Anomaly Detection and Explanation Discovery on Event Streams

Fei Song
Inria, France
Ecole Polytechnique, France
fei.song@inria.fr

Boyao Zhou, Quan Sun,
Wang Sun, Shiwen Xia
Ecole Polytechnique, France
{boyao.zhou,quan.sun,wang.sun,
shiwen.xia}@polytechnique.edu

Yanlei Diao
Ecole Polytechnique, France
University of Massachusetts
Amherst
yanlei.diao@polytechnique.edu

ABSTRACT

As enterprise information systems are collecting event streams from various sources, the ability of a system to automatically detect anomalous events and further provide human readable explanations is of paramount importance. In this position paper, we argue for the need of a new type of data stream analytics that can address anomaly detection and explanation discovery in a single, integrated system, which not only offers increased business intelligence, but also opens up opportunities for improved solutions. In particular, we propose a two-pass approach to building such a system, highlight the challenges, and offer initial directions for solutions.

ACM Reference Format:
Fei Song, Boyao Zhou, Quan Sun, Wang Sun, Shiwen Xia, and Yanlei Diao. 2018. Anomaly Detection and Explanation Discovery on Event Streams. In *Proceedings of ACM BIRTE Workshop (BIRTE'18).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

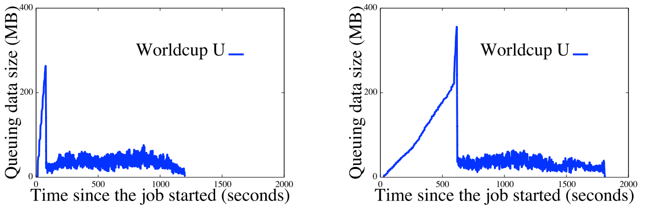
Enterprise information systems are collecting high-volume event streams from various sources such as financial data feeds, news feeds, application monitors, and system monitors. Among many of the needs to improve Business Intelligence (BI), the ability of a data stream system to automatically detect anomalous events from raw data streams and further provide human readable explanations for such events is of paramount importance.

In this work, we consider anomalies as the patterns in data that deviate from expected behaviors [5]. An anomaly can be detected by an automatic procedure, for which the state-of-the-art includes statistical, SVM, and clustering based techniques [3, 5, 6]. However, there is one key component missing in all these techniques: finding the **best explanation** for the anomalies detected, or more precisely, a human-readable formula offering useful information about what has led to the anomaly. The state-of-art methods mainly focus on detecting anomalies, but not providing useful information about what led to the anomaly. Without a good explanation, anomaly detection is only of limited use: the end user knows that something anomalous has just happened, but has limited or no understanding of how it has arisen, how to react to the situation, and how to avoid it in the future.

Our prior work offered concrete examples to illustrate the difficulty in generating explanations [18]. One example, as shown in Figure 1, is the case that an engineer runs the same analytical job over a large dataset every day and monitors the job progress in a dashboard. Figure 1(a) shows the normal progress that he sees every day. However, one day he starts to observe a different progress pattern, as shown in Figure 1(b). While the anomaly is already detected based on the visual difference, the user further needs an explanation that can help answer a series of questions: "*What is happening with the submitted job*?" "*Is the phenomenon caused by the bugs in the code or some system anomalies*?" "*Should I wait for the job to complete or re-submit it*?" "*What should I do to bring the job progress back to normal*?"

However, most data stream systems cannot generate explanations automatically, even if the anomaly has been signaled. Relying on the human expert to analyze the situation and find out explanations is tedious and time consuming, sometimes even not possible. In the above example, it will be very time-consuming for the engineer to pull a variety of Hadoop and system traces, parse them, correlate them based on the temporal relationship, try different ways of feature engineering, compare the features in the abnormal case with those in the normal case, and experiment with a set of data mining tools to finally derive an explanation.

BIRTE'18, August 2018, Rio de Janeiro, Brazil
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn



(a) Normal progress of the job. (b) Abnormal progress of the job.

Figure 1: Illustration of abnormal progress of a Hadoop job.

