of this paper is two-folded: firstly, we evaluate three unsupervised, reconstruction-based methods for anomaly detection (PCA, Autoencoder, LSTM-Encoder-Decoder) on the Yahoo! Webscope S5 benchmark dataset. Secondly, we compare our chosen models to a widely-used density-based approach and show that our reconstruction-based approaches outperform the related work.

CCS CONCEPTS

• **Computing methodologies** → **Anomaly detection**; • **Networks** → **Cloud Computing**; **Network monitoring**.

KEYWORDS

Anomaly Detection; Unsupervised Machine Learning, Deep Learning, Cloud Monitoring, AIOps

1 INTRODUCTION

The detection of anomalies in time series data is becoming increasingly important, especially since, on the one hand, more and more industries rely on the correct processing of data in their production chain. And on the other hand, the detection of anomalies in such data offers the possibility of early detection of any errors that indicate a major problem. In particular cloud-based services, which are indispensable in many industries nowadays, play a key role in this context. In the last years, the number of cloud-based products has risen to such an extent that the complexity of the underlying infrastructures increased exponentially. At the same time, a reliable and resource-optimized infrastructure is essential for both cloud users and cloud providers. Therefore, it is beneficial to evaluate novel methods from the field of machine learning and deep learning for the detection of anomalies in cloud metrics.

By definition [1], an anomaly is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism. These generating mechanisms are normally very domain-specific. In the case of network data [2] it can be for example intrusions, hardware failures or performance bottlenecks. The anomalies that in turn arise in monitoring metrics can be of different types and therefore require an appropriate data pre-processing and model selection to be detected. There are typically three main types of anomalies mentioned in the literature [3]: point anomalies, contextual anomalies and collective anomalies. Point anomalies in cloud metrics commonly manifest as global spikes in application latency or system resource utilization measurements, whereas contextual anomalies manifest either only under specific execution environments or as local spikes. A collective anomaly is defined by several accumulated observations each of which can be normal but their joint occurrence is anomalous, for example an unexpected streak of low-throughput values when compared to higher-throughput behaviour in the past.

The application of machine learning to detect these kind of anomalies in cloud-specific data is motivated on the one

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author.

ICCBDC'20, August 26–28, 2020, Virtual, United Kingdom

© 2020 Copyright is held by the owner/author(s).

ACM ISBN 978-1-4503-7538-2/20/08.

<https://doi.org/10.1145/3416921.3416934>

hand by the growing complexity of the underlying infrastructure which makes it almost impossible for human operators to represent all states by manual, deterministic rules. On the other hand, the patterns of cloud-specific data are caused by multiple interdependent components deployed in data centers servers with heterogeneous and equally interdependent resources. For detecting anomalies in such a constantly changing environment flexible methods are preferable. In principle, any supervised machine learning algorithm can be used to meet these requirements, but this would require a sufficient amount of well labeled data which is in practice hard to obtain. For this reason, we evaluate in this paper three *unsupervised, reconstruction-based* anomaly detection methods on the *Yahoo! Webscope S5* benchmark dataset: *Principal Component Analysis* (PCA) and two neural networks, an *Autoencoder* and an *LSTM-Encoder-Decoder*. This evaluation is the first contribution of our paper, as to our knowledge the related work has not considered them so far. Furthermore, as a second contribution we compare our results to a widely-used unsupervised, neighbour-based approach, namely *Local Outlier Factor* (LOF).

2 RELATED WORK

A general introduction to the topic of anomaly detection is given by [3]. Furthermore, [2] provides an overview about the application of machine learning for anomaly detection in computer networks. For the field of performance anomaly detection [4] is recommendable. Basically in any application domain, anomaly detection can be divided into two areas: statistical methods and machine learning. Statistical techniques fit a statistical model to the given data and then apply statistical inference to determine normal and anomalous data instances. Regression-based approaches for the detection of performance anomalies are for example [5], [6] and [7]. The key disadvantage of these approaches is their assumption of a particular data distribution.

This is counterbalanced by machine learning approaches which do not make any assumptions about the underlying distribution of data. In general, machine learning can be divided into supervised and unsupervised methods, which is also the case for anomaly detection in cloud metrics: in a supervised way it is fairly common to use decision trees [8–10] or Bayesian classifiers as proposed in [11–13]. These approaches rely heavily on accurately labeled data, and as an output produce class labels which may not be useful in cases where a continuous anomaly score is required.

