CloudDet: Interactive Visual Analysis of Anomalous Performances in Cloud Computing Systems

Ke Xu, Yun Wang, Leni Yang, Yifang Wang, Bo Qiao, Si Qin, Yong Xu, Haidong Zhang, Huamin Qu

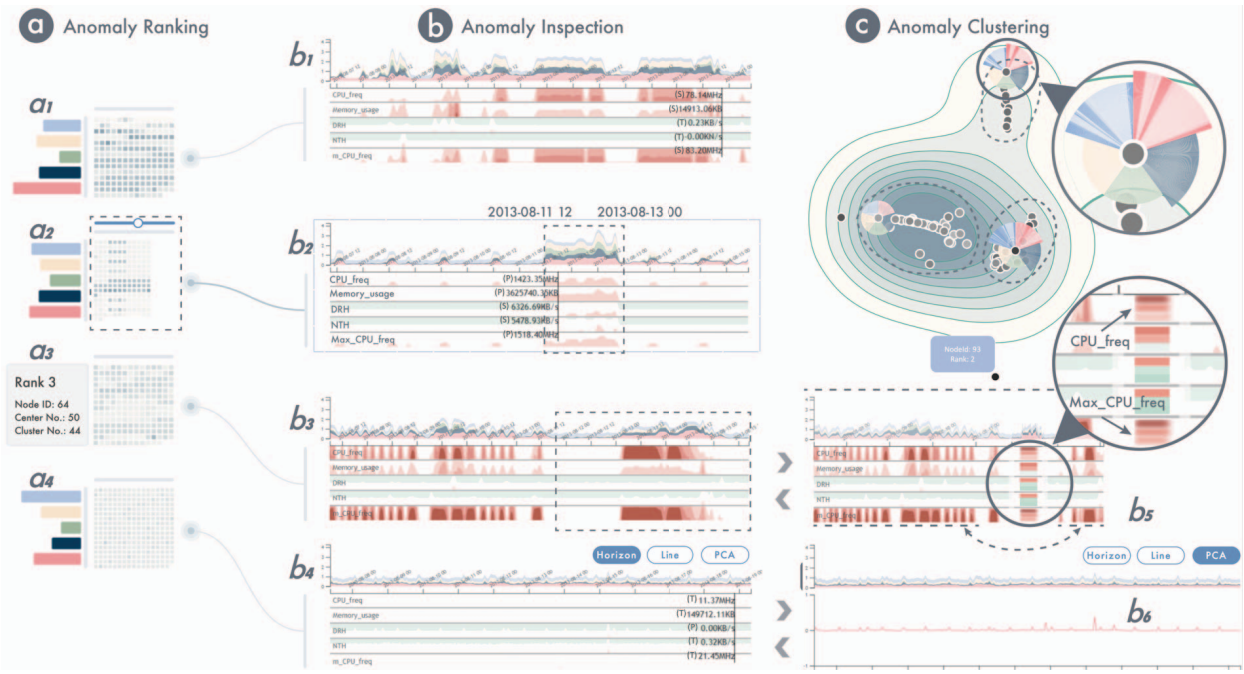

Fig. 1. CloudDet facilitates the exploration of anomalous cloud computing performances through three levels of analysis: (a) anomaly ranking, (b) anomaly inspection, and (c) anomaly clustering. The figure showcases some exploration results with Bitbrains Datacenter traces data. Node (b1) contains both short and long term spikes. Node (b2) shows a 12-hour periodic pattern for the performance metrics by observing the calendar chart in (a2), but encounters a spike in the process. Node (b3) shows many short-term and near-periodic spikes at the beginning and an abnormal long-term spike near the end. After collapsing the long-term one into a visual aggregation glyph in (b5), (b3) is updated and the latter temporal data "pop out", which shows a similar pattern as the beginning. Node (b4) shows a general periodic trend by using the PCA analysis in (b6). Most of the nodes are clustered into three groups in (c).

Abstract— Detecting and analyzing potential anomalous performances in cloud computing systems is essential for avoiding losses to customers and ensuring the efficient operation of the systems. To this end, a variety of automated techniques have been developed to identify anomalies in cloud computing. These techniques are usually adopted to track the performance metrics of the system (e.g., CPU, memory, and disk I/O), represented by a multivariate time series. However, given the complex characteristics of cloud computing data, the effectiveness of these automated methods is affected. Thus, substantial human judgment on the automated analysis results is required for anomaly interpretation. In this paper, we present a unified visual analytics system named CloudDet to interactively detect, inspect, and diagnose anomalies in cloud computing systems. A novel unsupervised anomaly detection algorithm is developed to identify anomalies based on the specific temporal patterns of the given metrics data (e.g., the periodic pattern). Rich visualization and interaction designs are used to help understand the anomalies in the spatial and temporal context. We demonstrate the effectiveness of CloudDet through a quantitative evaluation, two case studies with real-world data, and interviews with domain experts.

Index Terms—Cloud computing, anomaly detection, multidimensional data, performance visualization, visual analytics

1 Introduction

- *Ke Xu, Leni Yang, and Yifang Wang are with the Hong Kong University of Science and Technology. E-mail: {kxuak, lyangbb, ywangjh} @connect.ust.hk. This work is done when Ke Xu is an intern at MSRA.*
- *Yun Wang, Bo Qiao, Si Qin, Yong Xu, and Haidong Zhang are with Microsoft Research. E-mail: {wangyun, boqiao, Si.Qin, yox, haizhang} @microsoft.com. Yun Wang is the corresponding author.*
- *Huamin Qu is with the Hong Kong University of Science and Technology. Email: huamin@cse.ust.hk.*

Manuscript received 31 Mar. 2019; accepted 1 Aug. 2019.
Date of publication 16 Aug. 2019; date of current version 20 Oct. 2019.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2019.2934613

Cloud computing is becoming increasingly pervasive, with the extensive demand for big data analytics and discovery shifting many individuals and organizations towards cloud services. This move is motivated by benefits such as shared storage and computation service among a massive number of users. In order to maximally leverage the cloud, high availability and reliability are of utmost importance to the overall user experience. Therefore, it is important to monitor the compute nodes' usage and behavior, and then gain insights into the potential anomalous operations running in the cloud which might result in reduced efficiency or even downtime of the data center.

The most efficient way to detect and analyze anomalies in cloud systems is to monitor the general performance metrics of compute nodes (servers or virtual machines (VMs)), including CPU, memory, disk I/O, etc., in the form of a set of multivariate time series [39].

While monitoring applications hosted by these nodes also makes sense, the efficiency of this method is hindered by the scale, privacy and noise issues. Thus, tracking the performance metrics of compute nodes is more reasonable for exploring unusual behaviors. However, it is difficult to make automated anomaly detection with performance data in cloud systems. First, traditional methods for anomaly detection have mainly been approached through statistical and machine learning techniques [13, 27, 45], which face the inherent limitation that the ground truth required for training and performance evaluation is usually difficult to obtain. Second, the performance data generated by cloud systems are unstructured, and of high velocity. A very large-scale cloud data center often contains thousands of compute nodes that can record different metrics in a minute granularity. Third, the automated methods usually generate too many false positive results that aggravate the diagnosis burden in real-world scenarios. These characteristics stress the need for a more scalable and flexible way to identify and interpret anomalies in cloud computing.

Data visualization facilitates the analysis and evaluation of anomaly detection results via rich interaction techniques and intuitive representations of contextual information that provides additional evidences to support or refute the analysis conclusions [4, 38, 57]. However, most existing solutions are not specifically designed for anomaly detection in cloud computing, or are limited in capability for exploring large-scale multivariate time-series data. The main challenges challengess include: (1) Scalability: given the scale of compute nodes in cloud systems, there is a need to clearly show the anomalous patterns and to optimize the trade-off between system scalability and level-of-detail (LoD) of the performance data. (2) Interpretability: it is difficult to intuitively represent anomalous patterns and their corresponding raw data context with different semantics. (3) Multi-dimensionality: to comprehensively understand the data and anomalies, multi-faceted patterns should be conveyed, such as the temporal patterns of a compute node in terms of multiple metrics (e.g., the correlation between CPU and memory usage), or the distribution patterns among different nodes.

To address the challenges, we introduce CloudDet, an interactive visualization system to help cloud service providers efficiently detect and diagnose anomalies in large-scale cloud computing systems with the system-level performance data. Major research contributions include:

- **System.** We propose an integrated visual analytics system for the interactive exploration, detection, and diagnosis of anomalies with large-scale performance metric data in cloud computing systems. We formulate the design requirements through cooperation with cloud operations engineers. An unsupervised algorithm is proposed to facilitate the evaluation of anomalies based on the captured change patterns in a time series with ensemble analysis. The visualization and interaction designs support the visual analysis of anomalies at three levels: anomaly overview, ranking, and diagnosis.
- **Visualization and Interactions.** We propose a set of visualization and interaction designs to facilitate users' ranking, inspection and perception of most abnormal performances in cloud computing data. Specifically, we extend horizon graphs to make it visually scalable for displaying the overall pattern and making comparisons of different metrics. A glyph based on visual aggregation technique is introduced in the horizon graph to support a nonlinear time scaling. Another glyph is designed for data abstraction and anomaly highlighting in the spatial distribution context. Rich interaction designs are used in different views to promote data exploration.
- **Evaluation.** The effectiveness of CloudDet is demonstrated in multiple forms of evaluation. We first conduct a quantitative performance evaluation of the anomaly detection algorithm. The results show that our algorithm outperforms baseline algorithms in terms of accuracy with a comparable scalability. We then describe how CloudDet works through two case studies with real-world datasets, and collect feedback from experts in the cloud computing domain. All the evaluations validate the effectiveness of CloudDet in analyzing anomalous performances with cloud computing data.

