Using Consensus Clustering for Multi-view Anomaly Detection

Alexander Y. Liu and Dung N. Lam
Applied Research Laboratories
The University of Texas at Austin, P.O. Box 8029
Austin, Texas 78713-8029
{aliu, d.lam}@arlut.utexas.edu

This paper presents work on automatically characterizing typical user activities across multiple sources (or views) of data, as well as finding anomalous users who engage in unusual combinations of activities across different views of data. This approach can be used to detect malicious insiders who may abuse their privileged access to systems in order to accomplish goals that are detrimental to the organizations that grant those privileges. To avoid detection, these malicious insiders want to appear as normal as possible with respect to the activities of other users with similar privileges and tasks. Therefore, given a single type or view of audit data, the activities of the malicious insider may appear normal. An anomaly may only be apparent when analyzing multiple sources of data. We propose and test domain-independent methods that combine consensus clustering and anomaly detection techniques. We benchmark the efficacy of these methods on simulated insider threat data. Experimental results show that combining anomaly detection and consensus clustering produces more accurate results than sequentially performing the two tasks independently.

Keywords: insider threat, multi-view learning, anomaly detection, consensus clustering

I. Introduction

Malicious insiders are users, with legitimate access to a system, who abuse their access privileges in order to perform tasks that are detrimental to the organization that grants their access privileges. Malicious insider activity (including various forms of corporate espionage) can be difficult to detect. In particular, it may be difficult to fully capture insider activity in terms of definable models or rules. Thus, a useful tool for detecting potentially malicious insider activity is anomaly detection, where one makes the assumption that malicious insider activity can be detected as deviations with respect to some set of normal activity (e.g., the insider's normal work habits or the work habits of co-workers).

Certain straightforward anomalies (e.g., sudden, excessive printing and/or file transfer activity) can be detected using histograms or change detection, particularly if the insider is not actively hiding their actions. In the case where the malicious insider is trying to appear as innocuous as possible in order to avoid detection, the insider's activities may seem normal compared to those of peers when analyzing any one data source (or view of the data), while those same activities are anomalous when analyzed across multiple views of the data. Therefore, more sophisticated methods are necessary to detect anomalies that are apparent only when these multiple data sources are analyzed in concert. This paper describes a type of multi-view anomaly detection, where the goal is to find unusual combinations of activity across multiple sources of data.

A basic example of this type of anomaly is as follows. From analyzing file accesses of users, we find that users A, B, and C access only Resource *I*, while users D, E, F, and G access both Resources *I* and *II*. No anomalies can be detected from this single view. Analyzing another view representing email interactions among users, we find two social network clusters: {A,B,C,D} and {E,F,G}. By examining both views, it is apparent that user D is anomalous because other users in the first social network cluster access only Resource *I*, while user D accesses both resources. In this example, group {A,B,C,D} could represent a group of developers working on only one particular project *I*, while group {E,F,G} are working on both projects *I* and *II* (e.g., in a managerial role). Although it is common for users in the second group to access files for both projects, user D should not be accessing files for project *II*. It is necessary to combine information from both data sources to determine that user D is acting strangely with respect to other users.

An existing method called consensus clustering is able to leverage multiple views of the data and decide in which cluster each user belongs by summarizing multiple sets of cluster labels (e.g., taken from multiple sources of data) into a single set of labels. However, this method alone does not detect anomalous data points (i.e., users). To the best of our knowledge, no other work on finding anomalies of this type exists regardless of the domain.

This paper presents four methods that both identify the groups that each user belongs to (i.e., consensus clustering) and find any users that do not fit these discovered groups (i.e., anomaly detection). The four automated methods aggregate and capitalize on multiple sources of data to find insiders performing unusual combinations of behavior that would appear normal if examined individually. In particular, empirical results show that more accurate consensus clustering and anomaly detection is achieved by performing both tasks in a coupled fashion rather than independently.

DOI 10.1109/SPW.2012.18

To support analysis of multi-view data, a previously developed framework called Core-Facets is used for modeling various sources of audit data for user actions, such as file accesses, and storing this information into graph structures. The Core-Facets framework is a graph-based data warehousing model designed to merge, store, and prepare relevant data in the form of graphs [1]. Core-Facets facilitates linking heterogeneous data into a single graph representation (the core graph) and enables *faceting* (i.e., extraction of purpose-specific views of the core graph) for further analysis by graph-based or traditional data-mining techniques. In particular, Core-Facets enables the ability to store and analyze information from multiple sources of data, facilitating the capability to discover unusual combinations of activity based on the extracted facet graphs.

II. RELATED WORK

There has been extensive work on detecting insider threat, including many methods that use automated machine learning. Many past approaches have focused on using supervised classifiers, including a large body of work on masquerade detection, as well as Elicit, which uses Bayesian networks [2]. Many approaches have also used various types of anomaly detection, including recent work in [3] among others. Complementing these existing efforts, this paper focuses specifically on detecting a particular type of anomalous activity indicative of potentially malicious behavior: insiders who use an unusual combination of otherwise normal activities to accomplish their malicious tasks, particularly when the individual normal activities are split across multiple sources of audit data. For example, it may be unusual in an organization for a member of an engineering group to access a large number of payroll files. However, this is only unusual if we know both the user's role and the typical user behaviors of other users fulfilling that role.

