Graph-based Data Mining on Social Networks

Maitrayee Mukherjee
Department of Computer Science and Engineering
University of Texas at Arlington
Box 19015, Arlington, TX 76019

mukherje@cse.uta.edu

Lawrence B. Holder
Department of Computer Science and Engineering
University of Texas at Arlington
Box 19015, Arlington, TX 76019

holder@cse.uta.edu

ABSTRACT
In this research, we compare and contrast the salient features of illicit group information with legitimate group data. We describe how the graph-based knowledge discovery system, SUBDUE, when run in unsupervised discovery mode, finds structural patterns embedded within social network data. We also illustrate how SUBDUE, in supervised mode, learns distinguishing patterns between legitimate and covert groups, based only on the communication activities of the group members.

Keywords
Graphs, Social Networks

1. INTRODUCTION
Data Mining has emerged as a novel field of research and has valuable applications in the real world. In the early days, data was predominantly stored in the relational format, where each row represented an instance of a class or maybe one transaction. However, the advent of data stored in HTML/XML texts, ordered/unordered trees, symbolic sequences etc. provided impetus to the study of data mining on semi-structured data [11]. Research on data mining and machine learning of symbolic sequences [10] and ordered tree structures [20] also gained momentum. Multi-relational data mining whose main scope is to find patterns in expressive logical and relational languages from complex, multi-relational and structured data has also picked up greatly [12].

The need for mining structured data was apparent to the research community and one such approach focused on the topological view of data structures. Since the graph has a generic topological structure and is one of the most thoroughly researched data structures in Computer Science and Discrete Mathematics, state-of-the-art techniques in **graph-based data mining (GDM)** have had profound influence. GDM has tremendous utility because graph-structured data occur widely in practical fields like biology, chemistry, material science and communication networking [16].

KDD'04, August 22–25, 2004, Seattle, WA, USA.

Identifying illicit groups has become an important challenge for homeland security. Relationships (e.g., communications) among members of legitimate and illicit groups are another domain that can be easily represented as a graph. We accept the overhead of converting existing datasets to graphs and then do graph-based data mining on them to discover interesting and novel concepts. The world is drowning in data and security analysts face the stiff challenge of sifting through a plethora of data and zeroing in on the few suspicious bits of information. Hence, effective techniques of **social network analysis (SNA)** are critical to alleviate the problem of information overload. This is the main motivation of this paper. The communication patterns that we are looking for in this research are not big graphs, involving lots of actors, but small ones, that discriminate well between threat and non-threat groups and give us a lead on which networks to put under scrutiny. These kinds of signature patterns do not require knowledge of the entire terrorist network; hence the patterns that we found (described later) may be tested on data with varied levels of "observability". In the absence of real data, we have trained on well-researched, simulated data, with appropriate levels of observability, corruption and noise.

We focus on applying GDM to SNA, where participating actors are represented as vertices and communication links between actors as edges (see Figure 8). We brief the reader on social networks, SNA, graph-based representational techniques and how social networks of legitimate groups differ from those of illicit ones. We review the theoretical bases of GDM: a paradigm, of which the graph-based knowledge discovery system, SUBDUE [2, 3], is an implementation. We discuss two of the key data mining techniques implemented in SUBDUE: unsupervised pattern discovery and supervised concept learning. We illustrate, with examples and experiments, how we identify patterns within social network data represented as labeled graphs with the help of SUBDUE. We bring forth the concepts that SUBDUE learns to discriminate legitimate groups from illicit ones, based only on the communication patterns of the group members. Finally, we conclude and discuss avenues for future research.

2. SOCIAL NETWORKS
2.1 Social Network data
Social Network data focuses on the representation of actors and the relationships (or ties) between them. This kind of data, instead of revolving around a single actor, highlights the dynamics between groups of actors. This data is a shade different from conventional data, in databases and spreadsheets, which focuses primarily on individual actors and their attributes [14, 17].

2.2 Social Network Analysis (SNA)

SNA is the mapping and measuring of relationships and flows between people, groups, organizations or other information/knowledge processing entities. SNA uses attributed relational graphs (ARGs) where vertices represent people, organizations or any other object, edges represent relationships between objects and attributes store the details of each vertex and edge [14, 17].

2.3 Representing Social Networks

Formal methods are necessary to represent social networks because of their ability to depict huge datasets succinctly and systematically. Graphs and matrices are the most popular techniques because they lay down their own protocols which help us depict with ease and lead us to discern patterns in our data which would not have been possible to describe with words. They also allow the dull and repetitive workload to be shared by computers.

Both graphs and matrices have their own pros and cons. For this research, we have chosen graphs to represent our social networks for their ease of visualization and also because a graph can be easily represented as an adjacency matrix if required. Sociologists borrowed relevant graphical concepts from mathematicians and coined these graphics as "**sociograms**" [5].

Sociograms may be classified based on the different levels of measurements of the ties [5], as described below:-

A **binary** sociogram is one where the only thing that matters is whether a tie is present. An arrow represents the existence of a tie whereas its absence signifies no relationship whatsoever.

A **signed** sociogram uses signs on its ties; a plus (+) on the arrow indicates a positive choice whereas a minus (-) indicates its opposite.

A **valued** sociogram takes this concept a bit further by allowing us to put a measure of the strength of the relationship on the arrow.

Sociograms may again be classified based on the kinds of ties between actors [5], as described below:-

A **simplex** sociogram is one which depicts a single kind of relation between actors.

In a **multiplex** sociogram, there are multiple kinds of ties between actors.

In our research, we have opted for binary, simplex, "labeled" sociograms to depict communication patterns between actors. Figure 8 is an apt example.

3. CONTRASTING THREAT GROUP DATA WITH LEGITIMATE NETWORK DATA

Covert networks remain mingled with socially-oriented networks (like families, organizations etc.) in the real world. The buzz word for covert networks is "secrecy" and hence to discover such networks (technically, to discern distinctive patterns in the activities and communications of such illegitimate groups) can be very tricky and often misleading due to unavailability of authentic data or in some cases availability of "doctored" data. This issue has especially blown up in the recent past and after the September 11, 2001 tragedy, it has been in the limelight so much so that it is worthwhile to take a close look at the distinguishing properties of such networks.

(1) Social network graphs depicting covert threat groups may be incomplete due to missing actors (vertices) and links (edges) that investigators may fail to uncover [8].

