RCLens: Interactive Rare Category Exploration and Identification

Hanfei Lin, Siyuan Gao, David Gotz, Fan Du, Jingrui He, and Nan Cao

Abstract—Rare category identification is an important task in many application domains, ranging from network security, to financial fraud detection, to personalized medicine. These are all applications which require the discovery and characterization of sets of rare but structurally-similar data entities which are obscured within a larger but structurally different dataset. This paper introduces RCLens, a visual analytics system designed to support user-guided rare category exploration and identification. RCLens adopts a novel active learning-based algorithm to iteratively identify more accurate rare categories in response to user-provided feedback. The algorithm is tightly integrated with an interactive visualization-based interface which supports a novel and effective workflow for rare category identification. This paper (1) defines RCLens' underlying active-learning algorithm; (2) describes the visualization and interaction designs, including a discussion of how the designs support user-guided rare category identification; and (3) presents results from an evaluation demonstrating RCLens' ability to support the rare category identification process.

Index Terms—Visual analytics, information visualization, rare category detection, machine learning

1 Introduction

A variety of algorithms have been developed to identify predominant trends or classes of objects within complex datasets, including regression, classification, and clustering. These techniques aim to characterize the large-scale structures within a dataset that have high levels of support. This approach has a widespread use for prediction and other machine learning tasks. Conversely, outlier detection techniques are designed to identify individual data entities which are most distant from other entities in a dataset. Algorithms in this family seek to identify individual outliers (sometimes called anomalies) that represent exceptions to the overall structure of a dataset. This has applications, for example, in the security domain, where unusual data can be indicative of suspicious activities.

In between these extremes—between the identification of predominant patterns and individual outliers—is the task of rare category (RC) detection. Methods that robustly identify and accurately characterize rare categories can have widespread impact in fields as varied as network security, financial fraud detection, social network analysis, and personalized medicine. All of these use cases share the need to discover and characterize relatively small, underrepresented sets of rare but structurally similar data entities which are obscured within a larger but structurally different dataset. These rare categories could represent, for example, computer network intrusions, fraudulent financial transactions, artificial social network bots, or patients with rare diseases that require specialized treatments.

While widely useful, rare category detection is a technically challenging task that is currently an active research topic within the machine learning community (e.g., [1], [2]). One factor making this task so difficult is the lack of known categories in most real-world applications. Rather than seeking to identify additional occurrences of a known class, these systems are tasked with concurrently (1) identifying individual rare data entities (data entities which are neither "typical" nor an "outlier"), and (2) defining categories which link together rare data entities into groups with shared sets of characteristics. To address this challenge, an *active learning* approach is often adopted [3], [4], [5]. Active learning incorporates user input—often provided in the form of exemplary labels—to improve automated classification results. However, exploring unlabeled rare data entities, providing user-generated labels, and evaluating the response of the updated learning algorithm, are extremely challenging user tasks.

This paper introduces RCLens, a system designed specifically for user-guided rare category exploration and identification. RCLens adopts a novel active learning-based algorithm to iteratively identify more accurate rare categories. RCLens provides a unique visualization-based interactive workflow for sifting and revising candidate categories identified by the algorithm, with the user's interaction feeding back into the algorithm to produce improved labels.

In particular, the primary contributions for the work described in this paper include:

- *Active Learning Rare Category Identification Algorithm.* We define a new iterative active learning algorithm

- *H. Lin, S. Gao, and N. Cao are with iDV^x Lab, Tongji University, Shanghai Shi 200092, China. E-mail: hanfeil2@illinois.edu, siyuan.gao@yale.edu, nan.cao@gmail.com.*
- *D. Gotz is with the University of North Carolina at Chapel Hill, Chapel Hill, NC 27599. E-mail: gotz@unc.edu.*
- *F. Du is with the University of Maryland, College Park, MD 20742. E-mail: fan@cs.umd.edu.*
- *J. He is with Arizona State University, Tempe, AZ 85287. E-mail: jingrui.he@asu.edu.*

Manuscript received 27 July 2016; revised 11 May 2017; accepted 19 May 2017. Date of publication 6 June 2017; date of current version 25 May 2018. (Corresponding author: Nan Cao.)
Recommended for acceptance by B. Lee.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2017.2711030

for rare category identification. The algorithm intelligently suggests sets of rare data entities as new potential categories by distinguishing between both typical data entities and true outliers. The algorithm incorporates user feedback, as expressed via user interaction with the interface, to improve the identified rare data categories over time.

- *Visualization and Interaction Design for Rare Category Exploration and Refinement.* We propose novel visualization and interaction designs developed to meet both user and algorithmic requirements. The integrated design provides a highly usable visual environment for iterative rare category discovery and refinement.
- *System Implementation and Evaluation.* A prototype implementation of our approach is demonstrated in an accompanying video figure. The prototype has been evaluated with results highlighting RCLens' ability to support user-driven rare category identification.

The remainder of this paper describes these contributions in more detail. After a review of related work, the paper begins with a description of the rare category detection algorithm and the interface designs. The prototype system implementation is then described, along with the methods used for evaluation, and the evaluation results.

2 Related Work

In this section, we review techniques that are most relevant to our work in aspects of both analysis and visualization. In particular, we review the active learning-based analysis methods and the related visual analysis systems.

Compared to the traditional supervised learning techniques that passively train a model based on a set of prelabeled training instances, active learning can achieve a better accuracy with significantly fewer labeled training instances by actively querying oracles for labeling. In history, there were three major strategies for querying [6]. In the early stage, "membership query synthesis" was frequently used. Algorithms following this strategy [7], [8], [9] may ask an oracle to label any arbitrary instances without a specific target. These techniques are limited by their performances. Later, the "stream-based selective sampling" was developed to help solve this issue. These algorithms [10], [11], [12] scan each unlabeled instance and pick a small set based on different criteria for labeling. Recently, research has focused on the "pool-based" approach, which selects samples from a pool of unlabeled data instances based on certain measurements in a greedy fashion. When compared with the "stream-based" method, this approach is more efficient and flexible, which is why it is commonly used and furthermore adopted in our algorithm.

The proposed LOF-based Rare Category Detection (LOFRCD) algorithm, as an active learning-based rare category detection method, is related to, but different from those active learning based classification techniques, which have been studied and used either in general purpose [13] or specific applications such as text classification [14], [15], [16], image classification [17], speech recognition [18], and cancer diagnosis [19]. First, their goals are different. The goal of a classification algorithm is to separate data instances into categories; It terminates when all of the instances have been labeled. In comparison, the goal of a rare category detection algorithm is to separate rare categories from the majority class. It stops when no more rare categories could be found by the algorithm instead of all the data have been labeled. Second, rare category detection algorithms are designed to identify categories from highly skewed data as rare categories are usually very small, which can be difficult to handle with classification algorithms.

Our algorithm is also related to outlier detection (i.e., anomaly detection), which has been extensively studied for decades [20], [21] with the goal of identifying isolated instances in the data. These algorithms cannot be used for identifying rare categories which are usually isolated groups containing a set of similar data instances that are different from the majority. To address this issue, rare category detection algorithms were developed. For example, both Wu et al. [22] and Vatturi et al. [23] introduced the cluster based methods, in which the most compact and isolated clusters are treated as rare categories. These methods are inefficient for dealing with cases with ambiguities. This issue can be addressed by active learning. Pelleg et al. [5] introduced the first active learning framework for detecting rare categories. Later, their work was extended by the a series of density-based algorithms [2], [3], [24], [25], which detect rare categories by identifying the dramatic change of data densities based on the k-nearest neighbor search or reversed k-nearest neighbor search in either the feature space or a graph. Here, k is a parameter from prior knowledge. In comparison, our method also follows the k-nearest neighbor-based approach, but it is able to determine the best k automatically based on the local outlier factor [26]. Huang et al. [27] also introduced a method to adaptively select k values, however, their algorithm assumes a seed from the target category to start with, which is usually difficult to provide. Liu et al. [28] introduced two methods, iFRED and vFRED, for detecting rare categories based on wavelet transformation without a requirement of any prior knowledge. However, this algorithm has difficulty with high dimensional data, which is the major goal of the RCLens system.

To the best of our knowledge, RCLens is the first system that employs visualization techniques to support rare category detection based on active learning. However, there are several visual analysis systems that are designed for supporting outlier detection and classification, which are related to our work. For example, recently Zhao et al. introduced FluxFlow [29], which employs advanced analysis methods to detect anomalous information spreading patterns and intuitive visualization designs to help with the pattern summarization and interpretation. Cao et al. introduced the TargetVue system [30], which detects users with anomalous behaviors based on the local outlier factor and intuitive designs of behavior glyphs. However, none of these systems use active learning based-approach and none of them are developed for rare category detection. Visual analysis systems have also been developed for supporting active learning based classification [31], [32], though, the goal of these systems is to classify instances into categories instead of identifying rare categories.

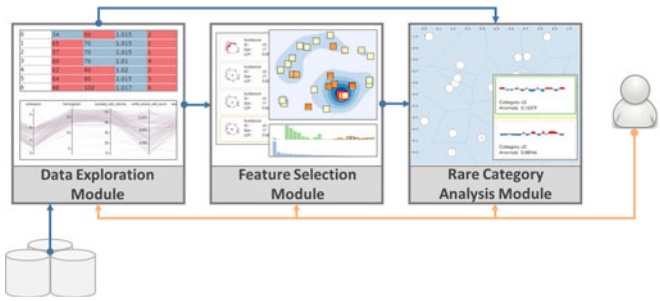


Fig. 1. RCLens system consists of three modules.

3 System Design and Overview

The RCLens system was designed to meet several real-world requirements for rare category identification. The development was guided by a close collaboration with two domain experts who specialized in anomaly detection and rare category analysis. Over the course of approximately six months, regular meetings were held in which detailed system design requirements were discussed and prototype systems were demonstrated to the experts for the purposes of gathering feedback. Improvements were made iteratively throughout the process. Below we describe the most critical design requirements (*R1-R3*) that were developed during these discussions, and which motivate our design.