Unsupervised approaches do not consider labels during training. However, it is useful to have them for evaluation purposes. For cloud metrics, LOF can be used as shown in [14–16]. This neighbour-based method is depending on a well-chosen distance measure and has a time complexity of $O(n^2)$. Its main drawback is the testing phase which is computationally intensive because for each new data point

a pairwise distance score has to be computed. Clustering-based approaches such as a Self-organizing Maps [17] or non-parametric, hierarchical clustering as in [18] are capable of identifying similar data points which in turn allows to mark outliers as anomalies. Again, most of these approaches highly depend on a proper distance measure while the test phase is fast since new data points are only compared with a few clusters.

Furthermore, PCA is prominent for its capability to reduce input dimensions. This property can be directly utilized for the detection of anomalies. A related approach from the area of anomaly detection in log files is for example [19]. The basic assumption here is to capture data regularities by selecting a proper amount of principal components which is relatively straightforward, especially when compared to the choice of a distance measure and appropriate amount of nearest neighbours as in the case of LOF. An additional advantage of PCA is its fast testing phase.

In the related work there are some approaches which are explicitly based on the Yahoo! Webscope S5 dataset [20]: In [21] PCA is used for dimension reduction and anomalies are detected by a density-based method which captures the main data patterns of each cloud metric to decide if a whole time series can be identified as anomalous. [22] uses a Variational Autoencoder as a generative neural network to detect anomalies in a multivariate fashion, while [23] is based on a statistical approach to fit a probability density function. Last but not least, [24] presents an online regression algorithm, [25] shows how to detect anomalies with a C-LSTM neural network in a supervised way and [26] deals with Restricted Boltzmann Machines to fit a probability distribution to the input data with the goal to identify anomalies by the stochastic free energy function.

To summarize there are plenty of different methods in place to detect the one or the other anomaly in cloud metrics. But in contrast to the methods we have chosen, the related work does so far not consider the potential of unsupervised, reconstruction-based models to identify anomalies based on a reconstruction error. In particular, models of this type have not yet been compared on the Yahoo! Webscope S5 benchmark dataset.

3 SELECTED ALGORITHMS

In this section we will provide background to all three reconstruction-based models and additionally introduce LOF since it is most widely used in the related work and will therefore later on serve as baseline in our comparison.

Local Outlier Factor

LOF was published by Breunig et al. [27]. Since then it is a widely used instance-based algorithm capable of identifying density-based anomalies. Its main advantage is the capability

Figure 1: LSTM-Encoder-Decoder architecture for unsupervised anomaly detection.

to detect local and global anomalies at the same time. Therefore, the concept of k -distance is introduced as the distance $d(\mathbf{x}, \mathbf{x}')$ between two vectors $\mathbf{x}, \mathbf{x}'$ from an input dataset $\mathbf{X}$ such that

- (1) for at least k vectors $\mathbf{y} \in \mathbf{X} \setminus \{\mathbf{x}\}$ it holds that $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{x}')$ and
- (2) for at most $k - 1$ vectors $\mathbf{y} \in \mathbf{X} \setminus \{\mathbf{x}\}$ it holds that $d(\mathbf{x}, \mathbf{y}) < d(\mathbf{x}, \mathbf{x}')$.

Then, the reachability distance of an vector $\mathbf{x}$ with respect to $\mathbf{x}'$ is given as

$$\text{reach-dist}_k(\mathbf{x}', \mathbf{x}) = \max\{k\text{-distance}(\mathbf{x}'), d(\mathbf{x}, \mathbf{x}')\} \quad (1)$$

and a local reachability density $lrd_k(\mathbf{x})$ is defined as the inverse of the average reachability distance of the vector $\mathbf{x}$ from its nearest neighbours. Finally, for identifying anomalies the local outlier factor

$$LOF_k(\mathbf{x}) = \frac{\sum_{\mathbf{x}' \in N_k(\mathbf{x})} \frac{lrd_k(\mathbf{x}')}{lrd_k(\mathbf{x})}}{|N_k(\mathbf{x})|} \quad (2)$$

is used as a measure for the degree to which $\mathbf{x}$ is abnormal.

