A Demonstration of the Exathlon Benchmarking Platform for Explainable Anomaly Detection

Vincent Jacob[†] Fei Song[†] Arnaud Stiegler[†] Bijan Rad[†] Yanlei Diao[†] Nesime Tatbul[*]
[†]Ecole Polytechnique, France [*]Intel Labs and MIT, USA
{vincent.jacob,fei.song,arnaud.stiegler,bijan.rad,yanlei.diao}@polytechnique.edu,tatbul@csail.mit.edu

ABSTRACT

In this demo, we introduce Exathlon – a new benchmarking platform for explainable anomaly detection over high-dimensional time series. We designed Exathlon to support data scientists and researchers in developing and evaluating learned models and algorithms for detecting anomalous patterns as well as discovering their explanations. This demo will showcase Exathlon's curated anomaly dataset, novel benchmarking methodology, and end-to-end data science pipeline in action via example usage scenarios.

PVLDB Reference Format:
Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, Nesime Tatbul. A Demonstration of the Exathlon Benchmarking Platform for Explainable Anomaly Detection. PVLDB, 14(12): 2827 - 2830, 2021.
doi:10.14778/3476311.3476355

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/exathlonbenchmark/exathlon.

1 INTRODUCTION

Recent advances in machine learning (ML) and data science led to a surge of interest in advanced analytics techniques over large and complex datasets. Detecting and explaining anomalous patterns in high-dimensional time series data is a prominent example. Despite growing efforts, the diversity of time series applications from IoT to finance, often noisy nature of the collected datasets, and the contextual variations in anomaly types and instances challenge the creation of robust and generalizable solutions. As a result, there is a wide variety of anomaly detection (AD) and explanation discovery (ED) techniques that differ in their functionality and performance. Lack of high-quality data repositories and benchmarking tools makes it hard to repeat, evaluate, and compare these current solutions as well as designing new ones in a well-informed manner.

As a community resource to support data scientists and engineers, we have recently proposed *Exathlon*, the first comprehensive benchmark for explainable AD over high-dimensional time series data [7]. Exathlon focuses on the familiar domain of metric monitoring in large-scale computing systems, and provides a benchmarking platform that consists of: (i) a curated anomaly dataset, (ii) a novel benchmarking methodology for AD and ED, and (iii) an end-to-end data science pipeline for implementing and evaluating AD and ED algorithms based on the provided dataset and methodology.

In this demo, we present our benchmarking platform and illustrate its practical use via two different scenarios: (i) *as an experimentation tool* by an ML researcher interested in developing and comparing two alternative deep learning (DL) models for AD in terms of their predictive performance and explainability, (ii) *as a data analysis tool* by an application engineer interested in choosing an AD and ED solution to deploy for improving the performance of his real-time e-commerce applications. Exathlon provides a modular and extensible data science pipeline and an interactive visual frontend to productively support these kinds of exploratory tasks. Our key goal is to introduce this new public resource to the database community for interactive experience and feedback.

Both benchmarking platforms and time series analytics tools have been subjects of past demos at database venues (e.g., [4, 5, 9, 14]). This demo will present a new platform for a novel benchmark. While there are several ML/DL-based analytics benchmarks (e.g., ADABench [10], DAWNBench [3]), there is only one public benchmark that specializes on time series AD (NAB [8]), yet with a much narrower focus than Exathlon (streaming AD over univariate time series). For ED as well as explainable AD, Exathlon is the first public benchmark to our knowledge. In what follows, we provide a brief overview of Exathlon and describe the two demo scenarios that we are planning to show at the conference.

2 EXATHLON OVERVIEW

Exathlon consists of three key components: a labeled dataset, an evaluation framework, and a data science pipeline. We summarize each, with more emphasis on the pipeline, which forms the basis of our platform implementation. Further details are in our paper [7].

