RISSAD: Rule-based Interactive Semi-Supervised Anomaly Detection

Jiahao Deng and Eli Brown

DePaul University, USA

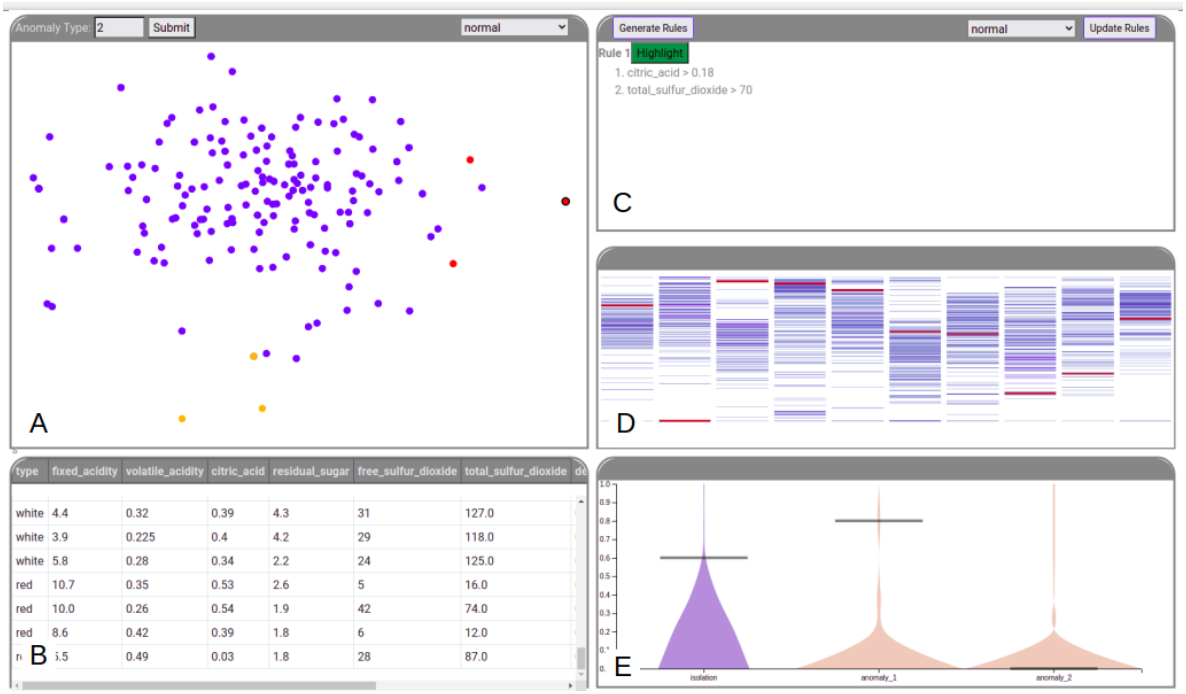

Figure 1: *The interface of the system:* ***A****. point view* ***B****. table view* ***C****. rule panel* ***D****.barcode view* ***E****. violin plot view*

Abstract

Anomaly Detection has gained more and more attention from researchers lately. Due to its nature of lacking reliable ground-truth labels, most of current state-of-art techniques mainly focus on unsupervised learning that does not provide an effective mechanism to interpret the results in a way that is understandable to general public. To address this problem, we present RISSAD: a visual tool that helps users not only to detect anomalies but also illustrate those anomalies in the form of explicit rules. The tool takes a semi-supervised learning approach based on algorithm that relies on partial-labeled dataset. The tool provides functionalities enabling users to label a sample anomaly by the its degree of isolation to the rest of population and how similar it is to existing anomaly types. We demonstrate the effectiveness on quantitative experiments simulated on existing anomaly datasets and a qualitative case study that illustrates a real-world usage scenario.

CCS Concepts

• ***Human-centered computing*** → *Visual Analytics;* • ***Machine Learning*** → *Anomaly Detection;*

1. Introduction

Anomaly detection plays an important role in many areas of research, including education [MXC*19],cyber-security [HLG14], mechanic engineering [GMESK99] ,financial fraud [HMYC18] and Journalism [SSW*17]. In general, An anomaly is vaguely defined as a data point that does not share the similar pattern with the rest of the population. However, this ambiguity in the definition leads to the lack of ground-truth labels in many datasets. Because of this, many traditional supervised learning algorithms such as decision tree, neural networks and multi-class support vector machine, will often perform poorly on problems where in which it is expensive to obtain labels for each training case. [CBK09].

