Submitted to *Management Science*
manuscript MS-14-00232

GOTCHA! Network-based Fraud Detection for Social Security Fraud

Dr. Véronique Van Vlasselaer
Department of Decision Sciences and Information Management, KU Leuven, Leuven, Belgium, Veronique.VanVlasselaer@kuleuven.be

Prof. Dr. Tina Eliassi-Rad
Department of Computer Science, Rutgers University, Piscataway, NJ, USA, tina@eliassi.org

Prof. Dr. Leman Akoglu
Department of Computer Science, Stony Brook University, Stony Brook, NY, USA, leman@cs.stonybrook.edu

Prof. Dr. Monique Snoeck
Department of Decision Sciences and Information Management, KU Leuven, Leuven, Belgium, Monique.Snoeck@kuleuven.be

Prof. Dr. Bart Baesens
Department of Decision Sciences and Information Management, KU Leuven, Leuven, Belgium, Bart.Baesens@kuleuven.be
School of Management, University of Southampton, Highfield, Southampton, SO17 1BJ, United Kingdom

We study the impact of network information for social security fraud detection. In a social security system, companies have to pay taxes to the government. This study aims to identify those companies that intentionally go bankrupt in order to avoid contributing their taxes. We link companies to each other through their shared resources, as some resources are the instigators of fraud. We introduce *GOTCHA!*, a new approach on how to define and extract features from a time-weighted network, and how to exploit and integrate network-based and intrinsic features in fraud detection. The *GOTCHA!* propagation algorithm diffuses fraud through the network, labeling the unknown and anticipating future fraud whilst simultaneously decaying the importance of past fraud. We find that domain-driven network variables have a significant impact on detecting past and future frauds, and improve the baseline by detecting up to 55% additional fraudsters over time.

Key words: fraud detection, network analysis, bipartite graphs, fraud propagation, guilt-by-association
History: This paper was first submitted on February 5, 2014.

---

1. Introduction

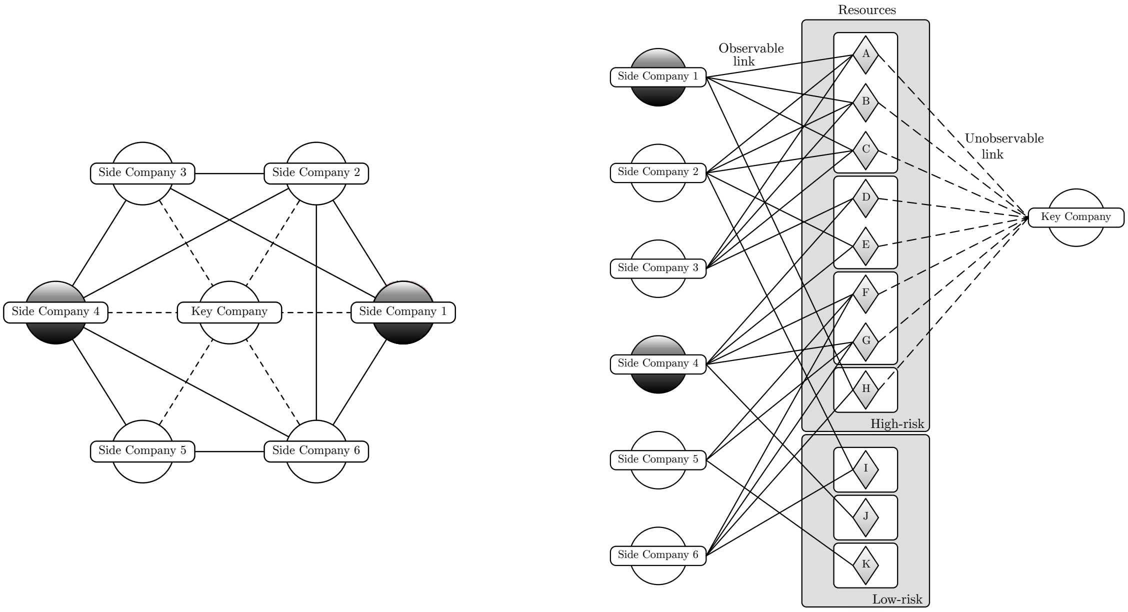
Fraud detection is a research domain with a wide variety of different applications and different requirements, including credit card fraud (Chan and Stolfo 1998, Quah and Sriganesh 2008, Sánchez et al. 2009), call record fraud (Fawcett and Provost 1997), money laundering (Gao and Ye 2007, Jensen 1997), insurance fraud (Dionne et al. 2009, Furlan and Bajec 2008, Phua et al. 2004) and telecommunications fraud (Hilas and Sahalos 2005, Estévez et al. 2006). The aforementioned problems generally exhibit the same characteristics, but the solution to each problem is rather domain-specific (Chandola et al. 2009). Data mining techniques – i.e., finding patterns and anomalies in large amounts of data – have already proven useful in risk evaluation (Baesens et al. 2003a,b), but fraud is an atypical example and requires built-in domain knowledge.

We introduce *GOTCHA!*, a new, generic, scalable and integrated approach on how (social) network analytics can improve the performance of traditional fraud detection tools in a social security context. We identify five challenges that concur with fraud. That is, fraud is an *uncommon*, *well-considered*, *time-evolving*, *carefully organized* and *imperceptibly concealed* crime that appears in many different types and forms. Whereas current research fails to integrate all these dimensions into one encompassing approach, *GOTCHA!* is the first to address each of these challenges together in one high-performance, time-dependent detection technique.

In short, *GOTCHA!* contributes to the fraud detection domain by proposing a novel approach on how to spread fraud through a (i) time-weighted network and features extracted from a (ii) bipartite graph (cfr. infra). We exploit dynamic network-based features derived from the direct neighborhood, and develop a new propagation algorithm that infers an initial exposure score for each node using the whole network. The exposure score measures the extent to which a node is influenced by fraudulent nodes. We integrate both intrinsic and network-based features into one scalable algorithm. We argue that fraud is a time-dependent phenomenon, and as a consequence *GOTCHA!* is designed such that a subject's characteristics and fraud probability can change over time.

We test the validity of our approach on a real data set obtained from the Belgian social security institution, which registers and monitors every active company in Belgium and keeps track of all resources, and their associations with companies.[1] In a social security system, companies have to pay employer and employee contributions to the government. Fraud occurs when companies *intentionally* go bankrupt in order to avoid paying these taxes. A new/existing company with (partly) the same structure is founded afterwards and continues the activities of the former company. We can compare the structures of companies through their resources.