R1 *Easy exploration of the raw data.* Easy access to the raw data and the ability to explore that data was the first requirement raised by both experts. This was due to its importance for providing evidence to support or refute the analysis results. The ability to filter data based on users' interests was also identified as a key aspect of exploration, helping users improve the analysis procedure and improve the analysis quality. Therefore, it was determined that the proposed system must provide interactive user interfaces designed to enable efficient exploration and selection of the raw underlying data. Moreover, the interfaces must support the identification of interesting subsets from the overall set of raw data.

R2 *Effective subspace investigation.* Detecting rare categories, or even outliers, from high dimensional data is a well-known challenge. The experts confirmed this point of view, and suggested that to overcome this issue, one must focus on a narrower subspace of the dataset using a subset of preselected features. From this feedback, it was determined that the system should facilitate efficient feature selection capabilities. This would help users define promising subspaces in which to investigate rare categories, assist in the reduction of noise in the data, and help uncover hidden relational patterns among different data instances.

R3 *Keep human in the analysis loop.* The lack of ground truth labels is a fundamental challenge for rare category detection. For this reason, human-judgments play a crucial role in the identification of anomalies and rare categories. Therefore, it is critically important for the system to allow analysts to supervise the rare category analysis procedure. This supervision process utilizes the users' domain knowledge and experiences. This requires an integrated design of algorithm, visualization, and interaction which helps users make correct judgments about the data and provide effective feedback to the underlying algorithms.

Based on these requirements, we developed RCLens, a visual analysis system for interactive rare category analysis and labeling. The system employs novel visualization designs as well as active learning-based analysis approaches to help identify and reveal rare categories. The system takes multidimensional data as input and processes that data to detect rare categories via three major modules (Fig. 1): (1) the *data exploration module*, (2) the *feature selection module*, and (3) the *rare category analysis module*. The *data exploration module* represents raw data via an interactive visualization to assist with data querying and filtering (*R1*). The selected data are then sent to the next module in the pipeline. Next, the *feature selection module*, analyzes the variances and the correlations of the data dimensions. Results from the analysis are visualized to provide visual cues which guide the user's feature selection and subspace investigation process (*R2*). Finally, the third *rare category analysis module performs analysis on the selected data, using the feature space constructed with the feature selection module. The putative rare categories are visualized and refined one by one through an iterative active learning procedure in which representative data instances and their k-nearest neighborhood are shown to users for labeling.* This approach satisfies the design requirement to involve human in the analysis loop (*R3*).

Use Case Scenario. The following scenario demonstrates how the proposed system can be used to support a rare category detection task. Consider a hypothetical system administrator named Bob who helps manage a large cloud computing resource (e.g., such as Amazon EC2). Bob's primary job is to detect intrusions based on server logs. Given the lack of ground truth data about intrusions within log files, Bob has to manually identify and label instances of log activity that correspond to attacks. These labels are then used for training automatic anomaly detection algorithms.

This is a tedious and time-consuming job, and difficult for users to perform reliably using a manual process. Bob would therefore like to use RCLens to help with the labeling task. To do that, he first extracts a set of features from the log instance and loads the transformed data into the RCLens system. After querying the log data using the data explorer, Bob selects a subset of log instances that have records of connection-requests issued using the TCP protocol. In the feature explorer, Bob further selects sets of highly independent features with large variances to help differentiate data instances. Based on these settings, the selected data are analyzed within the specified subspace. The rare category algorithm identifies the first instance that has the highest confidence to represent a rare category, which is then visualized together with its neighborhood as a candidate rare category in the category explorer for Bob to examine. Bob investigates this cluster and notices a few log instances which are less homogeneous than others in the same cluster. This appears visually as instances located a large distance from the categories center. Bob interactively removes those instances from the cluster displayed in the category explorer, then labels the remaining instances with a common label and feedback the result to the system.

The system checks these labeled instances and further expands the scope to involve more instances in the cluster. This process creates the first rare category which is shown in the category list. The system then iteratively produces additional recommended rare categories in the next iterations for Bob to visually inspect, refine, and label. On each iteration, Bob's refinement and labeling activity are integrated into the ongoing category analysis procedure.

The iterative process stops once all the suspicious categories have been labeled, i.e., no more rare categories can be found. Bob then explores the labeling results, comparing the identified categories. Bob merges similar categories together to form less granular labels, then exports the final results for future use (e.g., training advanced prediction models to detect future occurrences of similar intrusions).

4 Visual Explorers

The user interface for RCLens consists of three visualization-based components: (1) the data explorer, (2) the feature explorer, and (3) the category explorer. These visual explorers work together in a coordinated fashion to support the human-in-the-loop rare category labeling process described earlier in this paper. The data explorer first utilizes a query box to support rough data selection in the database, and also utilizes a familiar interactive parallel coordinates visualization and tabular heatmaps to help users further select a subset of data for analysis. These visualizations act as visual query controls, which are traditional visualization designs, that should be familiar to many users. In contrast, both the feature explorer and category explorer use more novel visual representations and support less familiar workflows. The remainder of this section describes these two visual explorers in more detail.

4.1 The Feature Explorer

The feature explorer is designed to help with interactive feature selection and subspace formulation. When compared with those automatical feature selection algorithms such as [35], [36], our design tends to provide a more flexible mechanism to help users select features in rich visual context. A desired subspace can be defined using a set of independent features which best characterize and separate the data instances. The design of the feature explorer employs three coordinated views to help with this task: a feature distribution view, variance bar charts, and the subspace list.

4.1.1 Feature Distribution View

This view provides an overview of the features in a dataset and the interactions between them. It assists with the identification of independent features, as well as the redundant feature elimination. This helps users define subspaces that are most promising for distinguishing data points in rare categories from other data instances. The feature glyphs, as well as their distribution shown in the view, are calculated based on the dataset selected from the data explorer. It remains unchanged until a new set of data is selected from the data explorer for analysis.

In the feature distribution view, multidimensional scaling is used to position glyphs—one for each feature—according to the pairwise feature correlations calculated based on the focal dataset under analysis (Fig. 2, Panel 3). This produces a view in which correlated features are clustered together while independent ones are positioned apart. To illustrate the distribution of data values with each feature, a pixel-based heat map is rendered within each feature's glyph (Fig. 3a). Within the heat map, each pixel indicates the feature value of an instance in the focal data. The colors of the pixels indicate feature values ranging from yellow (minimum value) to orange (maximum value). In this way, a feature heat map with mottled color indicates the variance of the corresponding feature is higher than that of a feature whose heat map in a uniform color. Pixels are reordered according to the similarities of the corresponding data instances and arranged into a line-by-line serpentine shape to preserve the order. The similarity of the data instances is determined via the following stress model:

$$min \sum_{i,j} \omega_{ij} ||x_i - x_j||^2,$$

where x_i and x_j indicate the coordinates of the ith and jth data instances in an N-dimensional visualization space. ω_{ij} indicates the similarity between instances i and j in the feature space. An optimal solution that minimizes the stress will place the data instances together if they have a large similarity. Intuitively, this model tries to minimize the distances between the instances with similar feature values. In this way, the data instances can be ordered based on similarities in 1D space, i.e., $N = 1$.

4.1.2 Variance Bar Chart Views

Color variation in the glyphs used in the feature distribution view provides some indicators of feature variance. This is complemented by a bar chart view of the feature variances. Each feature is represented by a bar, with the height encoding the feature's variance over the selected dataset. The bar chart is located beneath the feature distribution view (Fig. 2, Panel 4), and the two visualizations are linked such that clicking a bar will highlight the corresponding feature in the distribution view, and vice versa.

A second variance bar chart is included to help with the cases where individual features are highly correlated and difficult to separate. In these cases, our system employs principle component analysis (PCA), which computes orthogonal components that can be used in place of the dataset's raw features when defining a subspace for analysis. To help users determine the number of components to use in this process, the second bar chart illustrates the variance ratio of each component, that is, the percentage of the data variances preserved by each principle component calculated by PCA (Fig. 2, Panel 5). Users can select one or more components by brushing the bars in the bar chart to formulate a feature subspace. Here, each component computed by PCA is a combination of all of the data features, which can make the features difficult to interpret. To facilitate the interpretation, we reuse the above variance bar chart to help illustrate how the PCA component is formulated as a combination of different portions of the original features. In particular, when the mouse hovers over a bar in the component bar chart, the variance bar chart (the first bar