Faced with those challenges, many state-of-art techniques heavily rely on unsupervised learning algorithms such as Local Outlier Factor (LOF) [BKNS00], Isolation Forest [LTZ08] and One-Class SVM [SPST*01]. Despite some promising results delivered by these techniques in various situations, they generally do not provide a robust mechanism for interpretation of the results.

To address this issue with interpretability, efforts are being made to support the understanding of the results with various visual techniques across different application domains. [MXC*19] introduces a system that detects abnormal behaviors of users registered in Massive Open Online Courses. [LGG*17] builds a visual system to identify rare category based on active learning. [ZCW*14] contributes a timeline visualization tool to analyze anomalous user behaviors in social media platforms. Although these studies all made meaningful contributions to structure a pattern that helps users understand the data instead of simply applying a "black-box" machine learning technique, which is generally obscure to users, their target audiences are mostly experts and not the general public. Futhermore, the knowledge gained from the systems, to detect anomalies, is often not simple and intuitive enough to be passed down as clear rules and instructions.

To fill this gap, we built a rule-based visualization tool that generates rules for anomaly detection based on a limited number of labels. The tool contains two steps: in the first step, the user can label anomalous points based on their degree of isolation to the whole population and in the second step, the user will assign more points to each anomaly cluster based on both their degree of isolation to the population and similarity to each anomaly cluster defined by the user. Through evaluation, we have proved that for many datasets, our tool can create rules to detect and describe different types of anomalies within a limited number of interactions without sacrificing on accuracy compared to state-of-art detection algorithms.

2. Related Works

In this section, we will review some of the current most common anomaly detection algorithms and various visual techniques, which have been adopted in the past.

2.1. Anomaly Detection Algorithms

In general, most anomaly detection techniques are traced back to four categories: (1) classification-cased algorithms [HHWB02, MC03,WMCW03] (2) nearest neighbour-based Algorithms [BS03, BKNS00] (3) clustering-based algorithms [MLC07, SPBW12] (4) statistical-based algorithms [KK17,YTWM04].Owing to the differences in the underlying assumptions that the different algorithms are based on, an algorithm may outperform another algorithm in one case but produce inferior results in others. To combine the advantages of various techniques, ensemble approaches have gained popularity in recent years [VC17, ZDH*17]. Dimensionality- reduction such as multidimensional scaling (MDS) [Kru64] and principal component analysis (PCA) [SCSC03] is also used for anomaly detection given its advantage in reducing the computational cost and extracting features in latent space.

2.2. Anomaly Detection Visualization

Combined with detection algorithms explained in Section 2.1, visualizations are widely used to enhance a user's understanding of the problem and supplement the learning process of the chosen technique. For example, [AY19] builds a automated visual system to detect anomalous patterns in human behaviors with a modified Gaussian mixture model (GMM), which is a statistical-based algorithm. [LGG*17] proposes a visual system that detects rare categories using a nearest neighbour-based algorithm. The system also provides a scatter plot generated using Dimensionality Reduction to help users understand the overall distribution of datapoints. [XXM*18] creates a hybrid approach that ensembles multiple state-of-art anomaly detection algorithms and allow users to interact with data while learning the weights for each algorithm. The system also provides users a visual mechanism to select relevant features. To the best of our knowledge, our tool is the first of its type capable of creating rules through an interactive visual system.

3. Algorithm

Our algorithm is a direct modification to algorithm presented in [ZLZ*18]. The original algorithm **ADOA** is formed based on the assumption that anomalies are often isolated from the rest of population and are usually closed to each others and form distinct clusters. The algorithm is implemented in two stages: In stage one, the labeled anomalies are clustered using k-means. Then for each unlabeled point, its isolation score (*IS*) and similarity score (*SS*) are computed separately. *IS* is the probability of being an anomaly using isolation forest [LLYL02] while *SS* is defined as the normalized distance to the center of the nearest anomaly cluster. *IS* is a key measure of how isolated a sample is to the rest the population while *SS* measures how similar a sample is to each category of anomalies.