2 Related Work

In this section, we summarize the techniques that are most related to our work, including anomaly detection algorithms and visualizations for anomaly detection and temporal data.

2.1 Anomaly Detection

Anomaly detection has been extensively studied in recent years. Existing methodologies include classification-based algorithms (either supervised [24, 37, 56] or semi-supervised [15, 35]) , statistics-based algorithms [5, 61], distance-based algorithms [6, 8, 23], and spectral-based algorithms [48]. Different approaches have been taken to address the problems with time-varying, multivariate data [22], which is the type of performance metrics data in cloud computing systems. For example, one major category employs regression model-based algorithms to determine the anomaly score of time-series data [1, 20], and these have been extended to many subsequent models like the autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA). Although these regression-based algorithms are useful to deal with most applications with various data types, they have a limited capability in cloud computing systems due to the huge volume and variation of data.

A variety of anomaly detection techniques have been specifically designed for performance anomaly detection in cloud computing systems [2, 28, 45, 53]. Some approaches deal with specific types of problems in cloud systems. For example, Xu et al. focused on the anomalies during the execution of sporadic operations [59], and Yu et al. presented a diagnosis framework for Hadoop environment [62]. More general algorithms without domain-specific assumptions have also been introduced recently, such as techniques developed based on the statistical techniques [51, 52], entropy-based models [41] and isolation-based algorithms [10]. However, most of them are computationally . By contrast, scalable algorithms have also been proposed to facilitate the anomaly detection in cloud computing, e.g., implementing a probabilistic approach to detect abnormal software systems [46], adopting Holt-Winters forecasting to identify a violation in application metrics [30], and implementing a clustering method to find the anomalous application threads [63]. A common issue of these scalable techniques is that they require low-level access to application level information, while our approach only targets general performance metrics that can be obtained via sampling the state of the system. Another crucial limitation of these approaches is that many of them are specified for different anomaly patterns in data [10, 59], lacking a systematic and unified way to detect anomalies based on data features and to interpret the results derived from these techniques.

In this paper, we design an unsupervised algorithm that uses different strategies to evaluate anomalous performances based on their change patterns. The algorithm is also integrated in our system to help interpret normal and abnormal cases situated in cloud performance data.

2.2 Visualization for Anomaly Detection

Although the aforementioned algorithms generate numeric results of anomalies, they are limited in providing interpretation of the anomalies. The problem is further aggravated when there is a lack of a clear boundary and ground truth for normal/abnormal cases for evaluation. Therefore, the domain knowledge of human experts need to be involved in judging the cases. Visualization techniques have been applied to support interpretation and facilitate better decision making. In the traditional approach, statistical diagrams such as line charts and histograms are commonly used to represent anomalous trends in the raw data [31,33,34]. A variety of dimensionality reduction (DR) techniques are also applied for understanding how the data distribute in a multidimensional feature space, such as multidimensional scaling (MDS) [32], principal component analysis (PCA) [48] and t-distributed stochastic neighbor embedding (t-SNE) [36]. However, a crucial limitation is that these DR methods have a narrow focus that can not provide a systematic approach for other complicated applications.

Recently, more-unified visual analytics systems for anomaly detection have been proposed. For example, systems have been developed for temporal, multivariate data [49,58], for applications such as monitoring streaming traffic data [11], detecting the spreading of rumors [64]. More specific visualizations for performance analysis in cloud computing systems have also been introduced [18, 43]. In a very recent work, Cong et al. [57] proposed a visual analytic approach to detect anomalous executions with their call stack tree (CSTree) representation in high performance computing applications. However, it detects anomalies in the cloud system with event sequence data, like the log events,

which is application-level analysis rather than the general system-level performance analysis.

In our work, we analyze anomalous compute nodes by tracking the general, time-varying performance metrics (profile data) in cloud computing systems, such as CPU load, memory usage, and disk read/write load, etc. Novel visualization and interaction techniques are also introduced to help users interactively identify the anomalies by inspecting different performance metrics and the correlations among them.

2.3 Visualization for Temporal Data

The performance data tracked in cloud computing systems are represented by multivariate time series. Given the ubiquity of temporal data, researchers have broadly studied their visualization as applied to various applications [47]. Several surveys reported the state-of-the-art multivariate time series visualization techniques. Müller et al. [40] discussed the general aspects of time-dependent multivariate datasets and categorize the visualization techniques into static representations and dynamic representations . Aigner et al. [3] summarized the visual methods for analyzing temporal data from three aspects: temporal data abstraction, principal component analysis (PCA), and clustering.

Among the myriad approaches to temporal data visualization, multivariate time series are most relevant to our work. These approaches can be classed into four types [17,29]: (1) Line charts are the most common method to visualize the progression of values over time. Recently, Muelder et al. [39] designed behavioral lines to analyze the behavior of cloud computing systems by bundling different performance metrics over time [39]. However, placing multiple lines in the same space will reduce the fidelity of individual time series. (2) Stacked graphs are another approach which provides a summation view stratified by individual series [9]. Projects like NameVoyager [54] and sense.us [26] applied stacked graph to explore demographic data. The problems with stacked graph include difficulty in comparing individual series and misinterpretation of the space between curves. (3) Horizon charts are a space-efficient technique for time series visualization. Such charts were first proposed by Saito et al. [44], and then optimized in terms of graphical scalability and perception by Few et al. [19] and Heer et al. [25]. (4) More recently, many glyph-based designs have been developed to visualize the time series data in various applications [21]. For example, TargetVue [12] introduced three different glyph designs to detect anomalous user behaviors in their communication activities. Glyphs are an appropriate choice due to their effective use of screen space and expressiveness for temporal performance summary.

In our work, we enhance horizon charts by combining a glyph design that can visually aggregate the performance data to support the visual scalability in the time domain, thus, facilitating the comparison of focused data and the identification of anomalous nodes in large-scale cloud computing systems. Another glyph is designed to help the anomaly comparison based on the similarity of compute nodes. Various interaction techniques are introduced in our system to facilitate the exploration of multivariate time series from cloud computing systems.

3 System Overview

CloudDet was developed as part of a one-year anomaly detection project. This project is aimed at addressing the aforementioned challenges (scalability, interpretability, and multi-dimensionality) and satisfying real-world requirements formulated by three domain experts in anomaly detection and cloud system analysis. Weekly meetings were held for two months, at which detailed system design requirements were clarified by the discussion of anomaly detection problems in both cloud computing and visualization. One pilot system was developed to provide a rough evaluation of the anomaly detection algorithms. Below we describe the most critical design requirements (**R1–R4**) developed during these discussions.

R1 Support large-scale cloud computing data. The system should allow effective analysis and exploration of large-scale cloud performance data, enabling users to examine the anomalous cloud computing behaviors collected from thousands of compute nodes.

R2 Facilitate anomaly detection processes. Automatic anomaly detection algorithm should be integrated into the system to support the efficient identification of anomalies in historical data.The visual encoding should efficiently rank and highlight the abnormal nodes, as well as show when and how the anomalies occurred.

R3 Enhance the anomaly inspection with multifaceted pattern discovery. The system should create easy-to-understand designs to connect the anomaly detection results with auxiliary information from the raw performance data to form a semantic background against anomaly instances.

R4 Enable users to explore the data interactively. To facilitate expert users analyzing the detection results, it is necessary to incorporate flexible interactions that help them quickly explore a substantial number of performance tracking data.

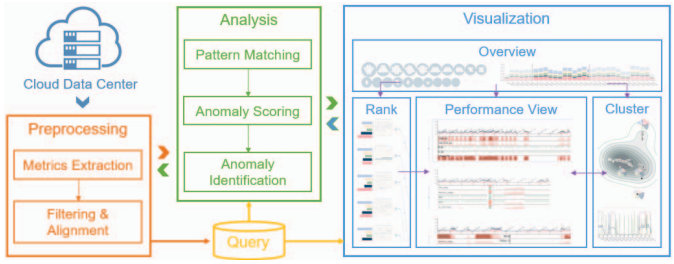

Fig. 2. CloudDet system overview and data processing pipeline.

Based on these requirements, we have designed CloudDet, an interactive data exploration system that enables experts to visually identify and analyze the anomalies in cloud computing performance. Fig. 2 illustrates the system architecture and the interactive analysis processing pipeline. The system consists of four major steps: (1) the data collection module, (2) the data preprocessing module, (3) the analysis module and (4) the visualization and interaction module. In particular, the data collection module collects the large-scale cloud computing performance data (**R1**). The data processing runs on the database, where different compute nodes' performance metrics are aligned and transformed into time-series data for anomaly scoring (**R2**). The analysis module runs an unsupervised anomaly detection algorithm to detect suspicious nodes and their abnormal periods for different metrics according to their temporal patterns, and ranks these nodes based on their anomaly scores (**R2**). The visualization and interaction module displays anomalous nodes, together with the corresponding contexts of raw metric data within several views. It provides a comprehensive visual summarization and interpretation of different patterns in cloud computing performance (**R3**). Rich interaction designs are created to support efficient anomaly exploration and diagnosis (**R4**). All these modules work together to form a scalable mechanism that enables an efficient procedure to reduce the information seeking space.