Anomaly Dataset. Exathlon's dataset has been systematically constructed based on real data traces collected from 93 repeated executions of 10 distributed streaming applications on a 4-node Spark cluster over a period of 2.5 months. Each of these executions includes 5 randomly selected applications running concurrently. During each execution, we collected metrics from both Spark's monitoring and instrumentation interface and the underlying operating system. All in all, each trace consists of a total of 2,283 metrics recorded once per second for 7 hours on average, constituting a multi-dimensional time series of 24.6GB in size. We first collected 59 *undisturbed traces* to characterize the normal execution behavior of our Spark applications; we then introduced various anomalous events, via a disruptive event generator (DEG), during Spark's job execution to generate 34 *disturbed traces*. There are 6 types of anomalous events: T1: bursty input, T2: bursty input until crash, T3: stalled input, T4: CPU contention, T5: driver failure, T6:

Proceedings of the VLDB Endowment, Vol. 14, No. 12 ISSN 2150-8097.
doi:10.14778/3476311.3476355

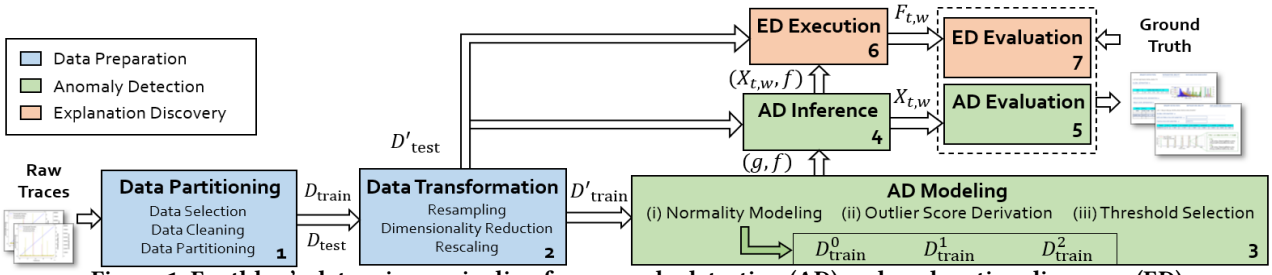


Figure 1: Exathlon's data science pipeline for anomaly detection (AD) and explanation discovery (ED)

executor failure, and a total of 97 anomaly instances in the whole dataset. For each of these anomalies, we provide ground truth labels for both the *root cause interval* (time period during which a DEG was running) as well as the corresponding *extended effect interval* (post-DEG time period until system returned to normal or crashed).

Evaluation Framework. Exathlon evaluates AD and ED algorithms in terms of both functionality and computational performance, using a well-defined set of criteria and metrics.

For AD, there are four criteria in increasing level of difficulty: *AD1: anomaly existence*, *AD2: range detection*, *AD3: early detection*, and *AD4: exactly-once detection*. These are all evaluated in terms of *range-based precision and recall* metrics [13]. Further, Exathlon offers 4 learning settings (LS1-LS4) to capture how well a learned AD model generalizes to different Spark workload characteristics, by selecting the training and test datasets accordingly. LS1 and LS3 train the AD models on a single-app trace basis, whereas LS2 and LS4 do so with multiple-app traces together. Similarly, LS3 and LS4 use only undisturbed traces in training and all disturbed traces in testing, while LS1 and LS2 allow the model to peek into small non-anomalous segments from the disturbed traces during training to capture some Spark workload context from the test traces.

Once an anomaly is flagged by an AD method, then its explanation can be generated using an ED method. ED methods can be *model-dependent* [11] or *model-free* [1, 15]. Furthermore, they can generate local explanations for each anomaly as well as global explanations for multiple anomalies appearing across a larger trace. Accordingly, for ED, we define two benchmarking criteria: *ED1: local explanation* and *ED2: global explanation*. These are measured using three key metrics: *conciseness* (number of features used in an explanation), *consistency* (common set of features in explanations of similar anomalies), and *accuracy* (precision/recall of an anomaly explanation vs. the ground truth if applied as a predictive model).

For computational performance, there are 3 criteria that can be evaluated by varying dimensionality and size of the dataset: AD training scalability, AD inference efficiency, and ED efficiency.

Data Science Pipeline. We designed and implemented an end-to-end pipeline for explainable time series AD. This pipeline includes all data processing steps necessary to turn our raw datasets into AD and ED results together with their benchmark scores. Our design is modular and extensible, making it easy to add new AD and ED techniques, as well as allowing the creation of multiple variants of the pipeline steps to experiment with and compare. For example, training data preparation for different AD learning settings (LS1-LS4) or scoring AD results for different criteria levels (AD1-AD4) can be easily configured, run, and compared in our pipeline.

