SNARE: A Link Analytic System for Graph Labeling and Risk Detection

Mary McGlohon
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA USA 15213
mmcgloho@cs.cmu.edu

Stephen Bay
Ctr for Advanced Research
PricewaterhouseCoopers, LLC
10 Almaden Blvd.
San Jose, CA USA 95113
stephen.bay@us.pwc.com

Markus G. Anderle
Ctr for Advanced Research
PricewaterhouseCoopers, LLC
10 Almaden Blvd.
San Jose, CA USA 95113
markus.g.anderle
@us.pwc.com

David M. Steier
Ctr for Advanced Research
PricewaterhouseCoopers, LLC
10 Almaden Blvd.
San Jose, CA USA 95113
david.m.steier@us.pwc.com

Christos Faloutsos
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA USA 15213
christos@cs.cmu.edu

ABSTRACT

Classifying nodes in networks is a task with a wide range of applications. It can be particularly useful in anomaly and fraud detection. Many resources are invested in the task of fraud detection due to the high cost of fraud, and being able to automatically detect potential fraud quickly and precisely allows human investigators to work more efficiently. Many data analytic schemes have been put into use; however, schemes that bolster link analysis prove promising. This work builds upon the belief propagation algorithm for use in detecting collusion and other fraud schemes. We propose an algorithm called SNARE (Social Network Analysis for Risk Evaluation). By allowing one to use domain knowledge as well as link knowledge, the method was very successful for pinpointing misstated accounts in our sample of general ledger data, with a significant improvement over the default heuristic in true positive rates, and a lift factor of up to 6.5 (more than twice that of the default heuristic). We also apply SNARE to the task of graph labeling in general on publicly-available datasets. We show that with only some information about the nodes themselves in a network, we get surprisingly high accuracy of labels. Not only is SNARE applicable in a wide variety of domains, but it is also robust to the choice of parameters and highly scalable–linearly with the number of edges in a graph.

Categories and Subject Descriptors

H.2.8 [**Information Systems**]: Database Applications—*Data Mining*

General Terms

Algorithms, Security

Keywords

Anomaly Detection, Social Networks, Belief Propagation

1. INTRODUCTION

Accounting irregularities, in which data are intentionally or unintentionally misrepresented, raise significant risk for corporations and investors. Settlement amounts awarded in investor lawsuits have been increasing [10], and so has the number of financial restatements in recent years [27]. Auditors undertake a variety of procedures to determine whether there is reasonable assurance that financial statements are fairly stated, so automated assistance for detecting risks of misstatement has the potential for making the audit process more efficient.

Most of the well-known techniques for detecting accounting irregularities, such as ratio analysis, operate at the financial statement level, a highly aggregated summary of a company's financial activity, and generally offer little useful guidance to an auditor beyond a broad indicator of risk at a company. We have been investigating analytics that operate at a much more detailed level, on the transactions recorded in a company's general ledger. Past methods in this domain [2] explored the potential of different classification methods, such as logistic regression, expectation-maximization, and naive Bayes, on individual accounts and transactions. In this paper we show how exploiting the link structure between accounts has the potential to greatly increase the accuracy of classification methods while making only a few assumptions. We will be applying belief propagation algorithms and link analysis to identify the risk of irregularities in corporate accounting.

Furthermore, we will show that this method is highly flexible to other tasks. Different domains will have different sources of knowledge about nodes in a network; however, our method allows a simple setting for domain experts to

KDD'09, June 28–July 1, 2009, Paris, France.

input this information without an understanding of the details of the algorithm.

Our contributions are the following: We introduce *SNARE* (Social Network Analysis for Risk Evaluation), which detects related entities that may be overlooked by using individual risk scores, it extends a well-known algorithm for graphical models into a useful application, and it may be flexibly applied to different domains. We show how it can be applied to the detection of fraud risk in general ledger accounting data as well as typical graph-labeling tasks in other domains such as web data and social networks.

2. RELATED WORK

Social networks have become more important as practitioners become increasingly aware of the significance of relations between entities in a network. It has been demonstrated that knowledge of social structure can allow one to help make inferences about an organization [3, 21], to identify individuals [17], or to predict adopters of consumer products [18]. Related work has used knowledge of social structures for detecting securities fraud [23]. The authors later improved the approach by showing that one can often infer links that are not explicitly stated [13], and successfully extended the methods using inferred knowledge [11].

Semi-supervised learning methods may also be useful for graph labeling, as addressed in [30]. Finding authority of a node is one specific labeling task addressed in the literature. One way of defining the authority of a node in a network is its "reputation for knowledge," that is, how reliable the source is. Guha et al. extends many of these ideas for reputation networks applied to eBay or Epinions [16]– rather than simply trusting someone's knowledge of a topic, one may also trust another's reliability as a seller on eBay or a recommender on Epinions. The authors use matrix methods and model a "web of trust", where both trust and distrust are propagated over edges (with different patterns of propagation). They were able to predict trust between individuals given a small amount of labeled data.

HITS[20] and Pagerank[24] address reputation for webpages. Other methods of propagation of trust and distrust are discussed in Ziegler et al. [31], particularly in relation to trust on the semantic web.

Other work identifies particular anomalous patterns and seeks to spot them in large graphs. Pandit et al. introduce *NETPROBE*, which uses belief propagation to model eBay as a tripartite network of "fraudsters", "honest users", and "accomplices". Upon deciding on this model, they then use loopy belief propagation to assign probabilities of each node being in the three states [25], by detecting bipartite cores. (Our work differs in that we do not identify a specific network structure such as a bipartite core, only that one can propagate labels using homophily. This allows for more flexibility to different domains.)

