Journal of Applied Security Research, 4:501–517, 2009
Copyright © Taylor & Francis Group, LLC
ISSN: 1936-1610 print / 1936-1629 online
DOI: 10.1080/19361610903176310

TECHNOLOGY

Graph-Based Analysis of Nuclear Smuggling Data

DIANE COOK and LAWRENCE HOLDER
Washington State University, Pullman, Washington, USA
SANDY THOMPSON, PAUL WHITNEY, and LAWRENCE CHILTON
Pacific Northwest National Laboratory, Richland, Washington, USA

Much of the data that is collected and analyzed today is structural, consisting not only of entities but also of relationships between the entities. As a result, analysis applications rely on automated structural data mining approaches to find patterns and concepts of interest. This ability to analyze structural data has become a particular challenge in many security-related domains. In these domains, focusing on the relationships between entities in the data is critical to detect important underlying patterns. In this study we apply structural data mining techniques to automate analysis of nuclear smuggling data. In particular, we choose to model the data as a graph and use graph-based relational learning to identify patterns and concepts of interest in the data. In this article, we identify the analysis questions that are of importance to security analysts and describe the knowledge representation and data mining approach that we adopt for this challenge. We analyze the results using the Russian nuclear smuggling event database.

KEYWORDS *Nuclear smuggling, data mining, graph representation, pattern discovery*

This work was supported by the United States National Nuclear Security Administration's Office of Nonproliferation Research and Development and conducted via a partnership between Washington State University and the U.S. Department of Energy's Pacific Northwest National Laboratory. The laboratory is operated by Battelle Memorial Institute for the U.S. Department of Energy under Contract DE-AC-05-76RLO-1830.

Address correspondence to Diane J. Cook, School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164-2752, USA. E-mail: cook@eecs.wsu.edu

INTRODUCTION

The success of machine learning and data mining for business and scientific purposes has fueled the expansion of its scope to new representations and techniques. Originally, data modeling and mining focused on data that could be represented as a vector of feature values. However, much collected data is structural in nature. Structural data contains entities as well as relationships between these entities. Compelling data in bioinformatics [28], network intrusion detection [15], Web analysis [1, 7], and social network analysis [5, 24] have become available that require effective handling of structural data.

The ability to learn structural concepts from data has also become a crucial challenge in many security-related domains. For example, the U.S. House and Senate Intelligence Committees' report on their inquiry into the activities of the intelligence community before and after the September 11, 2001 terrorist attacks revealed the necessity for "connecting the dots" [26], that is, focusing on the relationships between entities in the data, rather than merely on an entity's features or attributes. The ability to discover relationship-driven patterns can impact our ability to prevent future attacks and ensure national security.

Data mining and information technology is cited as a critical tool for ensuring the safety of our country [21]. In response to the government's call for science and technology to aid in the war against terror, researchers have found ways of using artificial intelligence technologies to address this challenge [2]. These projects include recognition of terrorist activity [8, 12, 17] and detection of outliers that may indicate suspicious behavior [16].

In this article, we focus on one particularly challenging area of security: nuclear smuggling and export control. The U.S. government has initiated a number of efforts to curtail nuclear smuggling including the State Department's Export Control and Related Border Security Assistance Program [25] and the NNSA's International Nonproliferation Export Control Program [20]. Data relevant to export control and weapons of mass destruction (WMD) proliferation can be acquired at a much higher rate than can be analyzed. Analysts need new and automated tools to deal with this glut of complex real-world data. The end goal of this research is to predict intent of smugglers in order to prevent terrorist events and the spread of WMD.

The goal of this project is to utilize artificial intelligence techniques to model and analyze nuclear smuggling data. We hypothesize that by analyzing the features and the structure of nuclear smuggling data, we can find patterns that consistently appear in nuclear smuggling events and can find interesting links between event components. We anticipate that the automated discovery of such patterns will aid analysts in understanding the nature of smuggling events. This outcome will be useful in more quickly and effectively detecting future smuggling events and in answering the real question faced by analysts: intent.

The outcome of this study will be the answers to four questions:

- Are there recurring structural patterns in nuclear smuggling data that can be discovered by an automated structural analysis algorithm?
- Can an automated analysis algorithm infer whether a link exists between a pair of smuggling events?
- Are there recurring structural patterns that occur in linked smuggling events that can be automatically discovered and that provide insights on the nature of smuggling strategies?
- Are there individual components that play a central role throughout the entire database (i.e., that act as hubs or authorities for nuclear smuggling activities)?

First, we will describe the database we use for this study and our planned approach for automated analysis. We will then describe the results of the study as applied to an actual nuclear smuggling database.

NUCLEAR SMUGGLING DATA

During the last two decades, there have been numerous reports of nuclear materials smuggling worldwide. Several databases have been created to document reported smuggling incidents [34]. We restrict our analysis to data that has been made publicly available. Analyzing nuclear smuggling incidents requires analyzing the structure of the incidents. Many nuclear materials can be exported for beneficial uses as well as for the creation of weapons, so called dual-use materials. For example, Americium 241 and Plutonium can serve as an alpha-particle source for smoke detectors as well as a component of nuclear weapons. Thus it is necessary to examine the context of the shipment, the path that it will take, and connections between export shipments and organizations involved in the export to detect illicit smuggling attempts.