4 Time Series Anomaly Detection

In this section, we introduce a novel unified algorithm that detects anomalies in time series based on the three important anomaly patterns from a business perspective (Fig. 3). The three components are numerically represented by integrating the periodicity detection results with STL decomposition. Then we use an ensemble-based method to to calculate the anomaly score of time-series data with respect to the three patterns in the data. The key procedures are described below.

4.1 Pattern Matching

The input time series, which are the metric tracking data of each compute node, are independently transformed into a set of historical data: for every data point d_n in the time series ($n \geq L$), the previous L points construct the temporal data $\mathbf{D} = \{d_{n-L}, d_{n-(L-1)}, ..., d_{n-1}, d_n\}$, where L is a user-defined parameter, as the number of recent historical data. Then the historical data at different points are fed into the pattern matching module, which contains the two following steps.

Periodicity Detection. As shown in Fig. 3(1), the first step is to detect the structural periodic changes and estimate the periodicity for each piece of historical data, which is an important parameter for the following steps. Here we employ a non-parametric, two-tier approach that provides both a high resolution and low false positive rate for

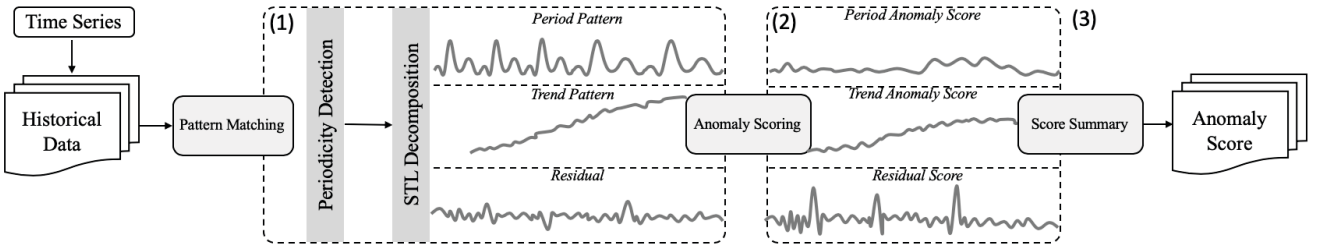

Fig. 3. Algorithm pipeline: (1) pattern matching for time series, (2) anomaly scoring, and (3) anomaly score summary.

periodic identification [50]. Specifically, the periodogram is firstly used to extract the candidate period, which is formulated as

$$P(f_{k/N}) = ||X(f_{k/N})||^2, \;\; k = 0, 1 ... \lceil \frac{N-1}{2} \rceil, \tag{1}$$

where P is the periodogram, indicating the expected power at each frequency $f_{k/N}$ (or equivalently at period N/k) of the input time series. The larger it goes, the more possible N/k is the candidate periodicity. $X(f_{k/N})$ is the Discrete Fourier Transform of the input time series at frequency $f_{k/N}$. However, the periodogram might provide false candidates or coarse estimation of the period due to the discrete increasing of frequency k/N. Therefore, a verification phase, autocorrelation (ACF), is required to make a more fine-grained estimation of the potential periodicities. ACF can examine the similarity of a time series to its previous values for different τ lags by calculating the following convolution:

$$ACF(\tau) = \frac{1}{N} \sum_{n=0}^{N-1} d(\tau) \cdot d(n+\tau), \;\; \tau = 0, 1 ... \lceil \frac{N-1}{2} \rceil. \tag{2}$$

If the candidate period from the periodogram lies on a hill (i.e., the local maximum) of ACF, we consider it as a valid period. Thus, for the historical data at the nth timestamp, we obtain the closest estimation of the period, denoted as T_n, which will be used as the parameter in the next step, and for anomaly scoring at the last step. [1]

STL Decomposition. The historical data is further split into three components by the seasonal-trend decomposition procedure based on loess (STL) [16], which is aimed to extract three temporal components for the anomaly score calculation. STL is proved to be fast and can cover most anomaly patterns in temporal cloud computing data. In addition, the three derived components are complementary in change patterns, and consistent in their unit and interpretability. We use the additive model of STL:

$$d_n = S_n + Tr_n + R_n, \;\; n = 1, 2 ... N, \tag{3}$$

where d_n, S_n, Tr_n and R_n means the data point, periodic component, trend component and residual component, respectively. Critical STL parameters for each historical data are (1) n_p, the number of observations per periodic cycle, which is T_n for daily data and $24T_n$ for hourly data; (2) n_i, the number of passes of the inner loop, which is 1; (3) n_o, the number of passes of robustness iterations of the outer loop, which is 5; (4) n_l, the smoothing parameter of the low-pass filter, which is $[n_l]_{odd}$; (5) n_s, the smoothing parameter of the low-pass filter, which is set as 15; and (6) n_t, the trend smoothing parameter, which is $[1.67 n_p]_{odd}$.

4.2 Anomaly Scoring

After getting the estimated period value (T_n) for each historical data and the numeric value (S_n, Tr_n and R_n) for the aforementioed three components, the respective anomaly scores (ASs) of three patterns are calculated in the anomaly scoring process based on their following detectors (as shown in Fig. 3(2)), expressed as follows:

(1) Periodic:

$$AS_{periodic} = min\left(\frac{|T_n - T_{n-1}|}{T_{n-1}}, 1\right), \tag{4}$$

where T_n and T_{n-1} denote the detected periods at the nth and $(n-1)$th timestamp at the periodicity detection step, respectively. As such, $AS_{periodic}$ describes the seasonal level shift.

[1] https://lukeluker.github.io/supp/PeriodicityDetection.py

(2) Trend:

$$AS_{trend} = min\left(\left|\frac{K_n - K_{n-1}}{K_{n-1}}\right|, 1\right), \tag{5}$$

where K_n is defined as the estimated slope of the trend component sequence $\mathbf{Tr} = \{Tr_{n-L}, Tr_{n-(L-1)}, ..., Tr_{n-1}, Tr_n\}$ through linear regression, and K_{n-1} represents the counterpart at the $(n-1)$th timestamp. Similarly, the trend level shift is indicated by AS_{trend}.

(3) Spike:

$$AS_{spike} = min\left(\left|\frac{R_n - \mu_{n-1} - 3\sigma_{n-1}}{3\sigma_{n-1}}\right|, 1\right), \tag{6}$$

where μ_{n-1} and σ_{n-1} denote the mean and variance with respect to the residue set $\mathbf{R} = \{R_{n-L}, R_{n-(L-1)}, ..., R_{n-1}\}$. This metric is used for the evaluation of unexpected outlier.

We take the minimum differences as scores for equations (4–6) so to scale the anomaly score into [0,1], with 1 as the most anomalous situation. The final anomaly score at the nth timestamp can be calculated by

$$AS = f(AS_{periodic}, AS_{trend}, AS_{spike}), \tag{7}$$

where the aggregation function f can be customized by various selectors, such as *min*, *max*, and different *weighted average*, etc. Our algorithm is incremental; i.e., at every timestamp, it only utilizes the most recent data, rather than long-term data, to update results. Long-term data are not reasonable as the data distribution may be changed by various operations and human interventions. By using only the most recent data, our model can quickly accommodate itself to data distribution drifts. In addition, the AS indicates a node's anomaly degree at every timestamp so that engineers are able to rank the nodes efficiently and reduce their efforts in anomaly detection and diagnosis. In the future work, we need a more optimal method to aggregate the anomaly scores of three components. Moreover, the algorithm is only applied to one metric data now, so the correlations among different metrics of a node can be considered to incorporate in the process.

5 Visualization

This section presents the design considerations of visualization components derived from the discussions with our expert users in Section 5.1 and a brief summary of the interface in Section 5.2, followed by the technical details of each component from Section 5.3 to 5.8.

5.1 Design Tasks

A list of design tasks was formulated to guide the visualization designs based on the requirements outlined in **R1–R4**. We discussed with the experts about the difficulties in determining the abnormal performance and the design ideas that are able to solve the problem. For example, the most commonly mentioned difficulty is to scale the large-size data and connect all the relevant information of the detected anomalies together to help experts decide whether it is a true anomaly or not. The traditional methods usually generate too many false positive results. To solve these challenges, the experts came up with some initial ideas (e.g., the stacked line charts) for visualization. In general, they desired a tool that can facilitate the identification, interpretation, and reasoning of detected cloud computing anomalies, thereby improving the maintenance of cloud centers. Guided by these considerations, we decided on a list of visualization tasks for two main purposes: (1) facilitating the exploration of data and the identification of anomalies and (2) enhancing the rationalization of the detected anomalies. We summarize these design tasks as follows.