Figure 1 shows an overview of Exathlon's full pipeline. First, the raw input traces are partitioned (1) and transformed (2) for preparing the datasets, D'_{train} and D'_{test}, for model training and testing, respectively. The *Data Partitioning* module (1) handles initial data selection, cleaning, and splitting, whereas the *Data Transformation* module (2) applies further resampling, dimensionality reduction, and rescaling as needed by the experiment. The resulting D'_{train} consists of undisturbed traces and is used for AD normality modeling along with outlier score derivation and threshold selection (3). This *AD Modeling* step (3) results in a pair of functions (g, f), respectively, that will assign outlier scores and binary predictions to the records in each trace of D'_{test}. These functions are for predicting anomalous ranges $X_{t,w}$ in D'_{test} through the *AD Inference* step (4), which are evaluated vs. the ground truth (real anomaly ranges) using the AD evaluation criteria (AD1-AD4) in *AD Evaluation* (5). After anomalous ranges are detected, they are provided as inputs to the *ED Execution* module (6) to derive explanations $F_{t,w}$. These explanations can then be evaluated according to the chosen ED evaluation criteria (ED1-ED2) by the *ED Evaluation* module (7).

We implemented this pipeline design using Python. To turn it into an easy-to-use platform, we also added some utilities, such as an interactive frontend to configure pipelines as well as visualizing and exploring the results on demand. We provide further details on this frontend as part of the demo scenarios described in the next section. The dataset, code, and documentation for Exathlon are publicly available at https://github.com/exathlonbenchmark/exathlon.

3 DEMO SCENARIOS

This demo will showcase Exathlon's benchmarking platform for explainable AD in action via two example usage scenarios.

3.1 Exathlon as an Experimentation Tool

Our first scenario shows how Exathlon can be used as an experimentation tool by data science and machine learning researchers.

Alice is an ML researcher interested in time series AD. With a focus on DL-based AD methods, she is curious to find out how forecasting-based approaches compare against reconstruction-based ones in terms of their accuracy and explainability. She picks one popular deep neural network (DNN) architecture from each category, Long Short-Term Memory (LSTM) [2] and Auto-Encoder (AE) [6] respectively, to implement and experimentally compare.

Data Preparation. The first step is to prepare the datasets to be used in model training and testing. Knowing that most AD approaches focus on capturing normality in their models, she decides

to use Exathlon's 59 undisturbed traces for training. The remaining 34 disturbed traces with anomalies will be used for testing. These dataset selection and partitioning steps can be handled using Exathlon's *Data Partitioning* module. Then Alice quickly realizes that there are too many dimensions in the data and decides to apply dimensionality reduction. Exathlon provides her with a few options as part of its *Data Transformation* module; she settles on custom feature-set selection, reducing the number of dimensions to 19.

Model Building and Evaluation. Once the datasets are ready, Alice can now train her LSTM and AE models in two separate pipeline deployments. Her models are implemented by extending the *AD Modeling* module of Exathlon's pipeline, including outlier score derivation and threshold selection. The resulting models are stored in a model repository provided by Exathlon. To see how they perform, each model is tested using the *AD Inference* module with the test dataset and evaluated using the *AD Evaluation* module.

AD Method Comparison. Exathlon's visual frontend reports the evaluation results. For example, Figure 2(a) shows the performance metrics reported for LSTM and AE evaluated at different AD levels of the benchmark. Alice observes that for AD1 (anomaly existence), LSTM outperforms AE in precision, recall, and F1-score. However, for AD2 (range detection) and AD3 (early detection), AE performs better than LSTM in recall and F1-score. Finally, for AD4 (exactly-once detection), LSTM loses even more performance than AE.