Principal Component Analysis

PCA is a widely used method for dimension reduction and mathematically defined as an orthogonal linear transformation that transforms input data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on. Let $\mathbf{X}$ be the input vectors of dimension l . In general, PCA captures dominant patterns in data to construct a (low) r -dimensional normal subspace S_d in the original l -dimensional space. The remaining $(l - r)$ dimensions form the abnormal subspace S_a . For anomaly

detection this property can be utilized by computing a reconstruction error based on the Euclidean norm

$$\mathcal{L}(\mathbf{x}) = \|\mathbf{x}_a\|_2, \quad (3)$$

where $\mathbf{x}_a$ is the projection of $\mathbf{x}$ onto the abnormal subspace S_a , and can be calculated as $\mathbf{x}_a = (\mathbf{I} - \mathbf{P}\mathbf{P}^T)\mathbf{x}$, where $\mathbf{P}$ is formed by the first r principal components chosen by PCA. The higher this reconstruction error is, the more abnormal a data point is.

Autoencoder

The first reconstruction-based neural network we use for unsupervised anomaly detection is an *Autoencoder* which is also often referred to as "non-linear" PCA due to its non-linear activation functions and its capability to perform dimension reduction. The vanilla version of an Autoencoder is defined by an input layer $\mathbf{x} \in \mathbb{R}^l$, a hidden layer $\mathbf{h} \in \mathbb{R}^p$ and an output layer $\hat{\mathbf{x}} \in \mathbb{R}^l$ which is of the same dimension as the input layer. The main idea is that Autoencoders map the input data by

$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}), \quad (4)$$

to a compressed, hidden representation ($p < l$), where σ is an activation function, for example ReLU or sigmoid, $\mathbf{W}$ is a weight matrix and $\mathbf{b}$ is a bias vector. The resulting hidden data representation is then used to reconstruct the original input data by

$$\hat{\mathbf{x}} = \hat{\sigma}(\hat{\mathbf{W}}\mathbf{h} + \hat{\mathbf{b}}), \quad (5)$$

where $\hat{\sigma}$, $\hat{\mathbf{W}}$, and $\hat{\mathbf{b}}$ are defined analogously as in (4). The whole Autoencoder is trained by backpropagation where we choose mean squared error (MSE)

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) := \frac{1}{l} \sum_{i=1}^l \left(x_i - \hat{x}_i \right)^2 \quad (6)$$

as the loss function to be optimized.

Table 1: Autoencoder Hyperparameter.

Autoencoder	
Hyperparameter	Value
Input/Output dimension	5
Dimensionality of hidden layers	(3, 2, 3)
Activation functions	Sigmoid
Loss function	MSE
Optimizer	SGD

LSTM-Encoder-Decoder

The second reconstruction-based neuronal network we use for unsupervised anomaly detection is a Long short-term memory network [28] composed as an encoder-decoder architecture. The additional benefit compared to the basic Autoencoder introduced in 3 is the capability to learn temporal relations from the input data. To achieve this the *LSTM-Encoder-Decoder* [29] has a chain-like structure and is based on the concept of an LSTM cell which is consisting of four neural networks interacting with each other. Figure 1a shows one LSTM cell and its internal architecture. For each time step t three input values are considered: the actual input value of a time series x_t , the previous cell state c_{t-1} and the previous hidden state h_{t-1} . The cell state acts as a memory layer where at each time step information are removed or added by linear operations. The hidden states are the outputs of each cell. Learning is then performed step-wise: Firstly, a sigmoid function σ called the "forget gate layer" is used to check what information from the previous output should be neglected

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f). \quad (7)$$

In the next step it is decided what new information is going to be stored in the cell state. Therefore, another sigmoid layer, the "input layer", decides which values should be updated

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (8)$$

and a tanh layer creates a vector of new candidate values, $\tilde{c}_t$, that could be added to the cell state

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c). \quad (9)$$

Then, the current cell state is defined by

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (10)$$

and the output becomes

$$\begin{aligned} o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \\ h_t &= o_t * \tanh(c_t). \end{aligned} \quad (11)$$

Similar as the previously introduced Autoencoder, the encoder maps an input sequence to a final hidden state h_l

which is in turn used as the input sequence for the decoder. Figure 1b shows the LSTM-Encoder-Decoder architecture we use for anomaly detection. The model is trained by using (6) again as the loss function to be optimized.

Table 2: LSTM-Encoder-Decoder Hyperparameter.