Many risk detection methods approach the problem by attempting to detect suspicious behavior in users. This approach has been successful for cellular phone fraud, where a caller's patterns are often disrupted by periods of inactivity. Here, most fraud schemes follow certain signatures, such that a rule-based system have lead to some successes [12]. Rule-based approaches have also been applied to the detection of money laundering[7]. A survey of related methods can be found in [6].

The literature contains many methods for detecting accounting irregularities which typically use a model-based approach [4, 5, 6, 9, 15]. However, many of these traditional approaches are limited by factors such as the diversity of fraud schemes, errors present in the training data, and access only to aggregated financial statement data instead of detailed transactions. To counter thihis problem, in previous work, authors set up a system called Sherlock[1] for detecting errors and fraudulent behavior in general ledger data [2]. Sherlock used classification methods for identifying suspicious accounts, by evaluating a set of features measuring different types of unusual activity. Methods such as naive Bayes, expectation-maximization, and logistic regression were used and compared. This work will approach the same problem of identifying accounts with high fraud risk from a social network analytic perspective.

This is the first work, to our knowledge, that has adapted generalized belief propagation to the accounting domain, and provided a framework to extend it into other domains for node labeling, incorporating both node and edge information. In this work, we are using data where all true labels are unknown from the start, and our results are verified by human investigation.

3. PROPOSED METHOD

We will address the following problem:

Given:

- A graph $G = (V, E)$, where entities (persons, accounts, blogs, etc.) are represented as vertices, or nodes, in the graph, and interactions (phone calls, account transactions, hyperlinks) between them are represented as edges.
- Binary class (state) labels $X = \{x_1, x_2\}$ defined on V.
- A set of flags for each node $v_i \in V$, based on node attributes (geographic location, name, etc.)

Output: A mapping $V \to X$ from nodes to class labels.

The labels X are binary categorical variables derived from the context (normal or irregular, conservative or liberal, etc.). We also note that while nodes and links can be related to social entities such as persons and relations or actions, the proposed methods can be applied to any sort of entities, such as accounts or webpages.

The basic premise of *SNARE* is to use neighboring labels to classify a given node. This premise has proven effective for many graph labeling tasks [19]. However, we also take into account domain knowledge, by assigning an initial risk scores to nodes prior to evaluating neighborhood associations between them. To measure risk by association, we then use *belief propagation* for passing risk to connected nodes. A detailed tutorial of belief propagation may be found in work by Yedidia [29].

Let us summarize the procedure. In a network for a given task, the true label for each node v_i is unknown. We are, however, given some local observations about the node, which we use as a local estimation of its risk, or *node potential* $\phi_i(x_c)$ of v_i for class x_c (the procedure for determining this will be described shortly). Information about this node is inferred from the surrounding nodes. This is obtained

[1] Sherlock is research in progress. As such, the methods we describe should not be interpreted as descriptive of PwC's current standard practice in analyzing general ledger data.

through iterative message passing to and from v_i to each neighbor v_j, where a message from v_i to v_j with its own assessment of v_j's believed class is denoted by m_{ij}. At the end of the procedure, the *belief* of a node v_i belonging to in class x_c is determined. The belief is an estimated probability, which can be thresholded into the classes (e.g. a $b_i(x_c) > .5$ implies v_i belongs to class x_c), or used relatively to compare risk scores between nodes (e.g. $b_i(x_c) > b_j(x_c)$ implies v_i is more likely to belong to x_c than v_j).

In more detail, messages are obtained the following way. Each edge e_{ij} has associated messages $m_{ij}(x_c)$ and $m_{ji}(x_c)$ for each possible class. $m_{ij}(x_c)$ is a message that v_i sends to v_j about v_j believed likelihood of belonging to x_c. Iteratively, messages are updated using the sum-product algorithm. Each outgoing message from a node to a neighbor is updated according to incoming messages from the node's other neighbors. Formally, the message-update equation is as follows:

$$m_{ij}(x_c) \leftarrow \sum_{x_d \in X} \phi_i(x_d)\psi_{ij}(x_d, x_c) \prod_{k \in N(i) \setminus j} m_{ki}(x_d) \qquad (1)$$

where $N(v_i)$ is the set of neighboring nodes to v_i. $\psi_{ij}(x_c, x_d)$ is the *edge potential* of an edge between two nodes i, j of classes x_c and x_d. $\psi_{ij}(x_c, x_d)$ is generally large if edges between x_c and x_d occur often, and small if not. Order of message-passing does not matter, provided all messages are passed in each iteration. We also normalize $m_{ij}(x_c)$ to avoid numerical underflow, as discussed in [8], so each edge's message vector sums to one: $\sum_c m_{ij}(x_c) = 1$.

Convergence occurs when the maximum change between any message between time ticks is less than some value (in our experiments 10^{-6}). Convergence is not guaranteed in general graphs (only for trees), but typically occurs in practice. Upon convergence, belief scores are determined by the following equation:

$$b_i(x_c) = k\phi_c(v_i) \prod_{v_j \in N(v_i)} m_{ji}(x_c) \qquad (2)$$

where k is a normalizing constant (beliefs for each class must sum to 1).

Adapting the message passing algorithm to our purposes has the following challenge: Find an effective yet intuitive way to choose node and edge potentials. We use two main concepts, *homophily* over edges and *node attributes* to influence probability of different classes.