Since a user's role and aggregate user behavior can change over time, it is difficult to maintain a list of predefined user roles, user tasks, and typical user behaviors for those roles and tasks. To address this challenge, these characteristics can be determined automatically using machine learning methods such as clustering. This problem requires a method of detecting unusual combinations of cluster memberships across multiple views, which can be used to find insiders who exhibit unusual combinations of behavior that are not suspicious when examined in isolation. As mentioned, no other work on finding anomalies of this type exists regardless of the domain. The following sections discuss past work on anomaly detection in various domains, followed by a discussion of consensus clustering and how these two capabilities can be combined. For completeness, work related to Core-Facets is also included.

A. Anomaly Detection

A large variety of approaches to anomaly detection have been proposed [4], including statistics-based approaches that assume a particular parametric distribution, one-class classifiers, and information-theoretic approaches. The anomaly detection problem is ill-posed in the sense that there are multiple valid definitions of what makes a particular data point unusual or anomalous. Thus, different anomaly detection algorithms can be useful for finding different types of unusual data points.

Most anomaly detection approaches assume the data is unstructured; that is, each data point is independent from all other data points. A number of existing approaches for finding anomalies on structured data such as graphs have been proposed. One such approach is OddBall [5], which finds four specific types of subgraphs centered on individual nodes: stars, clique, subgraphs with large weights to neighbors, and subgraphs with a "dominant edge" (i.e., one of the weights to a neighbor is extremely high). That is, OddBall finds subgraphs that are known to be unusual in a specific problem domain.

Another approach to find anomalies in graph-based data is to use clusters of subgraphs [6]. In this approach, a hidden mixture model is fit to the graph nodes based on node attributes. For example, given continuous node attributes, a mixture of Gaussians can be fit to the nodes. The hidden mixture model variable therefore indicates the "community" of a node that can be used to find "community-based anomalies" (nodes that are unusual because they link only to nodes from different mixtures). Note that this approach is one of the few graph-based anomaly detection approaches that can handle multiple attributes per node.

Other recent work includes research which uses a minimum description length principle to find anomalies in graphs containing the same single subgraph structure repeated throughout the graph [7]. In this approach, anomalies consist of changes to this subgraph structure. Modifications to this approach are likely necessary for extremely large graphs, graphs with a high-degree of variation in the subgraph structure, or graphs with multiple subgraph structures.

In this paper, we are interested in detecting anomalies that can only be identified by examining multiple views of the data. In this case, each view is encoded in a separate facet graph. There has been much recent work in machine learning on multi-view learning, a problem in which there are multiple sources of information for each data point. An often used example of multi-view learning is learning from video data, where the multiple views could include image data or text data from the closed captions. There is also recent interest in multi-view graph mining (e.g., [8] uses a multi-view approach to cluster data in a social network graph). However, we are unaware of any multi-view anomaly detection approaches.

B. Consensus Clustering

In machine learning, a clustering algorithm takes a group of unlabeled data points as input and creates a set of cluster labels. Typically, a single set of cluster labels is a mapping from each data point to a group ID, where the goal of most clustering algorithms is to give similar data points (for some measure of similarity) the same group ID, and to give dissimilar data points different group IDs.

The detection of groups or clusters of nodes in a single graph can be automated using existing graph clustering techniques in machine learning. These graph clustering algorithms (e.g., METIS [9], GraClus [10]) attempt to find balanced clusters of nodes that are highly linked to each other, and split nodes that are weakly linked. When applied to a single view of data, a clustering algorithm produces a cluster label for each data point.

Given sets of cluster labels from clustering different views of data (called a set of "ensemble cluster labels"), a common problem is to find a single set of labels (called the "consensus cluster labels") that best characterizes the entire dataset across all views. The result of consensus clustering is a new consensus cluster label for each data point based on the ensemble cluster labels of each data point. An advantage of finding a set of consensus cluster labels is robustness, since a single clustering on a single view of the data can be inaccurate due to noise in the data and/or shortcomings of the clustering algorithm itself such as convergence to local minima or improper parameter selection. In particular, many algorithms require the number of clusters to be found as input, which may be difficult to determine for arbitrary data.

Formally, the problem definition in consensus clustering is as follows. We are given a set of n_{facet} graphs (each representing a single view) and a method of independently clustering each graph. Due to non-determinism and variations in configuration parameters of the clustering algorithm, each of these n_{facet} graphs (or views) is clustered n_c times, resulting in $n_{facet} * n_c$ sets of ensemble cluster labels for the $n_d$data points (where a data point is a node in the graph). Let the $n_{facet} * n_c$ sets of ensemble cluster labels be represented by the ensemble cluster label matrix C_E with n_d rows and $n_{facet} * \ n_c$ columns. $C_E(i, j)$ is the cluster label of the ith data point for the jth set of cluster labels (the ordering of the cluster labeling in C_E is arbitrary). Let the set of consensus cluster labels be encoded as a vector C_L, where the cluster label of the ith data point is represented by $C_L(i)$.