LSTM-Encoder-Decoder	
Hyperparameter	Value
Number of LSTM cells	2-5
Dimensionality of h_l	2
Output activation function	tanh
Recurrent activation function	Sigmoid
Loss functions	MSE
Optimizer	Adam

4 EVALUATION

Yahoo! Webscope S5 Dataset

In the domain of anomaly detection for cloud computing infrastructures, there are almost no benchmark datasets publicly available. One exception is the Yahoo! Webscope S5 dataset¹. It consists of four data classes, each of which contains either a set of synthetic or real web traffic metrics tagged with anomalies. Table 3 gives an overview about all data classes. Data class A2 is the least complex one, where each synthetic time series is of length 1420 and each time step represents one hour of aggregated traffic. All metrics from this data class have a constant trend as well as a constant seasonality, only noise is added. On the contrary, all metrics in data class A3 and A4 are longer (1680 aggregated hours) and show a varying trend as well as three different seasonalities. Furthermore, the data classes differ with respect to the types of anomalies they contain: in A2 and A3 only contextual anomalies can be found. These anomalies can only be identified given a specific context, but not otherwise. In data class A4 additional change point anomalies are added. This type of anomaly indicates an anomalous behaviour on a more global scale, for example in terms of trend and seasonality. Data class A1 is the most diverse one due to its real nature. It consists of time series representing metrics of various Yahoo! services. In this case, anomalies are either of a contextual or collective nature which means that instead of isolated peaks or change points there are additionally whole sub-sequences of a metric consistently labeled as anomalous. Figure 2 shows some examples of the different data classes and their anomalies.

Before training our models, we normalize each metric by Z-score and MinMax scaling to make sure that the learning is not biased towards higher metric values. For Z-score scaling

¹<https://webscope.sandbox.yahoo.com/catalog.php?datatype=s>

Table 3: Summary of the four data classes from the Yahoo! Webscope S5 dataset. The contamination is the proportion of abnormal metric values in relation to all metric values of the entire data class. Note that all data classes are highly imbalanced.

Data class	Amount of metrics	Anomaly types (total frequency)	Contamination
A1 (real)	67	contextual and/or collective (1669)	0.0176
A2 (synthetic)	100	contextual (466)	0.0033
A3 (synthetic)	100	contextual (943)	0.0056
A4 (synthetic)	100	contextual (1045, containing 208 changepoints)	0.0062

(a) A4

(b) A1

Figure 2: Two examples of synthetic (a) and real (b) metrics, each of them from a different data class of the Yahoo! Webscope S5 dataset. Anomalies are marked in red. Note the different anomaly types, especially the changepoint anomaly (diamond, purple).

we compute for each metric x its mean value $\bar{x}$ and standard deviation σ_x . Then

$$x' = \frac{x - \bar{x}}{\sigma_x} \quad (12)$$

becomes the new, standardized metric. Additionally, we apply MinMax scaling which leads to

$$x'' = \frac{x' - \min(x')}{\max(x') - \min(x')}. \quad (13)$$

Since we are dealing with time series data, we use a sliding window of length $l = 5$ and a step size of 1 to transform each normalized metric into a set of feature vectors as input for our models. For the two neural network approaches we additionally over-sample our input data by duplicating each window several times to reach a critical amount of training data. Since each (real) metric shows a fairly unique behaviour, we train for all input datasets separately all four models.

Evaluation measures

In the domain of anomaly detection, we are mostly dealing with highly unbalanced datasets, which in turn require an appropriate choice of the evaluation measure. As widely used

in the related work, the F_1 -score is probably the best choice because, unlike accuracy, it combines precision and recall in such a way that it is not biased towards the majority (normal) class. We define the true positives (TP) as the amount of anomalies correctly detected. Furthermore, true negatives (TN) is the amount of windows that are identified as normal and indeed contain no anomaly. False positives (FP) are then those windows that are identified as anomalous but do not contain any anomaly and false negatives (FN) is the amount of anomalies that are not identified as such. Based on these quantities we define

$$Recall = \frac{TP}{TP + FN}, \quad (14)$$

$$Precision = \frac{TP}{TP + FP} \quad (15)$$

and finally

$$F_1\text{-score} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}. \quad (16)$$

Figure 3: F_1 -score distributions. Top row values are the averages marked with red stars.
1: LOF, 2: PCA, 3: Autoencoder, 4: LSTM-Encoder-Decoder.

As usual in the field of unsupervised anomaly detection, each of our four models returns a continuous score (Euclidean distance, MSE, LOF) that evaluates how far a data point is abnormal. The final decision to mark a data point as an anomaly is then always a trade-off between recall and precision. Thus, for each of our four models we iterate over a set of reasonable thresholds and compute the F_1 -score in each iteration step. The maximum F_1 -score gives us then the optimal threshold for the identification of anomalies.