(2) The difficulty in deciding who to include and who not to include is not a problem in legitimate networks like families and organizations. But in terrorist networks this may be a problem due to the secrecy maintained by the entire network of people. This is referred to as the "Fuzzy Boundary" problem [8].

(3) In legitimate networks, actors who are highly central are typically the most important ones. On the contrary, peripheral players (or "boundary spanners" as they are typically called) may be huge resources to a terrorist group although they receive very low network centrality scores. This is because they are well-positioned to be innovators, since they have access to ideas and information flowing in other clusters. Similarly, in an organization, these peripheral employees are in a position to combine different ideas and knowledge into new products and services. They may be contractors or vendors who have their own network outside of the company, making them very important resources for fresh information not available inside the company [5, 8].

(4) Terrorist networks are not static as new members are added to fortify the group and members are removed / killed / captured / compromised which shrinks the group size [8]. Most organizational networks are also dynamic but this is hardly the case with families.

(5) Covert networks trade efficiency for secrecy. A strategy for keeping cell members distant from each other, and from other cells, minimizes damage to the network if a cell member is captured or otherwise compromised [8]. Hence the shortest path in the social network graph is not usually the path taken for communication.

(6) Relationships between members belonging to a terrorist network and those not belonging to terrorist networks are rare and infrequent. Terrorists seldom make friends outside the trusted circle because eliminating boundary-spanning ties reduces the visibility into the network, and chances of leaks out of the network [8].

In the transcript (U.S. Department of Defense, 2001) Osama bin Laden's comment [8] about the September 11, 2001 attack is:

"Those who were trained to fly didn't know the others. One group of people did not know the other group."

Strong ties between previous contacts, which were formed years ago in school, college dormitories and training camps, keep the members linked. Yet, unlike normal social networks, these strong ties remain mostly dormant and therefore hidden to outsiders. In a covert network, because of their low frequency of activation, strong ties may appear to be weak ties. The less active the network, the more difficult it is to discover [8]. Legitimate networks, on the other hand, have a lot of incoming and outgoing ties as one might expect.

(7) Despite the need for secrecy, covert networks have goals to accomplish. Cell members must optimize between stealth and the need for intense task-based communication. It is during these periods of heightened activity, that these networks may shorten and become susceptible to discovery [8].

(8) More often than not, only one of the terrorists (may not be the group leader always) would speak for the whole group. Choosing somebody other than the leader to be the speaker makes sense just in case the leader is exposed. So we may find a multicast/broadcast from one particular person to the entire group or maybe to terrorists who spearhead other groups. These multicasts are often noticeable during terrorist activities and make the group vulnerable to discovery.

(9) The role of a "broker" [8] is a very powerful role in a social network as it ties two hitherto unconnected constituencies/groups together but of course, it is a single-point of failure. These broker-type roles are often seen in terrorist networks. Such nodes are also referred to as "cutpoints" [5].

We are aware of graph-based technologies being used for intelligence analysis by a research team at 21st Century Technologies in Austin, TX [1]. Graph-based algorithms enable security analysts to extract from a deluge of information a small subset that has suspicious characteristics. In this research, intelligence analysis relies on the fact that legitimate and illegitimate groups tend to exhibit different SNA metric values like geodesics, redundancy, small world phenomena, betweenness centrality, closeness centrality, clustering coefficient, node degree, diameter, girth etc. These metrics, combined with statistical pattern classification, arm the analyst with a tool for automatically pinpointing threatening group activities within a huge body of evidence.

Our approach is different in that we believe legitimate groups have different communication styles from illegitimate groups. Hence our research is based solely on the communication evidence of group members of social networks, without considering the network metric values and statistical patterns. We train on well-investigated, simulated intelligence analysis data to find discriminating patterns between threat and non-threat groups.

4. GRAPH-BASED DATA MINING (GDM)

4.1 Theoretical Bases of GDM

We briefly review the five theoretical bases of GDM [16].

(1) Sub-graph Categories: Sub-graphs are categorized into various classes (namely general sub-graphs, induced sub-graphs, connected sub-graphs, ordered trees, unordered trees and paths) and the approaches of graph-based data mining strongly depend on the targeted class.

(2) Sub-graph Isomorphism: Sub-graph isomorphism is the mathematical basis of substructure matching and/or counting.

(3) Graph Invariants: Graph invariants are the quantities (like the number of vertices, the degree of each vertex and the number of cyclic loops) to characterize the topological structure of a graph and they help to efficiently reduce the search space of the targeted graph structures. If two graphs are topologically identical, i.e., isomorphic, they also have identical graph invariants, though the reverse property does not hold.

(4) Mining Measures: These are various measures, similar to those in conventional data mining, to mine substructures in the graph, whose selection depends on the objective and the constraints of the mining approach. Some popular mining measures are support, information entropy, information gain, gini-index and minimum description length (MDL).

(5) Solution Methods: Approximately five types of search methods are used to solve the sub-graph isomorphism problem amidst a number of graphs. They are categorized into (1) heuristic search methods and (2) complete search methods, in terms of the completeness of the search. They are also classified under (1) direct and (2) indirect matching methods, in terms of the sub-graph isomorphism matching problem. The five types of search methods are: (1) conventional greedy search [3, 19], (2) inductive logic programming [13], (3) inductive database [6], (4) complete level-wise search and (5) support vector machine (SVM) [15].

4.2 Why we chose SUBDUE

Several approaches to GDM exist based on the task of identifying frequently occurring sub-graphs in graph transactions, i.e., those sub-graphs meeting a minimum level of support.

The FSG system (part of the PAFI system, University of Minnesota) [9] finds all frequent sub-graphs in large graph databases. FSG starts by finding all frequent single and double edge sub-graphs. Then, in each iteration, it generates candidate sub-graphs by expanding the sub-graphs found in the previous iteration by one edge. In each iteration the algorithm checks how many times the candidate sub-graph occurs within an entire graph. The candidates, whose frequency is below a user-defined level, are pruned. The algorithm returns all sub-graphs occurring more frequently than the given level.

gSpan (University of Illinois at Urbana-Champaign) [18] combines depth-first search and lexicographic ordering to find frequent sub-graphs. Their algorithm starts from all frequent one-edge graphs. The labels on these edges together with labels on incident vertices define a code for every such graph. Expansion of these one-edge graphs maps them to longer codes. Since every graph can map to many codes, the codes in the tree structure are not unique. If there are two codes in the code tree that map to the same graph and one is smaller than the other, the branch with the smaller code is pruned during the depth-first search traversal of the code tree. Only the minimum code uniquely defines the graph. Code ordering and pruning reduces the cost of matching frequent sub-graphs in gSpan.