These two scores (*IS* and *SS*) are then combined as a total weighted average (*TS*). The weight associated with each score is a hyper-parameter that can be tuned using cross-validation. Higher *TS* indicates a higher probability of being anomalous. A probability p is then computed for each unlabeled point using *TS* with formula explained in [ZLZ*18]. For user-labeled anomalous points, p is equal to 1.

In stage two, we first set p as the weights for each training case and then choose a multi-class classification model to classify all

unlabeled points into either one of the anomaly clusters or a normal class. Although the original paper chooses SVM as the training model, as we need to generate rules to describe each anomaly cluster, decision tree is selected for our purpose. Also, as explained earlier, the original algorithm automatically assigns a sample to its closest anomaly cluster when computing *SS*. However, to provide the user with more flexibility, we let the user make the choice of assigning a potential anomaly point to one of the anomaly clusters.

4. Description of RISSAD

In this section, we illustrate the interface and the workflow of our visual tool. Section 4.1 provides an description of each component offered by the tool. Section 4.2 illustrates the steps of how a user, with the help of the tool, can spot potential anomalies. Section 4.3 explains interactions and feedback with the rule panel (**C**).

4.1. Overview of the Components

Our visual tool (Fig. 1) constitutes 5 parts. Each part contributes a specific functionality to the overall workflow. **A** is the cluster view, it projects multi-dimensional data points onto a 2D scatter plot using *multidimensional scaling* [Mea92]. It provides a direct perspective on how points are overall distributed. **B** is the table view, it lays out the details of each data point corresponding to **A**. **C** is the rule panel that lists the rules learned from our interactions with the tool. The rules are descriptions of each anomaly cluster and the normal class. We can switch between each item by selecting the drop-down menu in the top right corner. This is explained in more detail in section 4.3. **D** is the barcode plot which offers a direct illustration of the distribution of points for each attribute. Each attribute of the original dataset contributes to one column, each thin line in the column represents a datapoint and they are ordered by the percentiles of that specific attribute from top to bottom. The top represents 0 percent while the bottom represents 100 percent. **E** is the violin plot view. The first violin plot marked in blue represents the distribution isolation scores of the entire dataset while the rest represent the distribution of the similarity scores for each anomaly cluster and they are computed based on the algorithm explained in section 3. Both isolation scores and similarity scores are normalized in the range of 0 to 1 using Min-Max Normalization. The y-axis is present at the left side indicating the exact scores with the bottom being 0 and the top being one.

4.2. Points Labeling

As illustrated in Section 3, the adopted algorithm requires two key parameters: isolation score (*IS*) and similarity score (*SS*) so our system are built revolving around them. In the beginning, all data points in scatter plot **A** are assumed to be in normal class and marked in blue. In the violin plot *E*, only the plot of the isolation scores exist at the beginning, as no anomalies are labeled and clustered at this point. When a user is interested in a point, he can place the mouse cursor over it without clicking it. This action will fire the following three events in three different components: (1) in table **B**, the corresponding row to the data point will be placed to the top and marked in light green. This illustrate the data point in detail. (2) In the barcode plot **D**, the thin lines in each column of each attribute corresponding to that specific data point are marked in red . This information illustrates the percentile of the sample point in each attribute. Another significant piece of information it provides is the overall distribution of the population for each attribute and how dense the neighboring points around that sample point are. Each line of the bar code plot is also bound with a mouse-over event. When activated, a tool tip at the bottom of the plot will indicate the percentile and attribute name associated with that line. (3) In the violin plot **E**, a black line will be placed on the plot to show the isolation score of the sample point. This is the most important event in our system, as the other two may rely on the user's prior knowledge to identify potential anomalies, but the violin plot can provide the user with an explicit value of the degree isolation. Higher scores lead to higher probability of being an anomaly. Based on the insight offered by those interactions, the user can then click all potential anomalous points and those points will be marked in red.