In this position paper, *we argue for the need of a new type of data stream analytics that can address anomaly detection and explanation discovery in a single, integrated system.* In the literature these two topics have been addressed in isolation: while anomaly detection has been studied intensively in the data mining community [3, 5, 6], explanation discovery with the goal of human-readable formulas has recently received attention in the database community [13, 15, 18]. On the latter topic, the line of work [13, 15] explains outliers for only group-by aggregate queries and finds a logical formula to describe a subset of tuples that contribute the most to the excessively high or low aggregate value of a specific group. It is hard to extend such work to explain anomalies detected by an arbitrary data mining algorithm. Our prior work [18] assumes that the normal and abnormal time periods are already given by the user (treated as ground truth) and finds explanations to best distinguish the abnormal periods from the normal ones. It does not handle the cases that such ground truth is not available.

The work closest to ours is MacroBase [1], a data analytics engine that helps the user prioritize attention over data streams, and offers modules for both outlier detection and explanation discovery. However, with a focus on fast data streams, it chooses to use simple methods that can provide high efficiency but may be insufficient for handling complicated anomalous events. For example, it performs outlier detection by a density-based method called MAD. While being simple and robust, MAD is suitable only for detecting point outliers, not contextual or collective outliers [5] which are often related to time-series and sequence data. As a example given in [5], a low temperature in winter might be normal, but the same temperature in summer would be an anomaly. To detect the latter as an anomaly, temporal dependency has to be accounted for. For explanation discovery, MacroBase finds an explanation in the form of conjunctive predicates, where each predicate compares an attribute to a categorical constant. It lacks the ability to generate richer explanations using logical operators (∨), relational operators (>, ≥, <, ≤), or sequential patterns.

To respond to the complexity of real-world anomalies, in this work we propose an integrated system for anomaly detection and explanation discovery that employs state-of-the-art techniques from both the Machine Learning and Database communities and addresses challenges in making them work effectively together. We posit that such an integrated approach will not only increase business intelligence, but also open up opportunities for improved solutions:

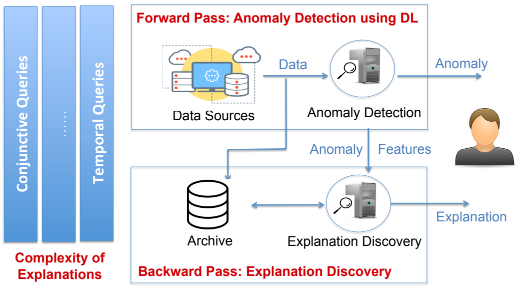


Figure 2: An integrated system for anomaly detection and explanation discovery

1. It will provide better understanding how these two tasks relate to each other. In particular, we propose *prediction accuracy* and *efficiency* as the criteria for an anomaly detection solution. In comparison, we propose *prediction accuracy* and *conciseness* as the desirable criteria for explanations, where conciseness follows the Occam’s razor principle – if there are two explanations, the simpler one is usually better (due to fewer assumptions made) and is easier for humans to understand. The above proposal points to an approach that the system can use any data mining tool to detect anomalies, subject to the accuracy and efficiency requirements, and then seek explanations by searching for a logical formula that approximates the anomaly detection tool in accuracy while minimizing the size of the formula. This approach will enable the system to embrace new data mining tools for anomaly detection, and at the same time offer a principled way to find explanations that correspond to the detected anomalies.

2. Since we formulate explanations as logical formulas, such formulas may belong to different language classes. As real world applications require the expressive power of explanations to increase from a simple class (e.g., conjunctive queries) to a broader class (e.g., temporal patterns with Kleene closure), an integrated system can gain insights from the relationship between anomaly detection and explanation discovery and tune techniques for them in a coordinated manner. As such, our solution space can be described by the increased complexity of explanations as one dimension and how to perform the two related tasks as another dimension, which is illustrated in Figure 2.

3. Existing techniques for anomaly detection and explanation discovery perform feature engineering separately. An integrated system has the potential to share feature engineering across the two tasks. For instance, (a subset of) the features extracted for anomaly detection may be reused for explanation discovery.

2 A TWO-PASS APPROACH

Towards our goal, we propose a new **two-pass approach** to support anomaly detection and explanation discovery in the same stream analytics system. This approach is motivated by our observation that though closely related, anomaly detection and explanation discovery often differ in the nature of computation (e.g., the optimization objective). It is hard to achieve both by a general-purpose procedure through one pass of data scan. Our prior work [18] showed initial evidence for this argument: We used logistic regression to perform both anomaly prediction and explanation discovery. While the model has good predictive power for some anomalous events, it does not serve as a good explanation. For instance, the learned model assigns non-zero weights to 30 out of 345 input features, and it is hard for the human to understand an explanation with 30 features (factors). Through manual exploration, the domain expert finally selected two features to construct an explanation; however, these two features are ranked low (23 and 24 out of 30) in the learned model.