The Apriori-based Graph Mining (AGM) system [7] searches the space of frequent sub-graphs in a bottom-up fashion, beginning with a single vertex, and then continually expanding by a single vertex and one or more edges. AGM also employs a canonical coding of graphs in order to support fast sub-graph matching. AGM returns association rules satisfying user-specified levels of support and confidence.

Few graph-based relational learning (GBRL) approaches have been developed to date. Two specific tools, SUBDUE and GBI [19], take a greedy approach to finding sub-graphs maximizing an information theoretic measure. SUBDUE searches the space of sub-graphs by extending candidate sub-graphs by one edge. Each candidate is evaluated using a minimum description length metric (discussed later), which measures how well the sub-graph

compresses the input graph if each instance of the sub-graph 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.

To summarize, we chose SUBDUE mainly due to two reasons. Firstly, we see that all the three approaches just mentioned: FSG, gSpan and AGM, find frequent substructures in a graph data set. SUBDUE, on the other hand, finds sub-graphs that compress the input data set well but which may not be frequent. We believe sub-graphs that compress well may reveal novel concepts in a database, whereas frequent sub-graphs may not always be interesting.

Secondly, all these three systems are based on graph transactions. SUBDUE does support graph transactions but on top of that, can also mine one large graph for interesting patterns.

4.3 Brief Introduction to SUBDUE

(http://ailab.uta.edu/subdue)

The Knowledge Discovery System SUBDUE [2] has been a pioneering work in the field of greedy search-based graph mining (Point 5, Solution Methods in Section 4.1). Structural data, represented as a labeled graph, serves as the input to SUBDUE, which writes out substructures, again as labeled graphs. A **substructure** is a connected sub-graph within the input graph (Point 1, Sub-graph Categories in Section 4.1). The found sub-graph can be considered a **concept**. An **instance** of a substructure in an input graph is a set of vertices and edges from the input graph, which match, graph-theoretically, to the graphical representation of the substructure. This algorithm is based on a computationally-constrained **beam** search.

For each unique vertex label, a substructure is defined as a vertex with that label and whose instances are all the vertices in the input graph with that label. A substructure is extended in all possible ways by a single edge and a vertex, or by only a single edge if both vertices are already in the sub-graph. At each expansion, candidate substructures are rated according to one of the three evaluation techniques, given below. The substructures are kept on a queue and are ordered based on their values, defined below. The search terminates upon reaching a user-specified **limit** on the number of substructures extended, or upon exhaustion of the search space. SUBDUE has been applied successfully to databases in domains like Image analysis, CAD circuit analysis, Chinese character databases, program source code, chemical reaction chains etc.

The value of a substructure S, in an input graph G, denoted by value(S, G), may be calculated according to any one of the following three evaluation techniques:-

(1) Minimum Description Length (MDL)

This is the default evaluation method used by SUBDUE (see Point 4, Mining Measures in Section 4.1). The MDL principle states that the best theory to describe a dataset is the one that minimizes the description length of the entire set of data. SUBDUE has implemented this principle in the context of compressing the input graph with the discovered substructure.

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 in the new vertex that represents the instance. The SUBDUE algorithm may be invoked again (termed **iteration**) on this compressed graph.

Here, value(S, G) = DL(G) / (DL(S) + DL(G|S)), where DL(G) is the number of bits (description length) required to encode the input graph G, DL(S) is the number of bits required to encode the discovered substructure S and DL(G|S) is the number of bits required to encode the input graph G after it has been compressed using substructure S.

In supervised concept learning (see Section 4.5), value(S, Gp, Gn) = [DL(Gp) + DL(Gn)] / [DL(S) + DL(Gp|S) + DL(Gn) - DL(Gn|S)], where Gp and Gn are the positive and negative graphs respectively.

(2) Size

The size measure is faster to compute than the MDL measure but is less consistent.

Here, value(S, G) = size(G) / (size(S) + size(G|S)), where size(G) = (#vertices(G) + #edge(G)), and (G|S) is G compressed with S.

In supervised concept learning (see Section 4.5), value(S, Gp, Gn)

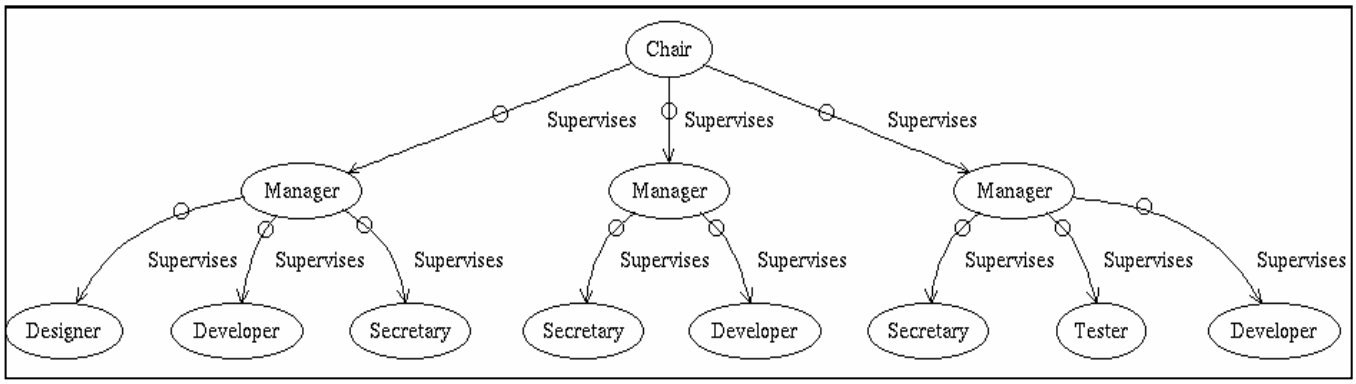

Figure 1. An organization chart showing the company hierarchy (produced using AT&T Graphviz).