In order to determine whether data mining algorithms can provide an automated analysis of nuclear smuggling data, we select the Nuclear Smuggling dataset as a testbed. The Nuclear Smuggling dataset consists of reports on Russian nuclear materials smuggling [18]. The database originally appeared as an index to a paper by Williams and Woessner [29] that investigated the dynamics of nuclear material smuggling in terms of suppliers, smuggling organizations, and end users. The information in the chronology is based on open-source reporting, primarily World News Connection (WNC) and Lexis-Nexis [27]. There are also some articles obtained from various sources that have been translated from Italian, German, and Russian. The research from which the chronology grew began in 1994 and the chronology itself first appeared as an appendix to a paper by Williams and Woessner [29] and later with an updated chronology [30]. DARPA's Evidence Extraction and Link Discovery (EELD) program updated the chronology to include incidents through March 2000.

In the Nuclear Smuggling database, each incident, or event, is described by a set of features. Links between events and event components are also included in the database. The database contains 45 relational tables. Each table fits into one of the following categories:

- Event objects. The event table (EV_EVENT) contains the basic information about each smuggling event.
- Entity objects. The entity tables contain information about the event location (E_LOCATION), material that was being smuggled (E_MATERIAL), the organization responsible (E_ORGANIZATION), persons involved in the incident (E_PERSON), the source of the material (E_SOURCE), and any weapon found on the person (E_WEAPON).
- Link objects. These tables explicitly represent a connection between events (LK_EVENT_EVENT), between entities (LK_LOCATION_LOCATION, LK_MATERIAL_LOCATION, LK_MATERIAL_MATERIAL, LK_MATERIAL_ORGANIZATION, LK_MATERIAL_WEAPON, LK_ORG_ORG, LK_ORGANIZATION_LOCATION, LK_ORGANIZATION_WEAPON, LK_PERSON_LOC, LK_PERSON_MATERIAL, LK_PERSON_OCC, LK_PERSON_ORG, LK_PERSON_PERSON, LK_PERSON_WEAPON, LK_WEAPON_LOCATION, LK_WEAPON_WEAPON), and between events and entities (LK_EVENT_MATERIAL, LK_EVENT_ORGANIZATION, LK_EVENT_PERSON, LK_EVENT_SOURCE, LK_EVENT_WEAPON).
- Feature encodings. These tables provide numeric encodings of feature values (L_CLASSIFICATIONS, L_CONCEALMENT, L_CONFIDENCE, L_COUNTRIES, L_EELD_COMPONENT, L_EVENTS, L_GENDERS, L_MATERIALS, L_METHODS, L_MOTIVES, L_OCCUPATIONS, L_ORGANIZATIONS, L_PLACES, L_RELATIONS, L_SOURCE_TYPES, L_WEAPONS).

The number of entries in each table varies from as few as 2–3 elements to as many as 800. There are 302 events described in the database, with almost 2,500 links defined between events and event components. As is indicated by the larger number of links, the database is highly structural and contains an abundance of information for each event. For this reason, a graph is a natural representation for the data. After representing the relational information as a graph, a graph-based relational learning algorithm can be used to discover recurring patterns and learn concepts from the data. Alternative representations would be to model the data and relationships as logical expressions. The data could then be analyzed using techniques such as Inductive Logic Programming, as was considered by Mooney et al. [19]. Because graph algorithms exist to perform the types of analyses we have targeted, we decided to employ a graph representation for this data.

GRAPH-BASED RELATIONAL LEARNING

There are a number of data mining algorithms that could be useful in analyzing this data. Because we want to identify prevalent patterns in the data, supervised learning algorithms will not be sufficient for this task. In addition, the data is inherently structural. Thus we need a structural discovery, or graph-based relational learning, approach.

Graph-based data mining is the task of finding novel, useful, and understandable graph-theoretic patterns in a graph representation of data. Several approaches to graph-based data mining identify frequently occurring subgraphs in graph transactions, that is, those subgraphs meeting a minimum level of support [13, 14, 22, 31].

We distinguish graph-based relational learning (GBRL) from graph-based data mining in that GBRL focuses on identifying novel, but not necessarily most frequent, patterns in a graph representation of data [11]. Only a few GBRL approaches have been developed to date. Two specific approaches, Subdue [3] and GBI [32], take a greedy approach to finding subgraphs maximizing an information-theoretic measure. Subdue searches the space of subgraphs by extending candidate subgraphs by one edge. Each candidate is evaluated using a minimum description length (MDL) metric [23], which measures how well the subgraph compresses the input graph if each instance of the subgraph were replaced by a single vertex. GBI continually compresses the input graph by identifying frequent triples of vertices, some of which may represent previously compressed portions of the input graph. Candidate triples are evaluated using a measure similar to information gain. Kernel-based methods have also been used for supervised GBRL [9]. Because we want to be able to perform unsupervised discovery as well as supervised learning on our structural data, and because we want to be able to easily interpret the results, we use the Subdue algorithm for this analysis.

The Subdue algorithm [3] encompasses several approaches to graph-based learning, including discovery, clustering, and supervised learning. Subdue uses a labeled graph $G = (V,E,L)$ as both input and output, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices, $E = \{(v_i, v_j) \mid v_i, v_j \in V\}$ is a set of edges, and L is a set of labels that can appear on vertices and edges. The graph G can contain directed edges, undirected edges, self-edges (i.e., $(v_i; v_i) \in E$), and multi-edges (i.e., more than one edge between vertices v_i and v_j). The input graph G need not be connected, but the learned patterns must be connected subgraphs (called substructures) of G.