The goal of consensus clustering is to maximize $\boldsymbol{f}(C_L, C_E)$, where $\boldsymbol{f}(\cdot)$ is some evaluation metric. Strehl et al. introduces the use of average normalized mutual information as the evaluation metric [11]; this evaluation metric is discussed in detail in section IV.B. Intuitively, average normalized mutual information is high when the groups in the consensus cluster labels agree with the groups in each column of the ensemble cluster labels.

Strehl et al. also introduces three methods for consensus clustering [11]. Of the three techniques, we found the MCLA approach to be most useful, but MCLA typically did not perform as well on our benchmark datasets as a technique introduced by Topchy et al. [12], who pose the problem of consensus clustering as a problem of fitting a mixture of multinomial models to the data. This technique can be extended to find anomalous points while finding consensus cluster labels, and such extensions are described in Section IV.

C. *Consensus Clustering and Anomaly Detection*

Data points that are anomalies when considering multiple views will give rise to weak consensus; i.e., they will not belong definitively to any consensus cluster. To the best of our knowledge, no work exists that simultaneously finds consensus clusters as well as anomalous data points. The closest description in terms of existing approaches would be a density-based clustering algorithm that worked on categorical data. In density-based clustering, the assumption is that only some of the data forms clusters in the form of dense regions of data in the feature space, while the remaining data is essentially "noise" that falls outside of these dense regions. Many approaches for density-based clustering exist, including the popular distance-based DBSCAN algorithm and various related algorithms [13]. However, density-based clustering algorithms are typically applied to numerical features for which a distance measure such as a Euclidean distance is appropriate.

D. *Graph Processing Frameworks and Scalable Tools*

Graph structures are useful because they are an intuitive representation for information characterized by numerous relationships, and can model unnormalized, heterogeneous data well. Modeling and analyzing data in the form of graphs has become prevalent in a growing number of areas, such as webmining, social network analysis, and bioinformatics [14]. In recent years, research on heterogeneous nodes [15] and heterogeneous relationships between nodes [16, 17] shows new interest in mining graphs that store a variety of information, as opposed to homogeneous graphs with one or two types of nodes and edges.

Although there are many individual efforts focusing on graph-based data fusion [18] and graph mining [14], there has been little work that offers a comprehensive method for managing graph data and preparing the data for analysis. A graph database called DEX [18] facilitates the integration of multiple data sources into large graphs. However, while DEX provides a high-performance graph database to be used for exploration and data retrieval, it does not offer different customizable views of the graph for analysis.

To extract views of the graph for multifaceted anomaly detection, the Core-Facets graph-based data warehousing framework is employed [1]. It uses a graph as the underlying model for merged heterogeneous data (enabling use of indirect connections across different data sources) and faceting to build purpose-specific graph views for analysis. The faceting mechanism, in particular, enables new research areas such as multi-view data-mining—that is, discovering new patterns by viewing the data from multiple different perspectives.

III. CORE-FACETS

We use the Core-Facets framework to create views from multiple sources of data in order to detect potential insider threats. The Core-Facets framework builds a heterogeneous core graph and uses a technique called faceting to dynamically extract data for a variety of analyses. Each node or edge in the core graph can have a different set of attributes that represent the semantic details of the data. To create a particular view of the data, a facet graph is created by extracting, filtering, abstracting, and transforming the core graph based on the semantics of the data or the graph structure. Multiple facets can then be employed for multi-view analysis at multiple semantic and temporal scopes.

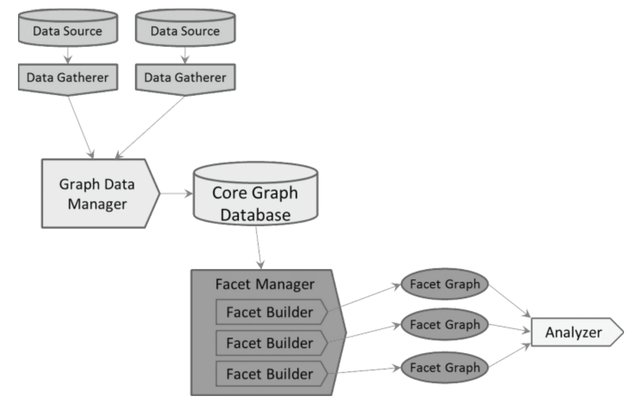

Figure 1: Dataflow Diagram of Core-Facets Framework

As shown in Fig. 1, Core-Facets consists of the following three phases:

1. Data Gathering (operational layer) – For each data source, a Data Gatherer retrieves desired data as specified by the user. The data is then formatted for import by the Graph Data Manager in the next phase.
2. Data Interpretation (data access and metadata layers) – The Graph Data Manager applies user-defined import and inference rules to the gathered data to build an intermediate graph. The rules map data entities to graph elements and attributes. The intermediate graph is then merged into the Core Graph Database.
3. Data Preparation (informational access layer) – User-defined Facet Builders create facet graphs from the core graph. The Facet Manager coordinates the interface between Facet Builders and the Core Graph Database. As shown in Figure 1, multiple facet graphs can be used by a multi-view analyzer.