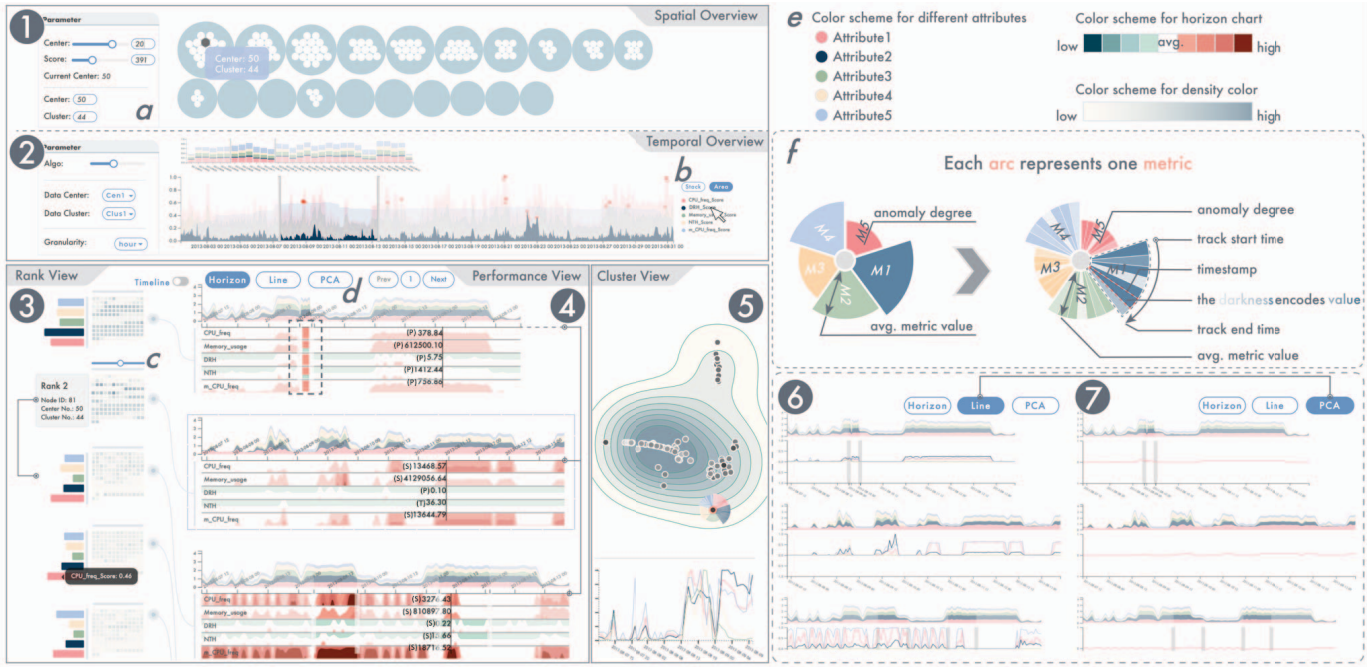

Fig. 4. The CloudDet system contains five interactive modules: (1) *Spatial Overview*, (2) *Temporal Overview*, (3) *Rank View*, (4) *Performance View* in *the horizon chart* mode, and (5) *Cluster View*. *The performance view* contains two other views/modes: (6) *the multi-line mode* and (7) *the PCA mode*. Users can select and filter data in (a), switch to different visualization modes in different views by buttons (b) and (d), and change the layout by the slider bar in (c). (f) is the explanation of the glyph in (5). (e) shows the color schemes used in different views.

T1 Show the overview of anomaly detection results for data query. A large-scale cloud computing system usually contains multiple data centers, with each center containing hundreds of clusters that host tens of thousands of servers. Hence, it is critical to provide visualization techniques that can summarize the cloud computing performances and anomaly detection results to help experts narrow down the search space.

T2 Rank the suspicious computing nodes dynamically. To reduce the search effort for suspicious nodes, visualization should be designed to aid the searching and filtering of anomalous performances by ranking the nodes whilst preserving the time context, e.g., displaying the periodic pattern of the operating activities.

T3 Browse the data flexibly in multiple ways. Despite the importance of the anomaly scores and ranks, the raw performance data that contain different metrics are of most concern for experts to identify an anomaly case. Therefore, rich interaction techniques should be designed to enable an efficient mechanism to explore the performance data and extract their anomalous patterns.

T4 Facilitate anomaly detection and interpretation. The visualization design of the system should consider the combination of the numeric anomaly detection results with the performance metric data. The visualization and interaction should enable users to make comparisons and display correlations over performance metrics at different levels from data summarization to focused context.

T5 Display the similarity of different computing nodes. In addition to displaying the temporal patterns, another key to understanding the anomalous cloud computing performance is to differentiate anomalous compute nodes from normal ones. To this end, the system should show the clustering of the nodes based on their similarities, revealing some common features that led to the anomaly.

5.2 User Interface

Guided by the above design tasks and the experts' feedback, we designed our visualization and interaction modules. The user interface (UI) of the CloudDet system, as shown in Fig. 4, consists of five modules: *the spatial overview* (Fig. 4(1)), indicating the general anomaly degree based on the cloud system hierarchy, from data centers to their sub-level data clusters; *the temporal overview* (Fig. 4(2)), displaying

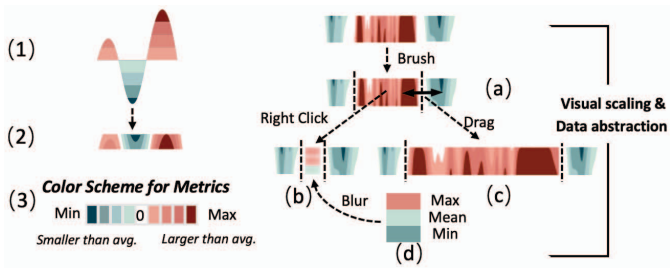

Fig. 5. (1)–(3) show the standard horizon chart design [25]. The horizon chart can be extended by: (a) selecting a specific period by brushing, and then (b) visually collapsing this period into a glyph for data abstraction and time scaling, or (c) stretching this period for detailed inspection.

the anomaly distributions over time for data filtering (**T1**); *the rank view* (Fig. 4(3)), showing the highly suspicious computing nodes in the descending order of their anomaly scores (**T2**); *the performance view* (Fig. 4(4)), associating the anomaly detection results with the raw performance metrics data and facilitating the anomaly inspection and correlation analysis via three visualization modes (the other two modes are *the multi-line mode* in Fig. 4(6) and *the PCA mode* in Fig. 4(7)) (**T3–5**); *the cluster view* (Fig. 4(5)), summarizing the activities of all the computing nodes (both normal and abnormal) and clustering them based on their similarity terms of data features (**T6**). All the views are interactively connected, illustrating different contexts of a set of top suspicious computing nodes. Different color schemes are designed to illustrate the different information (Fig. 4(e)). In particular, categorical colors are used to represent the information related to each of the performance metrics. A linear color scheme, ranging from white to grey, is used for indicating the anomaly score from low to high, while a dichotomous color scheme, ranging from dark blue, to white, to dark red, encodes the raw data values of the performance metrics, with red/blue encoding a value higher/lower than the average (white). More design details of each view are introduced in the following sections.

5.3 Initial Inspection and Data Selection

The CloudDet system contains two views for initial anomaly analysis and data query: (1) *the spatial overview* displays the overall anomaly

distribution based on the hierarchy of the cloud computing system; (2) *the temporal overview* depicts the variance of the summarized anomaly score with time. Both of them allow users to select a more detailed data subset for in-depth anomaly inspection (**T1**).

5.3.1 Spatial Overview

The spatial overview assists users to observe the general anomaly degree and query the data of a cloud computing system hierarchically (from data center to sub-level data cluster) through a bubble chart. Firstly, as shown in Fig 4(1), each blue outer bubble represents a data center and its interior white bubbles represent data clusters belonging to that center. According to the experts' feedback, abnormal instances are relatively rare and they hoped the system could directly show them the most likely anomalies. Thus, *the spatial overview* arranges data centers according to their abnormal scores in descending order, from left to right and top to bottom. The bigger a bubble is, the larger its anomaly score is. The calculation of the anomaly score is to sum the scores of all nodes in that center. In addition, the inner bubbles will only appear when their represented data clusters' sum of anomaly scores is larger than the human-set threshold. Both the number of data centers and the threshold value for data clusters can be set through two sliders (Fig. 4(a)). Based on the displayed information, users can query the data from a specific data center and cluster in two ways: (1) clicking on a white bubble of interest (the tooltip will show the name of the corresponding data center and cluster) to select the tracking records from its represented data cluster; (2) inputting a specific data center and data cluster ID by using the input boxes.

5.3.2 Temporal Overview

The temporal overview (Fig. 4(2)) aims at revealing an anomaly overview of the variation of all the important tracking metrics (e.g., CPU frequency and memory usage) over time, which provides users with a general impression of the whole data set in a temporal context. We present two types of charts, namely, an area chart and a stacked bar chart, to show the sum of all nodes's anomaly scores in terms of different metrics at every timestamp. The y-axis shows the anomaly score, and a categorical color scheme is used to represent different metrics. Specifically, the area chart overlaps the different metrics; thus the most anomalous metric at every time point can be highlighted, and the inter-pattern comparison and correlation discovery of multiple metrics can be fulfilled (**T5**). By contrast, the stacked bar chart emphasizes the total amount of anomaly scores for all the performance metrics at every time point. To assist users in catching the time period of interest more efficiently, we mark the top five points of each performance metric with red dots for reference. Different interaction designs are provided in this view: (1) brushing a time period in the overview for further analysis; (2) switching the view mode, as well as filtering out some metrics or highlighting others (tuning the corresponding color opaque) (Fig. 4(b)); (3) choosing three types of time granularities, namely, minutes, hours and days, as an input parameter to show the performance history at different levels of detail (Fig. 4(2)).