Results

In order to select an appropriate number of principal components, we have analyzed the components variance explained for each metric of each data class. For the real metrics from data class A1 the first two principal components explain in average approximately 80% of the data variances. Since this result is even better for the synthetic metrics, we select for each metric the first two principal components to span its linear, normal subspace.

Figure 3 summarizes our evaluation results. Apart from a few exceptions, PCA detects almost all point and contextual anomalies in data classes A2 and A3 best, even slightly better than the baseline model LOF. However, for data class A4, the average performance of PCA appears to be declining, although it is still better performing than LOF. A detailed comparison of the corresponding F_1 -score distributions for data class A4 shows less variance for PCA than for LOF which makes PCA also the more robust approach. Additionally, our evaluation shows that when no change point anomalies are present in data class A4, PCA and LOF perform significantly

better (0.1 on average in terms of F_1 -score) than when change point anomalies are taken into account. This indicates that both models have difficulties in detecting anomalies which are visible on a more global scale.

The LSTM-Encoder-Decoder performs best for data class A2. Its F_1 -score distribution is similar to that of PCA, but has fewer outliers. Remarkable for this data class is the poor performance of the Autoencoder which is in this case not able to learn in a robust way. In contrast, for data classes A3 and A4 the Autoencoder is capable to learn patterns, but the model obviously overfits the input data in such a way, that it does also learn to reconstruct anomalies. Here, the LSTM-Encoder-Decoder performs better but still worse than PCA and LOF.

For the real metrics in data class A1, the reconstruction-based neural networks perform best. Here, LOF can on average no longer detect the anomalies sufficiently. The distribution of the corresponding F_1 -score in figure 4 shows a very high variance which makes LOF for this data class not robust at all, whereas PCA performs significantly better again with an average F_1 -score of 0.75. Our results show the Autoencoder performs best and most robust for these real metrics. An exemplary result for data class A1 is given by figure 4.

A comparison of our evaluation results to the related work dealing with unsupervised approaches on the Yahoo! Webscope S5 dataset is summarized in Table 4. All of our reconstruction-based models outperform the related work on average.

(a) Real metric (anomalies marked with red stars)

(b) PCA

(c) Autoencoder

(d) LSTM-Encoder-Decoder

Figure 4: Example of reconstruction-based anomaly scores for a real metric from data class A1.

Table 4: Comparison between the average F_1 values obtained by our models and results from the related work considering unsupervised models.

A1	
	F_1
ES-CVAE [22]	~ 0.58
SORAD [24]	0.67
LOF	0.55
PCA	0.73
Autoencoder	0.76
LSTM-Encoder-Decoder	0.74

5 CONCLUSION

In this paper we evaluated three unsupervised, reconstruction-based anomaly detection models (PCA, Autoencoder and LSTM-Encoder-Decoder) on the Yahoo! Webscope S5 dataset to analyze how far they can detect performance anomalies in cloud-specific metrics without using labels for training. Additionally, we compared our results to LOF as a widely-used standard unsupervised, density-based model. Our results show that for less complex cloud metrics, PCA is the most robust and fastest way to detect anomalies, whereas neural networks without regularization tend to overfit the data drastically.

On the most complex, real metrics from data class A1, the capabilities of neural networks come into play: due to the higher amount of parameters, the models are better able to

learn what normal data patterns are and do not overfit the data as in the case of less complex data structures. Here our evaluation shows that an Autoencoder architecture is most promising as it performs best, even better than unsupervised approaches from the related work.

6 FUTURE WORK

For the future work, we suggest to optimize the chosen neural networks by different regularization techniques such as dropout, $L2$ or denoising to reduce the amount of false negatives. Furthermore, it would be meaningful to see how to extend our models in such a way that they are capable to detect anomalies in an online fashion. This extension could also go hand in hand with ensembling methods and/or transfer learning.