Unsupervised Discovery

Inputs to Subdue's discovery algorithm include the input graph (or a set of graphs), the beam length, and a limit on the total number of substructures considered by the algorithm. Subdue searches for a substructure that best compresses the input graph. A substructure in Subdue consists of a subgraph definition and all its occurrences throughout the graph. The initial state of the search is the set of substructures consisting of all uniquely labeled vertices. The only operator of the search is the *ExtendSubstructure* operator, which extends a substructure in all possible ways by a single edge or an edge and a neighboring vertex.

Subdue uses a beam search to identify candidate substructure concepts by applying the *ExtendSubstructure* operator to each substructure in the current state. The substructures are kept on a beam-limited queue and are ordered based on their description length (sometimes referred to as value) as calculated using the MDL principle.

The search terminates upon reaching a user-specified limit on the number of substructures extended, or upon exhaustion of the search space. Once the search terminates and Subdue returns the list of best substructures found, the graph can be compressed using the best substructure. The compression procedure replaces all instances of the substructure in the input graph by single vertices, which represent the substructure definition. Incoming and outgoing edges to and from the replaced instances will point to, or originate from the new vertex that represents the instance. The Subdue algorithm can be invoked again on this compressed graph. This procedure can be repeated a user-specified number of times, and is referred to as an *iteration*.

Subdue's search is guided by the minimum description length (MDL) [26] principle. The evaluation heuristic based on the MDL principle assumes that the best substructure is the one that minimizes the description length of the input graph when compressed by the substructure. The description length of the substructure S given the input graph G is calculated as $DL(S) + DL(G|S)$, where $DL(S)$ is the description length of the substructure, and $DL(G|S)$ is the description length of the input graph compressed by the substructure. Description length is calculated as the number of bits in a minimal encoding of the graph. Subdue seeks a substructure S that maximizes compression, calculated as

$$\text{Compression} = \frac{DL(G)}{DL(S) + DL(G|S)}.$$

As an example, Figure 1 shows the four instances that Subdue discovers of a pattern S_1 in the example input graph and the resulting compressed graph, as well as the pattern S_2 found in this new

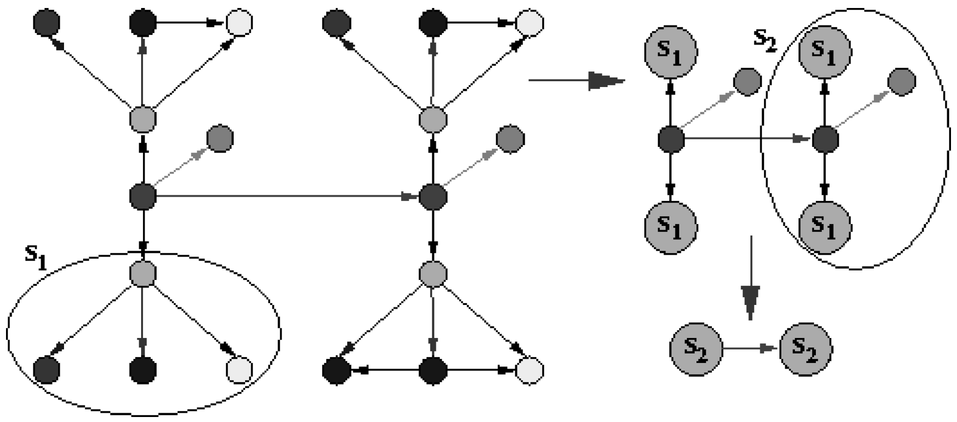

FIGURE 1 Example of Subdue's unsupervised discovery algorithm. A repetitive subgraph (S_1) is identified and used to compress the graph. New discoveries (in this case, pattern S_2) are made in subsequent iterations of the algorithm.

graph and the resulting compressed graph. To allow slight variations between instances of a discovered pattern (as is the case in Figure 1), Subdue applies an error-tolerant graph match between the substructure definition and potential instances.

Supervised Learning from Training Graphs

Extending a graph-based discovery algorithm to perform supervised learning [11] introduces the need to handle negative examples (focusing on the two-class scenario). The negative information can come in two forms. First, the data may be in the form of numerous small graphs, or graph transactions, each labeled either positive or negative. Second, data may be composed of two large graphs: one positive and one negative.