For purposes of explanation, we will have two classes, x_R for "risky" and x_{NR} for "non-risky". We will subsequently refer to $b_i(x_R)$ is the end probability of a node being risky after completion of the algorithm. A node with $b_i(x_R) = 1$ is certainly suspect, and $b_i(x_R) = 0$ is not suspect; most nodes will fall somewhere in between, on the continuum. *SNARE* will then produce a ranked list of the "risky" nodes, as candidates for further investigation.

For the edge potential term $\psi_{ij}(x_c, x_d)$ in the message-passing equations, we chose an identity function with a noise parameter ϵ. That is, if v_i is risky, v_j has a high probability of being risky, while allowing for some variance. The transition matrix is shown formally in Table 1.

Before beginning the message passing procedure, however, we must also assign a *node potential* to each individual node. The node potential represents the risk of a node without considering information from its neighbors. The initial node potential depends on the assumed distribution of class labels– when classes are evenly divided, default values $(\phi(x_{NR}), \phi(x_R)) = (0.5, 0.5)$ may be appropriate, while in cases where risk is sparse (as in most anomaly-detection domains) more skewed values such as $(\phi(x_{NR}), \phi(x_R)) = (0.9, 0.1)$ may be more reasonable.

$\psi_{ij}(x_d, x_c)$	$v_i = x_{NR}$	$v_i = x_R$
$v_j = x_{NR}$	$1 - \epsilon$	ϵ
$v_j = x_R$	ϵ	$1 - \epsilon$

Table 1: Transition matrix, or edge potentials for belief propagation.

However, a key component of *SNARE* is that the initial node potential is determined for each individual node by an process that can incorporate prior knowledge into the algorithm, for example in form of domain knowledge. In most domains where fraud is a challenge, there is rich information available about the potential fraudsters, such as geographic location, patterns of activity, or other flags for suspicious behavior. Therefore, we adjust node potential by assessing the risk to each individual node. There are many ways of doing this; the most useful for our purposes is the use of *flags*. A node may be flagged for having several different types of suspicious behavior, and the domain expert may assign different severity to these flags. Where applicable we chose to use additive risk, increasing with a sigmoid function:

$$F_i = \frac{1}{1 + exp(-1 * f_i)} \qquad (3)$$

where f_i is the total flagged risk, summed for all potential causes for suspicion. The node potential for node i, then, is $\phi_i(R) = F_i$ and $\phi_i(NR) = 1 - F_i$.[2]

When a node is highly flagged it also sends a stronger risk signal to its neighbors. However, if a flagged node's neighbors all have a low initial probability of being risky, the flagged node will be dampened. This is a reasonable action, since isolated flags are more likely to occur in error.

One key advantage of *SNARE* is that it will find risky associated nodes. Fraud schemes as they occur in accounting often involve many accounts, which often allow fraudsters to hide their actions. Since each account may have a very small risk score associated with it, traditional methods may not pinpoint the accounts as abnormal. However, *SNARE* will use the fact that the accounts interact with each other, and raise the associated risk of each account, allowing experts to more easily find the fraudulent behavior.

Since the flags are determined by the domain expert, this procedure can be successful on a wide variety of node labeling tasks, as we will show in the next section.

4. CASE STUDIES

We developed SNARE to help detect risks in accounting data, so we will primarily evaluate it on its ability to find misstated accounts in a company's general ledger.[3] However, since our G/L data is proprietary, and because we

[2] It may be possible to learn the appropriate flag increments through machine learning techniques; this is left for future work.

[3] Some of the terminology we use here is for the purpose of

believe SNARE is more generally useful, we also evaluate its performance for graph labeling using public data from social media and political campaigns. A description of the data and the problems addressed may be found in Table 2.

4.1 Detecting misstated general ledger accounts

The general ledger (G/L) of a company is an accounting record that summarizes its financial activity with double-entry bookkeeping. Within every G/L is a set of accounts which can be thought of as variables representing the allocation of monetary resources. Business events, such as the purchase of machinery, would result in a transaction that reduces the value of the the cash account but increases the value in the fixed asset account by an equivalent amount. The G/L is used to prepare the financial statements by aggregating the balances of the accounts and thus auditors are extremely interested in finding misstatements in this data.

Manipulation of records can be found by experts on both the G/L and financial statement level. There are many different fraud schemes [14, 28] for which experts have identified "red flags" that indicate suspicious behavior based on domain knowledge [9, 14, 22, 26]. For example, one fraud scheme is known as *channel stuffing*. In order to meet earnings expectation, fictitious sales are recorded to increase the revenue for the current quarter. These sales are typically not complete and are recorded solely to meet the earnings target. The company overloads their distribution channels to make it appear as if additional sales have been completed. This helps the company appear to meet its target. Such channel stuffing is usually followed by an increase in the number of returns at the beginning of the next quarter. In the general ledger, one could record the return of a sale by debiting revenue and crediting accounts receivable; thus to look for channel stuffing one might create a threshold test or red flag that highlights an account when there are an excessive number of these transactions.

In practice however, the creation of such a flag to detect channel stuffing or other schemes is fraught with difficulty and pitfalls. For instance with our example of channel stuffing one would need to determine what is an excessive amount of returns since some will always occur for normal business reasons. Setting the threshold too high could result in missing potential frauds, but setting the threshold too low could result in too many false positives. Furthermore, people who intentionally manipulate the G/L are often well aware of the red flags used by auditors and actively attempt to avoid detection. Thus, for example, they may try to hide the activity by spreading the returns over many accounts so as to not set off any thresholds. Our hope with SNARE is that we could set the thresholds relatively low so as to be more sensitive to risky activity and use belief propagation to aggregate risk in the network to identify misstated accounts with a low false positive rate.