= [size(Gp) + size(Gn)] / [size(S) + size(Gp|S) + size(Gn) - size(Gn|S)].

(3) Set Cover

The value of a substructure S is computed as the number of positive examples containing S plus the number of negative examples not containing S, this quantity divided by the total number of examples.

If this evaluation method is chosen, then the compression done at the end of each iteration in the MDL approach is replaced by just removing all positive examples containing S. The SUBDUE algorithm may be invoked again (i.e., iterated) on this reduced graph.

In our experiments, we have mostly relied on the MDL and the set cover approaches. The size measure has not been used in our research. Among the key features implemented in SUBDUE, we shall be applying the unsupervised pattern discovery and supervised concept learning techniques to this research.

4.4 Unsupervised Pattern Discovery

Knowledge is discovered in structural data by identifying common substructures (concepts represented as graphs) within the data (see Section 4.3) [2].

Figure 1 shows the hierarchy of an organization, which after being converted to a labeled graph, is fed to SUBDUE. Figure 2 shows the best substructure reported by SUBDUE in unsupervised discovery mode using the MDL evaluation technique discussed in Section 4.3. This substructure has been reported to have three instances in the input graph, which implies that, every manager, in the organization shown in Figure 1, supervises at least a developer and a secretary.

In the input graph in Figure 1, we see that a manager in this company may also supervise a designer or a tester but since it is not a pattern reported by SUBDUE, we suspect that some developers in this company are made to multiplex as designers or testers according to managerial wishes.

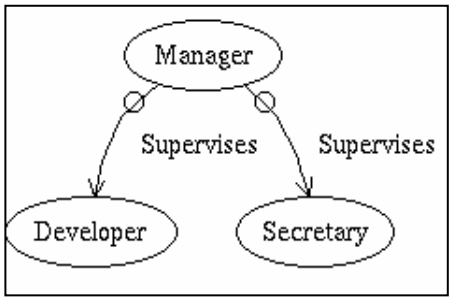

Figure 2. Best substructure reported by SUBDUE in Unsupervised Discovery Mode.

4.5 Supervised Concept Learning

SUBDUE has been extended to incorporate supervised graph-based concept learning [2, 4], which focuses on the two-class scenario. The inclusion of a negative graph enables SUBDUE to learn by example rather than by observation. Substructures that occur often in the positive graph, but not often in the negative graph, are more likely to represent the target concept.

The negative information may come in two forms. First, the data may be in the form of 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 3 shows a set of positive examples, denoted by G^+ and Figure 4 shows a set of negative examples, denoted by G^-.

One approach to supervised learning, namely the set-covering approach, is to find a sub-graph that appears often in the positive graphs, but not in the negative graphs. This amounts to replacing the information-theoretic measure with simply an error-based measure. This approach will lead the search towards a small sub-graph that discriminates well. However, such a sub-graph does not necessarily compress well, nor represent a characteristic description of the target concept. We can bias the search towards a more characteristic description by using the information-theoretic measure to look for a sub-graph that compresses the positive examples, but not the negative examples. This is the MDL approach and will lead the search towards a larger sub-graph that characterizes the positive examples, but not the negative examples.

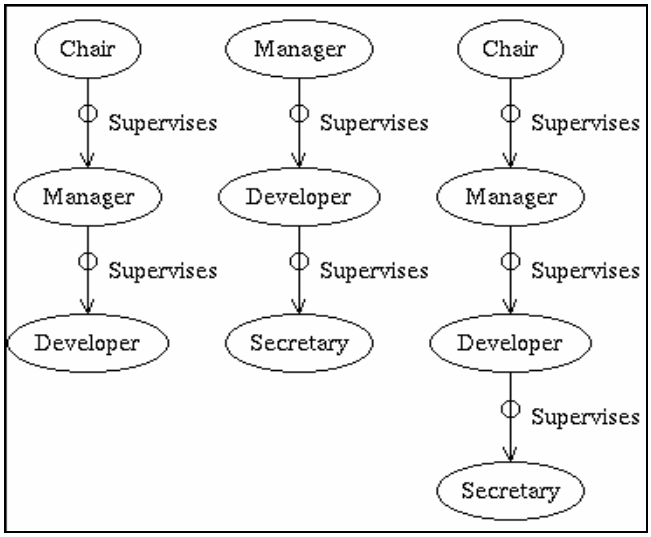

Figure 3. Set of positive examples, denoted by G^+, for supervised learning

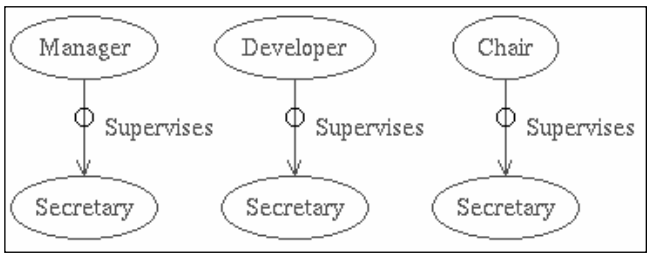

Figure 4. Set of negative examples, denoted by G^-, for supervised learning

In our example, the best substructure reported by SUBDUE using both these approaches is incidentally the same and is shown in Figure 5. This "Manager supervises Developer" pattern is present in all the positive examples in Figure 3 but not once in the set of negative examples in Figure 4. This sub-graph discriminates best as well as compresses the best.

Finally, this process can be iterated in a set-covering approach to learn a disjunctive hypothesis. If using the error measure, then any positive example containing the learned sub-graph would be removed from subsequent iterations. If using the information-theoretic measure, then instances of the learned sub-graph in both the positive and negative examples (even multiple instances per example) are compressed to a single vertex. Please see [4] for more information on graph-based supervised learning.

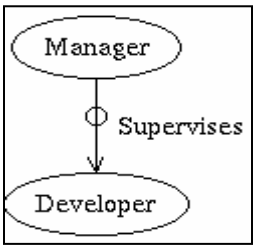

Figure 5. Best substructure learned by SUBDUE in Supervised Concept Learning mode

5. LEARNING STRUCTURAL PATTERNS EMBEDDED IN SOCIAL NETWORKS

5.1 The concept of a clique

Structural analysts are very interested in substructures, like dyads, triads and ego-centered groups, which may be present in the network. The solidarity and connections of large social structures can be built up out of such small, tight components in a bottom-up approach.