5.4 Anomaly Ranking

The rank view in Fig. 4(3) shows a list of compute nodes with high anomaly scores within the user-specified time period in the temporal view, which reduces users efforts in searching for suspicious nodes. To rank the nodes, we sum the anomaly scores of different metrics (equation (7)) for each node at every timestamp, and the larger the sum, the more abnormal the node. Each chart in this view represents one compute node and is placed in the increasing order of anomaly ranks, which consists of three components: an information card, an anomaly calendar view and a line connecting *the rank view* with *the performance view*. First, the information card employs a bar chart showing the average anomaly degree of different performance metrics. When clicking on this chart, the card will flip and provide textual information about the node, such as the node ID and the node rank. Second, the anomaly calendar view depicts the temporal pattern, especially the potential periodic patterns of a node. Each cell in this view represents one time unit according to the selected time granularity (minute, hour, day). The color, ranging from white to grey, encodes the sum of different metrics' anomaly score on one time unit, for example, it gets darker as the anomaly score increases. Various interaction techniques extend the functionality. Users can modify the arrangement of each calendar via tuning the slider bar for each node in Fig. 4(c). Thus potential periodicity of different nodes may be observed. Finally, the lines between *the rank view* and *the performance view* establish a correspondence between the general information and the detailed performance metrics of each node. With the buttons on the right of Fig. 4(d), the user can inspect different nodes according to their rankings (**T2**).

5.5 Performance View

The performance view (in Fig. 4(4)) displays the detailed performance metrics data, associated with anomaly detection results, to facilitate the anomaly inspection and correlation analysis among different metrics in time scale. Multiple visualization modes and interaction designs are provided in this view to improve the analysis efficiency. The charts are placed in increasing order of anomaly ranking, with the gap between two charts meaning the difference of anomaly scores between two represented nodes. The visualization for each node is composed of two sub-views. On the top, a stacked area view shows different metrics' anomaly scores and their comparison in proportion over time. A consistent categorical color scheme is used in this part to denote different metrics. By observing the height of the graph, users can easily detect when an anomaly occurs and which metric is abnormal. Thus they can quickly transfer to the suspicious time periods and metrics in *the performance view* for detailed analysis. On the lower half, there is a performance view that summarizes and displays the variation of different performance metrics data over time in three modes, allowing the understanding of correlations and temporal patterns with different semantics and scales.

Horizon Chart Mode. The horizon chart uses a space-efficient method to clearly show the overall trend of each performance metric and improve the comparison among different metrics over time. Each layer represents one metric in this view. Fig 5(1, 2) illustrates the construction of the standard horizon chart, which is extended from a normal line chart that displays the metric value changes over time. Specifically, we offset the negative values such that the zero point for the negative values is at the top of the chart. The metric value is indicated by a dichotomous color scheme ranging from green to white to red (Fig. 5(3)), with green/red encoding a value higher/lower than the average (white). The average value for a given metric is calculated by considering all the nodes in the data cluster. We can interpret the horizon chart with the following logic: (a) use the color to decide whether the data is positive or negative; (b) observe the darkest color to decide the general range of the data; (c) follow the height of the darkest part to understand the temporal trend of the metric value; and (d) read the textual information to learn the real value of the corresponding metric, and the dominant reason (three causes mentioned in Section 4.2) for the anomalies (**T3–4**).

Multi-line Mode. A multi-line chart is provided as a more conventional and familiar visualization type for users, compared with the horizon chart, to facilitate their understanding of the data. By clicking the "line" button shown in Fig. 4(d), the mode of *the performance view* will change to *the multi-line mode* (Fig. 4(6)), where each line presents one metric and the same categorical colors used in our system are applied. Also, each attribute's data is normalized and scaled to the range of [-1,1] because we need to make the units for different attributes consistent with the same y-axis (**T4**).

PCA Mode. The PCA (principal component analysis) view depicts the major trends of a node's performance (Fig. 4(7)). The node's performance data, which is a multivariate time series with several metrics (e.g., CPU frequency and memory usage), are projected to a one-dimensional time series with PCA analysis. Thereby, we can reduce the number of variables and make important trends in the data directly accessible, providing qualitative high-level insights for cloud computing performance and anomaly inspection [3] (**T4**).

Glyph Design. Given the large scale of cloud computing data in time scale, a visual aggregation glyph, following the "magnet" metaphor (Fig. 5(b)), is designed to facilitate anomaly inspection via conducting data abstraction for irrelevant or uninteresting periods in *the performance view*, as well as saving space in the time domain when users have to undertake analysis for a long period. After brushing a time period in *the performance view* (Fig. 1(a)), users can right click in the brushed area. Then this area will collapse into multiple glyphs (Fig. 1(b5)),

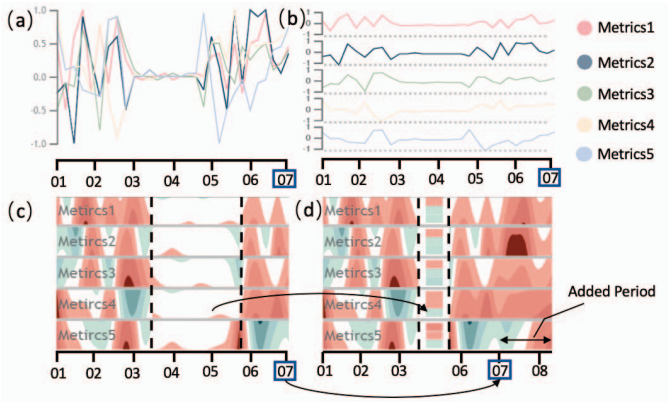

Fig. 6. Alternative designs for performance metrics: (a) multi-line chart, (b) small multiples, (c) horizon chart. Our design (d) extends (c) with glyph and the nonlinear time scaling for analyzing more data.

with each one representing one metric's data. Each glyph is trisected into three rectangles (Fig. 5(d)), and the color of the top, middle and bottom rectangles encodes the maximal, average and minimal values of the corresponding metric in the collapsed area. The color scheme is the same as that of the horizon chart (Fig. 5(3)). Moreover, the glyph filling color is blurred (Gaussian blur) to present uncertainty (std value) of the data abstraction for this area. In addition, users can also drag the border of the brushed area to narrow/broaden it, thus scaling *the performance view* and focusing on the information of interest (Fig. 5(c)). Finally, there is a dynamic axis which can follow the move of the mouse to immediately provide the real metric value for information reference.

Alternative Design Several design alternatives for *the performance view* were considered, as shown in Fig. 6. The first and most intuitive one was to use the multi-line chart containing all the metrics' performances in a single graph (Fig. 6(a)). However, this visualization has two problems: the severe visual clutter when data is large, and the inconsistency in measurement units for the different metrics. Another alternative was to draw line charts separately (Fig. 6(b)) for each metric and to align them horizontally. But this method consumes much more space for a single line chart if we want it to express the information as clearly as the horizon chart (Fig. 6(c)). The two methods both lack the ability to show the trends of multiple metrics clearly as horizon chart. However, a direct application of horizon chart does not support the scaling in time domain. Users would have difficulty in analyzing the overall changes or focusing on a detailed time period when the length of time series is very long. In this sense, the visual aggregation (glyph) and data abstraction techniques applied in the horizon chart enhance it by a nonlinear time scaling (Fig. 6(d)).

5.6 Cluster View

The cluster view (Fig. 4(5)) shows the spatial distribution of all the selected compute nodes using t-Distributed Stochastic Neighbor Embedding (t-SNE) based on their performance data, which are a multivariate time series. For each node, we construct a vector:

$$F = [m_{1,1}, m_{2,1}, ..., m_{r,1}, m_{1,2}, m_{2,2}, ..., m_{r,2}, ..., m_{1,n}, m_{2,n}, ..., m_{r,n}],$$

where $m_{r,n}$ means the rth metric value at the nth timestamp. Therefore, we conduct the t-SNE dimensionality reduction based on these feature vectors. The clustering analysis can aggregate the multivariate temporal data into subsets exhibiting a certain similarity. Each circular glyph in this view represents one compute node whose anomaly degree is encoded by the filling color of its inner dot, ranging from white to grey. The darker the color, the more abnormal it is. The anomaly degree is calculated by the Local Outlier Factor (LOF) and normalized to -1 (normal) and 1 (abnormal). To enhance the anomaly and clustering analysis, we render a contour map to reveal areas with different densities, which are indicated by the filling color from white (low density) to grey (high density). Intuitively, a normal node tends to lie in high-density areas where many others have similar behaviors. We use the kernel density estimation (KDE) to define the density at the user u's position (x_u), which is formulated as

$$f(x_u) = \frac{1}{nh}\sum_{1}^{n} K(\frac{x_u - x_i}{h}),$$

where K is the kernel function, $x_i (i \neq u)$ indicates the positions of the nodes and h is the kernel's bandwidth. From this view, we can also provide another perspective for anomaly diagnosis according to the inconsistent measurements of similar nodes. For example, some anomalous nodes (high LOF score) may be grouped in the white contour, which could represent a rare category that encounters the same problem (like network attacks) in cloud data centers. Furthermore, we design a glyph (Fig. 4(f)), following the "fan" metaphor, to facilitate comparison among different nodes by summarizing their general usage metrics (**T6**).

Glyph Design. This glyph contains several arcs which are equally segmented around the inner circle. The left one in Fig. 4(f) is our original design. Each arc illustrates one performance metric that can be denoted by the categorical colors that are consistently used in our system. The radius of the arc encodes the average value of its represented metrics across the selected time period. However, it loses the temporal information for each metric, so we redesigned the visualization as shown on the right in the figure. We enhanced each arc with a circular timeline in clockwise order. For example, if a node contains eight records for CPU frequency in the selected time period, then the corresponding arc is divided vertically into eight equal segments, with each segment representing one record. The lighter (closer to white) the segment is, the lower the record value is. Furthermore, users can click on a specific glyph to get the detailed raw metric data of the node at the bottom of this view, which is the same the multi-line chart employed in *the performance view*. In this way, we can help users immediately get the low-level but detailed performance data of a node of interest, as well as connect the visualization results of the horizon chart.