[1] Due to confidentiality issues, we will not elaborate further upon the exact type of resources, but the reader can understand shared resources in terms of the same address, equipment, buyers, suppliers, employees, etc.



(a) (b)

Figure 1 (a) Example of a spider construction. Company 1 and 4 are fraudulent. Resources are transferred towards other companies (solid line). The key company organizes the fraudulent setup, but its links to other companies are hidden (dashed line). (b) Bipartite graph of the spider construction. Companies are indirectly connected to each other through the resources.

A spider construction is a fraudulent setup with an active exchange of resources between the companies, i.e., fraudulent companies do not transfer all of their resources to only one other company as this might attract too much attention (see Figure 1a). They rather distribute their resources among many companies. Active companies that inherit resources from fraudulent companies, exhibit a high risk of perpetrating fraud themselves. In particular, we distinguish between the key and side companies. The *side companies* are the perpetrators of the fraud and have an observable link to each other through shared resources. The core of a spider construction is the *key company*, which is responsible for organizing the fraud, setting up many side companies and pruning away their profits, so that they go bankrupt. However, the key company has unobservable links, and therefore we can only detect the side companies. The main goal of *GOTCHA!* is to exploit the associations between companies and their resources to infer which companies have a high risk to commit fraud in the future. We believe that network-based knowledge might strongly improve the standard approaches, which only use intrinsic variables in the detection models.

In order to assess the added value of our approach, we compare *GOTCHA!* to three baselines: (1) an intrinsic model, only including intrinsic features; (2) a unipartite model, linking companies directly together by means of the resources they shared or transferred among each other; (3) a bipartite model, which starts from the same network representation as our *GOTCHA!* model, integrating both companies and resources (see Figure 1b). Yet, the model is not time-weighted. Our results show that an optimal mix between intrinsic and time-weighted network-based attributes contribute to a higher accuracy and more precise output than the baselines. Moreover, it appears that many regular (i.e., non-intentional) bankruptcy companies are also outputted and classified as high risk. This is a strong indication that the developed approach is also able to find those companies that committed fraud, but were not caught in the past. As a result, we argue that our approach is suitable for both *future* and *retrospective* fraud detection.

This paper is organized as follows: Section 2 motivates *GOTCHA!*'s fraud detection process and framework, as well as *GOTCHA!*'s contributions to existing research. Section 3 focuses on how network analysis is implemented for fraud detection. This section also discusses *GOTCHA!*'s propagation algorithm and how domain-driven networked features are defined and extracted from the network. Section 4 summarizes the modeling approach. Section 5 contains the results of *GOTCHA!* on social security fraud data. Section 6 concludes this paper.

2. Social Security Fraud Detection

2.1. Background

The Belgian Social Security Institution is a federal agency that monitors the tax contributions of every active company in Belgium. These contributions are used to fund the various branches in social security, such as family allowance funds, unemployment funds, health insurance, holiday funds, etc. Companies – or in general terms, the employers – need to pay employer and employee contributions to the government. Some companies, nevertheless, fail to redeem their obligations and file for bankruptcy. Recently, experts found evidence of fraudulent setups through bankruptcy.

In real data, we observe small "webs of fraud", the so-called *spider constructions*. A spider construction consists of (fraudulent) companies that are closely connected to each other through shared or transferred resources. Resources include address, equipment, buyers, suppliers, employees, etc. For example, two companies are associated with each other because they operate at the same location. The data reveals which resource is associated with which company for which specific time period. We observe that the profits of companies that belong to a fraudulent setup are often pruned away by a hidden key company (see Figure 1). Consequently, the company becomes insolvent and files for bankruptcy, leaving the government with unrecoverable debt claims. We see, however, that their operational resources move towards other currently legitimate or newly founded companies, e.g., 80% of the resources of the fraudulent company are re-used by a new or currently legitimate company. Those companies will continue the activities of the fraudulent company. The transfer (or sharing) of such resources induces the observable structure of spider

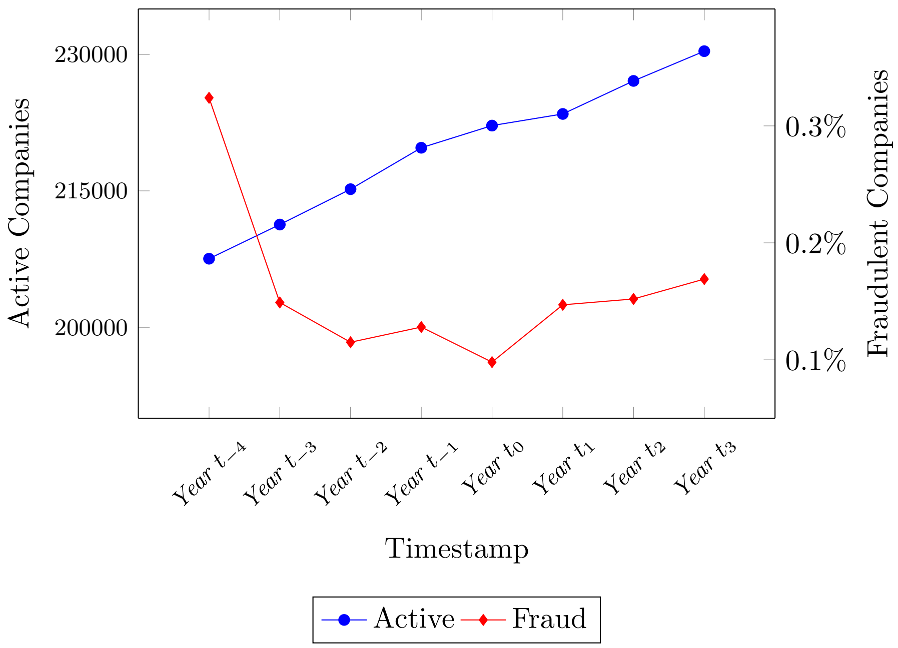


Figure 2 **Overview of the total number of active companies (blue curve) and fraudulent companies (red curve). The number of active companies is consistently growing. A similar trend can be noticed in the number of fraudulent companies.**

constructions. Companies that inherit (many) resources of fraudulent companies, exhibit a high risk of perpetrating fraud in the future as well. Figure 1b shows how (groups of) resources are exchanged between various companies, transferring fraudulent knowledge on how to commit fraud (Levin and Cross 2004) towards legitimate companies. We must note that *resource sharing* is nevertheless a normal activity in the corporate environment, complicating the detection process. Although the exact procedure of resource sharing is confidential, the reader can think in terms of e.g., the transfer or sharing of employees, equipment, buyers/suppliers, and addresses taken over by other employers, etc. The requirements of fraud experts are threefold: (1) curtailing the growth of existing spider constructions; (2) preventing the development of new spider constructions; and (3) detecting uncaught spider constructions, i.e., dense subgraphs in the network with many bankruptcies. In this work, we focus on requirement (1) and (2). Recall that we do not have information to associate key companies to their side companies. Therefore, we aim to find suspicious side companies.