Method Analysis. Alice then wants to analyze LSTM and understand why it starts to lose performance for AD2-AD4. She clicks on the "Separation Ability" tab to obtain additional profiling results. As shown in Figure 2(b), Exathlon's frontend offers separation metrics (Area Under Precision-Recall Curve, AUPRC) and separation plots at the global level (all test traces), at the application level (test traces of a specific Spark application), or at the trace level. Each separation plot shows the outlier scores assigned to the normal records, as well as those of the anomalous records of each anomaly type. At the global level, the outlier scores of normal records overlap significantly with some of the anomalous records, hence yielding poor separation power. To understand why, Alice asks for trace-level separation, for which Exathlon shows a table of trace-level AUPRC scores, one row for each trace. When Alice clicks on a specific row (trace), Exathlon shows a line plot of the outlier scores assigned to different records. She then sees that the outlier scores produced by LSTM often exhibit discontinuous *spikes*. For range detection (AD2), such frequent mixes of high and low values make it hard to produce continuous ranges of high outlier scores, penalizing recall when the outlier threshold is set high, or precision when the threshold is set low. Alice further wants to analyze the AE method to understand why its F1-score is not high. Similarly, she clicks on the "Separation Ability" tab to compare the separation plots at the global, application, and trace levels, as shown in Figure 2(c). While the trace level separation is quite good, as soon as the same AD model is run over different test traces, the outlier scores assigned to normal records in different traces start to spread widely, hence increasing the overlap with the scores of anomalous records.

Anomaly Type Analysis. Exathlon further allows Alice to filter the evaluation results by anomaly type, so that she can understand how well a particular AD method is able to detect anomaly types T1-T6.

Explainability Comparison. Besides the accuracy of the two AD models, Alice is also interested in comparing them in explainability, e.g., whether the anomalies detected by each model can be explained well. Among the available ED methods, Alice chooses LIME [11], a well-known method for explaining model predictions. For both methods, LSTM-LIME and AE-LIME, Exathlon's frontend reports a table of evaluation metrics for discovered explanations, including conciseness, consistency, and running time. In addition, Alice can choose a specific trace and examine the explanations returned for individual anomalies (e.g., see Figure 2(d)), as well as the related ED measures of such local explanations; she can further examine the ED measures of global explanations by choosing a subset of traces and anomalies of interest to her. From all of these results, Alice observes that AE-LIME generates more concise, locally stable, and globally consistent explanations for detected anomalies than LSTM-LIME. This observation is consistent with the earlier observation that LSTM often produces outlier scores in discontinuous *spikes*, hence preventing concise and consistent explanations.

