Visual Exploration of Time Series Anomalies with Metro-Viz

Philipp Eichmann
Brown University
peichmann@cs.brown.edu

Franco Solleza
Brown University
fsolleza@cs.brown.edu

Nesime Tatbul
Intel Labs and MIT
tatbul@csail.mit.edu

Stan Zdonik
Brown University
sbz@cs.brown.edu

ABSTRACT

This demo presents a novel data visualization solution for exploring the results of time series anomaly detection systems. When anomalies are reported, there is a need to reason about the results. We introduce Metro-Viz – a visual tool to assist data scientists in performing this analysis. Metro-Viz offers a rich set of interaction features (e.g., comparative analysis, what-if testing) backed by data management strategies specifically tailored to the workload. We show our tool in action via multiple time series datasets and anomaly detectors.

CCS CONCEPTS

• **Information systems → Data management systems**; **Temporal data**; *Main memory engines*; *Database web servers*; • **Human-centered computing → Interactive systems and tools**; *Graphical user interfaces*; *Visualization*; • **Computing methodologies → Machine learning**.

KEYWORDS

time series; anomaly detection; visual analysis

ACM Reference Format:
Philipp Eichmann, Franco Solleza, Nesime Tatbul, and Stan Zdonik. 2019. Visual Exploration of Time Series Anomalies with Metro-Viz. In *2019 International Conference on Management of Data (SIGMOD '19), June 30-July 5, 2019, Amsterdam, Netherlands.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3299869.3320247

SIGMOD '19, June 30-July 5, 2019, Amsterdam, Netherlands

ACM ISBN 978-1-4503-5643-5/19/06…$15.00
https://doi.org/10.1145/3299869.3320247

1 INTRODUCTION

Motivation. A broad range of applications from the Internet of Things (IoT) to DevOps monitoring involve a large number of sources generating huge volumes of *time series data* that collectively offer a view of the application space. Often, the application is not interested in the normal case, but rather, needs to be alerted to the abnormal case. These abnormal events are termed *anomalies*, and the algorithms that find them fall under the category of *anomaly detection* (AD) [6]. Time series AD systems are typically built based on machine learning (ML) models trained with previously collected datasets. This process is inherently an iterative and human-in-the-loop (HIL) process for various reasons:

Anomaly detection is a highly domain-specific problem. What constitutes an anomaly changes greatly from one application domain to another. This bears the need to develop, compare, and choose from multiple AD models and algorithms to find one that best suits a given application.

Time series anomalies can be complex. They typically fall under the category of collective and contextual anomalies [6], exposing themselves at different time granularities or aggregations. Anomalous patterns may vary over time due to the temporal and dynamic nature of the data. Multiple, potentially correlated attributes may require multi-variate analysis. Overall, interpreting and reasoning about time series anomalies is not a straightforward task.

Data or training meta-data can be noisy/unavailable. In most domains, anomalies are relatively rare or unique events, making it challenging to collect anomalous datasets. Often, training datasets capture a small number of anomalous patterns, which may be improperly labeled. In IoT, the presence of noisy sensor data worsens the issue. As a result, labels may be wrong, incomplete, ambiguous, or missing all together, misguiding the ML process. Human intervention can go a long way in quickly identifying and fixing such issues.

Pre-deployment testing/validation of anomaly detectors is safety-critical. Prediction accuracy is key in safety-critical domains (e.g., autonomous driving, patient monitoring). Multiple anomaly detectors must be run with multiple test datasets; their

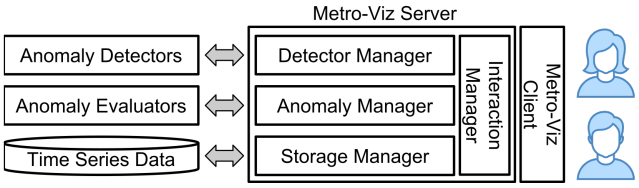

Figure 1: Architectural Overview

outputs must be compared; their statistical accuracies must be evaluated and cross-validated. Only after that can they be practically deployed in prediction serving systems.

Approach. To deal with these challenges, we believe that it is essential to build tools that will enable data scientists to productively interact with time series AD systems. In particular, tools for visually analyzing and experimenting with results of anomaly detectors can not only help compare and evaluate different algorithmic options, but also provide insights about anomalous patterns and what real-world phenomena they may correspond to. In this demo, we present Metro-Viz – a visual tool that we built for this purpose.

Novelty. Visual time series exploration has been extensively studied by both database and visualization communities. The most relevant examples include: RINSE for adaptive index-based data series exploration [15], Data Polygamy for exploring spatio-temporal datasets [5], ONEX for analyzing similarity-based time series clustering [12], Qetch for sketch-based time series querying [10], Track Xplorer for visual analysis of sensor activity tracking classifiers [3], and Steiger et al.'s visual pattern analysis system for sensor network anomalies [13]. The key novelty of Metro-Viz lies in its special focus on supporting data science tasks involved in building robust time series AD systems. It has been directly motivated by our own recent research on developing and evaluating such systems [8, 14], and builds on our past experience with data management architectures for IoT [11].