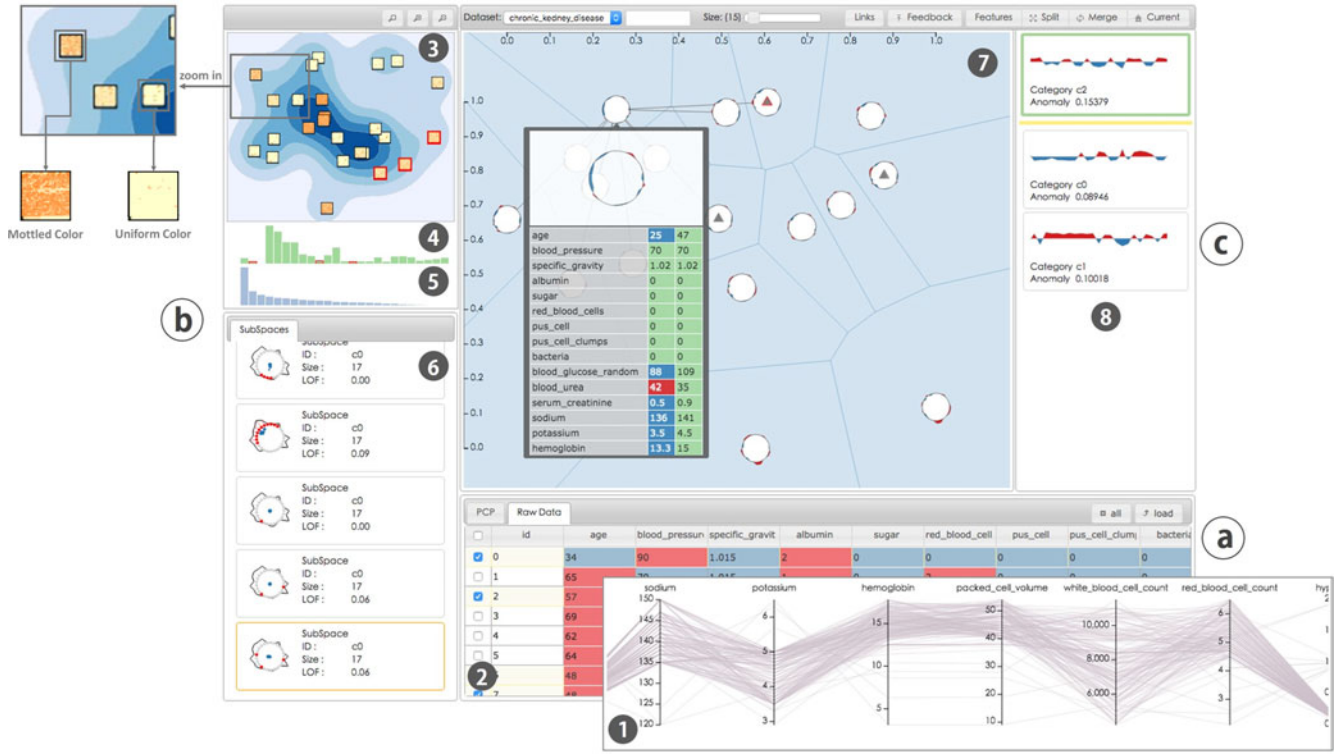


Fig. 2. The users interface of RCLensconsists of three major explorers: (a) the *data explorer*, (b) the *feature explorer*, and (c) the *category explorer*. Each explorer is a combination of multiple views. The *data explorer* consists of two views showing the raw data in a data table and in a parallel coordinates [33] (1,2), respectively. The *feature explorer* illustrates the distribution of features in a MDS view (3), the feature variance (4) and the variance ratios of the components computed based on principle component analysis(PCA) [34] (5), as well as the subspaces formulated by users (6). Finally, the *category explorer* consists of a Voronoi diagram illustrating a focused category (7), and a category list (8) showing the categories that are already labeled by users. A consistent color scheme is used across explorers, with dark red/blue indicating feature values that are above/below average, respectively. The black circles with numbers are used to highlight individual visualization panels throughout Section 4.

chart mentioned above) is updated such that the height of each bar indicates the portion of the corresponding feature in the focused component. The heights are reset to show the original variance measures when the mouse is removed from the bar it was hovering over. When a dimension subspace is generated by brushing the component bar chart, the system would then reduce the dimension of features and detect the corresponding categories based on this reduction result.

4.1.3 Subspace List

The subspaces produced either by selecting high variance-independent raw features or by selecting high variance-preserving components are visualized in a subspace glyph (Fig. 3b). Users can create multiple subspaces by selecting different sets of raw features or principle components. All defined subspaces are shown in the subspace list view (Fig. 2, Panel 6). Clicking on a tag in the list will select a subspace in which to execute the rare category analysis algorithm. By clicking on different tags, users can switch between different analysis results for comparison.

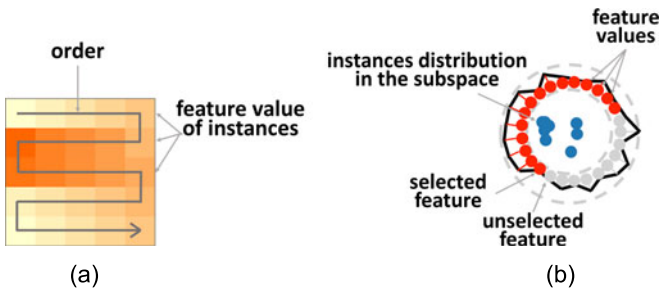


Fig. 3. Glyph design for the Feature Explorer view. (a) A feature node based on heap map with serpentine arrangement is used in the feature distribution view. (b) A subspace glyph showing the selected features is used in the subspace list view.

The design of the subspace glyph uses an extended version of star glyph, in which the features are radially arranged. When the subspace is formulated based on a set of raw features, the corresponding feature axises are highlighted in red as shown in Fig. 3b. When the subspace is formulated by principle components, the opacity of the feature axes are used to encode the mean feature portions in the selected components.The feature locations in each radial glyph across different subspaces are the same for users to better compare selected features. Inside the glyph, we generate a thumbnail to illustrate the data distribution in the subspace based on barycenter coordinates and the features or components that formulate the subspace.

4.2 The Category Explorer

The category explorer is the most critical component in the RCLensinterface. It is designed to explore, revise, and label rare categories based on the analysis results produced given the input multidimensional data and the subspace selected using the feature explorer. The interface is coupled with a novel active learning algorithm which iteratively detects rare categories, one category at a time. A Voronoi diagram is

used to represent each of the identified categories in which users can further refine the analysis results. The labeled categories identified by the user are shown in a list in which similar categories can be found and further merged.

4.2.1 Rare Category Detection

We propose a human-in-the-loop active learning-based approach to identify rare categories. Active learning techniques use an "oracle" to provide labels during the learning process to identify representative samples or resolve ambiguous data instances during the classification procedure. The goal is to improve the learning procedure over time to minimizing the oracle's efforts. The oracle can be a human user or any other source of trustworthy labels. Our algorithm design is inspired by Nearest-Neighbor-Based Rare Category Detection for Multiple Classes (NNDM) [3], the state-of-the-art active learning algorithm developed for rare category detection. NNDM identifies rare categories by estimating the density of the k-nearest neighborhood of each data instance. High-density data instances are subsequently selected for labeling. Making an appropriate selection for the value of the parameter k is essential for correctly detecting rare categories. This k is computed as np_i in NNDM, where n is the total number of data instances and p_i is prior knowledge that describes the proportion of the ith rare category in the dataset. However, in real applications, the number of rare categories as well as their proportions (p_i) within a dataset are usually unknown. This limits NNDM's application. To overcome this limitation, we introduce an adaptive algorithm based on the local outlier factor (LOF) algorithm [26] for rare category detection. More specifically, our new LOFRCD algorithm assumes no prior knowledge about the rare categories. Instead, the proposed method first evaluates if a given data point is an anomaly through the application of a modified version of LOF, then further refines the results to identify categories by considering different neighborhood conditions.

The Local Outlier Factor (LOF) Algorithm. LOF is one of the most widely-used outlier detection algorithms. It determines if an instance a is an outlier by comparing the k-neighborhood density of a to the k-neighborhood density of a's k-neighbors. This is formally defined as

$$LOF_k(a) = \frac{\sum_{b \in NN_k(a)} \frac{lrd_k(b)}{lrd_k(a)}}{k},$$

where $lrd_k(a)$ is a's local reachability density, which is defined as

$$lrd_k(t) = \left(\frac{\sum_{s \in NN_t(k)} dist_k(t,s)}{k} \right)^{-1},$$

and $dist_k(a,b) = \max(d_k(b), d(a,b))$ indicates the reachability distance between a and b, i.e., the euclidean distance between a and b, but no smaller than b's k-distance ($d_k(b)$). Note that although the parameter k is crucial to the analysis results, it is arbitrarily determined or selected based on a user's experience and intuition regarding a given dataset.

Assuming a proper k has been selected, one would intuitively suspect that a data instance's LOF score is close to 1 when it is considered similar to its neighbors (i.e., has similar lrd_k values). When the LOF score is smaller than 1, the data instance falls into a dense area of the dataset whose local density is larger than its neighbors, indicating a normal situation. Outliers, in contrast, are detected when the corresponding LOF score is larger than 1, which indicates the instance is isolated from its neighbors.

LOF determines outliers. However, outliers are not necessarily members of a rare category. Similarly, "inliers" are not necessary poor candidates for inclusion within a rare category. Therefore, the LOF approach is insufficient for rare category detection. To bridge the gap between outliers and rare categories, we propose an active learning approach which builds on core LOF concepts.

The LOF-based Rare Category Detection (LOFRCD) Algorithm. The LOFRCD algorithm detects rare categories by gradually enlarging an instance's k-neighborhood (i.e., increasing k) and examining the trend of the corresponding LOF scores. The algorithm is performed iteratively. In each iteration, a single instance with the highest confidence to represent a rare category is identified for a human oracle to label and adjust. To help the oracle make a better judgment, the algorithm puts the instance into its neighbor context by formulating a cluster based on the instance and its k-neighborhood. The whole cluster is shown to the oracle for labeling. Once it has been labeled as rare, the cluster would be fed back into the algorithm, which further expands the scope of the cluster to enclose more related but unlabeled instances. Once the procedure is completed, the data for the labeled clusters are removed for further consideration, and the remaining data are used for the next round of analysis. The algorithm terminates when no more undiscovered rare categories exist.

Design Rationale. The design of LOFRCD is based on observing the changing trend of an instance's local outlier factor as a function of the change in its neighborhood size. To better describe this finding we use instance a shown in Fig. 4a as an example. In particular, a belongs to a rare category C, which consists of a group of $|C|$ related instances that are isolated from the remaining parts of the data. We found a's LOF score, $LOF_k(a)$, depends heavily on the size of its k-neighborhood (i.e., the value of k). In particular, as shown in Fig. 4b, $LOF_k(a)$ decreases as k increases, as long as $k < |C|$. However, it increases along with k when $k > |C|$. Therefore, $k = |C|$, denoted as k_{inf}, is the inflection point within a's LOF-curve. When referring to Fig. 4a, we found it also reflects the boundary of C.