The first scenario is closest to the standard supervised learning problem in that we have a set of clearly defined examples. Figure 2 depicts a set of positive (G^+) and negative (G^-) examples. One approach to supervised learning is to find a subgraph that appears in many positive graphs, but in few negative graphs. This amounts to replacing the compression-based measure with an error-based measure. For example, we would find a subgraph S that minimizes the value

$$\frac{|\{g \in G^+ | S \not\subseteq g\}| + |g \in G^- | S \subseteq g\}|}{|G^+| + |G^-|} = \frac{FN + FP}{P + N},$$

where $S \subseteq g$ means S is isomorphic to a subgraph of g (although we do not need to perform a subgraph isomorphism test during learning). The first

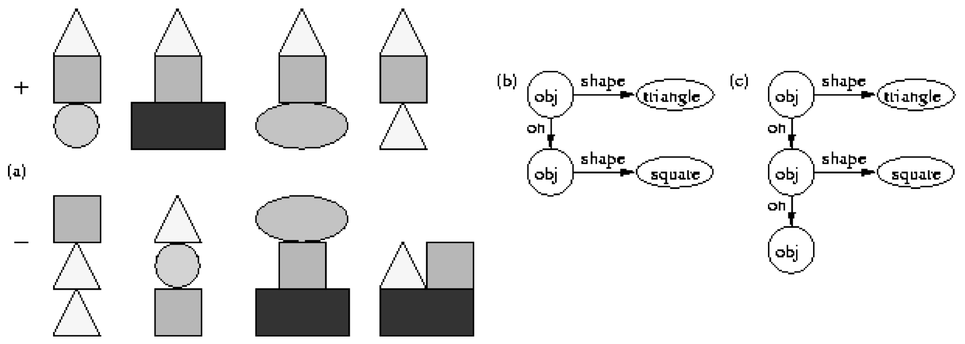

FIGURE 2 Visualization of graph-based data with (a) four positive and four negative examples, and (b, c) two possible graph concepts learned from the examples. Graphs that contain the learned concept are labeled as examples of the positive class, while other graphs are members of the negative class.

term of the numerator is the number of false negatives, and the second term is the number of false positives.

This approach will lead the search toward a small subgraph that discriminates well (e.g., the subgraph shown in Figure 2(b)). However, such a subgraph does not necessarily compress well, nor represent a characteristic description of the target concept. We can bias the search toward a characteristic description by using the compression-based measure to look for a subgraph that compresses the positive examples, but not the negative examples. If $DL(G)$ represents the description length (in bits) of the graph G, and $DL(G|S)$ represents the description length of G compressed by subgraph S, then we look for an S that minimizes $DL(G^{+}|S){+}DL(S){+}DL(G^{-}){-}DL(G^{-}|S)$, where the last two terms represent the portion of the negative graph incorrectly compressed by the subgraph. This approach will lead the search toward a larger subgraph that characterizes the positive examples, but not the negative examples (e.g., the subgraph shown in Figure 2(c)).

Finally, this process can be iterated in a set-covering approach to learn a disjunctive hypothesis. Using the error measure, any positive example containing the learned subgraph would be removed from subsequent iterations. Using the compression-based measure, instances of the learned subgraph in both the positive and negative examples (even multiple instances per example) are compressed to a single vertex.

DISCOVERING PATTERNS IN NUCLEAR SMUGGLING DATA

The first type of analysis we performed was to discover recurring structural patterns in the entire Nuclear Smuggling database. In order to apply the Subdue graph-based relational learner to this problem, we first had to

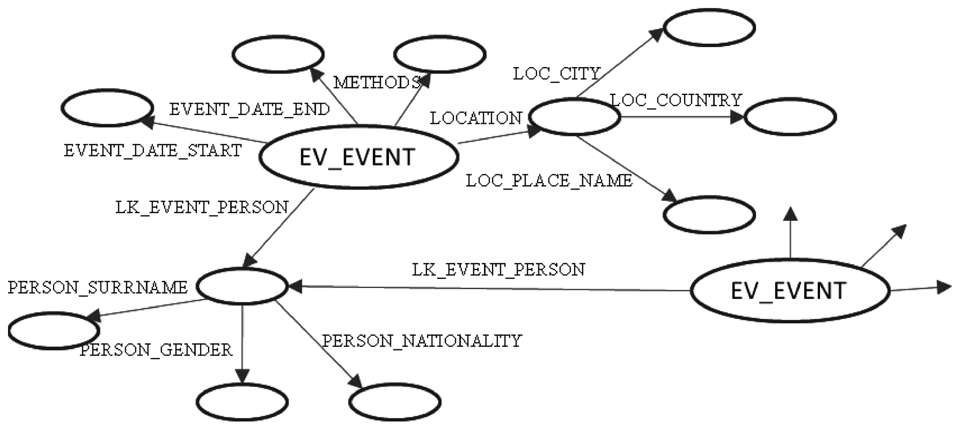

FIGURE 3 Graph representation for a pair of smuggling events. In this example, one event is described by the event start/end date, method, and location attributes. This event is connected to another event through a link to a common person that was involved in both smuggling events.

determine how to represent the information as a graph. Generally, entities are represented as nodes in a graph model, and relationships between the entities are represented as directed or undirected links between the nodes. For this dataset, entities include the smuggling events and the event objects as described earlier, while each of the relations described in the link tables corresponds to a directed link between entities in the graph. In addition, entities are described by a set of attributes. Some of these attributes are further described by sub-attributes (e.g., the attribute "location" is broken down into the sub-attributes "location place type," "location place name," "location city," "location country," and "location latitude/longitude"). In our graph representation, we included a link between each entity and its attribute values and between each attribute and its sub-attributes. Figure 3 shows a portion of the graph representation for a pair of smuggling events that are linked through a person that was involved in both events. Each event is centered at a node labeled "EV_EVENT" and is linked to the event attributes as well as possibly being linked to other events in the database. Creating a graph model of the entire set of nuclear smuggling events took less than one minute on a PC with 3 GB of memory. The resulting graph contained 25,951 vertices and 27,152 links.