Therefore, our two-pass approach is designed to handle anomaly detection and explanation discovery in two different passes of the data, as shown in Figure 2. In the forward pass, the live data streams are used to drive anomaly detection, and at the same time archived for further analysis. The detected anomalies will be delivered immediately to the user and the explanation discovery module. Then in the backward pass, explanation discovery runs on both the archived streams and feature sets created in the forward pass. Once the explanation is found, it is delivered to the user, with only a slight delay.

2.1 Anomaly Detection

There exist many anomaly detection algorithms; we refer the reader to the surveys [3, 5, 6] for details. Unlike most of prior work, our goal in anomaly detection includes not only the predication accuracy, but also the potential to assist in explanation discovery. This is the unique challenge in our setting. Below we first present the motivation for us to explore Deep Learning (DL) [9] as a framework for anomaly detection, and then discuss how to choose specific techniques to correspond to the complexity of intended explanations.

Anomaly detection in real-world applications raises two key issues. 1) *Feature space*: The vast amount of raw data collected from network logs, system traces, application traces, etc. does not always present a sufficient *feature set*, which are expected to be carefully-crafted features at an appropriate semantic level for anomaly detection algorithms to work. 2) *Modeling Complexity*: The labeled anomalies are often rare (in some cases non-existent), which indicates the need of *unsupervised learning or semi-supervised learning*. The effective model for anomaly detection may exhibit very complex (nonlinear) relationship with the features, which indicates that the detection algorithms must have good *expressive power*. The *generalization ability* is also critical to anomaly detection since the task is often to detect anomalies that have never happened before. To address both issues, we seek to explore Deep Learning as a framework that addresses feature engineering and anomaly detection in the same architecture.

As we discussed before, the logical formulas representing explanations can be divided into different categories. For the simple class (conjunctive queries), the explanations do not aim to include complex temporal relationships, and hence the dataset can be viewed as time-independent. In this case, the auto-encoder method may be a candidate for anomaly detection, while other more advanced methods may be added later. For a broader class where the explanations include temporal relationships, LSTM is more appropriate for anomaly detection because it inherently models temporal information.

Auto-encoder. A deep auto-encoder aims to learn an artificial neural structure such that the input data can be reconstructed via this structure [2, 7]. In addition, the hidden layer (of the narrowest width in the structure) can be used as a short representation (or essence) of input. The underlying assumption justifying using an auto-encoder for anomaly detection is that the model will be formed by normal data. Consequently, what it will learn is the mechanism for reconstructing data generated by the normal pattern. Hence, the data corresponding to the abnormal behavior should have a higher reconstruction error. In our work we applied auto-encoders in the Hadoop cluster monitoring task. Moreover, this deep auto-encoder extracts a short representation of the original data, which can be used as an extended feature set for explanation discovery in the backward pass.

Long Short-Term Memory. The second method uses Long Short-Term Memory (LSTM) [8]. It is an improved variant of RNNs (Recurrent neural networks), which overcomes the vanishing/exploding gradient difficulty of standard RNNs. It has the ability to process arbitrary sequences of input, and has been used recently to detect anomalies in time series. For example, in [10], the authors applied semi-supervised anomaly detection techniques. They first train a LSTM network of normal behaviors, then apply this network to new instances and obtain their likelihood with respect to the network. The anomaly detection is based on a pre-determined threshold. In this work, the measure used is the Euclidean distance between ground truth and prediction. Alternatively, in [4] the authors assumed that the error vectors follow a multivariate Gaussian distribution. In our work, we are adapting two methods to make them applicable in our setting.

Our initial results from cluster monitoring reveal that the LSTM outperforms the auto-encoder, in F-score and recall, for anomaly detection. Future research questions include: 1) *Tuning architectures and cost functions*: In current work, we only use standard architectures and cost functions for the neutral network architectures. It is worth investigating the best architecture and the customized cost function that can improve the detection accuracy for both methods. 2) *Tradeoffs*: We will further investigate for which workloads they provide better results for anomaly detection. Sometimes even if the intended explanation is a conjunctive query, LSTM still outperforms antoencoder for anomaly detection, which requires further understanding. 3) *Incremental training*: Most DL algorithms are designed for offline processing. However, in a stream environment, new data is arriving all the time and needs to be included in the training dataset. Ideally we need a mechanism to leverage the new data in a timely manner, but incremental training for DL methods is known to be hard. In ongoing work, we are exploring an ensemble method, i.e., to build a set of "weak" detectors on the new data, and then to perform anomaly detection using the combined result. Another idea is to randomly initialize the weights of the last layer and retrain with all data – it is a tradeoff between breaking local optima and reducing training cost.