A **clique** is a maximal complete sub-graph expanded to include as many actors as possible, in which every actor has a direct tie with each and every other member [5].

5.2 Relaxations of the clique definition

N-clique – Since the strict definition of a clique may be too strong to capture the meaning of the concept, it may be relaxed a bit to include an actor as a member of a clique if he/she is connected to every other member of a group at a distance greater than one. Usually the path distance of two is used. This corresponds to the "**Friend of a friend (FoaF)**" concept. This approach to defining substructures is called N-clique, where N stands for the length of the path allowed to make a connection to all other members. The problem with the N-clique approach is that it tends to find long and stringy groupings rather than the tight and discrete ones of the maximal approach. Sometimes, N-cliques can be found with a property that is undesirable for many purposes - the members of the N-clique get connected by actors who are not, themselves, members of the clique [5].

N-clan – An N-clan is a restricted form of N-clique which forces all ties between members of an N-clique to occur by way of the other members of the N-clique. This sometimes has an important bearing on sociological data [5].

K-plexes – The strong assumptions of a "maximal complete sub-graph" may also be relaxed by allowing actors to be members of cliques even if they have ties to all but K other members [5].

K-cores – A K-core is a maximal group of actors, all of whom are connected to K other members of the group. To be included in a K-plex, an actor must be tied to all but K other actors in the group. The K-core approach is more relaxed, allowing actors to join the group if they are connected to K members, regardless of how many other members they may not be connected to.

5.3 Unsupervised Discovery of K-plexes

We shall be concentrating on K-plexes for our experiments with the belief that if SUBDUE can successfully find K-plex structures, it can be used to mine for other such salient patterns embedded in social network data.

We introduce a couple of different classes of K-plex graphs:-

1) **KPlex_Inexact** - A class of K-plex graphs where each instance of the K-plex does not have the same structural pattern embedded.

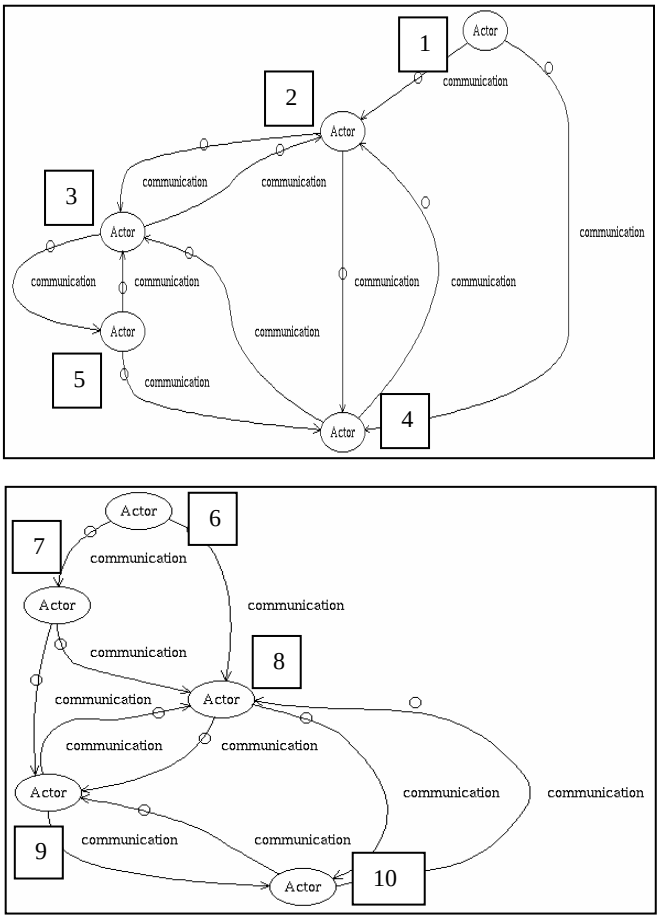

Figure 6. A KPlex_Inexact communication graph showing two instances of a 2-plex (Instance 1 on top, Instance 2 in bottom)

(Vertex Label: "Actor" and Edge Label: "Communication")

2) **KPlex_Exact** - A class of K-plex graphs that has the same structural pattern embedded in all the instances of the K-plexes.

Each class may of course have a number of sub-classes.

5.3.1 Unsupervised Discovery of KPlex_Inexact

Communication graphs of KPlex_Inexact will be more common in the real world where there is hardly any structure.

Actor 1 communicates with all (Actors 2 and 4) but two members (Actors 3 and 5) in Instance 1 (top graph in Figure 6); similarly for the other actors, thus making it a 2-plex. Again in Instance 2 (bottom graph in Figure 6), Actor 6 communicates with all (Actors 7 and 8) but 2 members (Actors 9 and 10); similarly for the other actors, thus making it another 2-plex. A self-connection is not considered a valid connection. This graph is a member of class KPlex_Inexact since each instance has different structural patterns, which means that Actor 6 in Instance 2 has different connections from Actor 1 in Instance 1, Actor 7 from Actor 2… and Actor 10 from Actor 5.

When the communication graph shown in Figure 6 was fed to SUBDUE, the best substructure, shown in Figure 7, was reported. Since the input graph is a member of KPlex_Inexact, SUBDUE, in its unsupervised discovery mode, finds neither the 2-plex of

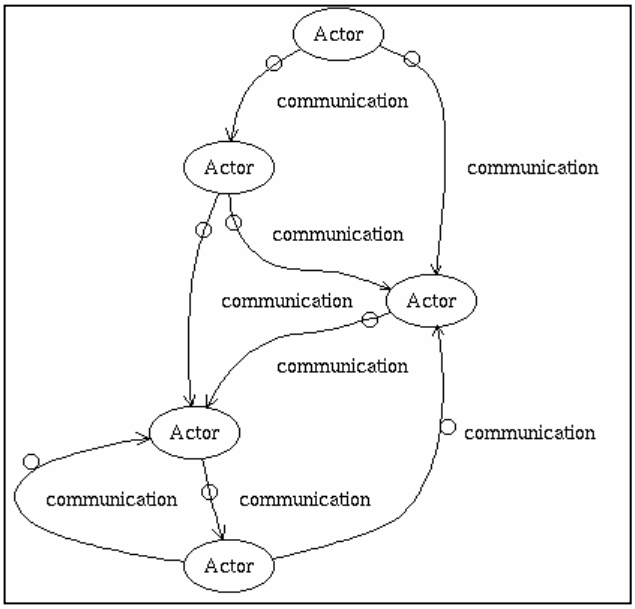

Figure 7. Best substructure reported by SUBDUE from the KPlex_Inexact communication graph shown in Figure 6.