Therefore, based on the above observation, we can easily and precisely detect the boundary of a rare category isolated from other data instances and centered at an instance a based on a's LOF-curve and the corresponding k_{inf} at the inflection point, once a is correctly identified. Therefore, the major goal of the LOFRCD algorithm is the identification of a data instance that has high confidence for representing the center of a new category that has not yet been identified.

Rare Category Identification. We estimate the confidence of each instance in terms of representing an unknown category from the following four different criteria. Instances with high confidence scores are chosen as the center of a potential rare category for the oracle to label. In particular, a representative instance, a of an unknown category should:

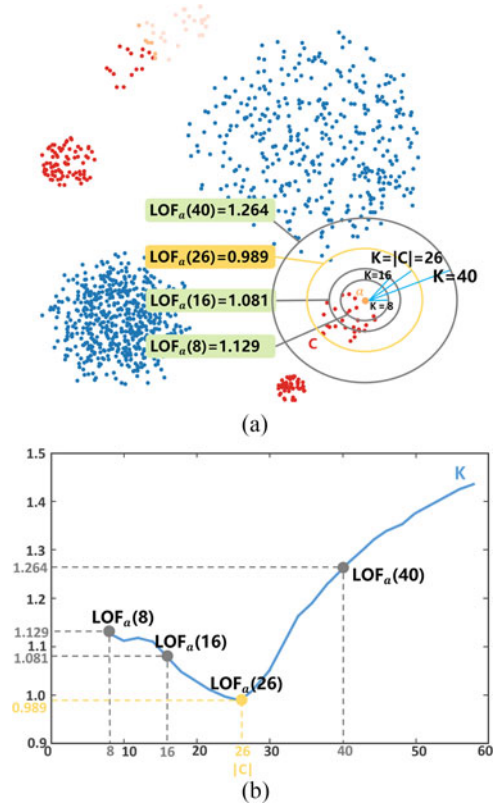


Fig. 4. (a) A sample dataset used for illustrating the LOFRCD algorithm design. (b) The LOF trend curve for a single instance as its k-neighborhood is enlarged.

- Represent an isolated minority class, and all of its neighbors should represent the same class. We estimate the isolation of a minority class via the following metric:

$$\mathscr{C}_1(a) = \frac{LOF_{k_{inf+1}}(a)}{LOF_{k_{inf}}(a)},$$

where k_{inf} represent the k value corresponding to the reflective point on a's LOF-curve, and k_{inf+1} represents the k value next to k_{inf} on the LOF-curve. A larger $\mathscr{C}_1(a)$ will indicate a more isolated underlying class. To ensure all a's neighbors (i.e., $b \in NN_{k_{inf}}(a)$) also represent the class, i.e., also have a large $\mathscr{C}_1$ score, the following metric is used:

$$\mathscr{C}_2(a) = \frac{\sum_{b \in NN_{k_{inf}}(a)} \mathscr{C}_1(b)}{k_{inf}},$$

which estimates the averaged $\mathscr{C}_1$ score of instances in the class centered at a, within the scope of a's k_{inf}-neighborhood (i.e., $NN_{k_{inf}}(a)$).
- Be close to all its neighbors within the category, and distant from neighbors outside the category. We define the following metric to help estimate this criterion:

$$\mathscr{C}_3(a) = \frac{d_{k_{inf+1}}(a)}{d_{k_{avg}}(a)},$$

where $k_{avg} = \frac{k_{inf}+k_{min}}{2}$ and k_{min} is the minimum k that has ever been tested for producing the LOF-curve. $d_k(a)$ indicates a's k distance (i.e., the distance between a and its kth nearest neighbor). In this way, a larger $\mathscr{C}_3(a)$ means that a is closer to its neighbors.
- Have a k_{inf} value that is similar to its k_{inf}-neighbors' k_{inf} values, which can be formally estimated by

$$\mathscr{C}_4(a) = exp - || \frac{k^a_{inf}}{\frac{\sum_{b \in NN_{k^a_{inf}}(a)}(k^b_{inf})}{k^a_{inf}}} - 1||.$$

Using these measures, the overall confidence for an instance, a, representing a new category can be determined by[1]:

$$\mathscr{C}(a) = \sqrt[4]{\mathscr{C}_1(a) \cdot \mathscr{C}_2(a) \cdot \mathscr{C}_3(a) \cdot \mathscr{C}_4(a)}.$$

In our implementation, only those data points which tend to be outliers (i.e., data instances whose LOF scores are larger than 1) are considered as potential category centers. This further ensures they represent rare categories instead of majority classes.

Building upon the $\mathscr{C}(a)$ metric, we define the LOFRCD algorithm for rare category detection as summarized in Algorithm 1. The algorithm first initializes a K-list (line 2) which contains all the k values to be tested for detecting the inflection point on the LOF-curve. After that, the algorithm calculates the neighborhood matrix M, where $M[i,j]$ indicates the jth nearest neighbor of the ith instance in the data and $M[i,:j]$ gives i's j-nearest neighbors. It is used for accelerating the calculation of LOF.

Algorithm 1. LOF Based Rare Category Detection

Data: U
Result: Rare Categories
1: **begin**
2: $K = \{2, 4, \ldots, 2^{log(|U|)}\}$
3: Calculating the $|U| \times |U|$ neighborhood matrix M
4: $U_{res} \leftarrow = \phi$
5: **while** *true* **do**
6: $U_r \leftarrow = \phi$
7: **for** $u \in U$ **do**
8: **if** $\exists k \in K$ *s.t.* $LOF_a(M[u,:k]) > 1$ **then**
9: $U_r = a \cup U_r$
10: **if** $U_r \neq \phi$ **then**
11: $a \leftarrow max_{u \in U_r} \mathscr{C}(u)$
12: $C \leftarrow \{a\} \cup NN_a(k_{inf})$
13: $C' \leftarrow VL(C)$ /*visualize C for labeling*/
14: $C' \leftarrow expand(C')$
15: $U \leftarrow U - C'$
16: $U_{res} \leftarrow C' \cup U_{res}$
17: $K \leftarrow refine(K, C')$
18: **else**
19: terminate
20: return U_{res}

In each iteration, the algorithm first detects those outliers by checking the full range of k values (line 7-9). After that, it

1. All the scores have been normalized into the range of $[0, 1]$ before multiplying them together. Experiments were conducted to verify the effectiveness of these metrics that are reported in the supplemental material: http://nancao.org/supp/rclens.pdf.

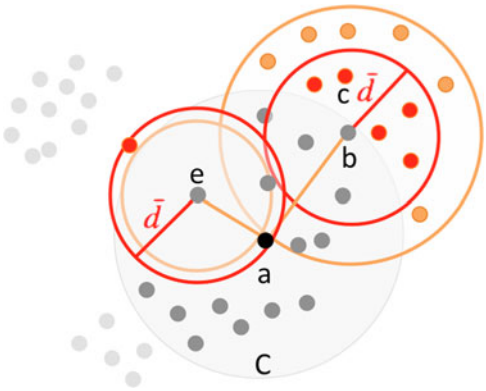


Fig. 5. The category expansion process grows the boundary of a category C to enclose as many closely related instances (the instances shown in red) as possible.

calculates the confidence score for each outlier and ranks out the instance with the highest confidence for labeling (line 11). This instance together with its k_{inf}-neighborhood formulates a class C, which is visualized and then labeled by the oracle. Here, we visualize the focal instance in its k_{inf}-neighborhood to show more context about the data distribution to help the oracle make a better decision (line 12, 13). The algorithm further expands the category based on the oracle's labeling results (i.e., C') to enclose as many related instances as possible, which helps reduce the total number of instances that need to be manually labeled (line 14).The labeled nodes are removed and the remaining data are used for the next round of analysis (line 15, 16) and will never be shown to the oracle again. The granularity of the K-list is later refined at the end of the iteration (line 17, Algorithm 2).

Algorithm 2. Refine()

Data: K, C
Result: Rare Categories
1: **begin**
2: $k = argmin_{k \in K}(k - |C|)^2$
3: **while** $|k - |C|| > \varepsilon$ **do**
4: $k = (k + |C|)/2$
5: K.insert(k)
6: return K

Finally, the algorithm terminates when it estimates that no more undiscovered rare categories exist, i.e., there are no more isolated instances can be detected ($U_{res} = \phi$) at all scopes determined by K.

Category Expansion. The category expansion algorithm ($expand(\cdot)$) is designed to extend the boundary of a category C in the feature space along the observed data distribution, with the aim of enclosing as many related instances within the current category as possible. This process is particularly useful when the underlying rare category is non-convex as shown in Fig. 5.

In particular, as shown in Fig. 5, for an instance b in the rare category C, the algorithm checks each instance c in b's k_b-neighborhood (i.e., $c \in NN_{k_b}(b)$, the orange circle centered at b) and adds c into the category C if it is close enough. Here, k_b indicates b is the k_bth nearest neighbor of C's center (denoted as a). We say the instance c is close enough to the category C, iff $d(b, c) < \bar{d}$ (the instances in the red circle centered at b in Fig. 5), where $d(b, c)$ indicates the euclidean distance between b and c and $\bar{d}$ is the averaged distance between the class center and all its neighbors in C. This algorithm takes the local data density into consideration. It ensures only the nearby instances following a similar local density can be added into C during the expansion. For example, as shown in Fig. 5, the original category C is non-convex, the expanding algorithm tends to expand and enclose more data instances along the direction where the data are more densely distributed, i.e., it tends to expand along the direction of $a- > b$ instead of $a- > e$. In addition, the expanding constrained by $\bar{d}$ (i.e., the red circles), which ensures the algorithm finds a tight boundary of the category.