Once a user decides that a sufficient number of points have been labeled already, he can then enter the number of anomaly types based on the observation of the scatter plot or his prior knowledge. 3 is the default size. Then **E** will be updated with violin plots of similarity scores for each anomaly type. In the previous step, the user checks points mainly based on their isolation scores. In the next step, similarity scores can be utilized for selecting more anomalous points while everything else remain the same. The user also has the flexibility to assign a point to any of the anomaly type by selecting the options in the drop-down menu. Each anomaly types is in marked in a different color. As more points are assigned to each anomaly type, the violin plots of the similarity scores are adjusted accordingly, as more points are added to each cluster.

4.3. Rules Generation

After anomaly points are labeled as described in Section 4.2, the user can check the **Generate Rules** button on the top left corner of rule panel **C**. The rules are created using algorithms explained in Section 3 listed as illustrated in Figure 2 b. Each rule has a highlight button originally marked in green. Once the user clicks the button, the button will be switched to yellow and the following two events will be activated: (1) points that associated with the rule will be highlighted with black borders as shown in Figure 2 a. (2) lines associated with rules in the barcode plot **D** will be marked in yellow as illustrated in Figure 2 These two events can help the user to understand each rule and how they reflect on the original data and further improve the rules by selecting or unselecting any points.

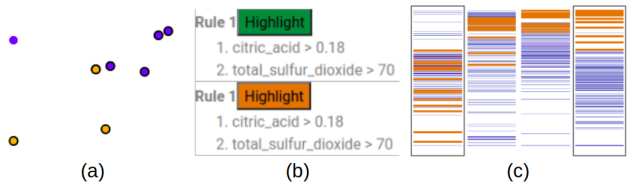

Figure 2: *An example interaction between Rule Panel (b), Scatter Plot (a) and Barcode Plot (c)*

5. Evaluation

The evaluation of our system consists of two parts: in Section 5.1, we run experiments to simulate user interactions with our systems based on different datasets. Performance of our proposed approach are measured against other algorithms. Then, in Section 5.2 we work through a case to illustrate how a user can solve actual real-world problem using our tool.

5.1. Simulation

In this simulation, our proposed algorithm (ADOA_Tree) is evaluated against 3 other algorithms: (1) Isolation Forest (IF), which represents an unsupervised approach (2) A regular Decision Tree (Naive_Tree) which represents a fully supervised learning approach and (3) the technique adopted (ADOA_SVM) by the original paper [ZLZ*18]. Rather than accuracy, we use *area under the curve* (AUC) as a performance metric, because the classifiers are probabilistic, AUC can be calculated based on a range of thresholds.

Since much of a user's work would be reviewing potential anomalies and considering the isolation and similarity scores, we focus our simulations on those two measures and we assume half of the labels provided by user interactions are based on isolation scores and the other half are based on the similarity scores. For both isolation and similarity scores, all points are ranked and the higher scoring points are labeled first. In each experiment we run , we create training and validation sets (70% vs. 30%). When training, we use further three-fold cross validation to tune the hyper-parameters. We have run 10 different experimental datasets and select 4 of them to demonstrate. Their results are illustrated in Figure 3. The x-axis represents the total number of labeled samples, and the y-axis represents the AUC score. Those experiments help us understand how performance changes as the number of labeled samples increases.

In most experiments, we find that our proposed approach outperforms Naive_Tree and IF and has similar performance with the ADOA_SVM. However, We observed that in some cases, commonly those with few labels, IF, which is unsupervised, performed better than all the supervised algorithms.

5.2. Case Study

Bob is a professional winemaker. His business has been facing fierce competition lately, in order to stay competitive, he decides to explore potential formulas for new wines. He collects wine dataset from the Kaggle website. The original dataset has 13 variables and 6497 cases, which is aimed for a classification problem with quality (0-10) as its target variable. Other variables are technical terms that describe different characters of wines such as alcohol level, ph level and density. As he is only interested in wines of a high quality, Bob selects cases with quality higher than or equal to 8. He then realizes that new formulas can be possibly discovered using anomaly analysis, as anomalies are often the ones associated with rare patterns. Even under the constraint that Bob is not proficient with statistics, our system offers a direct advantage of creating rules that are highly understandable and can be easily used as a guide for wine making.

Bob then works through the system as illustrated in Section 4.