Instance 1 nor the 2-plex of Instance 2 but is able to find a common sub-graph of both. The significance of this result may not be so obvious in the trivial example graph that we have used for demonstration purposes, but in huge datasets the discovered concept may be of great value.

This shows that SUBDUE is able to discover possibly interesting patterns/concepts in social network data.

5.3.2 Unsupervised Discovery of KPlex_Exact

When a communication graph of class KPlex_Exact is input, SUBDUE is able to discover all the instances of the entire K-plex pattern embedded.

5.4 Supervised Concept Learning

K-Plex communication graphs similar to the one shown in Figure 6 have been fed as positive examples and non K-plex graphs as negative examples to SUBDUE, which then runs in supervised concept learning mode. The goal is to be able to distinguish a K-plex pattern from other varieties strewn in a big graph. If the SUBDUE concept learner is able to discern the K-plex pattern, we believe it will be able to discover other kinds of patterns also in social network data.

Much consideration has been given into choosing a negative example for this experiment because choosing any random graph may lead a negative example to be a super-graph of the positive example K-plex, which may cause SUBDUE to report not-so-discriminating substructures. Hence the negative examples have been built by removing one edge from the positive K-plex examples. We term these negative graphs (K – 1)-plexes, as they are just one edge short of a K-plex. For example, in Figure 6, the communication edge between Actors 1 and 2 may be removed in Instance 1 and the edge between Actors 6 and 7 may be removed in Instance 2, to build the negative examples for this experiment.

The SUBDUE concept learner discriminated all the instances of the entire K-plex when the MDL evaluation technique is used. However, when the set cover approach is used, discrimination is perfect just like the MDL approach but the entire K-plex structure is not discovered. This is because the set cover approach tends to discover smaller substructures that discriminate well but may not compress as well. Please see Section 4.5 for relevant information.

6. DISTINGUISHING THREAT FROM NON-THREAT GROUPS

6.1 Description of the input data

The data originates from a simulator built by Information Extraction Technologies (IET) for the Evidence Assessment, Grouping, Linking, and Evaluation (EAGLE) program. This simulator builds a domain to simulate the evidence available about terrorist groups and their plans prior to their execution. The data is output in Lisp format.

The domain consists of threat and non-threat actors, threat and non-threat groups, targets, exploitation modes, capabilities, resources, communications, visits to targets and transfer of resources between actors, groups and targets. These events involve various forms of communication and transfer of assets.

We tweaked different simulator parameters to control the number of groups, the number of transactions etc. to generate various types of datasets for our experiments.

6.2 Data Extraction and Preparation

Instead of taking all available evidence from the data written by the simulator, we attempt to learn patterns distinguishing threat groups from non-threat groups based only on the communications activities between various actors.

Intra-group communication information was isolated as separate labeled graphs in two passes of the Lisp data. The first pass scans the ground truth to record all the threat and non-threat groups while the second pass records communication events which have taken place between actors belonging to the same group.

Communication networks easily lend themselves to a graphical representation as shown in the attributed relational graph in Figure 8. Hence, we were motivated to convert the dataset to a graph and do graph-based data mining on it. Each Actor (vertex) participates as the initiator (edge) or the respondent (edge) to a communication event (vertex), the latter having an attribute TwoWayCommunication (vertex).

6.3 Why Supervised Concept Learning and not Unsupervised Discovery

Finding interesting patterns in terrorist network datasets is a boon for society since the occurrence of similar patterns in other social networks might lead us to put the latter networks under scrutiny.

But there is always the outside possibility of this interesting threatening pattern appearing in a legitimate network.

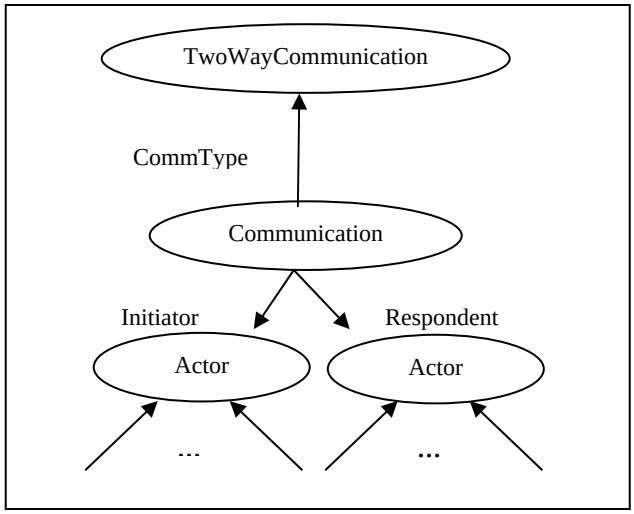

Figure 8. An attributed relational graph (ARG) to depict communication events between actors.

Hence, the goal of this research is to find patterns that can distinguish threat groups from non-threat groups using the SUBDUE supervised concept learner. Discovery of such signature patterns in other social networks will implicate the latter groups more convincingly.

6.4 Choice of Evaluation Technique

SUBDUE's default evaluation method is based on the MDL principle, which essentially says that the best pattern (or substructure) is the one that best trades off the size of the pattern and the size of the input graph after compressing away all the instances of the pattern. If there are negative graphs in the input, then the best pattern is the one that best compresses the positive graphs, but least compresses the negative graphs. This approach tends to prefer patterns that compress well, even though they may not discriminate well.

Thus, to distinguish threat from non-threat groups, we have chosen the set cover evaluation method. This method looks for patterns that discriminate well, which is the main objective in the threat group task, without regard for how well the patterns compress (see Sections 4.3 and 4.5 for relevant information).