The above algorithm expands the boundary of a detected category which increases the recall, however without user verification, the precision might be decreased, i.e., a trade-off between precision and recall always exists. Therefore, we made the expansion as an optional function of the algorithm which can be turned off by users when necessary.

Searching Range Refinement. The performance of the above algorithm highly depends on the input parameter D, which determines the range of k values to test when producing the LOF-curve of each data element. In our implementation, we initially set $K = \{2, 4, \ldots, 2^{log(|U|)}\}$. K is then gradually refined based on the oracle's feedback while labeling the data. Algorithm 2 describes the $refine(\cdot)$ function used in Algorithm 1, which refines the resolution of the initial K-list by inserting new testing points into the set K to best match the sizes of the rare categories being labeled by the oracle. Specifically, we find $k \in K$, s.t.,

$$k = \underset{k \in K}{argmin}(k - |C|)^2.$$

This formulation solves for the k that is the closest to representing the category size $|C|$ once C has been labeled by the oracle. We estimate the gap between k and C, inserting a new data point $(k + |C|)/2$ into K to increase the resolution of the LOF-curves. This refinement is performed iteratively until the gap between k and $|C|$ is smaller than a threshold. The refined K-list is then used in the next round of rare category identification.

Algorithm 2 uses the previously labeled categories for refining k's searching scope for two reasons. First, although we do not expect different rare categories are the same or follow the same distribution, we do expect that instances within the same rare category will follow the similar distribution. As discussed in "Category Expansion", Algorithm 1 usually underestimates the size of each rare category to ensure the precision. Therefore, a large or a non-convex rare category is usually separated into different parts and assigned with different labels. In this case, refining the searching range of k based on the previously labeled category can help with the detection of the next rare category or categories with the similar size or distribution. Second, refining the granularity of the searching range will also make the detection procedure increasingly precise. Especially, after finding those obvious ones, the remaining rare categories can only be found based on precise calculations. Therefore, gradually increasing the number of checking

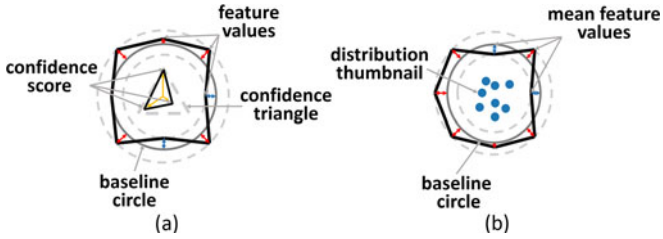


Fig. 6. Z-glyphs used in category explorer. (a) Z-glyph for representing an individual instance. (b) a collection glyph for merged circles.

points in the searching range based on the previous distribution of the data is a strategy for increasing the precision of the calculation. The effectiveness of Algorithm 2 is verified in experiments that are reported in the supplemental material: http://nancao.org/supp/rclens.pdf.

Complexity Analysis. The complexity of each algorithm iteration is $O(Nlog(N))$ (computing LOF scores) + $O(N)$ (calculating confidence scores) + $O(\sum_{i=0}^{k} m_i^2)$ (expanding the category) + $O(log(|K|))$ (refining the K-list) = $O(Nlog(N))$, where $N = |U|$ is the size of the dataset, m_i is the size of the ith category, and $|K|$ is the length of the K-list. Suppose there are k iterations in total, the algorithm complexity is $O(kNlog(N))$ (k iterations) + $O(Nlog(N))$ (calculating the neighborhood matrix M) = $O(kNlog(N))$. This performance can be achieved by using KD-tree to accelerate the calculation of neighborhood matrix and LOF scores.

4.2.2 Category View

A key aspect of the LOFRCD algorithm outlined above (Algorithm 1) is the use of the oracle's labels to improve the searching scope. Therefore, the interface that visualizes the initial rare categories detected by the algorithm for refinement and labeling is critical to the overall design. To assist with this category labeling process, a well-designed visualization should be able to represent the category in context, help the user correctly estimate the quality of the analysis results, and support rich interactions to drive interactive category refinement. The category view (Fig. 2, Panel 7) is designed for this purpose.

In the category view, glyphs representing the data instances are positioned spatially based on the values of the features via multidimensional scaling (MDS). The resulting visualization preserves high dimensional euclidean distance in the feature space in the low-dimensional visualization plane, thus showing k-nearest neighbor relationships for each instance in the feature space in a planer representation. This is critical as the k-nearest neighborhood is the key building block in the LOFRCD algorithm.

The individual glyphs adopt the Z-glyph design [30], [37], which has been extended as shown in Fig. 6 to illustrate a contextualized view of each data instance relative to category mean. In particular, in the glyph (Fig. 6a), the feature axises are radially aligned together, connecting by a baseline circle in the middle showing the mean feature values of the category. The feature values of the focused instance are shown surrounding the baseline and connected by a polyline. For the instance's feature values that are above average, the values are shown outside the baseline circle. In contrast, the feature values below average are shown within the circle. This design helps with the identification of outliers (i.e., the instances having a larger variance compared to the mean) in the focused category.

A confidence triangle is shown in the middle of the glyph with the lengths of the three axises indicate $\mathscr{C}_1(a)\mathscr{C}_2(a)$, $\mathscr{C}_3(a)$, and $\mathscr{C}_4(a)$ respectively,[2] illustrating the instance's confidence in terms of representing a new rare category from three different aspects as discussed in the last section.

The category view is highly interactive, allowing users to zoom in and out, and to pan their views to focus on specific sections of the dataset. To prevent occlusions when zooming out, glyphs are automatically aggregated into meta-glyphs based on averaged feature values when the boundaries of two or more glyphs begin to overlap. Similarly, the meta-glyphs are then split into multiple smaller glyphs when zooming in provides more room. The expert users are also able to switch between different glyph styles, with Z-Star used as the default. As shown in Fig. 6b, the meta-glyph is also visualized in form of a Z-glyph with the center illustrates a thumbnail of the distribution of its containing instances shown in the barycenter coordinates formulated by the surrounding feature axis. A k-slider is also implemented in our system, with which the oracle can manually adjust the size of k-nearest neighborhood shown surrounding the center point of the category.

With the above designs, the user can easily refine a category that is initially detected by the algorithm by eliminating outlier instances from the category, or by adding new closely related instances into the category. Once done with the editing, the user can assign a label to the current category by clicking the "feedback" button. The label could be anything provided by the user such as a category index number or a meaningful string indicating the type of the category. However, the algorithm will always take the label as a meaningless string, which is then used for labeling other closely related instances in the data during the next expanding procedure.

4.2.3 Category List

Once a rare category is labeled by the user and expanded by the algorithm, it is shown as a horizontal z-glyph arranged in a vertical list alongside the main category view (Fig. 2, Panel 8). In this glyph, the central horizontal baseline indicates the mean for feature values over the entire dataset. The category's mean feature values are shown as either (1) above the baseline when the value is larger than the overall dataset mean, or (2) below the baseline when the feature values are smaller than the overall dataset mean. In this way, the category glyphs in this view provide a profile of the rare category and its feature-by-feature differences from the overall majority of the data elements.

The glyph corresponding to the focus category (the one being displayed in the category view) is listed first in the list. Glyphs for the remaining categories are reordered based on their similarities to the focus. In this way, users can more easily identify similar categories for comparison. Optionally, users can merge similar categories together to form a single larger category. This operation is especially useful in case when what should have been one rare

2. The confidence triangle is only shown in the glyphs when the LOF-score of the corresponding instance is larger than 1.

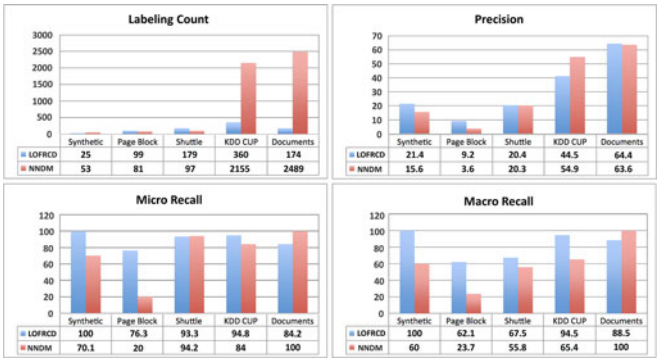


Fig. 7. The overall performances of LOFRCD in comparison to NNDM.

category is split into different parts during the detection procedure. This can happen in practice in ways that even the automatic category expansion algorithms cannot correct as shown in the example included in Fig. 5. Therefore, this manual category merging capability can help produce a more refined set of labels.

5 Evaluation

In this section, we evaluated the proposed algorithm and the RCLens system through quantitative estimations and an in-depth case study, respectively.

5.1 Algorithm Evaluation

In this section, we compared our algorithm to the baseline algorithm, NNDM, based on both synthetic and real data. The results demonstrate the power of the proposed algorithm.

Metrics. To ensure a comprehensive estimation of the algorithm, we compared our algorithm to NNDM based on four metrics: (1) *precision*, which estimates the algorithm's ability to correctly identify the members of rare categories; (2) *recall*, which estimates the algorithm's ability to correctly identify all rare categories. Here, precision and recall follow the standard definition introduced in [38]. Specifically, precision is given by the ratio between the number of "true positives" (i.e., the instances identified by the algorithm that indeed belong to a rare category) and the total number of instances identified by the algorithm. As for recall, we present both micro and macro average of recall to better illustrates our algorithm's performance. Micro average of recall is given by the ratio between all "true positive" and the total number of all rare category instances in the data. Macro recall is given by the average of recalls for each rare category. Having a high macro recall shows the algorithm has identified most of the instances in each rare categories. (3) *labeling count* (i.e., number of iterations), which estimates the cost of the oracle's effort; and (4) *labeling size*, which estimates the size of the labeled instances in each iteration in proportion to the size of full data.