2.2. Challenges

A first contribution of this research is the investigation and identification of the underlying reasons why fraud detection cannot be resolved by applying standard data analytics. We identify five challenges present in most fraud detection problems, and discuss how each challenge can be addressed. In general, the main challenges that characterize fraud are as follows:

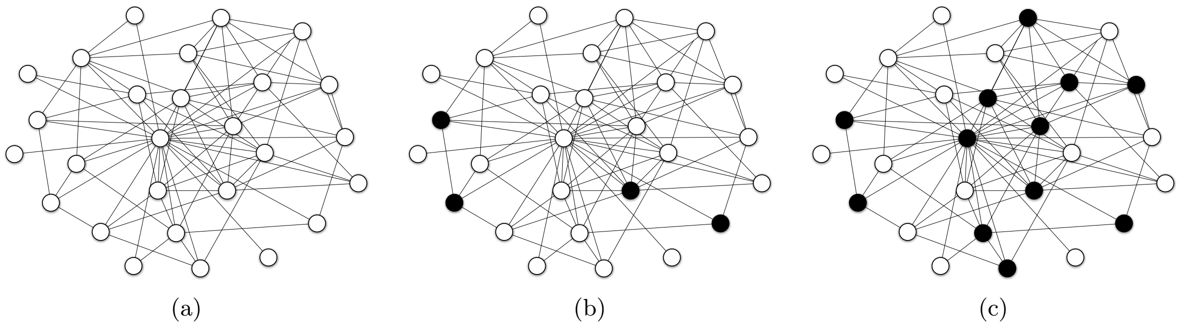


Figure 3 Real-life example of fraud propagating through a sub-network over time. Legitimate companies are unfilled, fraudulent companies are filled. The initial situation is represented in (a). When time passes, more nodes are influenced by fraudulent behavior of their neighbors (b), ultimately infecting almost the whole subgraph (c). This confirms the contagious effect of fraud.

Fraud is an *uncommon*, *well-considered*, *time-evolving*, *carefully organized* and *imperceptibly concealed* crime which appears in many different types and forms.

I. Uncommon Fraud detection techniques must deal with extremely skewed class distributions. Subject matter experts are often only able to identify a limited number of confirmed fraud cases. Rather than using unsupervised techniques, how can we use and learn from (sparsely) labeled data? Resampling techniques (Provost 2000, Chawla et al. 2011) are able to emphasize fraud and rebalance the data set.

Figure 2 depicts the number of active companies over 8 years (blue curve) and the percentage of fraudulent companies over the same time period (red curve) for the social security institution in our study.[2] Each year, approximately 230K companies are active with a fraud ratio between 0.09% and 0.18%, except for year t_{-4} where the fraud ratio is 0.32%.[3] For reasons of stability, *GOTCHA!* is applied to year $t_0 - t_3$.

II. Well-considered Complex fraud structures are carefully planned and well thought through. Fraud is present in all attributes. Labeling instances based on a single action (e.g., outlier detection) is often inaccurate and insufficient. We believe that integrating intrinsic and domain-driven network attributes helps to improve model performance.

III. Time-evolving Fraud evolves over time. Fraudsters learn from the mistakes of their predecessors and are highly adaptive (Jensen 1997). Models should be built for a varying temporal

[2] Due to confidentiality issues, the exact date of each timestamp is omitted.

[3] During *year* t_{-4} a fraud detection team was assigned and experts effectively started to report fraud. The peak in fraud detection is mainly due to catching up the piling backlog of old fraud cases and entering them in the system.

granularity, weighing information based on its recency (Rossi and Neville 2012). We estimate models for different timestamps, resulting in a time-dependent fraud probability.

IV. Carefully organized Fraudsters often do not operate by themselves, but are influenced by close allies and influence others in turn. They transfer knowledge on how to commit fraud without being detected. This is *homophily*. Homophily states that instances that are closely related to each other are likely to behave in the same way (Aral et al. 2009, Bapna and Umyarov 2012). A feasibility study (Park and Barabási 2007, Easley and Kleinberg 2010) on the social security data set indicates that fraudulent companies are indeed significantly more connected to other fraudulent companies (p-value ≤ 0.02 for $t_0 - t_3$ using a one-tailed proportion test).

V. Imperceptibly concealed Maes et al. (2002) formulated this as the presence of overlapping data. Fraudulent companies often have the same characteristics as legitimate companies. In the fraud detection domain, there is a need for extracting additional, meaningful features that uncover hidden behavior. We focus on influence. Influence is subtle and often subliminal. This challenge encompasses how to capture unobservable, subtle fraudulent influences from the external environment. We address this challenge by means of collective inference procedures, like network propagation techniques, to diffuse a small amount of fraudulent behavior through the network and infer a fraud *exposure score* for every node in the network.

Figure 3 illustrates how fraud spreads through a network over time, much like a virus. The closer the nodes are located to the region of a fraudulent source, the higher the probability of copying the fraudulent behavior. This phenomenon is known as the *propagation* effect (Prakash et al. 2010).

Sections 3, 4 and 5 of this paper explain in more detail how we address each of these challenges. In particular, Section 3.3 describes how we infer an initial exposure score for every company, and consequently label the unknown resources based on fraudulent influences from the whole network (*Challenge V*). In Section 3.4, each company is then featurized based on its direct resources (*Challenge IV*). Section 4 discusses how we integrate intrinsic and network-based features (*Challenge II*) and resample the data set using SMOTE (Chawla et al. 2011) (*Challenge I*). The proposed fraud detection technique estimates time-weighted features and a time-dependent fraud probability for every company (*Challenge III*), which is explained in Section 5.