6.5 Learning the signature pattern with SUBDUE concept learner

Each of the intra-group graphs (see Section 6.2) was fed to SUBDUE as an example, i.e., each threat-group communication graph was presented as a positive example whereas each non-threat group communication graph was presented as a negative example. The positive examples/threat-group graphs are marked by "XP" and the negative examples/non-threat group graphs are marked by "XN" in the labeled input graph fed to SUBDUE, as shown in the excerpts in Figure 9. The entry "v 1 Communication" indicates a vertex with id 1 and label "Communication" and the entry "v 2 TwoWayCommunication" indicates another vertex with id 2 and label "TwoWayCommunication". The entry "d 1 2 CommType" indicates a directed edge between vertices 1 and 2 with label "CommType". SUBDUE, when run in supervised concept learning mode, learns by example rather than by observation to find communication patterns that will distinguish threat from non-threat groups.

XP	**XN**
% UID-Group-3382 - Threat Group	% UID-Group-3106 - NonThreat Group
v 1 Communication	……..
v 2 TwoWayCommunication	d 5 3 Initiator
d 1 2 CommType	d 5 4 Respondent
v 3 Actor	v 7 Communication
v 4 Actor	v 8 TwoWayCommunication
d 1 3 Initiator	d 7 8 CommType
d 1 4 Respondent……	d 7 3 Initiator……..

Figure 9. Excerpts from the first positive and the first negative examples in the input.

Table 1 shows some statistics on the experiments that we conducted. Columns 1 and 2 show the number of threat and non-threat groups, designated as positive and negative examples respectively. These numbers are controlled by tweaking certain simulator parameters. Columns 3 and 4 are the number of vertices and edges in the input graph. The last two columns show SUBDUE's training set accuracy and its learning time for each dataset. We do not show the results for 3-fold cross-validation, because the testing and use of learned patterns requires sub-graph isomorphism and since the sub-graph isomorphism problem is NP-complete, the cost of testing became prohibitive.

6.6 Analysis of the best substructure

In this threat and non-threat group experiment, the SUBDUE concept learner discovered substructures that discriminate well between positive and negative examples and found a

Table 1. Size and performance statistics of three experimental datasets, generated by the EAGLE Simulator

Threat Groups	Non-Threat Groups	Total Vertices	Total Edges	Training Accuracy	Learning Time (seconds)
5	20	3,723	4,152	100%	17
15	85	10,447	12,351	96%	147
20	85	7,411	8,889	92%	35

communication pattern/substructure that has four instances in the positive graphs and none in the negative graph. This substructure has been visualized in Figure 10. The value of this substructure is 1 (see Section 4.3), meaning that it perfectly discriminates between the positive and negative group graphs. In other words, this substructure shows up only in positive graphs and not in negative graphs.

A **linear chain of communication** is apparent in Figure 10 where we find the communication initiated by the rightmost actor. The message is passed on in a linear chain-like fashion to the leftmost actor in the figure. This pattern discriminates a threat group from a non-threat group even intuitively because in the latter we may expect to find almost completely connected communication graphs, rather than chains.

The pattern that we discovered, shown in Figure 10, is evident in the map shown in Figure 11. This map, which has been reproduced with the kind permission of its owner Valdis E. Krebs, shows the pre-US ties between the terrorists, who were involved in the September 11, 2001 tragedy [8]. Each vertex in this map depicts a terrorist and each edge an old link between two terrorists, maybe in their training camp or college dorm days.

We reproduce a comment by the author of this article [8]:

"The network self-organized (via a network layout algorithm) into the shape of a serpent - how appropriate, I thought."

Since the ties in Figure 11 are "trusted", the author suspects that these links may have been used in connection to the terrorist attacks in the US to pass sensitive information and resources. But of course, in times of such heightened activity, this sparse map may fold over and connect to shorten the path lengths for brief periods of time (see Section 3, Point 7).

A lot of cell members in terrorist groups behave as mere "transmitters", i.e., they just send out the information that they receive without contributing much to the overall activity of that network (see Point 9 in Section 3). In our case, the actor in the middle may be one of these "broker" species. This re-emphasizes the fact that in terrorist networks, communication does not take place via the shortest path (see Section 3, Point 5). For example, the rightmost actor in Figure 10, instead of communicating directly with the leftmost actor, did so via the actor in the middle.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have discussed how semi-structured data necessitated the advent of graph-based data mining. Since graphs lend a topological structure to data and since communication networks lend themselves to a graphical representation, we have chosen to represent our social networks as labeled graphs and then do graph-based data mining on them to discover novel and interesting concepts. We have talked about social network data and how the data of legitimate groups differ from that of covert ones. The various representations of social networks have been touched upon and the significance of SNA for homeland security has been emphasized.

We have discussed the graph-based knowledge discovery system SUBDUE, along with its unsupervised pattern discovery and supervised concept learning approaches. We have shown the effectiveness of SUBDUE in discovering patterns embedded within social network data and also in learning concepts to discriminate legitimate groups from illicit ones, based only on the communication patterns of the group members. We have discussed a similar kind of research, involving identifying threat group activities, albeit with a different approach, being pursued by a research team in 21st Century Technologies.

Incorrect input data occur mostly in threat-group domains due to missing actors and links that investigators may fail to unearth and also due to the "Fuzzy Boundary" problem. Please see Section 3, Points 1 and 2, for a discussion of these issues. In the future, we may handle fuzzy groups, where an actor can belong to the group with a membership value in [0, 1], where 0 would mean that the actor does not belong to the group, 1 would mean that the actor surely belongs to the group and anything in between would mean

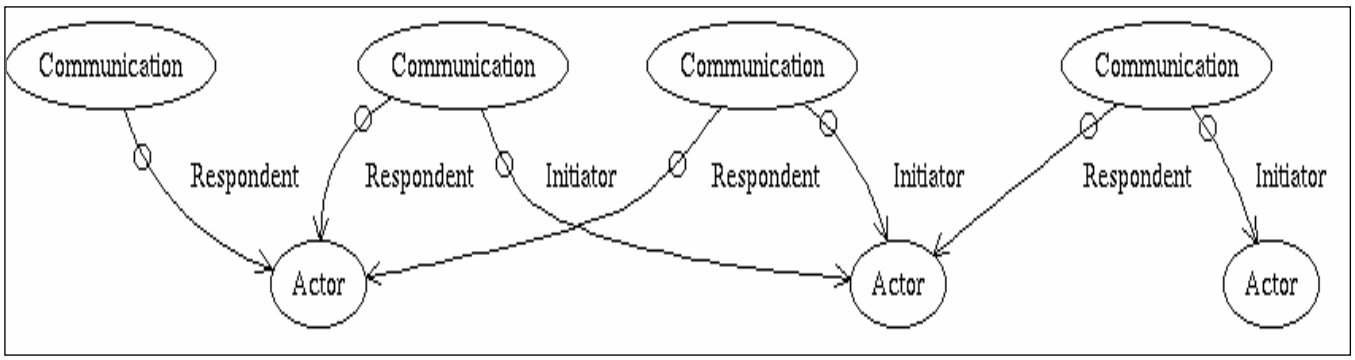

Figure 10. Graphical depiction of best substructure discovered by SUBDUE concept learner.

that the actor belongs to the group with a certain probability. We may also have to address the issue of identifying disconnected examples, where we would have one big graph, parts of which are designated as a part of a positive or negative concept. This big graph is called a "supervised graph", which is more realistic of a social network, where we have threat and non-threat groups inter-mingled. It annotates each graph component with its degree of participation in threat and non-threat groups.