Algorithm Settings. In the experiment, we performed NNDM under ideal conditions. More specifically, the total number of the rare categories and the size of each category were precisely given. In comparison, LOFRCD runs in more realistic conditions, in which no prior knowledge is given. Both algorithms stopped running when there was no more rare categories can be found in the data. Specifically,

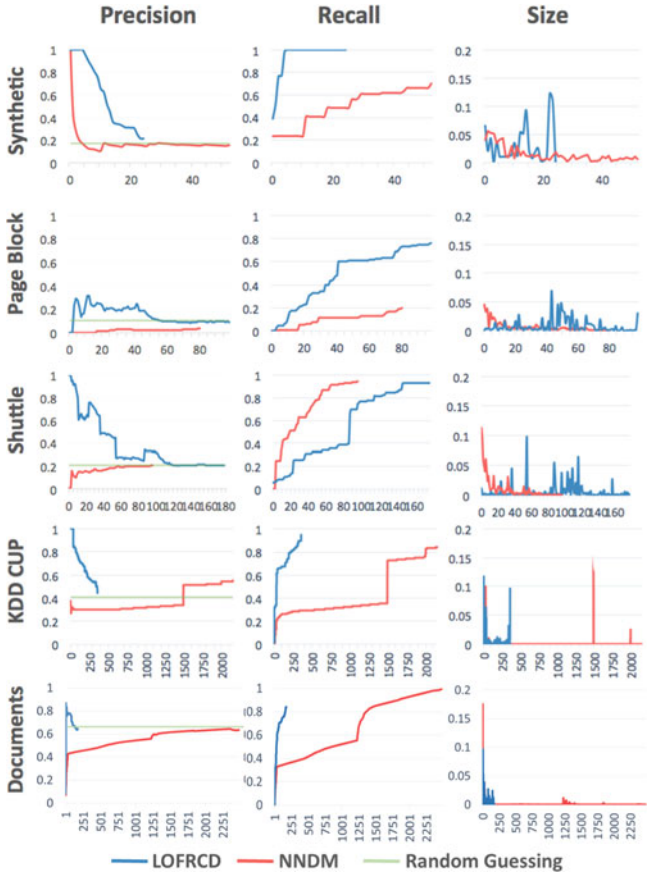


Fig. 8. The performances of each iteration respectively regarding to the precision, recall, and the size of labeled data instances.

NNDM stopped when at least one instance of each category had been found. LOFRCD stopped at a more strict condition in which all the potential category centers should be found. As the total number of rare categories was given, NNDM can always find all the rare categories. However, it was possible for LOFRCD to miss some rare categories when stop running. To test the algorithms' innate performance without human intervention, categories in the experiments were labeled automatically based on representative instances without any supervision from an oracle.

Datasets and Results. We used both synthetic data and real datasets for testing the proposed algorithm and compared the results of LOFRCD with the baseline technique NNDM. The overall experimental results are summarized in Fig. 7 and the detailed performance in each iteration is shown in Fig. 8. In general, our algorithm can achieve a higher precision and recall with fewer iterations. Next, we describe each dataset as well as the corresponding testing results in detail.

Synthetic Data. As shown in Fig. 4a, this dataset consists of four rare categories (in red) and two majority categories (in blue) with different sizes and distribution densities. The overall dataset contains 1,203 2-dimensional instances, with 204 of them belonging to rare categories. The testing results fully illustrate the advantage of our algorithm, it finds out all the rare categories within 25 interactions with the overall precession as 21 percent and the overall recall as 100 percent. When looking into the detailed iterations, we found our algorithm finds out all the rare categories within only six iterations (recall reaches 100 percent) and the precision of these

six iterations is 95.3 percent. In comparison, NNDM takes more time with a much lower precision and recall.

Page Block This dataset contains 5,473 10-dimensional data instances, which are separated into 5 classes. The majority class contains 89.77 percent of data instances and the remaining four classes only have 10.23 percent of the instances in total, which are treated as the rare categories. The testing results of LOFRCD and NNDM, as shown in Fig. 7 indicate that LOFRCD is much better than NNDM in terms of micro and macro average of recall, but only slightly better than NNDM in terms of precision. LOFRCD's benefits are revealed in Fig. 8. In each iteration, LOFRCD's precision is above the green line which indicates the precision of random guessing (i.e., the precision produced by randomly guessing whether or not an instance belongs to a rare category). [3] However, NNDM 's precision is below this line.

Shuttle. Another highly skewed public dataset[4], this dataset was also used to test the original NNDM algorithm. We used the full data for our experiments, consisting of 14,500 9-dimensional instances. The majority class contains approximately 80 percent of the data, with six additional categories. We removed two categories containing only 2 and 4 instances as the category size is too small to be considered a rare category. The results showed that LOFRCD and NNDM had a similar overall performance in terms of both precision and recall (Fig. 7), but Fig. 8 illustrates more details. NNDM kept a higher recall, but its precision was very low (even below the precision of random guessing). Especially, in the first few iterations, it labeled a relatively large set of data instances with a particular low precision. In comparison, the precision of our algorithm in the first several iterations is significantly higher than that of NNDM, which indicates the algorithm correctly detects rare categories since the very beginning. Later, its precision drops rapidly but the corresponding recall remains a high value, which indicates the major rare categories have already been found by the algorithm. We believe the drop of precision is due to the labeling of a large category around the 60th iteration. This also provides us a hint for terminating the algorithm.

KDD Cup '99. We also tested the algorithms using a random sample of the KDD Cup 1999 dataset. This sample contained 26,289 34-dimensional data instances, and included seven rare categories. The testing results showed that LOFRCD was about 6 times faster than NNDM in terms of labeling count, and our algorithm precision maintains higher than random guessing before most of the rare categories are detected while NNDM only reaches precision near random guessing. LOFRCD not only used less labeling counts, it also ended up with both higher micro average and macro average of recall compared with NNDM.

Documents. We used a full data set of news articles from the Reuters-21578 corpus,[5] a benchmark dataset commonly used for testing the performance of text classifiers. Our data sample contains 10,788 documents and 79 categories. Each document is assigned to one category. The distribution of the categories is highly skewed with 36 percent of the documents in the most common category. The rest are distributed across 78 rare categories. We excluded categories with less than 5 instances, and we ended up with 56 rare categories for testing. The testing results showed that LOFRCD is significantly better than NNDM in terms of labeling count. And our algorithm remains high precision above random guessing while NNDM is lower than it (Fig. 7). NNDM detected the last category only after the last iteration while our algorithm uses far less labeling count and higher precision to detect above 80 percent of the rare categories.

In general, the above experiments showed that LOFRCD has better precision than NNDM and are largely above the random guessing level at most time of the labeling process. In particular, LOFRCD's overall labeling count is significantly smaller than that of NNDM, signifying a meaningful reduction in the oracle's labeling efforts. In more realistic settings, where human users serve as the oracle, this can be a significant benefit. In addition, all the above comparison were made under the best condition of NNDM, i.e., the number of rare categories and the size of each category are precisely known. This is typically impossible in most real world applications.

5.2 Case Study and Expert Interview

To evaluate the RCLens system, we conducted a case study with domain experts based on real-world data. In this section, we introduce the study design, present the procedure and results, and discuss the findings.

5.2.1 Study Design

Participants. We invited a group of five expert users (1 female) to participate in our case study with the goal of estimating the usability of RCLens system. All of the users were from the information security team within the cloud computing department of a large international Internet company. One of the experts was the manager of the team and the rest were team members. Their major job duty was to detect and prevent intrusions based on server logs that are collected from their numerous servers. Their detection model was trained based on a large set of labeled data which were manually annotated. A lot of time and money were spent each year by a team of employees to manually label the data. They urgently need a tool that can help improve this process and make it less labor-intensive.

Dataset. We chose to use the KDD CUP'99 dataset [39] in the case study for two major reasons: (1) our experts were familiar with the dataset and its application domain, and (2) ground truth labels for the data were available to facilitate evaluation. The raw dataset contains 5 million server connection records and 22 types of illegitimate attacks. The attacks can be broadly classified into 4 categories as shown in Table 1. To ensure that the case study could be finished within a controlled period of time, a small random sample was generated. This sample contains 3,720 data instances of which 624 represent attacks. Those 624 instances represent 10 distinct attack types (i.e., 10 rare categories) with different sizes. As shown in Table 2, the random sampling roughly preserved the distributions found in the original dataset.

3. Even the precision is below random guessing the rare category algorithm is still useful as it still can largely capture the structure of a potential rare category from the data.

4. http://archive.ics.uci.edu/ml/datasets/Statlog+29

5. www.research.att.com/lewisreuters21578.html

TABLE 1
Types of Attacks

categories of attacks	types of attacks	description
DOS	back, land, neptune, pod, smurf, teardrop	Denial of service
Probing	ip_sweep, portsweep, nmap, satan	Surveillance and other probing
U2R	perl, rootkit, loadmodule, buffer_overflow	unauthorized access to local super user (root) privileges
R2L	ftp_write, guess_passwd, multihop, phf, imap, spy, warezclient, waremaster	unauthorized access from a remote machine

TABLE 2
Composition of Samples

attack types	categories	sample size	sample percentage
DOS	smurf	106	2.85
	Neptune	54	1.45
	back	51	1.37
	teardrop	51	1.37
Probing	ip_sweep	141	3.79
	satan	64	1.72
R2L	warezclient	66	1.77
	guess_password	39	1.05
	warezmaster	18	0.48
U2R	buffer_overflow	34	0.91
Total	-	624	16.77

Tasks. During the case study, the expert users were asked to label as many illegitimate server connections as possible and to classify these connections into categories while labeling. To ensure a full exploration of the system's functionality, the experts were required to answer a set of questions about RCLens such as:

- *T1* Which parts of the data are probably most problematic?
- *T2* In which feature or under which set of features are the connections more differentiable?
- *T3* Do the server connections shown in the category view represent the same type of connections? Are they illegitimate?
- *T4* Do the labeled connection categories shown in the category list contain different types of illegitimate connections?
- *T5* Have you found all major types of illegitimate connections in the data?