5.7 Interactions

The following interactions are designed to help with the data exploration and anomaly inspection. **Anomaly Inspection.** Users can compare different levels of cloud computing data with different scales in the spatial and temporal overview. They can right/double click and zoom the visualizations in *the performance view* to make a detailed inspection. A novel glyph summarizing a node's performances can be display or not in *the cluster view*. **Query and Filtering.** Users can load different subsets of data by using the query module or direct interaction on the views in Fig. 4(1, 2), as well as focus on a specific metric in *the temporal overview* by clicking the legend on Fig. 4(b) (**T1**). **Switching Context.** Users can switch between different visualization modes in *the performance view*. In *the temporal overview*, they can choose the area chart or stacked bar chart, and change the time granularity for all the corresponding views based on the analytic requirements. **Zooming and Scaling.** Four views support zooming for exploring a large set of data items, namely, *the spatial overview*, the calendar graph in *the rank view*. Particularly, users can brush a period of time for the node in *the performance view* and choose to expand it for more details or narrow it for information reduction. **Data Abstraction.** We design two novel glyphs to generalize different data information for data exploration and anomaly inspection, namely, the "magnet" in *the performance view* and the "fan" glyph in *the cluster view*. **Tooltips and Highlighting.** Tooltips are provided in *the spatial overview*, *the rank view* and *the cluster view* for more reference information. In addition, there is a dynamic axis in *the performance view*, following the moving of the mouse, to provide the raw value and dominant anomaly pattern of the performance data in real time. All the views are linked together.

6 Evaluation

We evaluated the effectiveness of CloudDet via a quantitative performance evaluation of the proposed algorithm in Section 6.1, two case studies with real-world data in Section 6.2, and follow-up interviews with domain experts in Section 6.3.

6.1 Quantitative Evaluation

We evaluated the effectiveness and scalability of the proposed algorithm through a quantitative comparison with five baseline methods: One-Class SVM [14], Robust Covariance Estimation (RCov), LOF [8], Bitmap Detector (BD) [55] and Twitter anomaly detection (Tad). They are selected as the most typical algorithms from five major categories of anomaly detection techniques, i.e., classification-based, neighbor-based, statistic-based, SAX (symbolic aggregate approximation)-based and

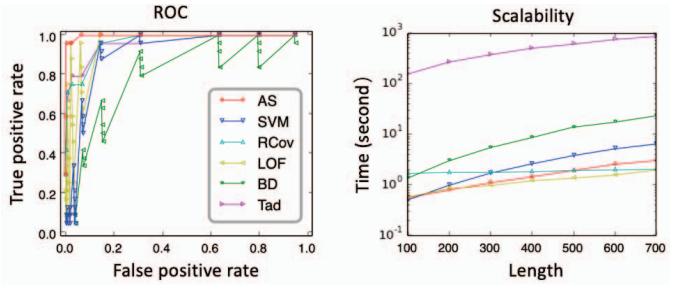

Fig. 7. Algorithm performance evaluation. Left: the result indicates that our algorithm (AS) outperforms others in accuracy. Right: our algorithm can scale and exhibit linearly with the varying length of time series.

S-H-ESD (Seasonal Hybrid ESD)-based methods, respectively. Other anomaly detection algorithms, like iForest [35], were discarded as their execution times are much longer than that of our proposed algorithm. We used the implementation of these models in the scikit-learn [42], luminal and Twitter anomaly detection package in Python.

Accuracy. The standard information retrieval metric ROC was used here for accuracy evaluation due to the extremely imbalanced ratio between positive and negative instances. The accuracy of the algorithm was verified using real_31.csv from the labeled Yahoo! S5 real time-series dataset [60]. This dataset contains mixed change patterns (both long-term and short-spike anomalies) with a 1.68% outlier percentage (24/1427), which can show the superiority of our algorithm better than using the dataset with a single anomaly pattern. We tested the accuracy of each algorithm with ten different values of its chosen parameter. In particular, our proposed algorithm took the number of recent "historical data" L as the adjustable parameter (see Section 4.1), which grew from 5 to 50, with 5 as the step length. Then we ran each parameter setting of the algorithm ten times based on different discrimination thresholds (proportion of anomalies) [0.005, 0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.64, 0.8, 0.95]. The thresholds grew exponentially, except for the last two. Therefore, we obtained 100 runs (10 parameter values $\times$ 10 discrimination thresholds) for each algorithm in the accuracy evaluation, which reduced the bias of detection results to different datasets. (Please refer to the supplementary material for more algorithm setting information and evaluation results. [2])

As shown in Fig. 7. Overall, our algorithm (denoted as "AS") outperformed the five baseline methods when there are different anomaly patterns in data. In particular, our algorithm had a higher true positive rate when the false positive rate was low (below 0.2), which means that our algorithm can reduce false positive anomalies when detecting the same number of anomalies as the baseline algorithms. This is important for cloud computing anomaly detection because the experts can save effort in anomaly diagnosis when the data scale is very large.

Scalability. The proposed algorithm must be scale-out and efficient to cope with large-scale data generated by cloud computing systems. To perform the evaluation, we tested the execution time of each algorithm by varying the length of the input time-series data. Specifically, in the experiments, each algorithm was executed on 50 time-series datasets (real_1 to real_50), with each dataset running ten times due to ten values for its adjustable parameters. We only varied the length of the input time series in different experiments, growing from 100 to 700 points with 100 as step length, to determine the execution time of the algorithm. The length was changed by selecting the first 100–700 data points of a dataset. The experiments aere conducted in an eight-core (Intel Core i7-4790 CPU@3.6GHz) Window 10 computer with 16 GB memory. The results are summarized in Fig. 7. The figure suggests that the execution time of our algorithm can scale and exhibits linearly with the length of the time series. "AS" is less scalable than RCov and LOF, probably due to the time spent for pattern matching before anomaly detection, but the difference is acceptable considering the higher accuracy in detection results compared with other algorithms. Although Tad performs well in accuracy test in general, it has a worst computing speed compared with others.

Limitations. Although the results show that our proposed algorithm has sound performance when considering the speed and accuracy together, the validity of the results needs further evaluation. There exist some factors that may affect the validity, such as the selection of parameters and their settings, the bias in the tested datasets and the insufficiency of the evaluation metrics. In particular, our algorithm may perform worse than the baselines when the anomalies merely have specific patterns like short-term spikes. Therefore, the results of the proposed algorithm could be affected by the aggregation strategy of the anomaly scores from the three components. A more optimal or automated aggregation method should be developed future work.[2]

[2]https://lukeluker.github.io/supp/vast19_cloudDet.pdf

6.2 Case Study & Expert Interview

We demonstrate the usefulness and usability of CloudDet by analyzing two real-world data sets. The study was conducted in a semi-structured interview with two domain experts for two purposes: to provide real abnormal cases to showcase the effectiveness of our system, and to give user feedback on the system design.

Study Set-up and Interview Process. We invited two experts who have a high level of experience in cloud computing related domains to be interviewed in the form of case study. The one for case study 1 is a data scientist from a company's cloud computing service team, who is responsible for the cloud system analysis and maintenance (**E1**). The expert for case study 2 is a researcher from the same company's data analytics group, and his main focus recently has been anomaly detection with cloud computing data (**E2**). Both of them had no prior knowledge about the data used in the case studies and were not collaborators on our project. Each interview lasted approximately 1.5 hours, with 30 minutes for the introduction of the system and 60 minutes for data exploration and anomaly detection by the experts themselves using our system. Notes and expert feedback were recorded in the process.

Case Study 1: Bitbrains Datacenter Traces. In the first case study, we used a one-month (2013/08/01 – 2013/08/31) dataset from Bitbrains [7], which contains the performance metrics of 500 VMs from a distributed data center. We resampled the data into hourly granularity and selected five typical metrics for the study, namely, average CPU rate (MHz), maximum CPU rate (MHz), memory usage (KB), disk read throughput (KB/s) and network transmitted throughput (KB/s). After loading the data into the system, the expert, **E1**, started by choosing the hourly granularity and observing *the temporal overview* in the area chart mode (Fig. 4(2)). Then he noticed the first time period in which many anomalous points were marked (**T1**), so he brushed this period (from Aug. 7, 12 am to Aug. 15, 12 am) to make a further inspection. The most abnormal node, compared with others via the horizon chart (Fig. 1(b1)), was not only abnormal in the drastic spikes (both short- and long-term) in the memory and CPU usage, but also quite sporadic and irregular in the spike occurrence time. Moreover, there existed inconsistency between the network throughput and other metrics (**T5**). After knowing that the data were collected from cloud service for business computation for enterprises, he believed that this node was very likely to be abnormal due to the unusual operations caused by users.