A. *Data Gathering*

Each data source (ranging from a basic log file to a complex database) can be regarded as a specific sensor with limited scope. For each data source, a Data Gatherer is defined to retrieve the data from that source. Using several Data Gatherers, data across multiple sources can be linked and merged together in the next phase. The data retrieval mechanism used by each Data Gatherer may be a basic network file transfer or periodic database queries that join several database tables. The Data Gatherers also convert the data into a common format regardless of the type of data so that the next phase can be agnostic to variations in data source formats.

B. *Data Interpretation*

The objective of the data interpretation phase is to build up the core graph. Gathered data is interpreted by the Graph Data Manager, which merges the data into the core graph. A domain ontology is required for interpreting the gathered data and building the graph as the ontology establishes consistent concepts, terminology, and semantic relationships among concepts. Based on the ontology, import and inference rules map the gathered data to concepts and relationships, constituting a graph, which is then merged into the Core Graph Database. At the end of this phase, the core graph is populated with data from all specified data sources, and it is ready to be used.

C. *Data Preparation*

A data analyzer does not typically use all the heterogeneous data modeled in the core graph due to the data's inherent complexities, but an analyzer should take advantage of the indirect links established by merging data from multiple sources. To enable this, faceting is used to extract only the data relevant for a given analysis while retaining useful indirect relationships among the data. A Facet Builder creates a facet graph that captures different semantic information as a view required by the analyzer(s). A facet graph is created by traversing the core graph and performing extraction, abstraction, filtering, and other transformations on the nodes and edges of the core graph. The Facet Builders interpret the semantics that are encoded in the graph as node and edge attributes. Formal semantic relationships (defined for the domain) specify hierarchical or compositional concepts and other relationships among the data to enable data filtering and abstraction.

Facet Builders can collapse a multi-hop path between nodes into a single edge, or collapse a subgraph into a single abstracted node to remove irrelevant details (e.g., by translating several low-level file modification event data points into high-level behavior described as "sending emails"). This faceting process is particularly useful in preparing input for graph analysis and data mining techniques that assume the data is homogeneous. Multi-view analyzers that process several facets at one time can use different facets to learn new patterns that are not apparent in a single facet. Facet graphs are also useful for partitioning the data by subgraphs or by time (e.g., a separate graph for each week) and for modeling specific semantic topics (e.g., process-to-file operations or computer-to-computer file transfers) that are relevant to an analyzer.

IV. MULTIFACETED ANOMALY DETECTION

As mentioned, in the machine learning literature, a single set of cluster labels on a set of data points is a mapping from each data point to a group ID. Given a set of ensemble cluster labels as input, the goal of a consensus clustering algorithm is to find the single set of consensus cluster labels that best summarizes the information in the ensemble cluster labels.

This paper presents four methods to find consensus cluster labels and anomalous points given ensemble cluster labels. The goal of these algorithms is to demonstrate the capability to use multiple facet graphs representing different views in order to characterize typical behavior and find atypical behavior across views, a potentially useful form of anomaly detection for insider threat detection. Briefly, the four methods are as follows:

- The Sequential method first performs anomaly detection on the independently clustered data points, then performs consensus clustering on the remaining non-anomalous data points. This is used primarily as a baseline method.

- The Greedy Iterative Removal method repeatedly iterates between selection of anomalous data points and consensus clustering, where the result of one algorithm feeds directly into the other algorithm.
- The final two methods employ an EM approach that performs anomaly detection within the consensus clustering algorithm, in effect creating a cluster of anomalous points during consensus clustering.

Formally, the problem definition is as follows. We use the same notation as in the formal definition of consensus clustering in Section II, with the following additions. Let $C_E(\mathfrak{D})$ be the matrix consisting of only the rows of ensemble cluster labels C_E corresponding to data points in $\mathfrak{D}$, an ordered set of data points. Similarly, let $C_L(\mathfrak{D})$ be the vector of consensus cluster labels corresponding to data points in $\mathfrak{D}$.

The goal of the methods presented in this paper is to find both $\mathfrak{D}_\alpha$ (a set of n_α data points consisting of the most anomalous data points) and $\mathfrak{D}_{normal}$ (a set of $n_d - n_\alpha$ data points consisting of the remaining data points) such that $\boldsymbol{f}(C_L(\mathfrak{D}_{normal}), C_E(\mathfrak{D}_{normal}))$ is maximized over all possible sets $\mathfrak{D}_{normal}$, where $\boldsymbol{f}(\cdot)$ is some evaluation metric. As in [11], average normalized mutual information is used as an evaluation metric. Unfortunately, directly finding the best set $\mathfrak{D}_{normal}$ is difficult using an exhaustive search due to the combinatoric nature of the problem. In the insider threat domain, we are particularly interested in whether the n_α points in $\mathfrak{D}_\alpha$ match known anomalies in benchmark datasets, so the capability of algorithms to identify the correct points in $\mathfrak{D}_\alpha$ is also evaluated.