2.2 Backward Pass: Explanation Discovery

In the backward pass, we aim to find an explanation for a detected anomaly. There are two parallel efforts on explanation discovery: In the database community, recent work [13, 15] aims to discover explanations of query results, but is limited to results of group-by aggregate queries. In the machine learning community, sensitivity tests [11, 12, 14] are designed to determine the importance of each input feature. They seek to provide interpretation for the prediction result on a specific input instance, and hence are more suitable for text and image processing, e.g., to identify important words in a sentence or items in an image.

To form a good explanation for a detected anomaly, we impose several requirements. The first requirement is the *simplicity in formality* of the explanation. An explanation involves complex non-linear functions may have very good prediction power but it is too complex for a human user to understand. To address this, we seek explanations that are easy to interpret by the user. The best choice, to the best of our knowledge, is a logical formula. As such, our work takes an initial step towards the integration of a logic-based approach to explanation discovery with a deep-learning based numerical approach to anomaly detection. Depending on the application needs, the logical formula can be drawn from a simple class known as *conjunctive queries* (CQ), or a higher descriptive class called *temporal pattern queries* (TPQ) which includes temporal patterns with Kleene closure, negation, value predicates, and aggregates [17]. For a given class, each explanation should satisfy two requirements, *prediction accuracy* and *compactness*. We expect the explanation to approximate the anomaly detection model in accuracy, and capture compactness by the size of the logical formula (e.g., the number of atomic predicates in the CQ class, or the number of sequential components in the TPQ class).

To develop a solution in the backward pass, we have the anomaly detection model, the raw input, and an extended feature set built in the forward pass. Now we consider measures that help drive the development and evaluation of an explanation. If we abstract the anomaly detection model as a labeling system, $L(X) = y \in \{0, 1\}$ where X is the input, L will label X as normal (y=0) or abnormal (y=1). The logical formula ϕ with approximately the similar detection accuracy to L is: $\phi(X) = z \in \{0, 1\}$. Then a good design of ϕ should satisfy: 1) 100% recall: the anomalies labeled by L should be labeled as anomalies by ϕ as well; 2) maximum precision: minimize the number of instances that will be labeled as anomalies by ϕ but as normal by L; 3) minimum of $||\phi||$.

Our prior work has developed an initial solution [18] that aims to find the most informative yet compact logical formula constructed from a manually engineered feature set. There are several relevant ideas in this work. First, given a feature set, we devise an *entropy-based, single-feature reward (distance) function* that characterizes the differentiating power over the normal and abnormal cases to which a given feature contributes. Second, we provide a formal definition of optimally explaining anomalies as a problem that maximizes the information reward provided by the explanation using a subset of the feature set. We model the problem of finding an optimal explanation from the feature set as a *non-monotone submodular maximization problem*, which is known to be NP-hard. Our prior work offers only a heuristic solution.

In ongoing work, we extend our solution as follows:

Feature Space. It is crucial to have a sufficient feature space that includes all necessary features for explaining observed anomalies. In general, there are two ways to generate features from raw inputs. One is to use deep learning (DL) to learn non-linear features, but with no clear semantic meanings. The other one is to use a small set of windowed aggregates to form new "smoothed" features. One issue to consider is how the DL-extracted features compare to the smoothed features for explanation discovery. A related question is how to make these two sets of features work together.

In addition, how to address the tradeoffs across the two tasks (anomaly detection and explanation discovery) is another issue to consider. Based on our experiments, RNNs have better performance in anomaly detection accuracy. However, autoencoders are potentially more suitable for explanation discovery. There are two advantages. First, autoencoders are designed to offer a compact representation of the original data, and being able to restrict the size of this representation is an advantage for explanation discovery. Second, autoencoders are designed to persist all the information from the raw data, while RNNs are determined by its specific optimization objective for anomaly detection and hence its internal layers (representation) may not provide sufficient information for explanation discovery.

Submodular Optimization. While prior work [18] can rank individual features based on their distinguishing power between the normal and abnormal cases, our solution to finding a minimum set of features to build a logical formula is only heuristic-based. It is beneficial to design an approximation algorithm that can evaluate different feature sets directly and find a sub-optimal solution with bounded errors.