To analyze general ledger data with SNARE we first need to create a network with nodes, edges, and initial risks. For our application, we construct the network as follows:

- Each account in the general ledger becomes a node in the network.
- For every pair of accounts (X, Y) in the general ledger, they are connected with an edge if there are transactions where the sum of the amounts debiting X and crediting Y exceeds a minimum threshold.
- The initial risks on the nodes is determined by performing a preliminary scan over the data to detect red flags as determined by domain experts. The red flags are given equal weight and taken together they determine the initial risk as defined by Equation 3.

For example, Figure 1 shows a partial network with nodes for accounts receivable, accounts payable, bad debt, non-trade A/R, and several revenue accounts. In our example of channel stuffing, thresholds for our red flags could be set low enough to flag multiple revenue accounts and SNARE would then propagate the risk to accounts receivable where the collected belief would be strong enough to implicate it. In the next two sections, we present results of SNARE on general ledgers with known misstatements and show that on real data it is effective at aggregating risk across the network.

4.1.1 GL1

In the first set of G/L data there were a total of $1,380$ accounts, $3,820$ edges, and $11,532$ red flags (nearly every node had at least one flag). From prior domain knowledge, 26 accounts were identified as being misstated. We applied *SNARE* to this network and the message-passing process converged after 6 iterations. Our initial node potentials were $\phi_i(Risky) = 0.1$ and $\phi_i(NotRisky) = 0.9$ for a node i with no flags, and additional flags changed node potential according to Equation 3, so key information is in the nodes' number of flags relative to each other.

Figure 2 shows the ROC curve for the *SNARE* approach under the assumption that the 26 identified accounts was the complete set of true positives (and all other accounts are true negatives). In addition to *SNARE* , we plotted to ROC curve for a default approach based on simply ranking the accounts by the number of tests flagged. From the graph, we note that *SNARE* dominated the default sum approach over all regions of the ROC curve. Furthermore, *SNARE* produced an extremely steep initial curve at low false positive rates. This is very promising as this is the region of the operating space most interesting from an application viewpoint.

4.1.2 GL2

The second set of G/L data contained $1,678$ nodes, $18,720$ edges, and $11,401$ red flags. Unfortunately, with this data set we had only coarse label information available that identified general groups of misstated accounts. For our experiments we treated all accounts in an identified group as being misstated, resulting in a total of 337 positive labels.

The results for *GL2* are shown in Figure 3. The results are not as strong as for the previous G/L, but this may be due to the noisy class labels. However, there is still significant improvement in the ROC curve compared with the default strategy of using the number of flags as a scoring mechanism.

Relevant non-proprietary risk-related data with a network structure is challenging to collect and institutions are reluctant to share data due to privacy concerns. Therefore, we will next show the use of *SNARE* for labeling nodes in using publicly available social network data.

conducting research in the area of accounting and is by necessity highly simplified and abbreviated. It not descriptive of how PricewaterhouseCoopers analyzes general ledgers.

Data	Problem description	Size (Nodes, Edges)	Classes	Flags
GL1	Identifying misstated accounts from a general ledger.	1,380 accounts, 3,820 edges (edge occurs if transaction)	1,354 Normal, 26 Misstated	Expert-identified flags of certain suspicious behaviors, 11,532 flags total on the 1,380 accounts.
GL2	Identifying misstated accounts from a general ledger.	1,678 nodes, 18,720 edges	1,305 Normal and 373 Misstated (noisy labeling, see Sec. 4.1.2).	Same as *GL1*, 11,401 flags total on the accounts.
PoliticalBlogs	Labeling political affiliation of blogs.	1,224 blogs joined by hyperlinks	636 Conservative and 558 Liberal	220 flags total, 171 unique blogs with nonzero flags. Blogs flagged based on key substrings in blog domain name.
Campaigns	Correctly classifying political candidates on a bipartite network of candidates and political action committees.	(2004 cycle) 1,357 nodes, 11,334 edges. Edge occurs if there was a donation from committee to candidate.	Republican or Democrat	Flags were on stated class of committees, so candidate labels were acquired only through propagation.

Table 2: Descriptions of data and problems

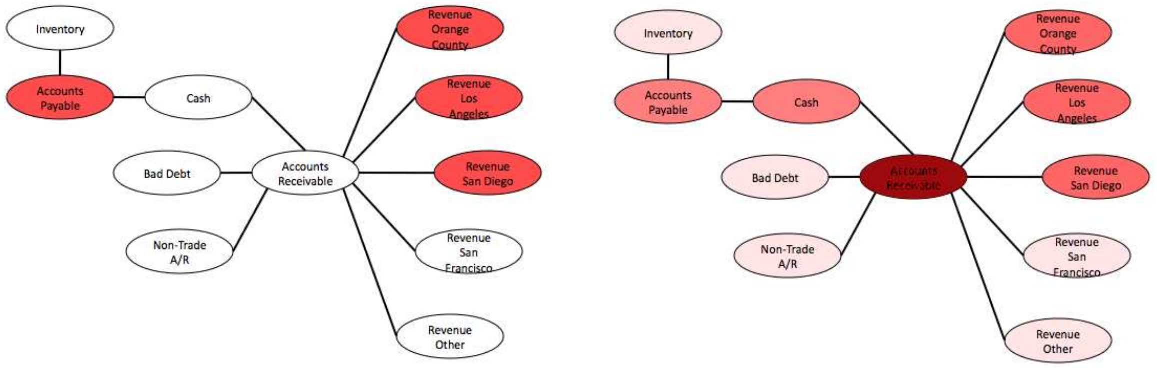

Figure 1: An example network with G/L accounts represented by nodes and edges connecting pairs of accounts with significant amounts debited/credited with each other, under a fraud scheme of *channel stuffing*. The left image shows flagged accounts in red (revenue accounts flagged by abnormal debits), before propagation. The image on the right is the relative risk scores based on beliefs after propagation. Notice that now, since *Accounts Receivable* had many flagged neighbors, it now has the highest risk in the network, while *Accounts Payable* had a lower relative risk, due to the influence of unflagged *Inventory*.