Thus, the main difference from the standard consensus clustering problem is the discovery of datasets $\mathfrak{D}_\alpha$ and $\mathfrak{D}_{normal}$ in addition to finding a set of consensus cluster labels. To the best of our knowledge, this is a new problem definition that has never been addressed.

A. *Sequential: Anomaly Detection then Consensus Clustering*

The Sequential method performs anomaly detection to identify anomalous data points and uses the remaining non-anomalous data points to determine consensus cluster labels. The advantage of this approach is the flexibility of using any existing anomaly detection method appropriate for categorical data and any existing consensus clustering approach. In the current work, a straightforward anomaly detector is used to look for the n_α data points consisting of the least frequently occurring combinations of cluster labels across views. Other forms of anomaly detection for categorical features are possible and are being tested as part of ongoing work.

Once these n_α data points are removed, consensus cluster labels are obtained using existing algorithms. In particular, the EM-based approach described in [12] is applied. (The graph-based clustering approach MCLA [11] was also tested, but we found the EM-based method performed better on our benchmark datasets and do not include these additional results).

B. *Greedy Iterative Removal*

The Greedy Iterative Removal method iterates between selection of anomalies and consensus clustering. On each iteration, a single additional data point is declared as anomalous and cannot be non-anomalous in subsequent iterations. In this method, the anomaly detector is tied to the objective function used in consensus clustering, but any existing consensus clustering algorithm can be used. As in the previous method, the EM method described in [12] is used for consensus clustering.

A more detailed description of this approach and the objective function is as follows. Strehl et al. describes an information theoretic objective function for measuring the quality of consensus cluster labels based on normalized mutual information [11]. Let $\mathcal{X}$ and $\mathcal{Y}$ be random variables, $I(\mathcal{X},\mathcal{Y})$ be the mutual information between $\mathcal{X}$ and $\mathcal{Y}$, and $H(\mathcal{X})$ be the entropy of $\mathcal{X}$. Then, the normalized mutual information $NMI(\mathcal{X},\mathcal{Y})$ between $\mathcal{X}$ and $\mathcal{Y}$ is defined as:

$$NMI(\mathcal{X},\mathcal{Y}) = \frac{I(\mathcal{X},\mathcal{Y})}{\sqrt{H(\mathcal{X}),H(Y)}}$$

In the case of consensus clustering, $\mathcal{X}$ and $\mathcal{Y}$ encode the probability that a particular point is in a particular cluster. For example, in a particular set of cluster labels C with k clusters over n_d data points, the random variable corresponding to C consists of k probabilities, where the probability that a point belongs to a particular cluster is estimated using maximum likelihood (i.e., the number of data points in C with that cluster label divided by the total number of data points n_d).

The objective function is then the average normalized mutual information between the consensus cluster labels C_L and each column of C_E. Intuitively, the consensus cluster labels that optimize this objective function are the labels that, on average, tend to group the same data points together in both the overall consensus clustering as well as the clusters in each view.

One can perform consensus clustering and anomaly detection in concert in a greedy manner as follows:

```
1. Initialization: $\mathfrak{D}_\alpha$ is an empty set, $\mathfrak{D}_{normal}$ contains all data points
2. Iterate until $|\mathfrak{D}_\alpha| = n_\alpha$, where $n_\alpha$ is a user-supplied input on the number of anomalies to find
   a. For all points $\boldsymbol{x_i}$ in $\mathfrak{D}_{normal}$:
      i. Remove $\boldsymbol{x_i}$ from $\mathfrak{D}_{normal}$; let this set be $\mathfrak{D}_{normal}'$
      ii. Perform consensus cluster labeling using all data points in $\mathfrak{D}_{normal}'$
      iii. Find the value of objective function $\boldsymbol{f}(C_L(\mathfrak{D}_{normal}'), C_E(\mathfrak{D}_{normal}'))$
      iv. Let this objective function value be $o_i$
   b. Remove data point $\boldsymbol{x_i}$ with lowest value $o_i$ from $\mathfrak{D}_{normal}$; add this data point to $\mathfrak{D}_\alpha$
```

Not surprisingly, this approach can be slow, particularly if the number of data points is large and the time complexity of the component consensus clustering algorithm is computationally expensive. Methods for speeding up this algorithm will be considered in future work.

C. EM: Multinomial Plus Uniform Distribution

For the following proposed EM methods, anomaly detection occurs within each EM iteration and hence, one cannot arbitrarily choose any anomaly detector or consensus clustering algorithm. At each iteration, a set of n_α points is flagged as anomalous. Unlike the Greedy Iterative Removal method, the set of anomalous data points can change as the EM process converges.

Topchy et al. presents a method of using EM to estimate a mixture of k multinomial distributions that will enable creation of a set of consensus cluster labels (i.e., the ith data point belongs to the jth cluster in the consensus cluster labeling if the ith data point has the highest probability of belonging to the jth multinomial distribution) [12]. An interesting advantage that can be used when estimating parameters of a mixture model with EM is that the components of the mixture model can be completely different probability distributions. That is, one does not need to constrain all the components to be modeled by multinomial distributions. This idea is used in a number of places, such as in [19], which describes methods called Bregman bubble clustering.