2 METRO-VIZ PROTOTYPE

We designed Metro-Viz based on the following goals:

Visual Exploration of Anomalies: Metro-Viz's overarching goal is to allow users to apply arbitrary AD algorithms on time series data and explore their results.

Statistical Accuracy Evaluation: Visual exploration complements, not replaces, statistical measurement when evaluating anomaly detectors. Thus, Metro-Viz should provide support for both forms of evaluation.

Interactive Hypothesis Testing: Interactive hypothesis testing allows users to simulate and understand complex scenarios not present in the data. Metro-Viz users should be able to test "what-if" scenarios to incorporate their domain expertise or expectations in their workflow.

Scalable Interaction: Metro-Viz users should experience the same level of interactivity over any arbitrary portion of the time series of interest at multiple time granularities [4].

Extensible Architecture: To manage the diversity of the domain, Metro-Viz should support multiple anomaly detectors, evaluators, and time series data in a modular fashion.

To realize these goals, we treat both usability and scalability as first-class citizens. As in Figure 1, Metro-Viz consists of two core components: the Metro-Viz Client, managing the interface to the user, and the Metro-Viz Server, managing access to all required data. We describe these in detail next.

2.1 Metro-Viz Client: User Interface

Figure 2 shows a screenshot from the current prototype of Metro-Viz. It provides the following UI features:

Time Series Viewer. After selecting a dataset from the Main Menu (Figure 2A), the data is displayed as a line graph in the Time Series Viewer (Figure 2B). The user can slide the window along the Time Axis (Figure 2C) to explore the line graph. S/he can also view the data at different levels of detail by selecting a time granularity and an aggregation function from the Main Menu (e.g., hourly sums). S/he can then apply one or more anomaly detectors on the selected data and granularity, by choosing them from the Main Menu. Detected anomalies are highlighted in the Time Series Viewer and hovering over each displays its properties (Figure 2D).

Set Operators and Consensus. The user can analyze multiple anomaly detectors' results using the Consensus Slider (Figure 2E) and the Set Operation Widget (Figure 2F). The Consensus Slider restricts the displayed anomaly ranges to only those above a certain level of agreement among the detectors, (e.g., ranges detected as anomalies by at least 5/7 detectors, see Figure 2H). The Set Operation Widget is for comparing the results of two sets of detectors using a Venn diagram that indicates the subsets to display (see Figure 2I).

Statistical Evaluation. The user can view summarized statistics in the Statistical Evaluation (Figure 2G,J). In presence of labeled data, s/he can also select accuracy metrics (e.g., precision/recall/f-score [14]) to evaluate each selected detector. Real anomaly ranges can be displayed in the Time Series Viewer in a similar manner as the predicted anomaly ranges.

Interactive Hypothesis Testing. Users can further understand anomaly detectors' characteristics by modifying the data displayed in the Time Series Viewer. A user can select several data points and change the amplitude of the selected subsequence. Similar to TimeSketch [7], users can also select a subsequence in the Time Series Viewer and replace it with a hand-drawn sequence (Figure 2K). These actions cause Metro-Viz to re-apply all selected anomaly detectors.