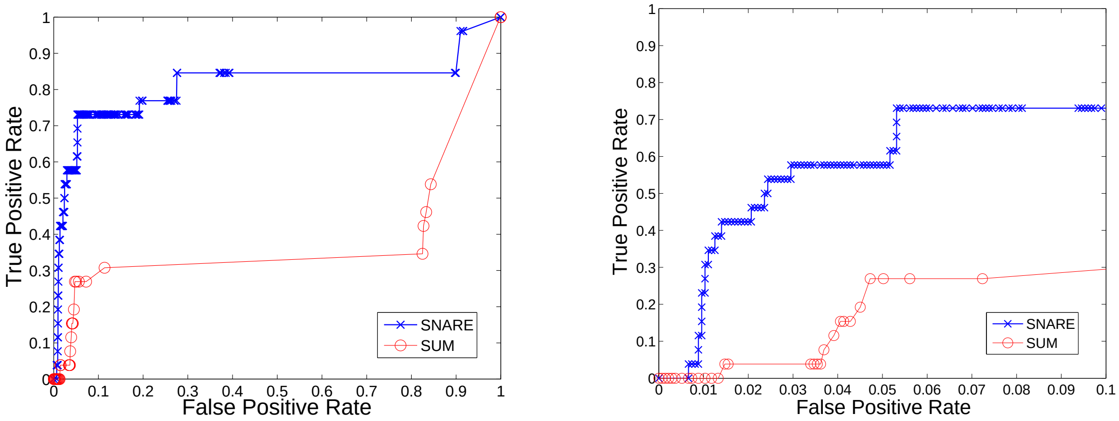

Figure 2: ROC curves for *SNARE* vs. SUM on GL1. The first graph shows the entire range and the second shows performance for false positive rates of less than 0.1.

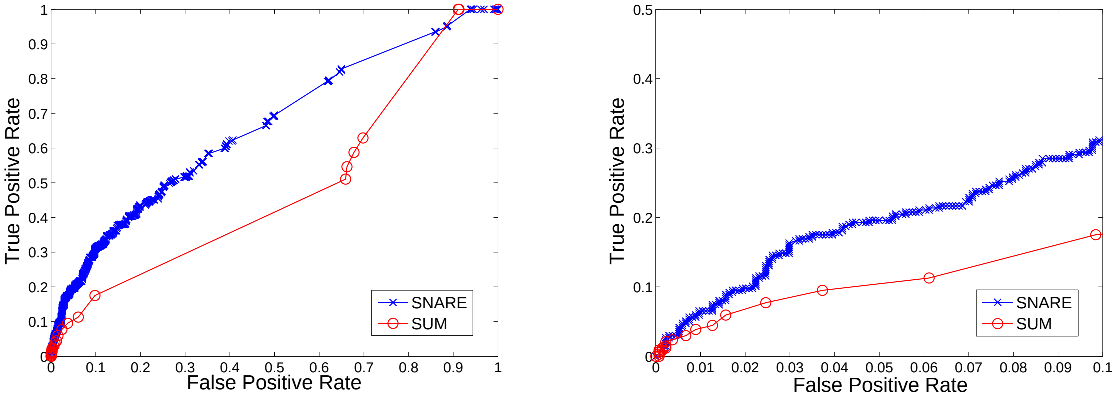

Figure 3: ROC curves for *SNARE* vs. SUM on GL2. The first graph shows the entire range and the second shows performance for false positive rates of less than 0.1.

4.2 Political blogs

The domain of social media presents the difficult task of automatically assessing political stance of a blog, news site, or other webpage. Doing so often requires analysis of sentiment in the text, which is both difficult and computationally expensive. Being able to do so by using the structure of the induced web graph can aid in this problem.

To this end, we tested *SNARE* on a network of political blogs, human-labeled as Conservative or Liberal. The data contained 758 Liberal blogs and 732 Conservative blogs, which were joined with edges based on hyperlinks made by the blog owners. (For details of building the network and labeling, see [1].) Of these, $1,224$ had degree greater than 0–558 Liberal and 636 Conservative, which we chose to focus on for our experiments. The network was relatively dense, with $16,718$ total edges.

In this case, node information was noisy. We chose to flag nodes as more likely to be Conservative/Liberal based on substrings in the blog title. We chose the following flags, and indicate each substring's prevalence in blogs human-labeled as Conservative and Liberal.[4] Of the connected nodes, 171 had flags. Some blogs had multiple flags, so we used additive risk score.