To discover recurring patterns, we ran Subdue on this graph input using the default parameter settings. Subdue spent over five hours performing 200 discover-and-compress iterations on the data. The initial discoveries were not that insightful, mostly describing the infrastructure of the graph. Because the relational tables include so many relation types and sub-relation types, the top-rated substructure that was discovered contains only a network of relation type descriptions and the relationships between them, as shown in

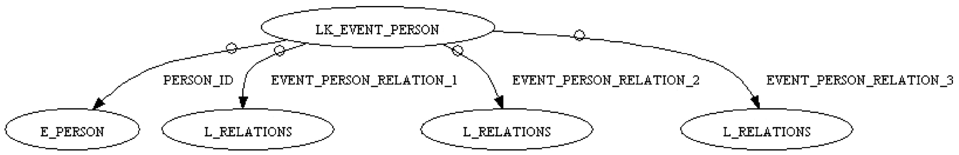

FIGURE 4 Highest-scoring substructure discovered by Subdue. This graph shows the patterns of an event described by a person involved in the event and three types of relationships between the person and the event.

Figure 4. This substructure does occur often in the database (there are 695 occurrences) and is fairly large. On the other hand, no information that is specific to any particular event is included in the discovery.

Fortunately, we can compress much of the graph's infrastructure away by using Subdue's option to replace the top substructure with a single node and repeat the discovery process. The first 50 iterations of Subdue resulted in substructures containing mostly infrastructure (links between attribute types and subtypes). After this infrastructure was compressed away, Subdue did find a number of insightful recurring patterns. For example, the pattern shown in Figure 5 showed up in 10 separate events. As the graph in Figure 5 shows, the pattern description is hierarchical—one of the nodes is labeled "SUB_1," which refers to the first substructure pattern that was discovered (shown in Figure 4). The original pattern occurrence was replaced by this single node, which became part of a larger pattern discovery. The complete pattern in Figure 5 describes features of individuals that are commonly players in a nuclear smuggling event. In particular, the person participates in a nuclear smuggling event, is a worker at a nuclear facility, and is from Romania.

Another pattern that recurred in the database is the substructure shown in Figure 6. This pattern linked together 7 different pairs of events. The events were linked because a participant or organization involved in one of the smuggling events was a rival of a participant or organization associated with another smuggling event.

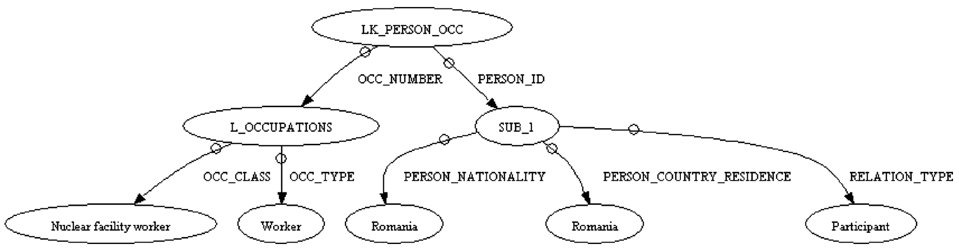

FIGURE 5 A substructure pattern discovered from the entire event database. This pattern describes features of individuals that participate in smuggling events. Features include the country of nationality and residence as Romania and occupation as a worker in a nuclear facility.

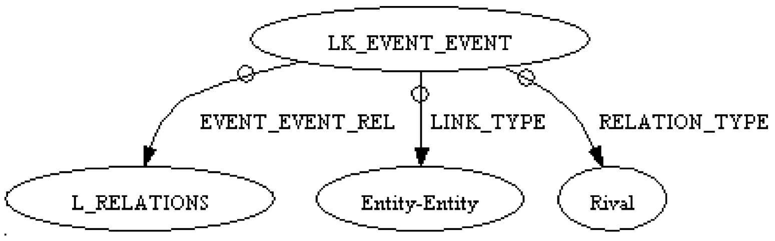

FIGURE 6 A substructure pattern discovered from the entire event database. This pattern describes a link between pairs of events involving participants that are rivals.

This second discovery was particularly interesting, because it indicated that there are connections between events and that some of these connections form common patterns. This type of structural pattern justifies our use of a graph model and a graph-based relational learner. Finding such common links between smuggling events can provide intelligence analysts with insights that aid in understanding the nature of smuggling events and in identifying individuals, organizations, locations, and materials that are likely to be part of a possible future smuggling event because of their relationship with other attempted smuggling events.

LEARNING TO RECOGNIZE LINKED EVENT PAIRS

To address our second question, whether we can learn to recognize linked event pairs, we formulated the problem as the following supervised learning problem: Given a pair of events, are they linked or not? Our goal is to learn a concept that maps pairs of events onto one of the values {Linked, Not_Linked}. In addition to learning a concept that accurately labels pairs of events, we also are interested in the concept description itself. We hypothesize that the concept description, if understandable, can provide some insights on how smuggling events are likely to be related.

In order to create sample data for our learning algorithm, we represented each pair of events that were directly linked in the database (appeared in the LK_EVENT_EVENT table) as a positive example graph for the Subdue graph-based relational learning algorithm. In order to not unduly bias the algorithm, we removed the explicit link between the two events, but other links between events and non-event entities were included in the graph. This makes the learning problem more difficult than if the direct event relationship were left intact [20], because the relationship needs to be inferred by features of the events themselves and other non-direct links (e.g., a common person, location, or weapon). To generate negative examples, we created a separate example for every pair of graphs that were not directly linked in the database.