2.3. Related Work

Although fraud detection algorithms are frequently discussed in the literature, only few research studies acknowledge the importance of network analytics in fraud detection. To the best of our

#	Reference	Fraud type	Challenges I	II	III	IV	V
1	(Goldberg and Senator 1995)	*money laundering*				X	
2	(Jensen 1997)	*money laundering*				X	
3	(Cortes et al. 2001)	*telecom fraud*			X	X	
4	(Chen et al. 2004)	*insurance fraud*				X	
5	(Galloway and Simoff 2006)	*law enforcement fraud*				X	
6	(Neville et al. 2005)	*security fraud*	X		X	X	
7	(Fast et al. 2007)	*security fraud*	X		X	X	
8	(Wang and Chiu 2008)	*online auction fraud*			X	X	
9	(Akoglu et al. 2010)	*various*				X	
10	(Yanchun et al. 2011)	*online auction fraud*				X	
11	(Gyöngyi et al. 2004)	*web spam*				X	X
12	(Chiu et al. 2011)	*online auction fraud*				X	
13	(Chau et al. 2006)	*online auction fraud*		X		X	X
14	(Pandit et al. 2007)	*online auction fraud*			X	X	X
15	(Gallagher et al. 2008)	*various*				X	X
16	(McGlohon et al. 2009)	*accounting fraud*				X	X
17	(Šubelj et al. 2011)	*insurance fraud*		X		X	X
18	(Akoglu et al. 2013)	*opinion fraud*				X	X
19	***GOTCHA!***	*social security fraud*	X	X	X	X	X

Table 1 Overview of all published papers related to fraud detection using network analytics.

knowledge, Table 1 gives an overview of all published papers related to fraud detection using network analytics. The table evaluates each paper according to the identified challenges in Section 2.2. All papers comply with *Challenge IV*, i.e., including network analysis in the detection process.

Methods 1-5 focus on one type of network feature to measure or visualize fraud and rely to a larger extent on human interaction for effectively guiding the fraud detection process. *GOTCHA!* is designed such that it derives multiple network-based features in order to judge the fraudulence of other instances. Methods 6-10 are more advanced; they analyze and combine multiple aspects of the direct neighborhood to decide whether a node in the network is fraudulent or not. Collective inference procedures for fraud detection are discussed in methods 11-18. Rather than only taking into account the direct neighborhood, *GOTCHA!* implicitly uses the indirect neighborhood to infer a label for the unknown nodes, both anticipating future fraud and forgiving past associations.

Except for Šubelj et al. (2011) and Chau et al. (2006), all fraud detection papers exclusively use network variables to detect fraud, neglecting instance-specific information. Although we believe that the network effects play an important role in accurately identifying fraud, individual instance behavioral information often also contains subtle signs of new types of fraud and should therefore

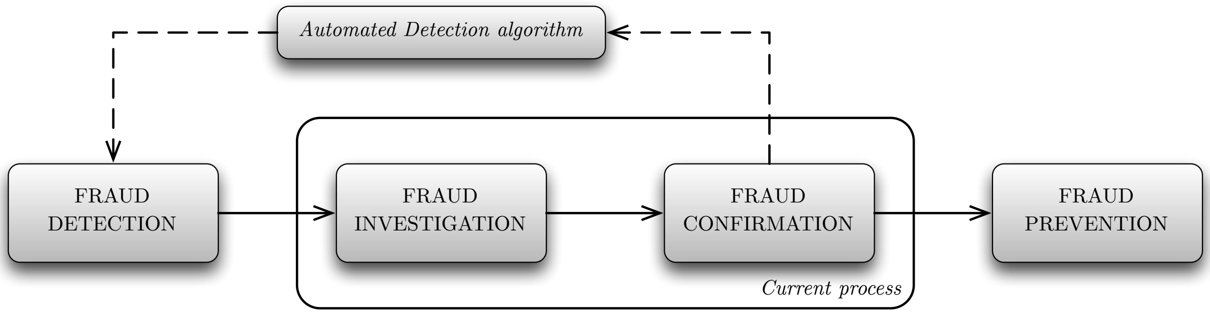


Figure 4 **Fraud detection process for the social security institution.**

not be disregarded and considered as a valuable indicator in the fraud detection process. Our paper differs from the work of Šubelj et al. (2011) and Chau et al. (2006) as they use intrinsic features only to bootstrap the network learning algorithms. In order to develop a comprehensible and usable technique for experts, we extend the intrinsic features with domain-driven network features. As such, we offer experts the opportunity to gain insights about the importance of each of the variables in the fraud detection process. Given that current research does not offer an encompassing approach, we developed *GOTCHA!*.

2.4. Proposed Fraud Detection Process

In order to make the *GOTCHA!* approach useable, it needs to be embedded in the global context of the fraud detection process. The goal of social security fraud detection is to define which companies are likely to commit fraud within a certain period of time. Currently, social security experts have mainly focused on manually inspecting random companies and determining whether they are involved in fraud or not. This section discusses how we propose to extend the current process. The fraud detection process is illustrated in Figure 4.

Fraud detection is the automated process of identifying high-risk instances. For reasons of generality, we use the term *Automated Detection Algorithms* to refer to any technique that is able to estimate a fraud detection model, such as tree models, linear or logistic functions, SVMs, ANNs, Bayesian learning, ensemble models, etc. (Hastie et al. 2001, Carrizosa et al. 2014). During *fraud investigation*, experts decide to agree or disagree with the high-risk companies identified by the model using their practical insights and knowledge. Note that, currently, experts are not guided as to which companies are potentially high-risk. This makes the fraud investigation process inefficient and time-consuming. The high-risk companies are passed on to the field auditors who finally confirm if their expectations are correct (*fraud confirmation*).