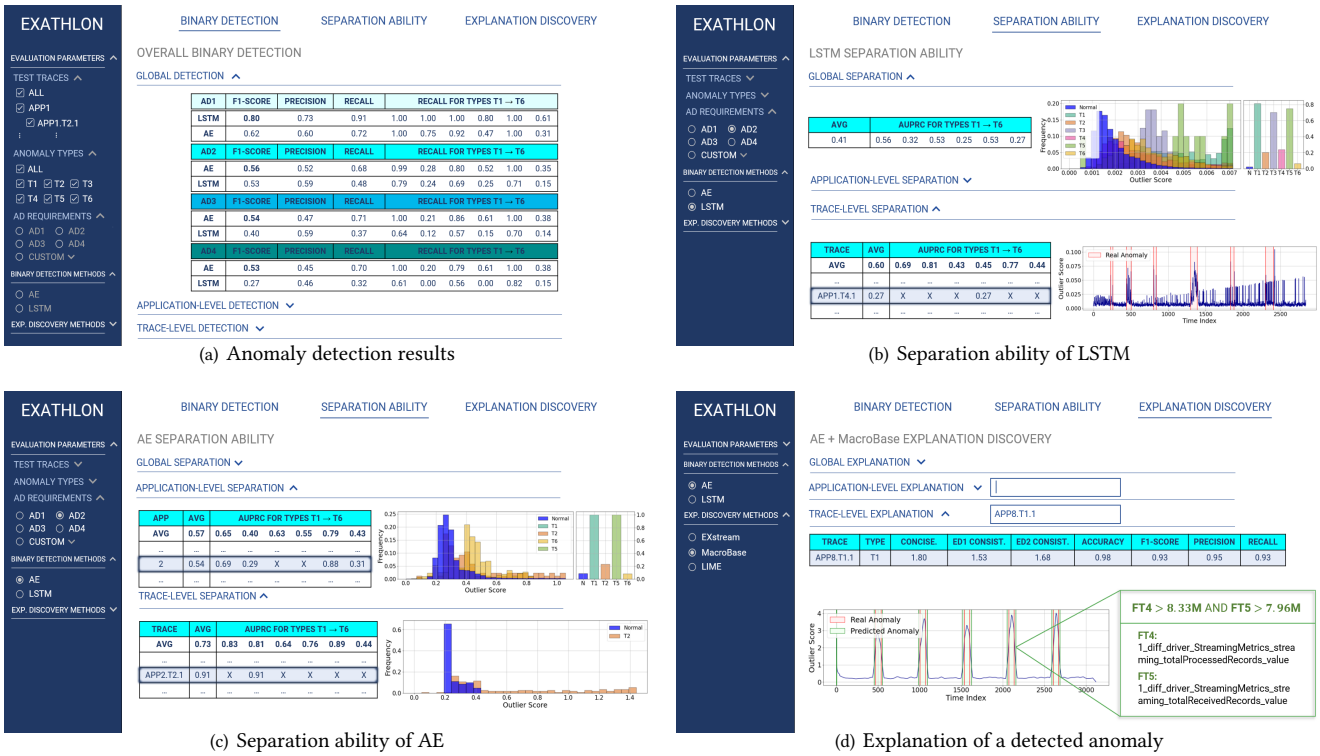
3.2 Exathlon as a Data Analysis Tool

Our second scenario demonstrates how Exathlon can be used as a data analysis tool by application users or domain experts.

Bob is an application performance engineer of an e-commerce platform, who wants to identify and remove performance bottlenecks for large-scale streaming applications running on Spark. He thinks the majority of such bottlenecks are due to input-related anomalies, such as bursty or stalled inputs (e.g., T1-T2-T3 anomalies in Exathlon). These anomalies lead to high latencies, which is unacceptable for real-time streaming applications that affect business decisions about sales strategies and inventory management. Bob aims to find the right AD and ED methods to use which can meet the following requirements: (1) *early online* AD, preferably with no duplicate detection results; (2) efficient *online* ED for each detected anomaly, which points to the root cause in the underlying system and hence can enable timely corrective action; (3) consistent explanations of anomalies of the same type, so that Bob can examine anomalies from multiple traces *offline* and recognize the common anomaly patterns in the system. Bob runs the Exathlon pipeline in order to find the most suitable AD and ED methods for deployment.

Consider the AD methods first. This demo will have a number of pre-trained AD models, including LSTM [2], AE [6], and BiGAN [12], with multiple variants trained under different benchmark settings (i.e., LS1-LS4). Their performance for T1-T2-T3 anomalies is reported in the Exathlon frontend like the table in Figure 2(a). By comparing AD3 (early detection) and AD4 (exactly-once detection) results, as well as checking detailed profiling results, Bob chooses the AE method as it meets requirement (1) the best.

Bob then investigates various ED methods, both model-free (e.g., EXstream [15], MacroBase [1]) and model-dependent (e.g., LIME [11]). Exathlon's frontend allows Bob to compare these methods in several ways: (*i*) *Form of explanation*: For each ED method, Bob selects a specific test trace and examines the individual explanations returned for detected anomalies. He sees that LIME returns explanations as feature importance scores between [0, 1], while EXstream and MacroBase return explicit logical formulas. An example explanation from MacroBase is shown in Figure 2(d) for a bursty input anomaly, which has caused significant scheduling delay because existing resources were not sufficient for handling the input rate. It states that the newly received (processed) records by



(a) Anomaly detection results

(b) Separation ability of LSTM

(c) Separation ability of AE

(d) Explanation of a detected anomaly

Figure 2: Exathlon's visual frontend

the running application were greater than 7.96M (8.33M) during the anomaly period (and one can check that such values should indeed be around 7.80M for this trace). Bob finds such explanations to be much more informative than feature importance scores. (*ii*) *Efficiency and local consistency*: Between EXstream and MacroBase, Exathlon's reported metrics show that EXstream can return an explanation with an avg. delay of 0.01 sec, while MacroBase requires 1-18 sec on avg. Further, explanations returned by EXstream are more succinct and locally stable in the face of small perturbation of data. Considering these, Bob prefers EXstream as it better meets requirement (2). (*iii*) *Global consistency*: Finally, Bob examines all the traces that contain a specific anomaly type and sees that MacroBase achieves the best global consistency scores. Since it is offline analysis, the longer running time of MacroBase is acceptable. Therefore, Bob chooses MacroBase to generate such explanations and summarizes them into a small set of patterns in his business report.