The resulting graph database contained 162 positive examples and 45,289 negative examples.

This imbalance in the number of examples creates two difficulties. First, in order to evaluate candidate concepts Subdue needs to determine if the corresponding graph pattern definition occurs in any negative examples. While this step does not require an NP-Complete subgraph isomorphism test, it does require a graph isomorphism test between the graph pattern definition and potential instances in the negative examples. Therefore, Subdue would take a very long time to complete the learning process. Second, the dominance of negative examples may unduly bias the results. In fact, the concept "all pairs are not linked" would achieve 99.6% classification accuracy, which is quite good.

To counteract this imbalance, we randomly sampled 162 negative examples from the initial collection and used this sample, together with the 162 positive examples, to form the input database. We then used 3-fold cross validation to obtain accuracy results for the learned concepts.

Subdue had a difficult time correctly labeling pairs of examples. While one pattern was discovered that was consistently associated with positive examples (linked events), the number of positive examples that contained this pattern was fairly low (on average about 30% of the positive examples contained this pattern whereas only 2% of the negative examples contained the pattern). However, no other consistent patterns were found among the remaining positive examples. The resulting average accuracy was 57%. The one pattern that performed well indicated that one of the pair of linked events had a participant who was a clear organizer of the smuggling event.

Because the explicit link between events comprising the positive example pairs was removed, Subdue was essentially trying to find a pattern in one or both events that indicated the "event would be linked to some other," and not necessarily trying to determine why this particular pair was connected. As a result, we instead turned our attention to analyzing patterns associated with known links between groups of events. This is described in the next section of the article.

DISCOVERING PATTERNS IN LINKED EVENTS

A unique aspect of this study is that we are focusing on security data that contains a distinct structural or relational component. If we can automatically discover patterns that describe the nature of these relationships, then analysts can use the information to better understand smuggling events. They can also identify suspicious activities that might indicate attempted smuggling events because of the presence of patterns that are common to linked smuggling events, people, organizations, or locations.

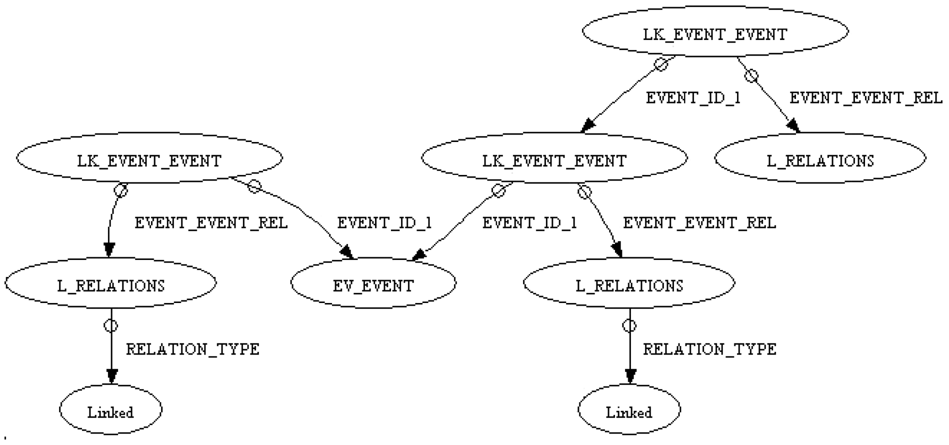

FIGURE 7 A substructure pattern discovered from the database of linked events. This pattern shows a chain of linked events.

In order to automate the analysis of linked events, we generated a third graph model of the nuclear smuggling data. This model represented in graph form the relational information for all smuggling events that contain links to one or more other smuggling events. The resulting graph contained 18,654 nodes and 19,740 edges. We ran 200 iterations of the Subdue discover/compress process on this graph and found a number of interesting results.

Figure 7 shows a type of pattern that was discovered from the database of linked events. In particular, this pattern reflects a chain of linked events. This is a particularly intriguing find, because all of the relations contained in the original database were only pair-wise relations. By applying graph analysis to a model of the data we were able to extend this pattern to an entire sequence of related events. The nature of the link between these events was not discovered in the original database (the link type is "Linked"). However, we can see that if the technique is applied to a more detailed database, the approach could discover chains such as a sequence of countries that the material was routed through as it was smuggled, a cluster of events that share a common participant person or organization, or a temporal event ordering that reflects an intended sequence of smuggled components, perhaps to be later assembled into a weapon.