String	Incidence	Flag
"con"	34 conservative, 9 liberal	+1
"right"	33 conservative, 2 liberal	+1
"rep"	19 conservative, 9 liberal	+1
"bush"	8 conservative, 6 liberal	+1
"lib"	11 conservative, 18 liberal	-1
"left"	3 conservative, 28 liberal	-1
"dem"	4 conservative, 28 liberal	-1
"kerry"	2 conservative, 6 liberal	-1

Since the number of Conservative and Liberal blogs was expected to be approximately equal, we used a default potential $(\phi(x_L), \phi(x_c) = \{0.5, 0.5\}$. With $\epsilon = 0.3$, 95% on nodes $(1,188$ of $1,247)$ were classified correctly. An additional 233 nodes ended with a belief score $b_{con} = 0.5$, which we did not consider to be classified one way or the other (though most of them were Liberal). Most of these were isolated nodes; fewer than 20 had a degree greater than 0. For isolated nodes we simply classified them based on the flag, which was 0 in most nodes.

SNARE presented improvements over using the flag method alone or through clustering based on structure. Often times the flag was misleading, such as in the case of `laughatliberals.com` or `johnkerrymustlose.com`, but the edge effects usually allowed *SNARE* to correct the classification, without needing to do sentiment analysis on the words. On the other hand, there were occasions where a few blogs of one class formed a sort of "appendage" on the main cluster of the opposite class, which typical graph clustering methods would fail to identify but were successfully labeled using *SNARE*. One example of this is the two blogs `enemykombatant.blogspot.com` and `democratvoice.org`. The former blog was connected to the Conservative cluster, but the flag on the latter blog, its neighbor, propagated into it, correctly labeling both blogs as Liberal. This is shown in Figure 4.

In fact, most misclassifications occurred on cases of unflagged blogs of one class only bordering on blogs of the opposite class, and in cases along the middle between the two clusters. These cases would be difficult to classify using node information or edge information alone.

4.3 Political campaign contributions

While labeling political party membership for individuals running for office is not typically a challenge, we used it as a way to test our approach to labeling nodes by leveraging connection structure.

We took subsets of data from the United States Federal Election Commission [5] from the election cycles of 1980 through 2006, that listed donations from political action committees to political candidates for President, Senate, and House of Representatives. We then built a bipartite network of committees and candidates, creating edges between

[4] Crawling the blogs themselves and using textual analysis would have potentially provided more accurate flags; however, we chose the more naive flag for experimental purposes, showing that even imperfect node information provides good results.

[5] `www.fec.gov/finance/disclosure/ftpdet.shtml`, downloadable in parsed format from `www.cs.cmu.edu/~mmcgloho/data.html`

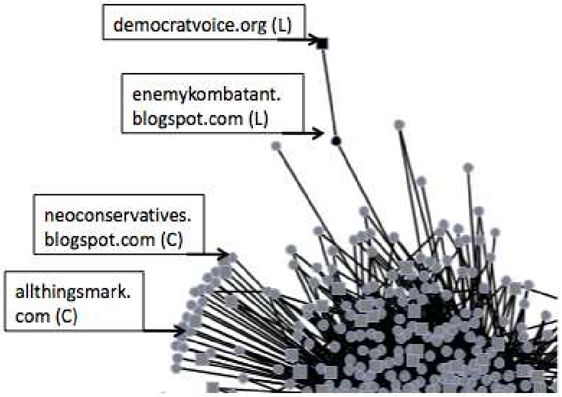

Figure 4: The political blog network, where human-labeled conservative blogs are shown in gray and liberal blogs shown in black. Flagged nodes (in either class) are shown as squares. This section highlights two outlier Liberal blogs connected to the cluster of Conservative blogs. Since democratvoice was flagged as Liberal, these two blogs were correctly classified with *SNARE*.

a committee and a candidate if a committee had, at some point, donated funds to the candidate. The largest cycle, 2004, contained $1,357$ nodes with positive degree (686 candidates and 671 committees) and $11,334$ edges. The classification task was to label a candidate as Democrat or Republican, based only on the committees it was connected to through donations.

Of the 671 committees, 583 were labeled with a party. We used these labels as flags (+1 or -1). From there, we ran *SNARE* on the bipartite graph to propagate labels to candidates. *SNARE* correctly labeled 659, mislabeled 12, and did not label 25, which gave an accuracy of 96 percent. With one exception (the earliest cycle, 1980, with an accuracy of 82%), all other cycles had above 90 percent accuracy.[6]

We find that varying parameters does not drastically affect accuracy, and the method is scalable to large graphs, as we will explain in the next section.

5. ANALYSIS

We next demonstrate the robustness of *SNARE* to different parameter ranges, analyze its computational efficiency, and compare the accuracy to to spectral clustering on the task of graph labeling.

5.1 Sensitivity of parameters

SNARE is very robust and easy to use. Some domain knowledge is necessary for determining the node potential for both flagged and unflagged nodes. Default node potential is typically set at the expected percentage from each class (for example, {0.9, 0.1} if one expects 90% of nodes in class 0 and 10% in class 1). Modifications of the sigmoid function tend to work well for additive risk for flagged nodes.

The edge potential parameter ϵ may be set in the range of $0 < \epsilon < .5$ without drastically affecting results. In *Campaigns*, we observed high sensitivity on the node potentials, and putting any bias on class tended to cause one class to dominate. This would seem natural, since the data were approximately split equally among the two classes, so any initial bias will dominate the final result. However, the ϵ parameter showed little sensitivity, and varying it between 0 and 0.5 affected results by less than 1 percent on both *Campaigns* and *PoliticalBlogs*. (Setting $\epsilon \geq .5$ would remove the homophily assumption, which would not be useful for tasks addressed here.) Figure 5 shows finer-grained results of varying parameters on blog data; even the smallest ϵ is effective, and accuracy does not change up to $\epsilon = 0.5$.