Observe the interactive nature of such a system: while experts feed the fraud detection algorithms with confirmed fraud, our algorithm guides the experts in turn where to look for fraud. In the end,

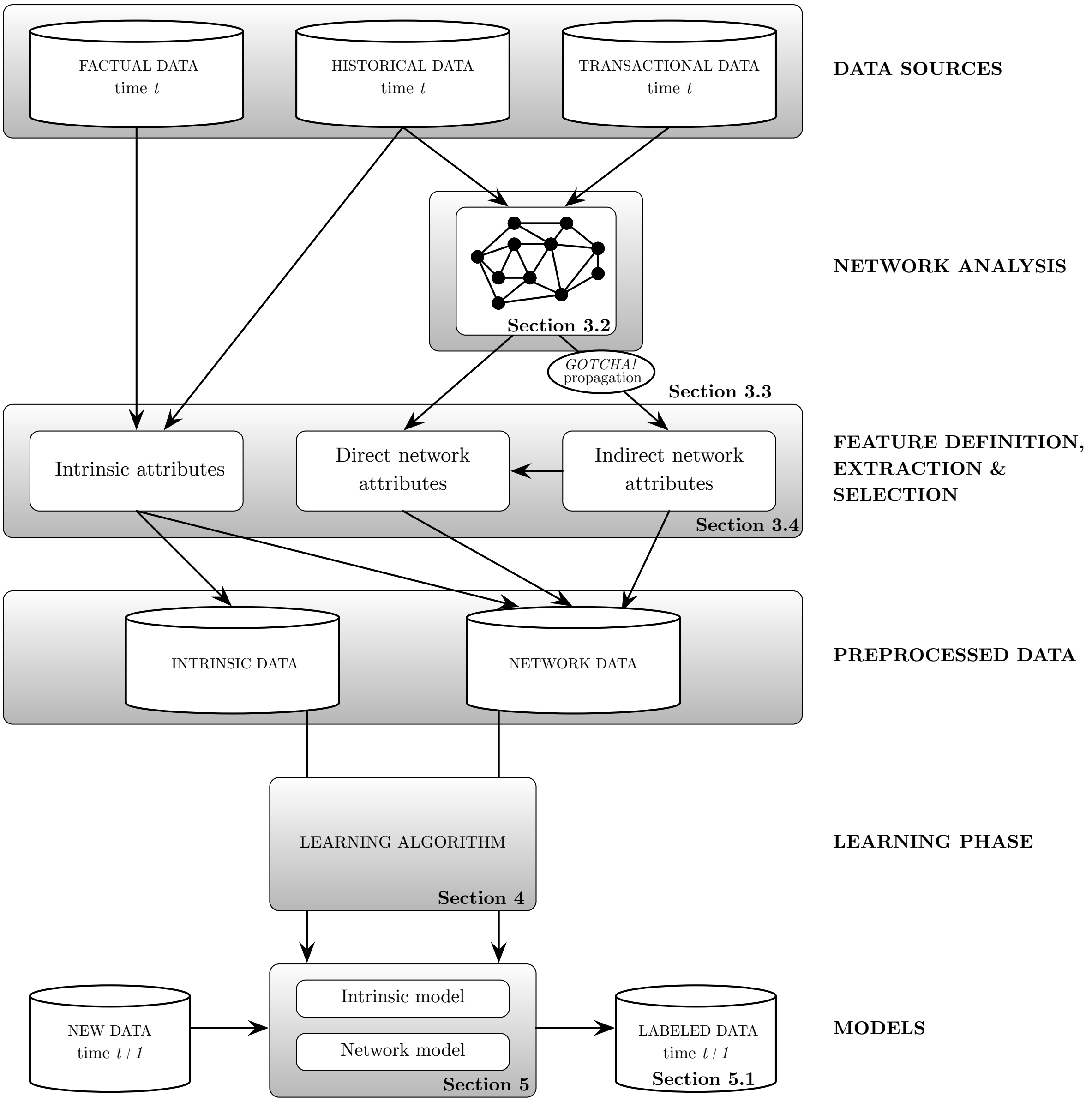


Figure 5 Proposed GOTCHA! framework for social security fraud detection.

the ultimate goal is to evolve towards *fraud prevention*, i.e., the ability of detecting fraud before it is even committed (Bolton and Hand 2002).

This paper studies the fraud detection phase by proposing *GOTCHA!*. The next section will discuss the fraud detection process in more detail. We expect that our process is more efficient and systematic than experts merely following their own intuition. Our estimated models give a good indicator which companies are likely to commit fraud (see Section 5).

COMPANY	Intrinsic Variables								Network Variables					Fraud?
	Regional		Sectorial		Historical		Legal		Direct			Indirect		
	province	...	sector	...	age	...	form	...	degree	quadrangles	...	exposure	...	
6357	P4	...	catering	...	3	...	Corp.	...	16	0	...	0.12	...	***No***
3904	P7	...	transport	...	1.5	...	PLLC	...	8	0	...	0.01	...	***No***
3041	P5	...	cleaning	...	0.7	...	LLC	...	56	8	...	0.65	...	***Yes***
7932	P2	...	catering	...	8	...	Corp.	...	93	7	...	0.03	...	***No***
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Figure 6 **Example of a preprocessed data set.**

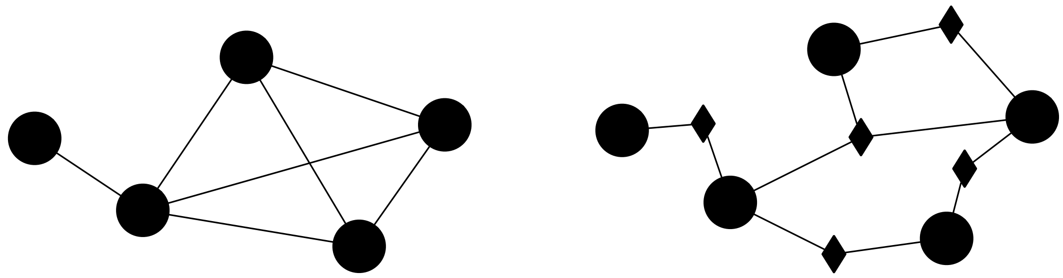
2.5. *GOTCHA!*'s Fraud Detection Framework

Figure 5 illustrates in greater detail our proposed framework for the fraud detection phase (see Figure 4) in a social security context. We start from three data sources. A factual data source contains company-specific information such as regional, sectorial and legal characteristics of each company. Historical data log changes in company information, e.g., when a company changes its legal seat. Transactional data record which resources are associated to which companies, including the time period. Those data sources are transformed into relevant company-specific and network-centric attributes. Transactional data form the basis to create the global network structure representing the relationships between companies and resources as a bipartite graph (Section 3.2). As historical relationships between companies and resources contain important information, we use the historical data sources to reconstruct historical links and add them to the network, weighing the links based on their recency. While the past and the present is explicitly implemented in such a network, future behavior can be estimated by exploiting both direct effects as well as collectively inferring fraud through the whole network (Section 3.3). Approximately 350K active and non-active companies and 5 million resources are considered in the network.

According to Verbeke (2012), variables can be classified into two categories:

Definition 1. A **local** or **intrinsic variable** represents intrinsic information of a company as if it was treated in isolation. Those variables include regional, sectorial, historical and legal characteristics.