FINDING IMPORTANT ENTITIES

The goal of our last task is to find the most important, or authoritative, entities in the database. We define an important entity as one that links to a large number of other important entities and to which a large number of

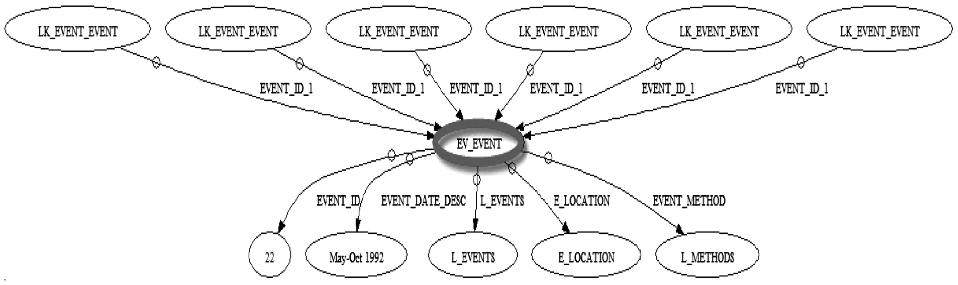

FIGURE 8 A subgraph of the dataset containing the top-ranked node (highlighted in the middle of the graph) and a portion of the neighborhood surrounding the node.

other important entities link. In this way an important entity acts both as a hub (i.e., points to data elements of importance) and as an authority (i.e., is pointed to by important data elements). The method for finding important nodes in a graph was introduced by Brin and Page [1] and was used by them to determine a Web page's relevance or importance.

For this task, we used a graph implementation of the PageRank algorithm that is included in the igraph [4] suite of graph tools. The PageRank algorithm calculates a probability distribution over nodes in the graph. Each node starts with its own estimated value of relative importance and propagates this value evenly over its outgoing links. The final probability values are calculated using an iterative method that corresponds to the principal eigenvector of the normalized graph adjacency matrix.

Once we calculated the PageRank values of all of the nodes in our graph model, we sorted them to highlight the nodes with greatest importance. Of the 25,952 nodes in the graph, 20,869 of the nodes had a value close to 0.0000. Another 4,548 nodes had a value of 0.0001, 491 nodes had a value of 0.0002, and 39 nodes had a value of 0.0003. Of greatest interest were the top four nodes in the graph. The top-ranked node had a value of 0.0005 and the next three had a value of 0.0004.

Figure 8 shows a portion of the neighborhood around the top-ranked node. The entire neighborhood (of distance one) around this node contains 40 nodes and edges, and is too big to show here. This node contains links to and from 9 other events. In addition, there are links between the node and descriptors of people, locations, and smuggled materials. Clearly, this node is a central source of information in the nuclear smuggling database. The next three nodes in the ranking also represented smuggling events and while they had fewer incoming and outgoing links than the top-ranked node, the types of links were similar in nature to those for the top-ranked node.

Analyzing the top-ranked smuggling event (#22) in detail will likely give experts insight on the smuggling strategies. In this case, event 22 describes the theft of 100 kg of uranium in 1992 from the Chepetskiy Metallurgical

Combine in Glazov, Russia. This event is linked to five other events, four of which have a “rival” relationship to event 22 and describe the seizure of some of this uranium. Four other events link to event 22 and describe deals or transfers related to this uranium. From these related events additional events are linked that describe how the remainder of the uranium was eventually sold, distributed, or seized. We see here that the theft of a significant amount of nuclear material can result in the separation of the material into smaller amounts that are then distributed in varied routes, generating a network of nuclear smuggling events.

This analysis shows that a graph representation is useful for capturing the structure of security data such as nuclear smuggling events. In addition, existing AI-based graph algorithms such as Subdue and PageRank are effective at identifying nodes, patterns, and concepts of interest for understanding nuclear smuggling activity.

CONCLUSIONS

Preventing the proliferation of nuclear materials and weapons is an important challenge to national and global security. Like many security problems, nuclear non-proliferation involves a vast network of people, places, organizations, and materials. Tools for assisting analysts in their investigations in these areas must be able to reason not only about the attributes of entities involved in events, but also the relationships between these entities. In this work we have explored the investigation of nuclear smuggling events using a network, or graph-based, perspective. Specifically, we have shown the benefits of using graph-based pattern learning to identify characteristic patterns of nuclear smuggling events and distinguishing patterns between linked and non-linked events. These patterns indicated the types of facilities, locations, and people involved in nuclear smuggling events, as well as the typical presence of an organizer behind linked events. A graph-based approach to this problem also allows the application of network analysis tools for identifying entities central to the problem. In this case we found certain events involving the theft of a large amount of nuclear material initiate a series of events for dividing and selling this material.

The information discovered from our analysis gives insight into the techniques and routes used to smuggle nuclear material, which can then be used to better protect existing materials and better respond to new events. This analysis also shows that other security-related problems may benefit from such a graph-based approach. As a next step for this research, we will address the question of whether structural analysis of the nuclear smuggling database identifies patterns that enable analysts to predict intent when presented with a new event. This is the long-term goal of this research project.

In addition to the analyses shown here, numerous other graph-based analysis tools can be applied to the nuclear smuggling data and similar problems. Graph-based anomaly detection [6] can detect small, unexpected deviations to normative patterns in a graph that may indicate an attempt to hide illicit behavior by mimicking normal patterns. Dynamic graph mining [33] looks for patterns in how a graph changes or evolves over time and can be used to detect emerging criminal networks or predict how smuggling routes might change over time. These and other graph-based techniques hold promise for analyzing security data and will be pursued in the future.