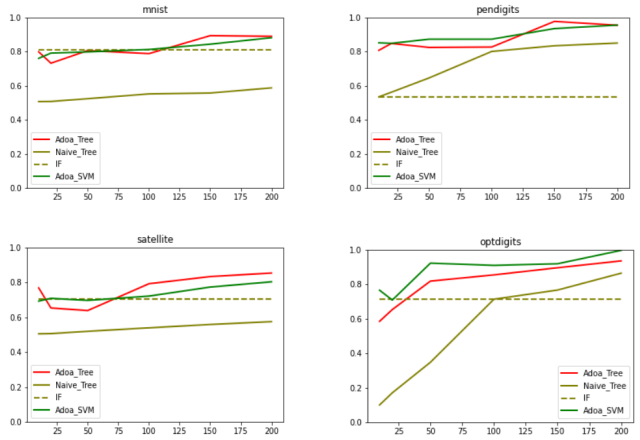

Figure 3: *Experiment Cases*

He begins with the scatterplot in Figure 1,**A** and notices several points that are clearly separated from the majority. The violin plots of *isolation* and*similarity* scores in Figure 1 **E** confirm the status of these points to be the likely outliers. To obtain optimal results, he only selects points with isolation scores over 0.8. This produces a set of three points shown in pink. He notices two of those points are significantly closer to each other, implying that there may be two groups of anomalies. He sets the *Cluster Size* to 2, and presses *Submit* to request a clustering. Specifically, he selects points with both isolation score and similarity score higher than 0.6. three points are selected during the step before rules are generated. Figure 1 shows the exact view after all points selected.

He then click the highlight buttons next to each rule to find other potential anomalous samples of each type. After checking with the table **B** and **A**, he finds that most of the highlighted points belong to red wine while the most highlighted points of the second anomaly cluster belong to white wine. This is an extremely valuable information as Bob realizes factors for making both red and white wines of exceptional characters.

6. Conclusion

In this paper, we present a visual tool RISSAD that aims to help users detect and understand anomalies. The tool provides users with visually effective functionalities to help user identify potential anomalous samples based on isolation and similarity and scores and label them. Most importantly, it creates clear and understandable rules through a semi-supervised approach. Through both quantitative and qualitative evaluations, we find that the tool can achieve its intended purpose without sacrificing much on accuracy.

References

[AY19] ARAKAWA R., YAKURA H.: Rescue: A framework for real-time feedback on behavioral cues using multimodal anomaly detection. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (2019), pp. 1–13. 2

[BKNS00] BREUNIG M. M., KRIEGEL H.-P., NG R. T., SANDER J.: Lof: identifying density-based local outliers. In *Proceedings of the 2000*

ACM SIGMOD international conference on Management of data (2000), pp. 93–104. 2

[BS03] BAY S. D., SCHWABACHER M.: Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (2003), pp. 29–38. 2

[CBK09] CHANDOLA V., BANERJEE A., KUMAR V.: Anomaly detection: A survey. *ACM computing surveys (CSUR) 41*, 3 (2009), 1–58. 2

[GMESK99] GUTTORMSSON S. E., MARKS R., EL-SHARKAWI M., KERSZENBAUM I.: Elliptical novelty grouping for on-line short-turn detection of excited running rotors. *IEEE Transactions on Energy Conversion 14*, 1 (1999), 16–22. 2

[HHWB02] HAWKINS S., HE H., WILLIAMS G., BAXTER R.: Outlier detection using replicator neural networks. In *International Conference on Data Warehousing and Knowledge Discovery* (2002), Springer, pp. 170–180. 2

[HLG14] HONG J., LIU C.-C., GOVINDARASU M.: Integrated anomaly detection for cyber security of the substations. *IEEE Transactions on Smart Grid 5*, 4 (2014), 1643–1653. 2

[HMYC18] HUANG D., MU D., YANG L., CAI X.: Codetect: financial fraud detection with anomaly feature detection. *IEEE Access 6* (2018), 19161–19174. 2

[KK17] KWAK S. K., KIM J. H.: Statistical data preparation: management of missing values and outliers. *Korean journal of anesthesiology 70*, 4 (2017), 407. 2

[Kru64] KRUSKAL J. B.: Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika 29*, 1 (1964), 1–27. 2