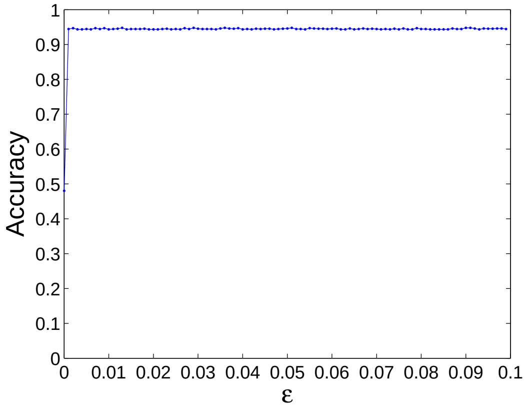

Figure 5: A demonstration of the robustness of *SNARE*, by varying the ϵ for *PoliticalBlogs* data, between 0 and 0.1. Note that even the smallest ϵ is effective. Accuracy results are similar for ϵ up to 0.5 (omitted to avoid redundancy).

5.2 Computational performance

The most costly operation of *SNARE* occurs during the message-passing. Each iteration runs in $O(|E|)$ time, where $|E|$ is the number of edges in the network. Our experiments also reached convergence in relatively few iterations (less than 10 for all datasets). Other negligible computational costs are in assessing node potentials and calculating beliefs (both $O(N)$), and in all cases convergence occurred within 10 message-passing iterations.

Since the data varied in structure, we chose to run scaling experiments only on *Campaigns*. To sample, we took different window-sizes of election cycles, for every possible cycle, and timed the completion of *SNARE* 100 times apiece. A plot of average time vs. number of edges in the graph is shown in Figure 6, including the best linear fit.

5.3 Comparison to existing work

To compare our performance to the state of the art, we also run spectral clustering on our data, which is an unsupervised method for node labeling. For *Campaigns* and *PoliticalBlogs* the data were already well-clustered, and visual analysis could cluster reasonably successfully. Spectral clustering, however, performed less well than *SNARE* even on these data sets.

On *PoliticalBlogs*, attempting to find two clusters failed. However, clustering results were better by allowing for a third cluster that did not fit with the other two. The two

[6] In fact, using very sparse flags (randomly selecting 10 committees from each class to flag) produced comparable results.

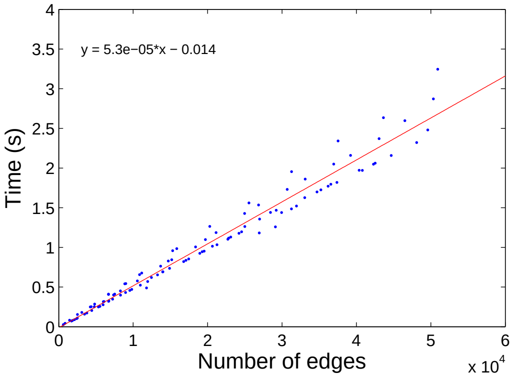

Figure 6: Scalability results for *Campaigns* data. *SNARE* scales linearly, with a 50,000 edge graph converging in under 3.5 seconds.

major clusters roughly corresponded to the conservative and liberal sectors. In full, of 1224 non-isolated blogs, 1133 were correctly classified. There were 83 misclassifications, and 8 in the third "undecided" cluster. This gave an accuracy of 92.5%, slightly less than *SNARE*.

On *Campaigns*, results were similar. There were two distinct clusters roughly corresponding to the parties. There were 617 correct classifications, 19 incorrect, and 60 unclassified, for 88.5% accuracy.

However, for data sets such as the general ledger data where the nodes do not form very clear clusters, spectral clustering does not perform well. In this type of data *SNARE* has a distinct advantage.

6. CONCLUSION

We successfully applied link analysis to the domain of risk detection for accounting data and produced results that were a significant improvement over a the method that flags suspicious accounts. Formerly, an automated system simply flagged entities that appeared risky, with some sense of priority. Using link analytic methods, one can rerank the risk of an account not only based on irregularities in a single account, but also in other accounts with which it shares transactions. Also, a group of accounts that are closely related and have distributed risk may be identified while under individual flags they would fall below the threshold. In many other domains there may be a cluster of related entities (for example, collaborators in a social network), where the collection of evidence from each party may put the collective risk above the threshold.

We also show that *SNARE* is successful for the task of node labeling in networks in general. While risky nodes may be relatively sparse in a graph, we show that by adjusting initial belief scores one can generalize to domains where labels are more evenly divided between two classes. *SNARE* also has the capability of considering prior node-specific domain knowledge for flags– while we used accounting-specific flags in *GL1* and *GL2*, we chose text flags in *PoliticalBlogs* and committee information in *Campaigns*.

The *SNARE* system is simple to implement and extend to other domains, and may be particularly useful for other types of fraud detection that ordinary graph clustering methods may have difficulty with, such as link farms or botnets in the web graph, or fraud in mobile phone networks.

In summary, our contributions are the following:

- We have introduced *SNARE*, which uses belief propagation, taking into account both domain knowledge as well as network effects for labeling nodes in a graph, for risk detection and other applications. *SNARE* has the following characteristics:
- **Flexible:** We have applied *SNARE* to a variety of domains, including a sample of general ledger accounting data as well as public datasets (blog labeling, election contributions).
- **Accurate:** *SNARE* has a high labeling accuracy, compared to simply using flags for accounting irregularity detection (up to 6.5 lift, more than twice that of the default heuristic), and performs better than spectral clustering (with up to 97% accuracy).
- **Scalable:** The algorithm is very efficient, running in linear time with the number of edges in the graph– 50,000 edges completed in 3 seconds.
- **Robust:** *SNARE* is robust with a variety of parameters, so it requires almost no tweaking of parameters to work correctly. It is therefore flexible, simple to implement, and can be applied to many other domains, in addition to those we have already introduced.