REFERENCES

1. S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, *30*, 107–117, 1998.
2. H. Chen and F.-Y. Wang. Artificial intelligence for homeland security. *IEEE Intelligent Systems*, *20*(5), 12–16, 2005.
3. D. Cook and L. Holder. Graph-based data mining. *IEEE Intelligent Systems*, *15*(2), 32–41, 2000.
4. G. Csárdi and T. Nepusz. The igraph software package for complex network research. *Proceedings of the International Conference on Complex Systems*, 2006.
5. P. Domingos and M. Richardson. Mining the network value of customers. *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, 57–66, 2001.
6. W. Eberle and L. Holder. Anomaly detection in data represented as graphs. *Intelligent Data Analysis*, *11*(6), 663–689, 2007.
7. G. W. Flake, S. Lawrence, C. L. Giles, and F. Coetzee. Self-organization of the web and identification of communities. *IEEE Computer*, *35*(3), 66–71, 2000.
8. T. Fu and H. Chen. Analysis of cyberactivism: A case study of online Free Tibet activities. *Proceedings of the IEEE International Conference on Intelligence and Security Informatics*, 2008.
9. T. Gartner. Kernel methods for mining graph data. In *Mining Graph Data* (D. Cook & L. Holder, eds.), Wiley, 2006.
10. J. Gonzalez, L. B. Holder, and D. Cook. Graph-based relational concept learning. *Proceedings of the International Conference on Machine Learning*, 219–226, 2002.
11. L. Holder and D. Cook. Graph-based relational learning: Current and future directions. *ACM SIGKDD Explorations*, *5*(1), 90–93, 2003.
12. L. Holder, D. Cook, J. Coble, and M. Mukherjee. Graph-based relational learning with application to security. *Fundamenta Informaticae, Special Issue on Mining Graphs, Trees, and Sequences*, *66*(1-2), 83–101, 2005.
13. A. Inokuchi, T. Washio, and H. Motoda. Complete mining of frequent patterns from graphs: Mining graph data. *Machine Learning*, *50*(3), 321–354, 2003.
14. M. Kuramochi and G. Karypis. Finding frequent patterns in a large sparse graph. *Data Mining and Knowledge Discovery*, 2005.
15. W. Lee, S. Stolfo, and K. Mok. A data mining framework for building intrusion detection models. *Proceedings of the IEEE Symposium on Security and Privacy*, 120–132, 1998.

16. S. Lin and H. Chalupsky. Discovering and explaining abnormal nodes in semantic graphs. *IEEE Transactions on Knowledge and Data Engineering*, *20*(8), 1039–1052, 2007.
17. H.-M. Lu, D. Zeng, and H. Chen. Bioterrorism event detection based on the Markov switching model: A simulated anthrax outbreak study. *Proceedings of the IEEE International Conference on Intelligence and Security Informatics*, 76–81, 2008.
18. S. J. McKay, P. N. Woessner, and T. J. Roule. Evidence extraction and link discovery (EELD) seedling projects, database schema description, version 1.0. Technical Report 2862, Veridian Systems Division, August 2001.
19. R. Mooney, P. Melville, L. Tang, J. Shavlik, I. Dutra, D. Page, and V. Costa. Relational data mining with inductive logic programming for link discovery. *Proceedings of the NSF Workshop on Next Generation Data Mining*, 2002.
20. National Nuclear Security Administration. Nuclear nonproliferation. http://nnsa.energy.gov/nuclear_nonproliferation/.
21. National Research Council. *Making the Nation Safer: The Role of Science and Technology in Countering Terrorism*. National Academy Press, 2002.
22. S. Nijssen and J. Kok. A quickstart in frequent structure mining can made a difference. *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, 647–652, 2004.
23. J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scientific, 1989.
24. M. F. Schwartz and D. C. M. Wood. Discovering shared interests using graph analysis. *Communications of the ACM*, *36*(8), 78–89, 1993.
25. U.S. Department of State. The EXBS Program: Export control and related border security assistance. http://www.state.gov/t/isn/export/ecc/20779.htm.
26. U.S. Senate and H.C. on Intelligence. Joint inquiry into intelligence community activities before and after the terrorist attacks of September 11, 2001, December, 2002.
27. L. R. Tang, R. J. Mooney, and P. Melville. Scaling up ILP to large examples: Results on link discovery for counter-terrorism. *Proceedings of the KDD Workshop on Multi-Relational Data Mining*, pp. 107–121, 2003.
28. J. T. L. Wang, M. J. Zaki, H. T. T. Toivonen, and D. Shasha. *Data Mining in Bioinformatics*, Springer, 2004.
29. P. Williams and P. N. Woessner. Nuclear material trafficking: An interim assessment. *Transnational Organized Crime*, *1*(2), 206–238, 1995.
30. P. N. Woessner. Chronology of radioactive and nuclear materials smuggling incidents: July 1991–June 1997. *Transnational Organized Crime*, *3*(1), 114–209, 1997.
31. X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. *Proceedings of the International Conference on Data Mining*, 2002.
32. K. Yoshida, H. Motoda, and N. Indurkhya. Graph-based induction as a unified learning framework. *Journal of Applied Intelligence*, *4*, 297–328, 1994.
33. C. You, L. Holder, and D. Cook. Graph-based data mining in dynamic networks: Empirical comparison of compression-based and frequency-based subgraph mining. *IEEE International Conference on Data Mining (ICDM) Workshop on Analysis of Dynamic Networks*, 2008.
34. L. Zaitseva and K. Hand. Nuclear smuggling chains. *American Behavioral Scientist*, *46*(6), 822–844, 2003.