Definition 2. A **network variable** aggregates information that is contained by the neighborhood of each company. We assume that behavior of a company's neighbors has an influence on the company itself. Those variables include the degree, triangles and propagated exposure score (see Section 3.4 for details), and can be classified as direct and indirect network variables depending on whether they are derived from the direct neighborhood or take into account the full network



(a) (b)

Figure 7 **Overview of a unipartite (a) and a bipartite (b) graph.**

structure.

Figure 6 gives an example of the preprocessed data, and features of each category. We derive regional, sectorial and legal variables from the factual data source; the historical features are extracted from the historical data. The transactional data source is the basis for the creation of the network variables and specifies which resources are assigned to which companies for which time period (see Section 3).

In the remainder of this paper, we will use the terms intrinsic and network variables to indicate whether the variables are generated by instance-specific or network-centric information. The data preprocessing phase derives intrinsic, direct and indirect network attributes. Rather than using plain relational classifiers as proposed by (Macskassy and Provost 2007) to predict fraud, the network data set imposes a mix of intrinsic and domain-driven network attributes. A learning algorithm will then estimate the corresponding models (Section 4). Those models are used to evaluate fraudulent behavior of companies (Section 5).

3. Network Analytics for Fraud Detection

3.1. General Concepts and Notations

Our proposed approach is based on fundamentals from graph theory, incorporating *Challenge IV* of Section 2.2. Boccaletti et al. (2006) define graph theory as the natural framework for the exact mathematical treatment of complex networks. Formally, a complex network can be represented as a graph. A graph consists of a set of *vertices* $v \in \mathcal{V}$ and *edges* $e \in \mathcal{E}$. Vertices – also referred to as nodes or points – are connected by edges – also known as links or lines. A standard graph can thus mathematically be represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and is shown in Figure 7a. A graph can be either *directed* or *undirected*, depending on the direction imposed on the edges. When edges define the capacity or the intensity of a connection (Boccaletti et al. 2006), the graph is said to be *weighted*. Mathematically, a graph is represented as a matrix. The adjacency matrix $\mathbf{A}_{n \times n} = (a_{i,j})$ is the corresponding matrix representation of size $n \times n$ of a graph, with n being the total number of vertices and $a_{i,j} = 1$ if a link between node i and j exists, and $a_{i,j} = 0$ otherwise. The weight matrix $\mathbf{W}_{n\times n} = (w_{i,j})$ captures the link weight of the relationships between the nodes.

Most networks contain only one node type. Certain applications, however, require implementing a second entity. Such networks are bipartite graphs, as shown in Figure 7b. In contrast to unipartite graphs, a bipartite graph consists of two types of vertices $v_1 \in \mathcal{V}_\mathbf{1}$ and $v_2 \in \mathcal{V}_\mathbf{2}$. An edge $e \in \mathcal{E}$ exclusively connects objects from different classes to each other. For each edge in a bipartite graph, the following property holds:

$$e(v_1, v_2) \in \mathcal{E} | v_1 \in \mathcal{V}_\mathbf{1} \text{ and } v_2 \in \mathcal{V}_\mathbf{2} \tag{1}$$

This property enforces that two instances of the same class are never directly connected, but always connect through an object of the other class. The adjacency matrix of an undirected bipartite graph is formally written as $\mathbf{A}_{n\times m} = (a_{i,j})$, with $a_{i,j} = 1$ if a link between node $i \in \mathcal{V}_\mathbf{1}$ and node $j \in \mathcal{V}_\mathbf{2}$ exists, and $a_{i,j} = 0$ otherwise. The corresponding adjacency matrix has a size of $n \times m$, with n and m the number of objects in set $\mathcal{V}_\mathbf{1}$ and $\mathcal{V}_\mathbf{2}$ respectively. The weight matrix is $\mathbf{W}_{n\times m} = (w_{i,j})$.

3.2. Time-weighted Bipartite Networks

Reality is often difficult to capture in mathematical formulations or even a graphical representation. Network analysts, in consideration with field experts, should carefully choose and agree upon the right design of the network, reflecting the reality in the best possible way. It is particularly important to bridge the richness of experts' knowledge to the technical limitations of network analytics by selecting the most relevant data features for the analysis.

We argued in Section 1 that in a social security fraud detection problem companies are related to their resources. The goal of fraud detection is to find high-risk companies, but the resources are an important indicator as they help in executing the company's (fraudulent) activities. Resources are transferred from company to company. If a currently legitimate company inherits resources from a fraudulent company, this substantially increases the fraud risk of that company. Hence, we create a bipartite graph (or bigraph) connecting companies to their past and present resources. We work with undirected networks as fraud can pass from a company to a resource, and vice versa.

For computational reasons, the graphical representation is mapped into a weight matrix $\boldsymbol{W}$ with size $c \times r$, where c and r specify the number of companies and resources respectively. The strength of the relationship between a company and resource is exponentially weighted in time:

$$\begin{cases} w_{i,j} = e^{-\gamma h} & \text{if a relationship exists between company } i \text{ and resource } j \\ w_{i,j} = 0 & \text{otherwise} \end{cases} \tag{2}$$

with γ the decay constant[4], and h the time passed since the resource was linked to the company, with $h = 0$ representing a current relationship. The value of the decay constant γ indicates the rate at which past information declines, and is chosen (by mutual agreement with the experts) such that only limited past information is taken into account. Particularly, if experts say that the associations can be considered as irrelevant after x days, then we choose γ such that the decay function goes to zero for time values greater than x, i.e., $f(t > x) \approx 0$. For example, if one decides that information of only 5 years back should be taken into account, then $\gamma \approx 1$.

The matrix $\boldsymbol{W}$ is time-dependent. To incorporate the time-evolving characteristics of fraud (cfr. *Challenge III* in Section 2.2), we create a matrix $\boldsymbol{W}_t$ for each timestamp $t \in \{t_0, t_1, t_2, t_3\}$, representing the interrelated structure at time t. The social security bigraph contains approximately 350K active and non-active companies and 5 million active and non-active resources in every timestamp of analysis. In each timestamp, the network density is around 4.5×10^{-6}.