Constructing Formulas. Once we have selected the minimum relevant feature set, we still need to search for the most appropriate formula. As the application needs for the expressive power of explanations increase from conjunctive queries to temporal pattern queries, the complexity of searching for the optimal logical formula will also increase. For CQ queries, we essentially search through conjunctive formulas built on the most informative features selected through submodular optimization, as described above. For TPQ queries, we need an efficient way to search through the automata models that formally define temporal patterns [17] where the most informative features will be used construct formulas that guide the transition between different automata states.

Approximation to Deep Models. Model induction is an alternative approach to discovery explanation. Recent work [16] has proposed to construct a neural network model which can be easily approximated by a decision tree (which can sometimes be large themselves). To do so, it adds a penalty term to the cost function for backpropogation; more precisely, it introduces a tree penalty term that can force the deep model constructed to be easily approximated by a decision tree, while ensuring that this decision tree has the minimum average length.

Analogously, we could try to customize this approach to construct an anomaly detection model which can be translated directly into a logical formula, or at least be easily approximated by a logical formula with much less complexity. For a simple case, if we restrict the explanation as a Disjunctive Normal Form (DNF), and each predicate of the form ($v\ o\ c$) (v is a feature, c is a constant, and o is one of five operators $\{>, \geq, =, \leq, <\}$), then the explanation can be formed by the paths leading to the leaves labeled as anomalies. Even with this simplified version, we need a definition of the penalty term which is different from [16]: our optimization goal is to optimize the measures described before. For example, if we aim to simplify the explanation, then the penalty term should try to minimize the number of attributes involved in the explanation (paths). A more general issue is how to extend to broader logical formula classes: for other logical systems such as FO, how to transform the decision tree into a logical formula is a major research question.

3 CONCLUSIONS

We sketched our vision for a new type of data stream analytics that addresses anomaly detection and explanation discovery in an integrated system. We argued that such a system opens up opportunities for improved solutions. In particular, we proposed a two-pass approach that performs anomaly detection on live streams and runs explanation discovery over the archived streams and derived feature sets. We sketched a number of directions for solutions, including using deep neutral networks for anomaly detection, and further deriving explanations from these neutral networks through intelligent search over possible logical formulas while minimizing the size of such formulas.

REFERENCES

[1] Peter Bailis, Edward Gan, et al. 2017. MacroBase: Prioritizing Attention in Fast Data. *SIGMOD*, 541–556.

[2] Yoshua Bengio. 2009. Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning* 2, 1 (2009), 1–127.

[3] Monowar H. Bhuyan, D. K. Bhattacharyya, and Jugal K. Kalita. 2014. Network Anomaly Detection: Methods, Systems and Tools. *IEEE Communications Surveys and Tutorials* 16, 1 (2014), 303–336.

[4] Loïc Bontemps, Van Loi Cao, et al. 2017. Collective Anomaly Detection based on Long Short Term Memory Recurrent Neural Network. *CoRR* abs/1703.09752 (2017).

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM Comput. Surv.* 41, 3 (2009), 15:1–15:58.

[6] Manish Gupta, Jing Gao, et al. 2014. Outlier Detection for Temporal Data: A Survey. *IEEE Trans. Knowl. Data Eng.* 26, 9 (2014), 2250–2267.

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

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[9] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[10] Pankaj Malhotra, Lovekesh Vig, et al. 2015. Long Short Term Memory Networks for Anomaly Detection in Time Series. *ESANN*, 89–94.

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

[12] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-Precision Model-Agnostic Explanations. *AAAI*.

[13] Sudeepa Roy, Laurel Orr, and Dan Suciu. 2015. Explaining Query Answers with Explanation-Ready Databases. *PVLDB* 9, 4 (2015), 348–359.

[14] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. *ICML*, 3319–3328.

[15] Eugene Wu and Samuel Madden. 2013. Scorpion: Explaining Away Outliers in Aggregate Queries. *PVLDB* 6, 8 (2013), 553–564.

[16] Mike Wu, Michael C. Hughes, et al. 2018. Beyond Sparsity: Tree Regularization of Deep Models for Interpretability. *AAAI*.

[17] Haopeng Zhang, Yanlei Diao, and Neil Immerman. 2014. On complexity and optimization of expensive queries in complex event processing. *SIGMOD*, 217–228.

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