REFERENCES

- [1] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.
- [2] Dhruva Kumar Bhattacharyya and Jugal Kumar Kalita. *Network anomaly detection: A machine learning perspective*. Chapman and Hall/CRC, 2013.
- [3] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.
- [4] Olumuyiwa Ibidunmoye, Francisco Hernández-Rodríguez, and Erik Elmroth. Performance anomaly detection and bottleneck identification. *ACM Computing Surveys (CSUR)*, 48(1):1–35, 2015.
- [5] Terence Kelly. Detecting performance anomalies in global applications. In *WORLD5*, volume 5, pages 42–47, 2005.
- [6] Qi Zhang, Ludmila Cherkasova, and Evgenia Smirni. A regression-based analytic model for dynamic resource provisioning of multi-tier applications. In *Fourth International Conference on Autonomic Computing (ICAC’07)*, pages 27–27. IEEE, 2007.
- [7] Hui Kang, Xiaoyun Zhu, and Jennifer L Wong. DAPA: Diagnosing application performance anomalies for virtualized infrastructures. In *2nd USENIX Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE 12)*, 2012.
- [8] Song Fu. Performance metric selection for autonomic anomaly detection on cloud computing systems. In *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011*, pages 1–5. IEEE, 2011.
- [9] Gueyoung Jung, Galen Swint, Jason Parekh, Calton Pu, and Akhil Sahai. Detecting bottleneck in n-tier it applications through analysis. In *International Workshop on Distributed Systems: Operations and Management*, pages 149–160. Springer, 2006.
- [10] Jason Parekh, Gueyoung Jung, Galen Swint, Calton Pu, and Akhil Sahai. Issues in bottleneck detection in multi-tier enterprise applications. In *2006 14th IEEE International Workshop on Quality of Service*, pages 302–303. IEEE, 2006.
- [11] Yongmin Tan et al. Online performance anomaly prediction and prevention for complex distributed systems. 2012.
- [12] Xiaohui Gu and Haixun Wang. Online anomaly prediction for robust cluster systems. In *2009 IEEE 25th International Conference on Data Engineering*, pages 1000–1011. IEEE, 2009.
- [13] Rob Powers, Moises Goldszmidt, and Ira Cohen. Short term performance forecasting in enterprise systems. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 801–807, 2005.
- [14] Tao Wang, Wenbo Zhang, Jun Wei, and Hua Zhong. Workload-aware online anomaly detection in enterprise applications with local outlier factor. In *2012 IEEE 36th Annual Computer Software and Applications Conference*, pages 25–34. IEEE, 2012.
- [15] Tao Wang, Jun Wei, Wenbo Zhang, Hua Zhong, and Tao Huang. Workload-aware anomaly detection for web applications. *Journal of Systems and Software*, 89:19–32, 2014.
- [16] Tian Huang, Yan Zhu, Qiannan Zhang, Yongxin Zhu, Dongyang Wang, Meikang Qiu, and Lei Liu. An lof-based adaptive anomaly detection scheme for cloud computing. In *2013 IEEE 37th Annual Computer Software and Applications Conference Workshops*, pages 206–211. IEEE, 2013.
- [17] Daniel Joseph Dean, Hiep Nguyen, and Xiaohui Gu. Ubl: Unsupervised behavior learning for predicting performance anomalies in virtualized cloud systems. In *Proceedings of the 9th international conference on Autonomic computing*, pages 191–200, 2012.
- [18] Li Yu and Zhiling Lan. A scalable, non-parametric anomaly detection framework for hadoop. In *Proceedings of the 2013 ACM Cloud and Autonomic Computing Conference*, pages 1–2, 2013.
- [19] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pages 117–132, 2009.
- [20] N Laptev and S Amizadeh. Yahoo anomaly detection dataset s5. URL <http://webscope.sandbox.yahoo.com/catalog.php>, 2015.
- [21] Rob J Hyndman, Earo Wang, and Nikolay Laptev. Large-scale unusual time series detection. In *2015 IEEE international conference on data mining workshop (ICDMW)*, pages 1616–1619. IEEE, 2015.
- [22] Suwon Suh, Daniel H Chae, Hyon-Goo Kang, and Seungjin Choi. Echo-state conditional variational autoencoder for anomaly detection. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 1015–1022. IEEE, 2016.
- [23] Olumuyiwa Ibidunmoye, Thijs Metsch, and Erik Elmroth. Real-time detection of performance anomalies for cloud services. In *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, pages 1–2. IEEE, 2016.
- [24] Markus Thill, Wolfgang Konen, and Thomas Bäck. Online anomaly detection on the webscope s5 dataset: A comparative study. In *2017 Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–8. IEEE, 2017.
- [25] Tae-Young Kim and Sung-Bae Cho. Web traffic anomaly detection using c-lstm neural networks. *Expert Systems with Applications*, 106:66–76, 2018.
- [26] Cristina Monni, Mauro Pezzè, and Gaetano Prisco. An rbm anomaly detector for the cloud. In *2019 12th IEEE Conference on Software Testing, Validation and Verification (ICST)*, pages 148–159. IEEE, 2019.
- [27] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.
- [28] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [29] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.