In particular, one can combine ideas from [12] and [19] as follows. Instead of estimating a mixture of k multinomials as in the algorithm in [12], the goal is to estimate parameters for k multinomials and a $(k+1)^{\text{th}}$ component consisting of a uniform distribution. As motivated in [19], the n_α points with the highest probability of belonging to the $(k+1)^{\text{th}}$ distribution can be regarded as noise or anomalies, and points with the highest probability of belonging to the remaining k multinomials can be considered as points belonging to clusters describing normal behavior. Note that this is the only method that estimates the value n_α, while the other methods take n_α as an input parameter.

D. EM: Multinomial Plus Unknown Distribution

A drawback of the previous approach is that it imposes a uniform probability distribution on the $(k+1)^{\text{th}}$ component, which represents the anomalous data points. A uniform distribution may or may not be an appropriate model. In contrast to the previous method, to assign data points to the $(k+1)^{\text{th}}$ component, this EM method selects the n_α data points that fit the remaining k multinomial components most poorly. These n_α worst-fitting data points correspond to the data points with the highest entropy calculated over the probabilities of belonging to the k multinomial components. Specifically, this EM method reassigns the n_α worst-fitting points to the $(k+1)^{\text{th}}$ component during the M-step (rather than calculating fit to a uniform distribution). There is nothing to be estimated for the $(k+1)^{\text{th}}$ component during the E step, as the prior of the $(k+1)^{\text{th}}$ component is fixed to n_α/n_d, and n_α is a user-supplied input parameter.

E. Discussion

Note that all of our approaches inherit a number of flaws from the approaches on which they are based. We plan to address these and other short-comings in future work. In particular, the number of clusters (i.e., the number of typical groups of behavior) needs to be specified in all proposed approaches. Methods to automatically determine the number of clusters in consensus clustering have been proposed. Integrating these was not an initial priority because, in our experiments, even if the incorrect number of clusters in the data is provided to the algorithm, the anomalous data points can still be found.

All but one of the methods has a parameter that can control n_α, the number of data points considered to be anomalous, while the remaining method (EM with uniform distribution) automatically discovers n_α. For insider threat, both of these approaches are potentially reasonable, and the choice of approach depends specifically on the characteristics of the environment in which the data is being collected. In our experiments, we limit the number of data points to be flagged as anomalous to a relatively small number (i.e., 5) in order to simulate an environment where the number of false positives needs to be low.

V. Experimentation

In order to test the proposed methods, data with known anomalies is needed. Two sets of multi-faceted graph data were created that can be processed in the Core-Facets framework. The graph data is created using an extension to recent work in generating realistically structured graphs. Although initial work in graph generation focused on relatively small graphs [20], a series of newer work [21, 22] has examined issues related to graph generation and simulation of large-scale graphs with similar properties to extremely large, real datasets. In particular, [21] describes a recursive graph generator named "RTG" that is compared using eleven metrics to two real datasets.

RTG is limited to creating either unipartite or bipartite single-view graphs. Moreover, RTG may produce several disconnected subgraphs, rather than a single connected component. In order to create multi-view data with more than two types of nodes, a modified version of the RTG algorithm was created; this version of RTG is run several times for different node types, and all generated graphs are merged into a single core graph. This modified version of the RTG algorithm includes the capability to remove duplicate edges and self-referencing edges, and also includes the ability to connect all nodes to the largest connected subgraph using preferential attachment.

In each of the two datasets created for this experiment, there are four facets. Each facet contains two groups of nodes; in particular, there are a large number of edges connecting nodes from the same group, and a small number of edges connecting nodes from different groups. Thus, in all facet graphs, there exist nodes in the two different groups that are connected to nodes in the other group, meaning that separation of the nodes into the correct groups is not trivial and requires a good graph clustering algorithm. Also note that because of the RTG generator, there is a hierarchical cluster structure within each group of nodes (i.e., subclusters within a cluster). This artifact may cause a clustering algorithm to identify the subclusters rather than a single cluster, depending on the parameters specified to the algorithm. Finally, there is one anomalous node that is highly linked to one group in two of the facets and to the other group in the remaining two facets. In total, there are 6340 nodes per facet and an average of 8155.5 edges per facet in the first dataset, and 3550 nodes per facet and an average of 5667.75 edges per facet in the second dataset.

The GraClus algorithm [10] was used to cluster each facet graph. In particular, since the number of appropriate clusters may not be known in practice, each facet was clustered into 2, 3, 4, 5, and 6 clusters, resulting in 20 sets of cluster labels as input to the algorithms (i.e., C_E contains 20 columns). As mentioned, the number of clusters to create for the consensus cluster labels is also uncertain. Thus, the experiments use a range of 2, 3, 5, and 10 clusters. Empirically, the results show that even if the number of target clusters does not match the number of clusters in the actual data (i.e., when the number of consensus clusters is greater than 2) that the anomalous data points can still be found using certain methods.