Another type of anomalous periodic performance was also noticed by **E1**. Different from the above case, this node, as shown in Fig. 1(b2), had a periodic pattern in CPU usage while all the other metrics were stable. Then the expert tuned the slider bar at *the rank view* and found the cycle was about 12 hours (Fig. 1(a2)), which could be caused by regular analytic tasks run by the system customers However, there was a sudden increase in all metrics from Aug. 11 to Aug. 13 (Fig. 1(b2)), then felling to the same periodic performance as before. Similarly, when the expert switched to the next five nodes, he found an instance that had many near-periodic and short-term spikes at the beginning but an abnormal long-term increase near the end (Fig. 1(b3)). To check whether this node returned to its previous behaviors after the anomalous periods, he brushed this period and right clicked it to transfer it to the "magnet" glyph for space scaling. Then the horizon chart was updated and the data after this period were shown (Fig. 1(b5)). As expected, the node returned to similar periodic behavior as before. The expert also inspected the glyph and found that the part representing CPU frequency was blurred heavily, which means that this abnormal period was mainly unstable in CPU usage. Finally, when navigating in *the performance view*, the expert found a node which seemed quite stable and normal with no spikes in the horizon chart (Fig. 1(b4)), while its anomaly ranking was high (10th). Then he chose *the PCA mode* to see whether

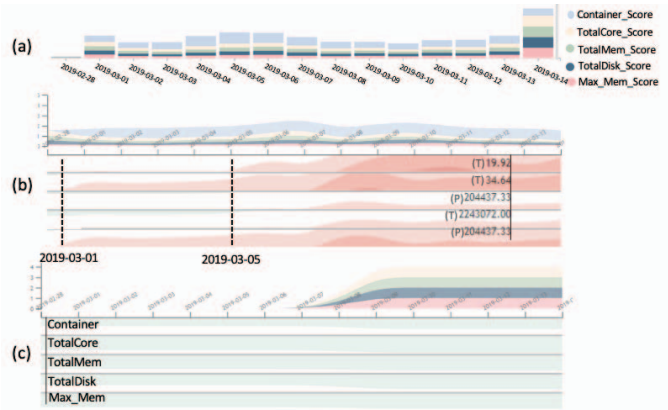

Fig. 8. Identified anomalous patterns with live cloud system data.

there was any general trend in the data (Fig. 1(b6)). *The PCA mode*, however, showed a clear periodic pattern in the performance.

Finally, the expert selected other time periods via *the temporal overview* and found some interesting patterns in *the cluster view* (Fig. 1(c)). For example, the distribution of nodes (Aug.6–17) showed three different clusters of nodes based on their behaviors. To compare the three clusters, the expert clicked some represented nodes in each cluster and obvious differences were observed. Compared with the normal nodes, one abnormal cluster (top-right) had a large red arc whose color was not distributed consistently, which means the highly changed CPU metric was the dominant reason for anomaly. By contrast, the other rare category had both large red and blue arc, which means they were abnormal due to their high value of several metrics. The expert thought that the nodes in the same cluster may perform similar tasks or belong to the same servers in the cloud computing systems (**T6**).

Case Study 2: Live Cloud System Data. We conducted the second case study with live cloud computing data (provided by the collaborator on the project) collected from about 1,000,000 nodes in the previous two weeks (2019/03/01 – 2019/03/14), and we processed these time series into hourly resolution. In particular, the data were collected from more than 100 data centers, with each center containing about 20 data clusters, and each data cluster containing about 500 compute nodes. The performance metrics of the dataset were all about general resource allocations of a compute node, including container (VM) count, total core (CPU), total disk, average memory and maximum memory. **E2**, started by using *the spatial overview* to select a data cluster for detailed inspection (**T1**). In this view (Fig. 4)(1)), he visualized the top 20 anomalous data centers and chose one anomalous data cluster from the first data center (**T2**). Then the visualization switched to *the temporal overview* (Fig. 8)(a)). Considering that the resource allocation usually does not change frequently, he decided to conduct anomaly analysis with day granularity. The expert noticed a sudden increase in anomaly scores at the last day by observing *the temporal overview* (Fig. 8)(a)), which means that special operations might have been undertaken on that day. Then he switched to *the rank view* and *the performance view* (**T2, 4**). After a brief inspection of the top-ranked notes, he quickly noted some nodes, like Fig. 8(c), that have few changes in all resource allocations during the previous two weeks. He thought this was abnormal for a live system, and conjectured a physical maintenance or system breakdown for these nodes. In addition, when he compared some top-ranked nodes, he found another common issue that some nodes' metrics did not have synchronized behaviors like normal nodes. For example, starting from 03/01 in Fig.8(b), the core and memory increase whilst the container number remains unchanged, which indicates that the resources allocated for each container increased. Moreover, around 03/05, there was a large increase in container number but a small increase in other metrics, which could be caused by adding some small containers into the node (**T5**). Such evolution in a compute node was abnormal due to the frequent inconsistent adjustment in the resource allocation compared with others in the same data cluster.

6.3 User Feedback

The experts from the case studies (**E1**, **E2**) provided a wealth of insightful feedback, which are summarized into three categories.

Automated Anomaly Detection. The experts showed a transition in attitude towards the automated anomaly detection algorithm. Before the case studies, neither of them trusted the algorithm because the traditional methods generated a large number of false positives in their past experiences. **E1** stated, "We don't have time to check the results one by one... even a few percentages of wrong [false positive] results for cloud computing is large in number." However, after inspecting the top-ranked results in *the rank view* and *the performance view*, both experts placed increased trust in the algorithm. For example, they commented that "the algorithm can detect anomalies showing different anomalous patterns." Inspired by the PCA analysis for different metrics in our system, **E2** further suggested that "the algorithm could be more powerful if it can make a comprehensive anomaly detection by considering different metrics with different time granularity."

System. Both experts regarded the system as useful and user-friendly for analyzing cloud computing anomalies. "It's useful!" was the most common refrain from **E1**, and he felt the findings were exactly the sort of information that cloud computing systems would use in anomaly analysis. "It would be useful for maintenance and resource allocation of cloud computing systems," he said, "We usually implement some resource allocation algorithms to the cloud system, but have difficulty in evaluating them... This tool can monitor our system and identify warnings [anomalies] from performance data." Moreover, **E2** commended the workflow of the system as clear and logical because the system is consistent with the conventional way to analyze cloud system data. However, both experts felt the system "too comprehensive" and "a little overwhelming" at the beginning of use. Although a short introduction was sufficient to bootstrap their analyses, **E2** suggested that a tacit tutorial be included to guide independent users, and that the system could also display the information based on an exploration-based approach.

Visualization and Interaction. Most visualization and interaction designs were thought to meet the design tasks. Both experts stated that "the performance view is the essential part" because it displayed "the overall trend" of tracking metrics and also supported "a clear comparison" among different metrics and nodes. **E2** stated, "I used to take the line chart to analyze the results... It's quite chaotic when there are too many metrics or long time periods." **E1** mentioned the novelty of showing the similarity among compute nodes in *the cluster view*. He said, "I never thought of this," and "It provides a new perspective to understand cloud computing anomalies." **E2** thought the glyph in the horizon chart as "interesting and useful" because it can "put aside uninterested information and save space for more data." Moreover, they regarded the interactions in our system as helpful, which allowed a quick way to navigate and scale the visualizations when needed.

7 Conclusion and Future Work

We have presented CloudDet, an interactive visualization system that enables expert users to identify and analyze anomalous performances in cloud computing systems. The system supports the exploration process in both spatial and temporal context by incorporating a novel anomaly detection algorithm, multiple coordinated contextual views and rich interaction designs. We demonstrated the power of CloudDet through a quantitative evaluation for algorithm performance, two case studies using real-world data and follow-up interviews with domain experts. While these results are promising, there are several new directions for future work. First, the algorithm can still be optimized in many aspects like the aggregation strategy for different components' anomaly scores. Second, we plan to deploy our system on cloud computing platforms so as to discover ways to improve it through more user feedback. To improve the qualitative evaluation, we intend to compare CloudDet with established application monitoring (APM) tools used by cloud operators and implement it with more datasets. To turn the approach into a supervised one, we can tune the anomaly detection model responsively by feeding user corrections (e.g., re-rankings and anomaly labeling) back to an active learning process in real-time. Finally, we will evaluate the usability of our system and glyph designs via a formal user study such that simplifying our system by removing the uncritical parts.

8 Acknowledgements

We would like to thank all the reviewers and domain experts for their comments. This work is partially supported by a grant from MSRA (code: MRA19EG02).

REFERENCES

[1] B. Abraham and A. Chuang. Outlier detection and time series modeling. *Technometrics*, 31(2):241–248, 1989.

[2] B. Agrawal, T. Wiktorski, and C. Rong. Adaptive real-time anomaly detection in cloud infrastructures. *Concurrency and Computation: Practice and Experience*, 29(24):e4193, 2017.

[3] W. Aigner, S. Miksch, W. Müller, H. Schumann, and C. Tominski. Visual methods for analyzing time-oriented data. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):47–60, 2008.

[4] L. Akoglu, H. Tong, and D. Koutra. Graph based anomaly detection and description: A survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.

[5] V. Barnett and T. Lewis. *Outliers in Statistical Data*. Wiley, 1974.

[6] S. D. Bay and M. Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 29–38. ACM, 2003.

[7] Bitbrains. Gwa-t-12 bitbrains dataset. "`http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains`".

[8] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: identifying density-based local outliers. In *ACM Sigmod Record*, vol. 29, pp. 93–104. ACM, 2000.

[9] L. Byron and M. Wattenberg. Stacked graphs–geometry & aesthetics. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1245–1252, 2008.

[10] R. N. Calheiros, K. Ramamohanarao, R. Buyya, C. Leckie, and S. Versteeg. On the effectiveness of isolation-based anomaly detection in cloud data centers. *Concurrency and Computation: Practice and Experience*, 29(18):e4169, 2017.

[11] N. Cao, C. Lin, Q. Zhu, Y.-R. Lin, X. Teng, and X. Wen. Voila: Visual anomaly detection and monitoring with streaming spatiotemporal data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):23–33, 2018.

[12] N. Cao, C. Shi, S. Lin, J. Lu, Y.-R. Lin, and C.-Y. Lin. Targetvue: visual analysis of anomalous user behaviors in online communication systems. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):280–289, 2016.