5.2.2 Procedures

We started the case study with a tutorial explaining the "rare category" concept, the major features of RCLens, and the study dataset. We then provided a detailed demonstration of the system followed by a practice session in which the experts were asked to use the system on their own. After fully exploring the systems functionality, users were asked to answer the questions enumerated above by exploring the data with the RCLens system. Each expert performed the study separately on different computers at the same time, with group discussions encouraged. A post-study questionnaire and a semi-structured group interview were conducted to collect feedback and comments after the users were finished using the system. The questionnaire and interview included questions on aspects of the visual design, the active learning technique, the overall system's usefulness and ease of use, and general pros and cons. During the study, moderators were available to answer any of the experts' questions to avoid any confusion or misunderstanding. The study lasted approximately 1.5 hours (10 minutes for introduction[6], 10 minutes for practice, 40 minutes for finishing the required tasks, and 30 minutes for the post-task interview and questionnaires).

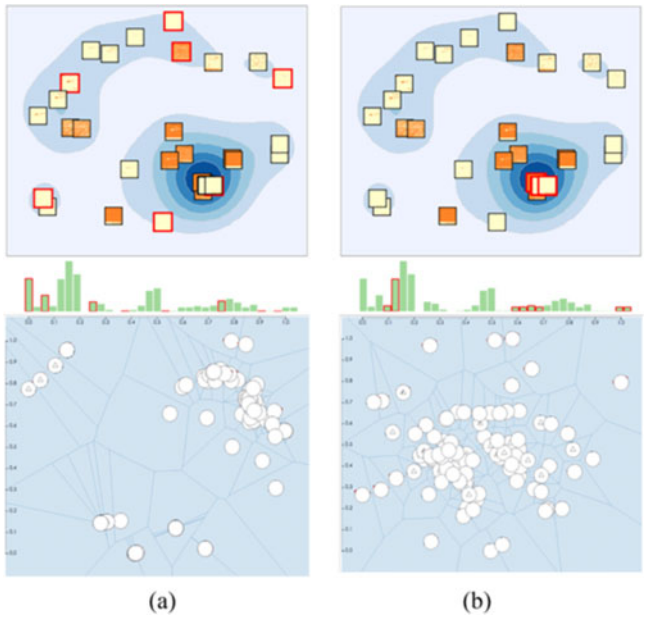


Fig. 9. Subspace exploration. (a) The subspace formulated by highly correlated features and the distribution of data instances. (b) The subspace formulated by irrelevant features and the corresponding data distribution.

During the study, we did not provide any prior knowledge to the participants. At the end, the experts were able to find out at least one rare category for each attack type (7 were detected in total), although they missed several unobvious ones due to the limited time. To detect these rare categories, more than 10 different subspaces were explored and at least 3 of them were chosen for the analysis. We found the group discussion helped a lot on their performance. As RCLens is not specifically designed to detect rare Internet connections, their discussion mainly focused on how to determine the types of the suspicious connections that they found.

5.2.3 Findings

We report findings from the case study, focusing on both the feature selection and rare category detection capabilities in the RCLens system.

6. This 10-minutes doesn't include the conference call that we made one day before for preparing the study. During the call, we also briefly introduced the system.

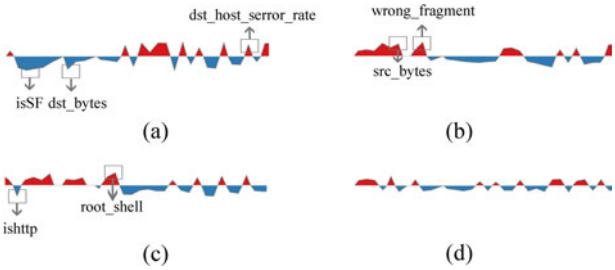


Fig. 10. Four categories detected by the domain experts in a row based on the RCLens system. The first three are consist of three different types of illegitimate connections, i.e., (a) satan, (b) back, and (c) warezclient. The last one (d) shows a normal category.

Fig. 9 compares the data distribution in two different subspaces formulated by our experts during the study. In particular, the data instances (i.e., server connections) are clearly separated into clusters in a subspace that is formulated by a set of independent features (Fig. 9a) but overlapped in another subspace formulated by a group of highly correlated features (Fig. 9b). The first one was chosen for the subsequent rare category detection step.

Fig. 10 illustrates the horizon z-glyphs of four categories that were detected by the experts during the study. The z-glyphs of the first three categories illustrate irregular shapes that are significantly different from the mean value, and the corresponding distributions of the data inside these categories are also highly skewed. This produces the irregular patterns in the corresponding z-glyphs (Figs. 10a, 10b, and 10c) which are indicative of a rare category. These categories were later verified to be the rare categories that contained specific types of illegitimate connections. In particular, these categories represent attacks from the "satan", "back", and "warezclient" categories, respectively. In comparison, Figs. 10d and 11d both illustrate a "normal" category, containing near-mean feature values and less interesting connections, respectively.

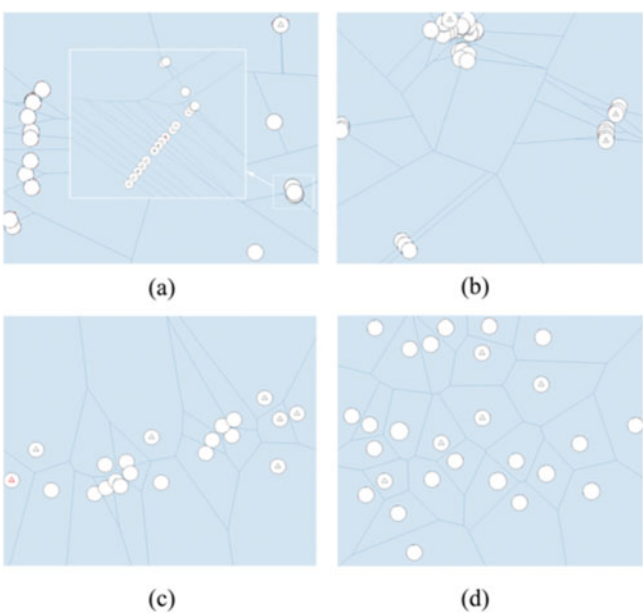


Fig. 11. The instance (i.e., connection) distributions in each of the categories shown in Fig. 10.

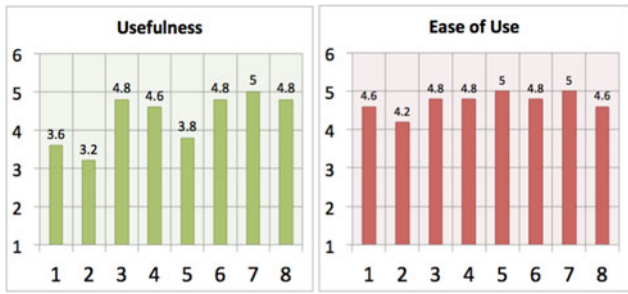


Fig. 12. The questionnaire results. The scores are ranged from 1 (very useless or very difficult to use) 5 (very useful or very easy to use). 1 to 8 on the *x*-axis indicates eight different views shown in Fig. 2.

5.2.4 *Subjective Feedback*

After the case study, the participants completed a post-study questionnaire to rate for the usability of each of the eight views (Fig. 2). The results (Fig. 12) show that overall our system useful and easy to use.

In general, the participants liked RCLens system and confirmed its novelty. They said: *"None of the visualization tools that we used before were able to identify anomalies while considering their categories." "This accelerates the data labeling process,"* and that *"we should introduce it to our data preprocessing team"*, referring to the contractors that they had hired to manually label the data. In the data exploration and analysis process, the experts first selected an interesting subset of data by querying the connections based on their IP address, then further filtered the data for analysis via the parallel coordinates panel. Some of the experts particularly liked this two-step data selection process. One analyst said: *"It is nice to roughly select the data first and refine and filter them later (on the parallel coordinates); this is especially helpful when the data details initially are unknown."*

The experts were also impressed by the feature explorer. They believed that the feature distribution view was helpful for identifying independent features and the subspace glyphs were also informative in terms of showing a thumbnail of the subspace. *"With these views, I can easily find the most informative features and investigate the distributions of the data within different feature spaces,"* said one expert, and the others agreed. When investigating "illegitimate connections" in the category explorer, the experts felt the zoom-to-aggregate function was helpful for quickly summarizing a recommended rare category and identifying outliers in the category. They commented that *"once the nodes are merged (into meta nodes), I can easily read the summarized features from a tooltip, which is useful for understanding the category."* They also appreciated the idea of merging similar categories together in the category list after knowing the algorithm's details and believed the horizontal z-glyph was very "intuitive and efficient" for representing a rare category.

One limitation that was commonly mentioned in the case study was that our system could not handle temporal data, which limited its usage. Another commonly mentioned issue was that sometimes the category interpretation was not easy. This was a foreseen problem as RCLens was not specifically designed for detecting illegitimate connections. It was designed for a more general purpose, i.e., finding rare categories in multidimensional data without domain

specific assumptions. In addition, the users also felt the data explorer was not well designed as *"manually exploring the raw data (even based on query) is not easy and scalable"* and the parallel coordinates sometimes is "too clutter to help with the data filtering." These are indeed problems of the current system design, which will be addressed in the future. Two of the experts also mentioned concerns about the scalability of the data explorer as the parallel coordinates were not scalable to present very large numbers of variables in high-dimensional datasets. Indeed, this is a limitation of the current prototype, but the experts also believed that the two-step query mechanism partially addressed this issue.