A. Evaluation Metrics

Since the proposed methods rely on EM, which is known to be non-deterministic due to randomization during initialization, our results are averaged over five runs. To evaluate the quality of consensus cluster labels, the averaged NMI evaluation metric from [11] is used. Note that past studies using non-deterministic consensus clustering methods would often run their algorithm multiple times and simply keep the method with the highest average NMI. However, we found that methods with higher average NMI did not necessarily perform better at finding outliers. Thus, the results were averaged instead of simply taking the result with highest average NMI.

Since the main goal of this paper is to find unusual insider activity, we are primarily concerned with finding the correct anomalies in our datasets. Thus, the proposed methods were compared by evaluating the percentage of runs where the proposed method was able to find the known anomaly within the top n_α anomalous data points, where n_α is set to 5 for methods that require n_α as input. This is identical to the recall metric used in information retrieval.

The results also report the precision of each algorithm, which is the percentage of data points that are actually anomalous out of all points labeled as anomalous. In our datasets, only a single anomaly was injected, so a precision of 0.2 is the highest possible precision when n_α is equal to 5. When n_α is given and the number of true positives is fixed to two possible values (one if the anomaly is found, zero otherwise), the precision is redundant given the recall. However, precision is necessary to examine for those methods which automatically determine n_α in order to ensure these methods do not generate too many false positives. In particular, results show that the method that automatically determines n_α (EM with uniform distribution) tends to be less precise than the best-performing method.

B. Results

Results for the graph data are shown in Tables 1 and 2. The Greedy Iterative Removal method is always able to find the known anomalous node in the top 5 outliers, and is the only method that is able to do so in all cases. In particular, the Greedy Iterative Removal method is successful even if the number of consensus clusters (under table column "# of CCs") is different than the number of clusters in the actual dataset.

Table 1: Results for graph dataset 1 (Note that due to the experimental setup, the max possible precision for all methods other than EM: uniform is 0.20.)

Algorithm	# of CCs	Average NMI	Average Precision	Average Recall
Sequential	2	0.995	0.00	0.00
	3	0.899	0.00	0.00
	5	0.738	0.00	0.00
	10	0.639	0.00	0.00
Greedy Iterative Removal	2	0.996	0.20	1.00
	3	0.847	0.20	1.00
	5	0.710	0.20	1.00
	10	0.642	0.20	1.00
EM: uniform	2	0.987	0.07	1.00
	3	0.843	0.09	1.00
	5	0.712	0.06	0.60
	10	0.637	0.00	0.00
EM: unknown	2	0.996	0.20	1.00
	3	0.853	0.00	0.00
	5	0.728	0.00	0.00
	10	0.627	0.00	0.00

Table 2: Results for graph dataset 2 (Note that due to the experimental setup, the max possible precision for all methods other than EM: uniform is 0.20.)

Algorithm	# of CCs	Average NMI	Average Precision	Average Recall
Sequential	2	0.991	0.00	0.00
	3	0.872	0.00	0.00
	5	0.776	0.00	0.00
	10	0.623	0.00	0.00
Greedy Iterative Removal	2	0.994	0.20	1.00
	3	0.910	0.20	1.00
	5	0.731	0.20	1.00
	10	0.647	0.20	1.00
EM: uniform	2	0.982	0.08	1.00
	3	0.867	0.10	1.00
	5	0.771	0.53	0.80
	10	0.649	0.20	0.20
EM: unknown	2	0.993	0.20	1.00
	3	0.876	0.00	0.00
	5	0.811	0.00	0.00
	10	0.635	0.00	0.00

The EM with uniform distribution method can work well, but doesn't always catch the known anomalous node (i.e., less than perfect recall) and tends to have poorer precision than the greedy method. Interestingly, the EM method that fits a mixture of multinomials and a single unknown distribution does not seem particularly robust and works well only when the number of consensus cluster labels matches the actual number of clusters in the dataset. Finally, note that the Sequential method, which attempts to do anomaly detection separately from consensus clustering, never finds the correct anomalies. Thus, even if our primary goal is anomaly detection, it is beneficial to couple characterization of normal and anomalous points when looking for unusual combinations of activity.

It is interesting to note that the methods which result in highest averaged NMI do not necessarily result in the best precision and recall for finding anomalous data points. Given the relatively small percentage of anomalies (i.e., one data point out of thousands of data points), it is quite possible to achieve very high averaged NMI but find none of the anomalies, which happens for the Sequential method in many cases. Note that the Greedy Iterative Removal method does result in competitive averaged NMI with the methods that produce the best averaged NMI, and so seems to be the overall best method on the two datasets used in our benchmark.

VI. Summary

This paper presents methods of finding potentially malicious insider activity across multiple data sources. Specifically, the objective of the methods is to find unusual combinations of otherwise normal activity, where the activity of the insider is indistinguishable from activities of normal users when examining a single source of data, but the combination of activities across data sources is unusual. These multiple data sources can be captured as multiple facets in a Core-Facets framework.