Currently we take a snapshot of static terrorist data, convert it to a labeled graph and feed it to SUBDUE. But for networks as dynamic as terrorist groups, this approach may not yield the most interesting concepts (see Section 3, Point 4). An incremental version of SUBDUE is being built that can update the best substructures based on new data, without rerunning from scratch.

Ties between terrorists are kept under cover, and when unearthed, importance should be attached to the strength of the tie. Strong ties may lead to more interesting concepts than medium/weak ones. Hence, we plan to incorporate signed and valued sociograms besides the binary ones that we are using (see Section 2.3).

In future, we will run more experiments and learn on data with different levels of observability, corruption and noise.

8. REFERENCES

[1] Thayne Coffman, Seth Greenblatt and Sherry Marcus: Graph-based technologies for Intelligence Analysis. Communications of the ACM, Volume 47, Number 3, March 2004.

[2] D. J. Cook and L. B. Holder: Graph-Based Data Mining, IEEE Intelligent Systems, 15(2), 32-41, 2000.

[3] D. Cook and L. Holder: Substructure discovery using minimum description length and background knowledge. J. Artificial Intel. Research, 1:231-255, 1994.

[4] J. Gonzalez, L, Holder, D. Cook: Graph-Based Relational Concept Learning, Proceedings of the Nineteenth International Conference on Machine Learning, 2002.

[5] Robert Hanneman: Introduction to Social Network Methods, Department of Sociology, University of California, Riverside. (URL: http://faculty.ucr.edu/~hanneman /SOC157/Software/NETTEXT.PDF).

[6] T. Imielinski and H. Mannila: A database perspective on knowledge discovery. Communications of the ACM, 39(11):58-64, 1996.

[7] A. Inokuchi, T. Washio and H. Motoda, An Apriori-based Algorithm for Mining Frequent Substructures from Graph Data, Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases, 2000.

[8] Valdis E. Krebs: Uncloaking Terrorist Networks (URL: http://www.firstmonday.dk/issues/issue7_4/krebs). First Monday, volume 7, number 4, April 2002.

[9] M. Kuramochi and G. Karypis, An Efficient Algorithm for Discovering Frequent Subgraphs, IEEE Transactions on Knowledge and Data Engineering, 2002.

[10] H. Mannila and H. Toivonen. Discovering generalized episodes using minimal occurrences. In 2nd Intl. Conf. Knowledge Discovery and Data Mining, pages 146 -151, 1996.

[11] A. Mendelzon, A. Mihaila, and T. Milo. Querying the world wide web. Int. J. Digit Libr., 1:54 -67, 1997.

[12] MRDM'01: Workshop multi-relational data mining. In conjunction with PKDD'01 and ECML'01, 2002. (URL: http://www.kiminkii.com/mrdm/).

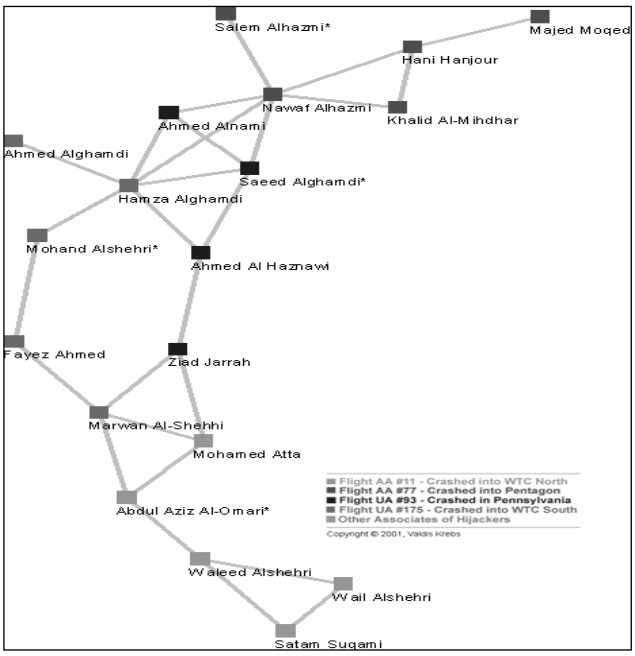

Figure 11. Old ties between terrorists involved in the September 11, 2001 attack (Reproduced with permission from [8]).

[13] S. Muggleton and L. De Raedt: Inductive logic programming: Theory and methods. J. Logic Programming, 19(20):629-679, 1994.

[14] John Scott: Social network analysis: a handbook. London; Thousands Oaks, Calif. : SAGE Publications, 2000

[15] V. Vapnik: The Nature of Statistical Learning Theory. Springer Verlag, New York., 1995.

[16] T. Washio and H. Motoda: State of the Art of Graph-based Data Mining, SIGKDD Explorations Special Issue on Multi-Relational Data Mining, Volume 5, Issue 1, 2003.

[17] Stanley Wasserman and K. Faust: Social network analysis: methods and applications. Cambridge; New York: Cambridge University Press, 1994.

[18] X. Yan and J. Han, gSpan: Graph-Based Substructure Pattern Mining, Proceedings of the IEEE International Conference on Data Mining (IEEE ICDM), 2002.

[19] K. Yoshida, H. Motoda, and N. Indurkhya: Graph-based induction as a unified learning framework. J. of Applied Intel., 4:297-328, 1994.

[20] M. Zaki. Efficiently mining frequent trees in a forest. In 8th Intl. Conf. Knowledge Discovery and Data Mining, pages 71-80, 2002.