3.3. *GOTCHA!*'s Fraud Propagation Algorithm: Defining high-risk nodes in the network

This section handles *Challenge V* (see Section 2.2). In particular, we answer the following questions: (1) Which *resources* are often involved in fraud and exhibit a high risk to entice other companies to perpetrate fraud as well? (2) Which *companies* are sensitive to fraud? More specifically, we need a score that indicates which *resources* are coincidentally associated with fraudulent companies (low-risk) and which resources systematically pop up when fraud is detected (high-risk). For example, assume an address that was previously used by a fraudulent company is taken over by another company. What would you say about the riskiness of that resource? Would the resource riskiness change if you knew that the address was already used by many fraudulent companies previously, or if the address was the location of only one fraudulent company many years ago? Similarly, we derive a score that gives a primary indication of how the *company* is affected by the fraudulent influences from its neighborhood.

Given a time-weighted bipartite graph of companies and resources, we infer an *exposure score* for every node (i.e., resource and company) in the network. The exposure score expresses the extent to which the node is affected by fraud. As only companies are directly attributed to fraud, we start from the label of the few confirmed fraudulent companies. The bipartite graph allow to spread fraudulent influence through the network and define an exposure score for each company and resource. As such, each company can be analyzed based on its own exposure score and the links to high- and low-risk resources.

[4] Due to confidentiality issues, we will not elaborate on the exact value of γ.

We start from the Personalized PageRank algorithm (Page et al. 1998), one of the popular applications of the *Random Walk with Restarts* (RWR) method (Gleich 2014). In the fraud domain, this algorithm is already exploited by Gyöngyi et al. (2004) and is the basis of their TrustRank algorithm to detect web spam. We extend the Personalized PageRank algorithm in order to meet the following domain-specific requirements:

1. *Bipartite graphs*: fraud contaminates both companies and resources.
2. *Focus on fraud*: only fraud – and no legitimate effects – propagates through the network.
3. *Dynamics*: fraud is evaluated upon its recency.
4. *Degree-independent propagation*: high-degree companies spread proportionally more fraud than low-degree companies.

In general, the Personalized PageRank algorithm computes an exposure score for each node which depends on (a) the exposure scores of the node's neighborhood and (b) a random jump towards another node in the network. Mathematically, this can be written as,

$$(\vec{\xi}) = \alpha \cdot \boldsymbol{A}(\vec{\xi}) + (1-\alpha) \cdot \vec{v} \tag{3}$$

with $(\vec{\xi})$ a vector containing the exposure scores of the nodes, $\boldsymbol{A}$ the corresponding column-normalized adjacency matrix, α the restart probability and $\vec{v}$ the restart vector. The restart vector $\vec{v}$ is uniformly distributed over all nodes, and normalized afterwards.

Solving Equation 3 requires a matrix inversion. This is often not feasible to compute in practice. The most widely used way to compute the relevance score is by the power iteration method, which iterates until convergence (Tong et al. 2006). Convergence is reached until the change is marginal, or after a maximum number of iteration steps. Next, we discuss how we integrate the fraud-specific domain requirements into the algorithm.

Requirement 1 Equation 3 is developed for unipartite graphs. We want to assess the extent to which fraud affects both companies and resources. Starting from the weighted adjacency matrix $\boldsymbol{W}_{c\times r}$ of the bipartite graph with c companies and r resources (see Section 3.2), the matrix is transformed to a unipartite representation, according to (Tong et al. 2008),

$$\boldsymbol{Q} = \begin{pmatrix} 0_{c\times c} & \boldsymbol{W} \\ \boldsymbol{W'} & 0_{r\times r} \end{pmatrix} \tag{4}$$

Matrix $\boldsymbol{Q}$ is a symmetric matrix with $c+r$ rows and columns. Introducing zeros enforces that resources exclusively connect to companies and vice versa. The column-normalized matrix is

$\boldsymbol{Q_{norm}}$, a matrix where all columns sum to 1. The iterative propagation procedure for bipartite graphs can then be written as,

$$(\vec{\xi}) = \alpha \cdot \boldsymbol{Q_{norm}}(\vec{\xi}) + (1-\alpha) \cdot \vec{v} \tag{5}$$

Note that $\boldsymbol{Q_{norm}}$ is a dynamic matrix, representing both present and past relationships. All active and non-active companies are included. This allows us to integrate and exploit all connections (ever established) among companies and resources. The vectors $\vec{\xi}$ and $\vec{v}$ are of size $c+r$, containing the exposure scores and restart probabilities of the companies and the resources.

Requirement 2 The goal is to focus on fraud and exclusively propagate fraudulent influence through the network. A similar approach is taken in Provost et al. (2009) to compute brand affinity, measuring the proximity of a node to the seed nodes. Seed nodes are nodes that already are enticed about the product or, in our case, into fraud. Given information provided by seed nodes, how will this affect the other currently legitimate companies and resources in the network? We use the restart vector to personalize the ranking towards fraud and stress the fraudulent influences of the seed nodes. The restart vector specifies which nodes (here: companies) committed fraud, where $v_j = 1$ if entry j is a fraudulent company and $v_j = 0$ if entry j is a resource or a legitimate company. Although there is a lack of evidence of confirmed fraud nodes, the algorithm is able to cope with only few labeled nodes by emphasizing fraud in the restart vector.

Requirement 3 Fraud is dynamic. Recently caught companies are a more important source of spreading fraud than companies detected many years ago. The restart vector reflects the fraudulent influence a certain company can disperse, and should depend on the recency of the fraud. The more time passed since fraud was detected, the lower a particular fraudulent company's influence. Inspired by the half-time decay of nuclear particles, we exponentially decay the relevance of fraudulent activities over time,

$$\begin{cases} v_j = e^{-\beta h} & \text{if entry } j \text{ is a fraudulent company} \\ v_j = 0 & \text{otherwise} \end{cases} \tag{6}$$

with β the decay constant (see Section 3.2 for details), and h the time passed since the company was detected fraudulent where $h = 0$ represents a current fraud company.