Experiments show that simply keeping a histogram of group memberships across facets and looking for unusual combinations of group memberships (i.e., the Sequential method) cannot find anomalous nodes. Instead, one should characterize normal behavior in the data while finding anomalous data points using extensions to existing methods for consensus clustering, even if the primary goal is to find anomalous data points. In our test datasets, the Greedy Iterative Removal method performed well at finding the anomalous insider activity. Future work will focus on further testing this approach, particularly on additional datasets, as well as improving the efficiency and capabilities of the proposed approaches. In particular, the ability of the methods to rank anomalies in addition to identifying a point as anomalous should be considered.

References

[1] D. N. Lam, A. Liu, and C. Martin, Graph-Based Data Warehousing Using the Core-Facets Model, 11th Industrial Conference on Data Mining, New York, NY, 2011.

[2] M. A. Maloof and G. D. Stephens, ELICIT: A System for Detecting Insiders Who Violate Need-to-know, in *Recent Advances in Intrusion Detection, Proceedings of the 10th International Symposium (RAID 2007)*, vol. 4637, *LNCS*, C. Kruegel, R. Lippmann, and A. Clark, Eds. Gold Coast, Australia, 5-7 September: Springer, 2007, pp. 146-166.

[3] V. H. Berk, G. Cybenko, I. G.-d. Souza, and J. P. Murphy, Managing Malicious Insider Risk through BANDIT, *Hawaii International Conference on System Sciences*, 2012.

[4] V. Chandola, A. Banerjee, and V. Kumar, Anomaly detection: A survey, *ACM Computing Surveys*, vol. 41, 2009

[5] L. Akoglu, M. McGlohon, and C. Faloutsos, OddBall: Spotting Anomalies in Weighted Graphs, in *14th Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Hyderabad, India, 2010.

[6] J. Gao, F. Liang, W. Fan, C. Wang, Y. Sun, and J. Han, On Community Outliers and their Efficient Detection in Information Networks, in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Washington, D.C.: ACM, 2010, pp. 813-822.

[7] W. Eberle, L. Holder, and D. Cook, Identifying Threats Using Graph-Based Anomaly Detection, in *Machine Learning in Cyber Trust: Security, Privacy, and Reliability*, J. J. P. Tsai and P. S. Yu, Eds. New York, NY: Springer, 2009, pp. 73-107.

[8] D. Greene and P. Cunningham, Multi-view clustering for mining heterogeneous social network data, in *31st European Conference on Information Retrieval, Workshop on Information Retrieval over Social Networks*, vol. 5478, *LNCS: Information Systems and Applications*, M. Boughanem, C. Berrut, J. Mothe, and C. Soule-Dupuy, Eds. Toulouse, France, 2009.

[9] G. Karypis and V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, *SIAM Journal on Scientific Computing*, vol. 20, pp. 359-392, 1999.

[10] I. Dhillon, Y. Guan, and B. Kulis, Weighted Graph Cuts without Eigenvectors: A Multilevel Approach, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 1944-1957, 2007.

[11] A. Strehl and J. Ghosh, Cluster Ensembles - A Knowledge Reuse Framework for Combining Multiple Partitions, *Journal of Machine Learning Research*, vol. 3, pp. 583--617, 2002.

[12] A. Topchy, A. K. Jain, and W. Punch, Clustering ensembles: models of consensus and weak partitions, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1866-1881, 2005.

[13] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, 2nd Intl Conf on Knowledge Discovery and Data Mining, 1996.

[14] C. Aggarwal and H. Wang, Managing and Mining Graph Data, in *Advances in Database Systems*, vol. 40, A. Elmagarmid and A. P. Sheth, Eds.: Springer, 2010, pp. 600.

[15] L. Tang, H. Liu, and J. Zhang, Identifying evolving groups in dynamic multi-mode networks, *IEEE Transactions on Knowledge and Data Engineering*, 2010.

[16] D. Cai, Z. Shao, X. He, X. Yan, and J. Han, Community mining from multi-relational networks the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases, Portugal, 2005.

[17] L. Tang and H. Liu, Uncovering cross-dimension group structures in multi-dimensional networks, in *SIAM Intl Conf on Data Mining, Workshop on Analysis of Dynamic Networks*. Sparks, NV, 2009.

[18] N. Martínez-Bazan, V. Muntés-Mulero, S. Gómez-Villamor, J. Nin, M.-A. Sánchez-Martínez, and J.-L. Larriba-Pey, DEX: High-performance exploration on large graphs for information retrieval, 16th ACM Conf on Information and Knowledge Management, Portugal, 2007.

[19] G. Gupta and J. Ghosh, Bregman bubble clustering: A robust framework for mining dense clusters, *ACM Trans. Knowl. Discov. Data*, vol. 2, pp. 8:1-8:49, 2008.

[20] D. Chakrabarti and C. Faloutsos, Graph mining: Laws, generators, and algorithms., *ACM Computing Survey*, vol. 38, 2006.

[21] L. Akoglu, M. McGlohon, and C. Faloutsos, RTM: Laws and a recursive generator for weighted time-evolving graphs, in *ICDM*, 2008.

[22] E. Even-Bar, M. Kearns, and S. Suri, A network formation game for bipartite exchange economies, in *SODA*, 2007.