[LGG*17] LIN H., GAO S., GOTZ D., DU F., HE J., CAO N.: Rclens: Interactive rare category exploration and identification. *IEEE transactions on visualization and computer graphics 24*, 7 (2017), 2223–2237. 2

[LLYL02] LIU B., LEE W. S., YU P. S., LI X.: Partially supervised classification of text documents. In *ICML* (2002), vol. 2, pp. 387–394. 2

[LTZ08] LIU F. T., TING K. M., ZHOU Z.-H.: Isolation forest. In *2008 eighth ieee international conference on data mining* (2008), IEEE, pp. 413–422. 2

[MC03] MAHONEY M. V., CHAN P. K.: Learning rules for anomaly detection of hostile network traffic. In *Third IEEE International Conference on Data Mining* (2003), IEEE, pp. 601–604. 2

[Mea92] MEAD A.: Review of the development of multidimensional scaling methods. *Journal of the Royal Statistical Society: Series D (The Statistician) 41*, 1 (1992), 27–39. 3

[MLC07] MÜNZ G., LI S., CARLE G.: Traffic anomaly detection using k-means clustering. In *GI/ITG Workshop MMBnet* (2007), pp. 13–14. 2

[MXC*19] MU X., XU K., CHEN Q., DU F., WANG Y., QU H.: Moocad: Visual analysis of anomalous learning activities in massive open online courses. In *EuroVis (Short Papers)* (2019), pp. 91–95. 2

[SCSC03] SHYU M.-L., CHEN S.-C., SARINNAPAKORN K., CHANG L.: *A novel anomaly detection scheme based on principal component classifier*. Tech. rep., MIAMI UNIV CORAL GABLES FL DEPT OF ELECTRICAL AND COMPUTER ENGINEERING, 2003. 2

[SPBW12] SYARIF I., PRUGEL-BENNETT A., WILLS G.: Unsupervised clustering approach for network anomaly detection. In *International conference on networked digital technologies* (2012), Springer, pp. 135–145. 2

[SPST*01] SCHÖLKOPF B., PLATT J. C., SHAWE-TAYLOR J., SMOLA A. J., WILLIAMSON R. C.: Estimating the support of a high-dimensional distribution. *Neural computation 13*, 7 (2001), 1443–1471. 2

[SSW*17] SHU K., SLIVA A., WANG S., TANG J., LIU H.: Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter 19*, 1 (2017), 22–36. 2

[VC17] VANERIO J., CASAS P.: Ensemble-learning approaches for network security and anomaly detection. In *Proceedings of the Workshop on Big Data Analytics and Machine Learning for Data Communication Networks* (2017), pp. 1–6. 2

[WMCW03] WONG W.-K., MOORE A. W., COOPER G. F., WAGNER M. M.: Bayesian network anomaly pattern detection for disease outbreaks. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)* (2003), pp. 808–815. 2

[XXM*18] XU K., XIA M., MU X., WANG Y., CAO N.: Ensemblelens: Ensemble-based visual exploration of anomaly detection algorithms with multidimensional data. *IEEE transactions on visualization and computer graphics 25*, 1 (2018), 109–119. 2

[YTWM04] YAMANISHI K., TAKEUCHI J.-I., WILLIAMS G., MILNE P.: On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Mining and Knowledge Discovery 8*, 3 (2004), 275–300. 2

[ZCW*14] ZHAO J., CAO N., WEN Z., SONG Y., LIN Y.-R., COLLINS C.: # fluxflow: Visual analysis of anomalous information spreading on social media. *IEEE transactions on visualization and computer graphics 20*, 12 (2014), 1773–1782. 2

[ZDH*17] ZHANG X., DOU W., HE Q., ZHOU R., LECKIE C., KOTAGIRI R., SALCIC Z.: Lshiforest: A generic framework for fast tree isolation based ensemble anomaly analysis. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)* (2017), IEEE, pp. 983–994. 2

[ZLZ*18] ZHANG Y.-L., LI L., ZHOU J., LI X., ZHOU Z.-H.: Anomaly detection with partially observed anomalies. In *Companion Proceedings of the The Web Conference 2018* (2018), pp. 639–646. 2, 4