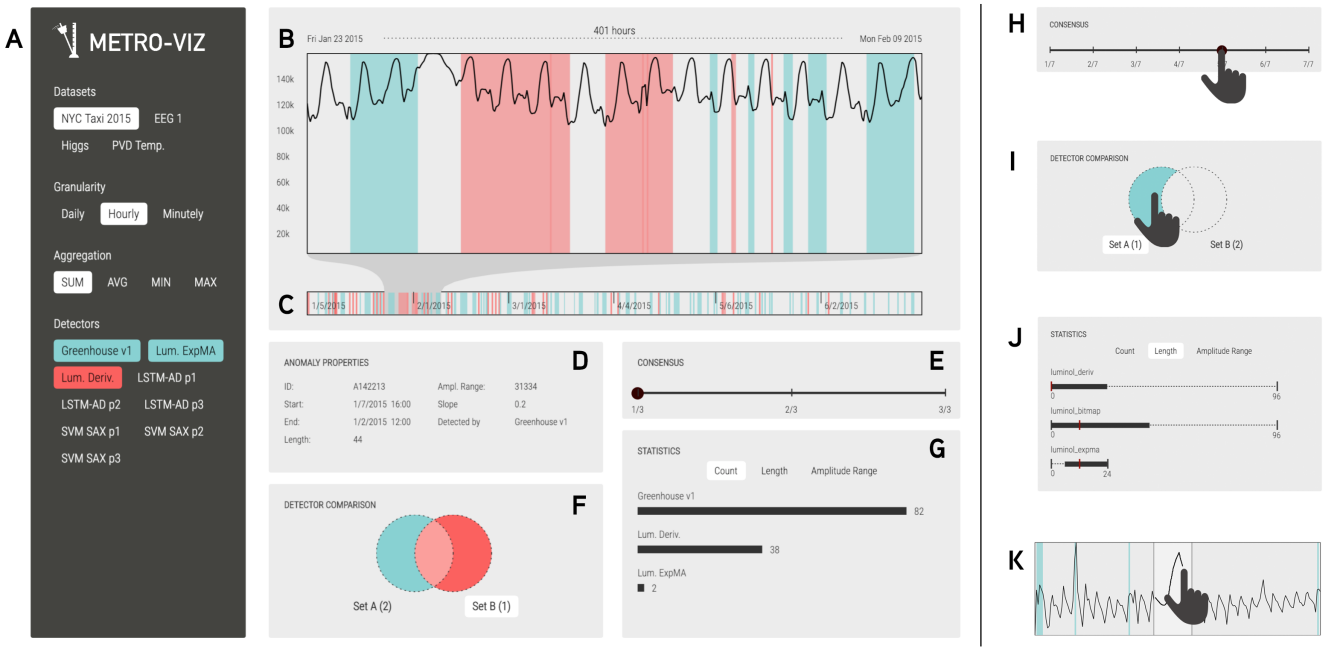

Figure 2: Left: Screenshot of Metro-Viz's User Interface, Right: Selected User Interface Components

2.2 Metro-Viz Server: Data Management

Metro-Viz's UI embodies a unique workload that defines the design space of our data management strategies to ensure a usable and scalable interaction experience. Most fundamentally: (i) data is accessed in units of time windows, and (ii) what the user is mostly interested in seeing is one active window at a time into the time series through the Time Series Viewer. A window is a contiguous subsequence of a time series and has four basic properties: (i) the start time of the window, (ii) the granularity of each observation in the window (e.g., hourly), (iii) the aggregation function used to generate each such observation (e.g., hourly sum), and (iv) the total number of observations in the window (e.g., 24 hourly sums). Each of the server components manages these windows based on the requests it receives from the UI. For example, the strategies used by the Interaction Manager when a user pans from left to right in the Time Axis differ from those used when the Detector Manager begins detecting anomalies. These strategies are described below.

Interaction Manager (IM). User actions in the Time Series Viewer and the Time Axis are managed by the IM. When the user pans along the Time Axis, the IM infers the four properties of the windows it provides to the Client. It then requests these windows from the Storage Manager (SM). Depending on user's behavior, it can request the SM to pre-fetch or cache certain windows. For example, when a user starts panning in one direction, the IM requests to pre-fetch windows in that direction. If the user modifies data during hypothesis testing, the IM generates a new version of the window which is passed to the SM to cache. If this is performed while detectors are selected, the window is also passed to the Detector Manager for re-detection.

Detector Manager (DM). When a user selects a detector from the Main Menu, the DM manages both the available detectors and window management strategies for the current AD task. The DM allows users to plug in different anomaly detectors through a simple interface. These detectors are also shown in the Main Menu. Because AD can be an expensive task, the DM performs *on-demand detection*: it does not greedily perform AD on the entire time series data; instead, it performs detection on the current window immediately and proceeds to detect on windows close by, moving further out. The DM performs this detection in a separate thread for each detector selected. If the user jumps to a location in the time series where detection has not been performed, the DM immediately aborts its current detection task and restarts at the new window. Similarly, if the user modifies the data through the Time Series Viewer, the DM only runs re-detection on the windows affected by this modification. The DM determines the windows it needs based on the properties of the detector. For example, some detectors consider an interval prior to an observation in determining if that observation is anomalous. For each window, the DM returns a result window of boolean values indicating if an observation is anomalous. These are stored as highly compressed and computationally efficient bitmaps, and passed to the Anomaly Manager.

Anomaly Manager (AM). When the user compares results of anomaly detectors using the Consensus Slider or the Set Operation Widget, the bitmap representation allows the AM

to perform highly efficient set operations. Bitmaps also facilitate calculating the properties of the anomaly ranges for statistical evaluation. For example, in presence of labeled data (also stored as bitmap windows), accuracy statistics can be calculated from the result of bit-wise operations between the result windows and the labeled data windows. These operations are once again performed on each window, beginning with the window being displayed in the Time Series Viewer to maintain interactive latency in the UI.