6 Conclusion

In this paper, we introduce RCLens, a visual analytics system designed to support user-guided rare category exploration and identification. RCLens adopts a novel active learning-based algorithm to iteratively identify more accurate rare categories in response to user-provided feedback. That algorithm is tightly integrated with an interactive visualization-based interface that supports a novel and intuitive workflow for rare category identification.

A prototype was developed based on the methodology presented in this paper, and evaluated using both (1) quantitative experiments to compare against prior methods, and (2) a case study with a team of network security analysts to evaluate the usability and applicability of the approach. The evaluation results show that RCLens performs as well (or better than) previous methods in ideal conditions, while providing additional capabilities that are important to real-world use cases. The case study showed that the workflow supported by our approach is both applicable to real-world scenarios and well-received by practitioners in the field.

Future work will focus on improving the efficiency of the algorithm to make it even more scalable, extending the algorithm to support time series data, and designing classication-based rare category detection techniques. Moreover, we will add more statistical information in the visualization panels to guide the data exploration procedure. In addition, new domain-specic context view "plugins" will be explored as a means to better tailor the system to unique domain-specic challenges.

Acknowledgments

We would like to thank all the experts who helped with the system evaluation. We also would like to thank Dr. Sabrina Lin from IBM T. J. Watson Research Center for her initial contribution to the algorithm design. This work is a part of the research supported from NSFC grant No. 61602306, NSF Grant No. IIS-1552654 and No. DMS-1557593, ONR Grant No. N00014-15-1-2821, IBM SUR Award, and IBM Faculty Award. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

References

[1] J. He, "Rare category detection," in *Analysis of Rare Categories*. Berlin, Germany: Springer, 2011, pp. 17–74.
[2] J. He and J. Carbonell, "Prior-free rare category detection," *Learning*, vol. 2, 2009, Art. no. 5.
[3] J. He and J. G. Carbonell, "Nearest-neighbor-based active learning for rare category detection," in *Proc. Advances Neural Inf. Process. Syst.*, 2007, pp. 633–640.
[4] T. M. Hospedales, S. Gong, and T. Xiang, "Finding rare classes: Active learning with generative and discriminative models," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 2, pp. 374–386, Feb. 2013.
[5] D. Pelleg and A. W. Moore, "Active learning for anomaly and rare-category detection," in *Proc. Advances Neural Inf. Process. Syst.*, 2004, pp. 1073–1080.
[6] B. Settles, "Active learning literature survey," *University Wisconsin, Madison*, vol. 52, no. 55–66, 2010, Art. no. 11.
[7] D. Angluin, "Queries and concept learning," *Mach. Learn.*, vol. 2, no. 4, pp. 319–342, 1988.
[8] D. Angluin, "Queries revisited," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2001, pp. 12–31.
[9] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," *J. Artif. Intell. Res.*, vol. 4, pp. 129–145, 1996.
[10] D. Cohn, L. Atlas, and R. Ladner, "Improving generalization with active learning," *Mach. Learn.*, vol. 15, no. 2, pp. 201–221, 1994.
[11] I. Dagan and S. P. Engelson, "Committee-based sampling for training probabilistic classifiers," in *Proc. Int. Conf. Mach. Learn.*, 1995, pp. 150–157.
[12] V. Krishnamurthy, "Algorithms for optimal scheduling and management of hidden Markov model sensors," *IEEE Trans. Signal Process.*, vol. 50, no. 6, pp. 1382–1397, Jun. 2002.
[13] T. M. Hospedales, S. Gong, and T. Xiang, "A unifying theory of active discovery and learning," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 453–466.
[14] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Large-scale text categorization by batch mode active learning," in *Proc. ACM Int. Conf. World Wide Web*, 2006, pp. 633–642.
[15] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proc. ACM Int. SIGIR Conf. Res. Develop. Inf. Retrieval*, 1994, pp. 3–12.
[16] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *J. Mach. Learn. Res.*, vol. 2, pp. 45–66, 2002.
[17] C. Zhang and T. Chen, "An active learning framework for content-based information retrieval," *IEEE Trans. Multimedia*, vol. 4, no. 2, pp. 260–268, Jun. 2002.
[18] G. Tur, D. Hakkani-Tür, and R. E. Schapire, "Combining active and semi-supervised learning for spoken language understanding," *Speech Commun.*, vol. 45, no. 2, pp. 171–186, 2005.
[19] Y. Liu, "Active learning with support vector machine applied to gene expression data for cancer classification," *J. Chemical Inf. Comput. Sci.*, vol. 44, no. 6, pp. 1936–1941, 2004.
[20] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surveys*, vol. 41, no. 3, 2009, Art. no. 15.
[21] V. J. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artif. Intell. Rev.*, vol. 22, no. 2, pp. 85–126, 2004.
[22] J. Wu, H. Xiong, P. Wu, and J. Chen, "Local decomposition for rare class analysis," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2007, pp. 814–823.
[23] P. Vatturi and W.-K. Wong, "Category detection using hierarchical mean shift," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 847–856.
[24] J. He, Y. Liu, and R. D. Lawrence, "Graph-based rare category detection," in *Proc. IEEE Int. Conf. Data Mining*, 2008, pp. 833–838.
[25] H. Huang, Q. He, J. He, and L. Ma, "RADAR: Rare category detection via computation of boundary degree," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*, 2011, pp. 258–269.
[26] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," *ACM Special Interest Group Manage. Data Rec.*, vol. 29, pp. 93–104, 2000.
[27] H. Huang, K. Chiew, Y. Gao, Q. He, and Q. Li, "Rare category exploration," *Expert Syst. Appl.*, vol. 41, no. 9, pp. 4197–4210, 2014.
[28] Z. Liu, K. Chiew, Q. He, H. Huang, and B. Huang, "Prior-free rare category detection: More effective and efficient solutions," *Expert Syst. Appl.*, vol. 41, no. 17, pp. 7691–7706, 2014.
[29] J. Zhao, N. Cao, Z. Wen, Y. Song, Y.-R. Lin, and C. M. Collins, "# FluxFlow: Visual analysis of anomalous information spreading on social media," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 12, pp. 1773–1782, Dec. 2014.
[30] N. Cao, C. Shi, S. Lin, J. Lu, Y.-R. Lin, and C.-Y. Lin, "TargetVue: Visual analysis of anomalous user behaviors in online communication systems," *IEEE Trans. Vis. Comput. Graph.*, vol. 22, no. 1, pp. 280–289, Jan. 2016.

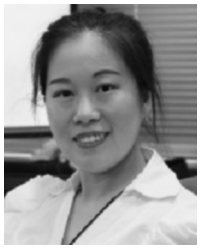
[31] M. Babaee, S. Tsoukalas, G. Rigoll, and M. Datcu, "Visualization-based active learning for the annotation of SAR images," *IEEE J. Sel. Topics Appl. Earth Observations Remote Sens.*, vol. 8, no. 10, pp. 4687–4698, Oct. 2015.
[32] T. Iwata, N. Houlsby, and Z. Ghahramani, "Active learning for interactive visualization," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2013, pp. 342–350.
[33] Inselberg A. Parallel coordinates[M]. Springer US, 2009.
[34] I. Jolliffe, *Principal Component Analysis*. Hoboken, NJ, USA: Wiley, 2002.
[35] T. M. Phuong, Z. Lin, and R. B. Altman, "Choosing SNPs using feature selection," in *Proc. IEEE Comput. Syst. Bioinf. Conf.*, 2005, pp. 301–309.
[36] W. Duch, *Filter Methods*. Berlin, Germany: Springer, 2006, pp. 89–117. [Online]. Available: http://dx.doi.org/10.1007/978–3-540-35488-8_4
[37] N. Cao, Y.-R. Lin, and D. Gotz, et al., "Z-Glyph: Visualizing outliers in multivariate data," *Inf. Vis.*, pp. 1–19, 2017, http://journals.sagepub.com/doi/abs/10.1177/1473871616686635
[38] J. W. Perry, A. Kent, and M. M. Berry, "Machine literature searching x. machine language; factors underlying its design and development," *Amer. Documentation*, vol. 6, no. 4, pp. 242–254, 1955.
[39] A. A. Olusola, A. S. Oladele, and D. O. Abosede, "Analysis of KDD'99 intrusion detection dataset for selection of relevance features," in *Proc. World Congr. Eng. Comput. Sci.*, 2010, vol. 1, pp. 20–22.

Hanfei Lin received the bachelor's (with Hons.) degree from the Computer Science Department, East China Normal University. She is working toward the graduate degree at the University of Illinois Urbana-Champaign. She was a research assistant in Intelligent Big Data Visualization (iDV^x) Lab.

Siyuan Gao received the bachelor's (with Hons.) degree from the Math School, Zhejiang University. He is working toward the PhD degree at Yale University. He was a research assistant in Intelligent Big Data Visualization (iDV^x) Lab.

David Gotz is an associate professor in the School of Information and Library Science, University of North Carolina at Chapel Hill, an assistant director of the Carolina Health Informatics Program, and an associate member of the UNC Lineberger Cancer Center. His research interests include data visualization, visual analytics, data science and analysis, and medical informatics.

Fan Du received the bachelor's (with Hons.) degree from Zhejiang University. He is working toward the PhD degree in computer science at the University of Maryland, College Park. He was a research assistant in Intelligent Big Data Visualization (iDV^x) Lab. His research focuses on data visualization and human-computer interaction, especially on analyzing healthcare data, and user activity logs.



Jingrui He is an assistant professor of computer science with Arizona State University. Her research interests include heterogeneous machine learning, rare category analysis, active learning and semi-supervised learning, with applications in social network analysis, healthcare, and manufacturing.

Nan Cao is a professor with Tongji University. He is also the founder and director of Intelligent Big Data Visualization (iDV^x) Lab. His research interests include data visualization, visual analysis, and data mining. He creates novel visual analysis techniques for supporting anomaly detection in complex (i.e., big, dynamic, multivariate, heterogeneous, and multi-relational) data.