Acknowledgments

The authors would like to thank Polo Chau for his helpful consultation. This work was conducted in collaboration with the PricewaterhouseCoopers' Center for Advanced Research (CAR), directed by Sheldon Laube and Glenn Ricart. In addition to their support and feedback, we would like to acknowledge the assistance of our colleagues within the Center for Advanced Research working on the Sherlock system: Jeff DeLisio, Mave Houston, Li Chen, Rohit Kumar and Kamaljeet Kaur. This material is based upon work supported by the National Science Foundation (NSF), Grants No. IIS-0705359, IIS-0808661, and CNS-0721736. This work is also partially supported by an IBM Faculty Award, a Yahoo Research Alliance Gift, a SPRINT gift, with additional funding from Intel, NTT and Hewlett-Packard. Mary McGlohon was partially supported by a Yahoo! Key Technical Challenges Grant. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or other funding parties.

7. REFERENCES

[1] L. Adamic and N. Glance. The political blogosphere and the 2004 U.S. election: Divided they blog, 2005.

[2] S. Bay, K. Kumaraswamy, M. G. Anderle, R. Kumar, and D. M. Steier. Large scale detection of irregularities in accounting data. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 75–86, Washington, DC, USA, 2006. IEEE Computer Society.

[3] R. Behrman and K. Carley. Modeling the structure and effectiveness of intelligence organizations: Dynamic information flow simulation. In *Proceedings of the 8th International Command and Control Research and Technology Symposium.*, 2003.

[4] T. Bell and J. Carcello. A decision aid of assessing the likelihood of fraudulent financial reporting. *Auditing: A journal of practice and theory*, 19:169–184, 2000.

[5] M. Beneish. The detection of earnings manipulation. *Financial Analysts Journal*, 55(5):24–36, 1999.

[6] R. Bolton and D. Hand. Statistical fraud detection: A review, 2002.

[7] R. J. Bolton and D. J. Hand. Unsupervised profiling methods for fraud detection, 2001.

[8] T. Cohn. *Scaling Conditional Random Fields for Natural Language Processing*. PhD thesis, University of Melbourne, 2007.

[9] P. M. Dechow, W. Ge, C. R. Larson, and R. G. Sloan. Predicting material account manipulations. *AAA 2008 Financial Accounting and Reporting Section (FARS)*, 2008.

[10] D. Dooley and G. Lamont. PwC 2005 securities litigation study. Technical report, PricewaterhouseCoopers LLP, 2006.

[11] A. Fast, L. Friedland, M. Maier, B. Taylor, D. Jensen, H. G. Goldberg, and J. Komoroske. Relational data pre-processing techniques for improved securities fraud detection. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 941–949, New York, NY, USA, 2007. ACM.

[12] T. Fawcett and F. J. Provost. Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1(3):291–316, 1997.

[13] L. Friedland and D. Jensen. Finding tribes: identifying close-knit individuals from employment patterns. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 290–299, New York, NY, USA, 2007. ACM Press.

[14] W. Golden, S. Skalak, and M. Clayton. *A Guide to Forensic Accounting Investigation*. John Wiley & Sons, Hoboken, N.J., 2006.

[15] H. Grove and T. Cook. A statistical analysis of financial ratio red flags. *Oil, Gas and Energy Quarterly*, 53(2):3212–3346, 2004.

[16] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 403–412, New York, NY, USA, 2004. ACM.

[17] S. Hill and F. Provost. The myth of the double-blind review?: author identification using only citations. *SIGKDD Explor. Newsl.*, 5(2):179–184, December 2003.

[18] S. Hill, F. Provost, and C. Volinsky. Network-based marketing: Identifying likely adopters via consumer networks. *Statistical Science*, 22(2):256–275, 2006.

[19] D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *KDD '07: Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004.

[20] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[21] S. A. Macskassy and F. Provost. Suspicion scoring based on guilt-by-association, collective inference, and focused data access. In *Proceedings of the NAACSOS Conference*, June 2005.

[22] C. W. Mulford and E. E. Comiskey. *The Financial Numbers Game: Detecting Creative Accounting Practices*. John Wiley & Sons, Hoboken, N.J., 2002.

[23] J. Neville, Ö. Şimşek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg. Using relational knowledge discovery to prevent securities fraud. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 449–458, New York, NY, USA, 2005. ACM Press.

[24] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[25] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 201–210, New York, NY, USA, 2007.

[26] H. Schilit. *Financial Shenanigans: How to Detect Accounting Gimmicks and Fraud in Financial Reports*. McGraw-Hill, 2002.

[27] S. Skalak and C. Nestler. Global economic crime survey 2005. Technical report, PricewaterhouseCooper LLP, 2005.

[28] J. Wells. *Corporate Fraud Handbook: Prevention and Detection*. John Wiley & Sons, Hoboken, N.J., 2004.

[29] J. S. Yedidia, W. T. Freeman, and Y. Weiss. *Understanding belief propagation and its generalizations*, pages 239–269. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[30] X. Zhu. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2005.

[31] C. N. Ziegler and G. Lausen. Propagation models for trust and distrust in social networks. *Information Systems Frontiers*, 7(4-5):337–358, 2005.