Storage Manager (SM). All windows and result windows are managed by the SM. It is the only component that accesses the raw on-disk time series data when required. It generates windows from the time series data and pre-fetches and caches these based on the other components' needs. Over time, the SM builds a collection of windows of the time series at various granularities and aggregations. These windows serve as pre-aggregated views of the time series data. The SM then re-uses these windows where possible. For example, if the in-memory cache already contains hour-granularity windows aggregated by summing minute-granularity values, the SM would prefer calculating day-granularity windows by summing the hourly windows, instead of using the minutely values or the data on disk. When the SM's cache is full, it flushes these windows to on-disk storage. By not discarding any window, the SM minimizes the need to access the raw time series data as well as re-calculation.

3 DEMONSTRATION SCENARIO

We will demonstrate Metro-Viz using several domain-specific datasets. First, we will guide users through a hypothetical scenario. The user can then examine any of the available datasets and detectors in Metro-Viz to gain a sense of its usability and scalability.

In the guided demonstration, we use the NYC taxi data, simulating a smart city application that shows the number of pick-ups per minute for a six-month period starting in January 2015 [2]. We use a labeled subset of this dataset to evaluate the detected anomalies [1].

Let us now walk through a typical iterative AD workflow, where a civic studies researcher is looking at the total passenger count over time in New York City. The researcher wants to compare various AD algorithms and reason about specific anomalies. In this scenario, she works with a dataset and two LSTM-based anomaly detectors, Greenhouse (zero-positive) [8] and LSTM-AD (not zero-positive) [9], and tries to answer the following questions: How well do the two detectors find the interesting elements in the data? How many of the anomalies are FPs? Are there any correlations between the results of the two detectors?

She first explores the data for trends or periodicity, getting an intuitive sense of what the data looks like. She pans along the Time Axis, observing these trends in the Time Series Viewer. She does the same at the granularity of a day. She then selects two detectors and observes the detected anomalies. To compare the detectors, she assigns each to a set in the Set Operation Widget and selects one set, then the other. Doing so updates the highlighted anomalies.

She then looks at Statistical Evaluation to gain an overall sense of the detectors' performance. She notices that Greenhouse seems less sensitive to anomalies with small amplitude ranges, and that LSTM-AD is more likely than Greenhouse to detect short anomalies. She decides to test this by modifying a non-anomalous range to be similar in shape and length to the anomalies detected by LSTM-AD. Similarly, she simulates an anomaly detected by Greenhouse in a non-anomalous range. Curious whether the anomalous days remain at the hourly granularity, she clicks on the hourly granularity causing Metro-Viz to apply the detectors to hourly data. With this iterative workflow, she can begin exploring the detectors, granularity, and aggregation that best suit her application.

ACKNOWLEDGMENTS

We thank Junjay Tan for his contribution. This research has been funded in part by Intel and by NSF grant IIS-1514491.

REFERENCES

[1] [n. d.]. Numenta Anomaly Benchmark. http://github.com/numenta/.
[2] [n. d.]. NYC OpenData. http://opendata.cityofnewyork.us/.
[3] Marco Cavallo and Cagatay Demiralp. 2017. Track Xplorer: A System for Visual Analysis of Sensor-based Motor Activity Predictions. In *IEEE VIS DSIA Workshop*.
[4] Sye-Min Chan et al. 2008. Maintaining Interactivity While Exploring Massive Time Series. In *IEEE VAST Symposium*.
[5] Yeukyin Chan et al. 2017. Querying and Exploring Polygamous Relationships in Urban Spatio-Temporal Data Sets. In *ACM SIGMOD*.
[6] Varun Chandola et al. 2009. Anomaly Detection: A Survey. *ACM Computing Surveys* 41, 3 (2009).
[7] Philipp Eichmann et al. 2015. Evaluating Subjective Accuracy in Time Series Pattern Matching using Human-Annotated Rankings. In *ACM IUI Conference*.
[8] Tae Jun Lee et al. 2018. Greenhouse: A Zero-Positive Machine Learning System for Time Series Anomaly Detection. In *SysML Conference*.
[9] Pankaj Malhotra et al. 2015. Long Short Term Memory Networks for Anomaly Detection in Time Series. In *ESANN Symposium*.
[10] Miro Mannino and Azza Abouzied. 2018. Qetch: Time Series Querying with Expressive Sketches. In *ACM SIGMOD Conference*.
[11] John Meehan et al. 2017. Data Ingestion for the Connected World. In *CIDR Conference*.
[12] Rodica Neamtu et al. 2017. Interactive Time Series Analytics Powered by ONEX. In *ACM SIGMOD Conference*.
[13] Martin Steiger et al. 2014. Visual Analysis of Time Series Similarities for Anomaly Detection in Sensor Networks. *Computer Graphics Forum* 33, 3 (2014).
[14] Nesime Tatbul et al. 2018. Precision and Recall for Time Series. In *NeurIPS Conference*.
[15] Kostas Zoumpatianos et al. 2015. RINSE: Interactive Data Series Exploration with ADS+. *PVLDB* 8, 12 (2015).