[13] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15, 2009.

[14] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[15] Y. Chen, X. S. Zhou, and T. S. Huang. One-class SVM for learning in image retrieval. In *Proceedings of International Conference on Image Processing*, pp. 34–37. Citeseer, 2001.

[16] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning. STL: A seasonal-trend decomposition. *Journal of Official Statistics*, 6(1):3–73, 1990.

[17] N. Ezzati-Jivan and M. R. Dagenais. Multi-scale navigation of large trace data: A survey. *Concurrency and Computation: Practice and Experience*, 29(10):e4068, 2017.

[18] M. Farshchi, J.-G. Schneider, I. Weber, and J. Grundy. Experience report: Anomaly detection of cloud application operations using log and cloud metric correlation analysis. In *IEEE 26th International Symposium on Software Reliability Engineering*, pp. 24–34. IEEE, 2015.

[19] S. Few. Time on the horizon visual, 2008. Online at `https://www.perceptualedge.com/articles/visual_business_intelligence/time_on_the_horizon.pdf`.

[20] A. J. Fox. Outliers in time series. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(3):350–363, 1972.

[21] J. Fuchs, F. Fischer, F. Mansmann, E. Bertini, and P. Isenberg. Evaluation of alternative glyph designs for time series data in a small multiple setting. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 3237–3246. ACM, 2013.

[22] M. Gupta, J. Gao, C. C. Aggarwal, and J. Han. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 26(9):2250–2267, 2014.

[23] V. Hautamaki, I. Karkkainen, and P. Franti. Outlier detection using k-nearest neighbour graph. In *Proceedings of the 17th International Conference on Pattern Recognition*, vol. 3, pp. 430–433. IEEE, 2004.

[24] S. Hawkins, H. He, G. Williams, and R. Baxter. Outlier detection using replicator neural networks. In *International Conference on Data Warehousing and Knowledge Discovery*, vol. 2454, pp. 170–180. Springer, 2002.

[25] J. Heer, N. Kong, and M. Agrawala. Sizing the horizon: the effects of chart size and layering on the graphical perception of time series visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1303–1312. ACM, 2009.

[26] J. Heer, F. B. Viégas, and M. Wattenberg. Voyagers and voyeurs: supporting asynchronous collaborative information visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1029–1038. ACM, 2007.

[27] V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[28] O. Ibidunmoye, F. Hernández-Rodriguez, and E. Elmroth. Performance anomaly detection and bottleneck identification. *ACM Computing Surveys*, 48(1):4, 2015.

[29] K. E. Isaacs, A. Giménez, I. Jusufi, T. Gamblin, A. Bhatele, M. Schulz, B. Hamann, and P.-T. Bremer. State of the art of performance visualization. *EuroVis*, 2014.

[30] A. I. Jehangiri, R. Yahyapour, P. Wieder, E. Yaqub, and K. Lu. Diagnosing cloud performance anomalies using large time series dataset analysis. In *IEEE 7th International Conference on Cloud Computing*, pp. 930–933. IEEE, 2014.

[31] A. Kind, M. P. Stoecklin, and X. Dimitropoulos. Histogram-based traffic anomaly detection. *IEEE Transactions on Network and Service Management*, 6(2):110–121, 2009.

[32] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[33] P. Laskov, K. Rieck, C. Schäfer, and K.-R. Müller. Visualization of anomaly detection using prediction sensitivity. In *Sicherheit*, vol. 2, pp. 197–208, 2005.

[34] J. Lin, E. Keogh, and S. Lonardi. Visualizing and discovering non-trivial patterns in large time series databases. *Information Visualization*, 4(2):61–82, 2005.

[35] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *IEEE 8th International Conference on Data Mining*, pp. 413–422. IEEE, 2008.

[36] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11):2579–2605, 2008.

[37] M. V. Mahoney and P. K. Chan. Learning rules for anomaly detection of hostile network traffic. In *IEEE 3rd International Conference on Data Mining*, pp. 601–604. IEEE, 2003.

[38] S. McKenna, D. Staheli, C. Fulcher, and M. Meyer. Bubblenet: A cyber security dashboard for visualizing patterns. In *Computer Graphics Forum*, vol. 35, pp. 281–290. Wiley Online Library, 2016.

[39] C. Muelder, B. Zhu, W. Chen, H. Zhang, and K.-L. Ma. Visual analysis of cloud computing performance using behavioral lines. *IEEE Transactions on Visualization and Computer Graphics*, 22(6):1694–1704, 2016.

[40] W. Müller and H. Schumann. Visualization for modeling and simulation: visualization methods for time-dependent data-an overview. In *Proceedings of the 35th Winter Simulation Conference: Driving Innovation*, pp. 737–745. Winter Simulation Conference, 2003.

[41] A. Navaz, V. Sangeetha, and C. Prabhadevi. Entropy based anomaly detection system to prevent ddos attacks in cloud. *arXiv preprint arXiv:1308.6745*, 2013.

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[43] M. Peiris, J. H. Hill, J. Thelin, S. Bykov, G. Kliot, and C. Konig. Pad: Performance anomaly detection in multi-server distributed systems. In *IEEE 7th International Conference on Cloud Computing*, pp. 769–776. IEEE, 2014.

[44] T. Saito, H. N. Miyamura, M. Yamamoto, H. Saito, Y. Hoshiya, and T. Kaseda. Two-tone pseudo coloring: Compact visualization for one-dimensional data. In *IEEE Symposium on Information Visualization.*, pp. 173–180. IEEE, 2005.

[45] A. Sari. A review of anomaly detection systems in cloud networks and survey of cloud security measures in cloud storage applications. *Journal of Information Security*, 6(02):142, 2015.

[46] K. Shen, C. Stewart, C. Li, and X. Li. Reference-driven performance anomaly identification. In *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, pp. 85–96. ACM, 2009.

[47] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *The Craft of Information Visualization*, pp. 364–371. Elsevier, 2003.

[48] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, Miami Univ Coarl Gables FL Dept of Electrical and Computer Engineering, 2003.

[49] J. Tao, L. Shi, Z. Zhuang, C. Huang, R. Yu, P. Su, C. Wang, and Y. Chen. Visual analysis of collective anomalies through high-order correlation

graph. In *IEEE Pacific Visualization Symposium*, pp. 150–159. IEEE, 2018.

[50] M. Vlachos, P. Yu, and V. Castelli. On periodicity detection and structural periodic similarity. In *Proceedings of the SIAM International Conference on Data Mining*, pp. 449–460. SIAM, 2005.

[51] C. Wang, V. Talwar, K. Schwan, and P. Ranganathan. Online detection of utility cloud anomalies using metric distributions. In *IEEE Network Operations and Management Symposium*, pp. 96–103. IEEE, 2010.

[52] C. Wang, K. Viswanathan, L. Choudur, V. Talwar, W. Satterfield, and K. Schwan. Statistical techniques for online anomaly detection in data centers. In *12th IFIP/IEEE International Symposium on Integrated Network Management and Workshops*, pp. 385–392. IEEE, 2011.

[53] T. Wang, W. Zhang, C. Ye, J. Wei, H. Zhong, and T. Huang. Fd4c: Automatic fault diagnosis framework for web applications in cloud computing. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(1):61–75, 2016.

[54] M. Wattenberg and J. Kriss. Designing for social data analysis. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):549–557, 2006.

[55] L. Wei, N. Kumar, V. N. Lolla, E. J. Keogh, S. Lonardi, and C. A. Ratanamahatana. Assumption-free anomaly detection in time series. In *SSDBM*, vol. 5, pp. 237–242, 2005.

[56] W.-K. Wong, A. W. Moore, G. F. Cooper, and M. M. Wagner. Bayesian network anomaly pattern detection for disease outbreaks. In *Proceedings of the 20th International Conference on Machine Learning, 2003*, pp. 808–815, 2003.

[57] C. Xie, W. Xu, and K. Mueller. A visual analytics framework for the detection of anomalous call stack trees in high performance computing applications. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):215–224, 2019.

[58] K. Xu, S. Guo, N. Cao, D. Gotz, A. Xu, H. Qu, Z. Yao, and Y. Chen. ECGLens: Interactive visual exploration of large scale ecg data for arrhythmia detection. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, p. 663. ACM, 2018.

[59] X. Xu, L. Zhu, I. Weber, L. Bass, and D. Sun. POD-diagnosis: Error diagnosis of sporadic operations on cloud applications. In *IEEE/IFIP 44th International Conference on Dependable Systems and Networks*, pp. 252–263. IEEE, 2014.

[60] Yahoo. S5-a labeled anomaly detection dataset. "`https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70`".

[61] K. Yamanishi, J.-I. Takeuchi, G. Williams, and P. Milne. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Mining and Knowledge Discovery*, 8(3):275–300, 2004.

[62] L. Yu and Z. Lan. A scalable, non-parametric anomaly detection framework for hadoop. In *Proceedings of the ACM Cloud and Autonomic Computing Conference*, p. 22. ACM, 2013.

[63] X. Zhang, F. Meng, P. Chen, and J. Xu. Taskinsight: A fine-grained performance anomaly detection and problem locating system. In *IEEE 9th International Conference on Cloud Computing*, pp. 917–920. IEEE, 2016.

[64] J. Zhao, N. Cao, Z. Wen, Y. Song, Y.-R. Lin, and C. Collins. #FluxFlow: Visual analysis of anomalous information spreading on social media. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1773–1782, 2014.