ACKNOWLEDGMENTS

This work was supported by the European Research Council (ERC) Horizon 2020 research and innovation programme (grant n725561).

REFERENCES

[1] Peter Bailis, Edward Gan, Samuel Madden, Deepak Narayanan, Kexin Rong, and Sahaana Suri. 2017. MacroBase: Prioritizing Attention in Fast Data. In *ACM SIGMOD Conference*. 541–556.

[2] Loïc Bontemps, Van Loi Cao, James McDermott, and Nhien-An Le-Khac. 2016. Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Networks. In *FDSE Conference*. 141–152.

[3] Cody Coleman, Daniel Kang, Deepak Narayanan, Luigi Nardi, Tian Zhao, Jian Zhang, Peter Bailis, Kunle Olukotun, Christopher Ré, and Matei Zaharia. 2019. Analysis of DAWNBench, a Time-to-Accuracy Machine Learning Performance Benchmark. *ACM SIGOPS Operating Systems Review* 53, 1 (2019), 14–25.

[4] Fabien Duchateau, Zohra Bellahsene, and Ela Hunt. 2007. XBenchMatch: A Benchmark for XML Schema Matching Tools. In *VLDB Conference*. 1318–1321.

[5] Philipp Eichmann, Franco Solleza, Nesime Tatbul, and Stan Zdonik. 2019. Visual Exploration of Time Series Anomalies with Metro-Viz. In *ACM SIGMOD Conference*. 1901–1904.

[6] Geoffrey Hinton and Ruslan Salakhutdinov. 2006. Reducing the Dimensionality of Data with Neural Networks. *Science* 313, 5786 (2006), 504–507.

[7] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. 2021. Exathlon: A Benchmark for Explainable Anomaly Detection over Time Series. *Proceedings of the VLDB Endowment (PVLDB)* 14, 12 (2021).

[8] Alexander Lavin and Subutai Ahmad. 2015. Evaluating Real-Time Anomaly Detection Algorithms - The Numenta Anomaly Benchmark. In *ICMLA*. 38–44.

[9] Rodica Neamtu, Ramoza Ahsan, Charles Lovering, Cuong Nguyen, Elke A. Rundensteiner, and Gábor N. Sárközy. 2017. Interactive Time Series Analytics Powered by ONEX. In *ACM SIGMOD Conference*. 1595–1598.

[10] Tilmann Rabl, Christoph Brücke, Philipp Härtling, Stella Stars, Rodrigo Escobar Palacios, Hamesh Patel, Satyam Srivastava, Christoph Boden, Jens Meiners, and Sebastian Schelter. 2019. ADABench - Towards an Industry Standard Benchmark for Advanced Analytics. In *TPCTC*. 47–63.

[11] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *ACM SIGKDD Conference*. 1135–1144.

[12] Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. 2017. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. In *IPMI Conference*. 146–157.

[13] Nesime Tatbul, Tae Jun Lee, Stan Zdonik, Mejbah Alam, and Justin Gottschlich. 2018. Precision and Recall for Time Series. In *NeurIPS Conference*. 1924–1934.

[14] Robert Ulbricht, Claudio Hartmann, Martin Hahmann, Hilko Donker, and Wolfgang Lehner. 2016. Web-based Benchmarks for Forecasting Systems: The ECAST Platform. In *ACM SIGMOD Conference*. 2169–2172.

[15] Haopeng Zhang, Yanlei Diao, and Alexandra Meliou. 2017. EXstream: Explaining Anomalies in Event Stream Monitoring. In *EDBT Conference*. 156–167.