Requirement 4 Fraudulent companies infect their surrounding resources directly. However, low-degree companies have fewer links through which fraud can propagate and affect the resources

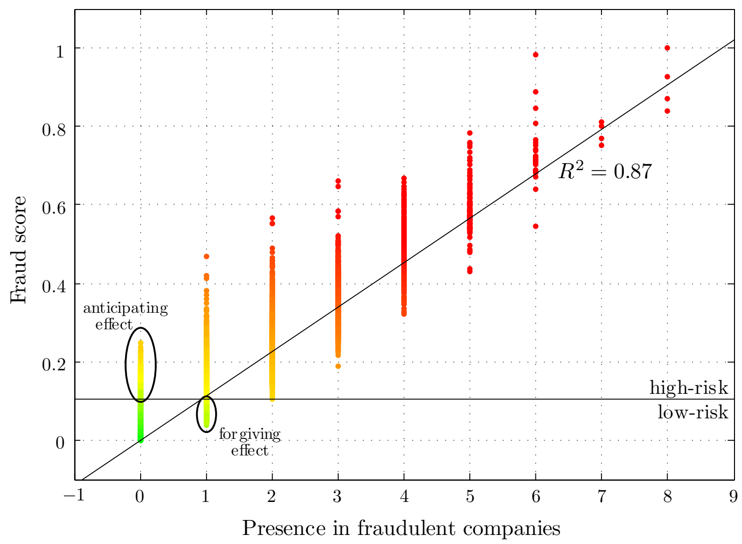


Figure 8 Each resource is associated with its propagated exposure score and its presence in fraudulent companies. The resources are colored according to their riskiness (red indicates high risk, green is low risk). The horizontal line represents the boundary dividing the resources in a low-risk and high-risk category. Note that only 0.28% of all resources are labeled as high-risk.

more strongly. High-degree companies have many links, resulting in a marginal impact on the neighboring nodes. In realistic situations, this assumption does not hold. The influence of high-degree companies should be equally treated as low-degree companies, as high-degree companies have a wider range to influence other companies. Hence, fraud propagation has to be proportional to a node's degree, and

$$\vec{z} = \vec{v} \odot \vec{d} \tag{7}$$

with $\vec{z}$ the degree-adapted restart vector, which is the element-wise product of the restart vector $\vec{v}$ and the degree vector $\vec{d}$ denoting the degree of each entry. The normalized vector is $\vec{z}_{norm}$.

After $k+1$ iterations, the exposure score for each company and resource equals

$$\vec{\xi}_{k+1} = \alpha \cdot \boldsymbol{Q_{norm}} \cdot \vec{\xi}_k + (1-\alpha) \cdot \vec{z}_{norm} \tag{8}$$

with α the restart probability[5], $\boldsymbol{Q_{norm}}$ the column-normalized adjacency matrix, $\vec{z}_{norm}$ the normalized degree-adapted restart vector, $\vec{\xi}_k$ a vector containing the exposure scores of all nodes after k iterations, and $\vec{\xi}_0$ the initial distribution. Note that the final scores are independent of the initial values of $\vec{\xi}_0$(Page 2001). We repeat the process for 100 iterations in order to make sure that

[5] based on Page et al. (1998), we choose $\alpha = 0.85$

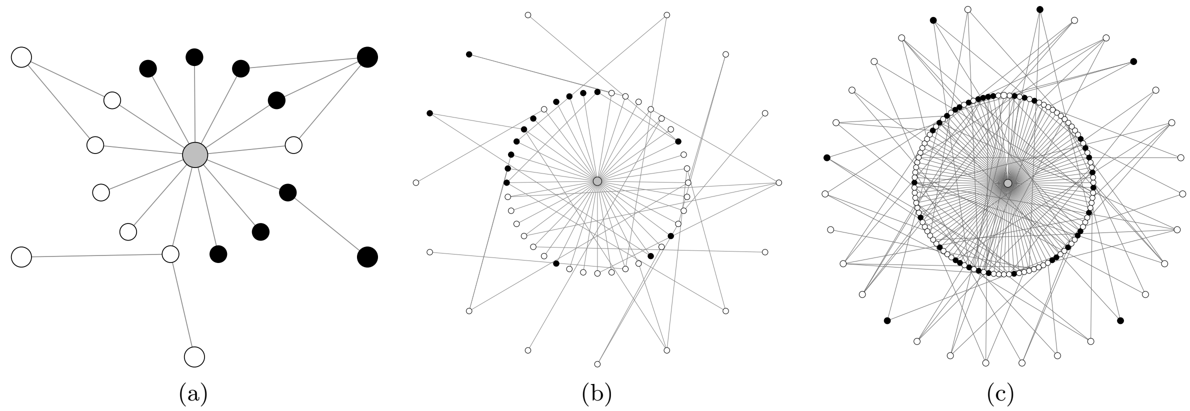


Figure 9 Various egonets for micro- (a), small- (b) and medium-sized (c) companies. The company is the center (i.e., the ego) of the egonet and is surrounded by its resources (i.e., the alters). High-risk resources are labeled in black, low-risk nodes are white-colored. All central companies (egos) are still active at the time of analysis.

potential changes in the final exposure score are only marginal.

Apart from a company score, the *GOTCHA!* propagation algorithm also assigns an exposure score to each resource. Note that the interpretation of the exposure scores of both companies and resources is the same: it expresses the extent to which the company/resource is exposed to fraud. Figure 8 shows the exposure scores of the resources compared to their presence in fraudulent companies (for year t_0). In general, 87% of the variation in the resources' exposure score is explained by their presence in fraudulent companies. While certain resources were never associated with fraudulent companies before, they receive a relatively high exposure score. This means that, although those resources are not directly contaminated by fraudulent activities, they are surrounded by a huge amount of fraud. We call this the *anticipating effect* of *GOTCHA!*'s fraud propagation. On the other hand, some resources have been involved in fraudulent companies, but received a low fraud score. Due to the incorporation of the recency of fraud in the propagation algorithm, there is a *forgiving effect* present. When time evolves and resources were not involved in fraud again, their fraudulent influence decreases and is only marginal.

In agreement with social security fraud experts, *GOTCHA!* considers resources involved in at least two fraudulent companies always as high-risk. The minimum exposure score of the resource connected to at least two fraudulent companies is chosen as the cut-off value to distinguish between low- and high-risk resources. The horizontal line in Figure 8 illustrates this cut-off value. Resources located above the cut-off line are marked as high-risk. Note that this corresponds to only 0.28% of all resources.

Having an estimated probability of the riskiness of the resources, we are now able to characterize each company based on its connectivity to high- and low-risk resources.

Feature	Description	Unipartite	Bipartite	*GOTCHA!*
DIRECT FEATURES				
Neighborhood Degree	number of first-order neighbors that are of			
high-risk	- high-risk	*X*	*X*	*X*
low-risk	- low-risk	*X*	*X*	*X*
relative	proportion of high-risk neighbors	*X*	*X*	*X*
Time-Weighted Degree	time-weighted[6] number of first-order neighbors that are of			
high-risk	- high-risk			*X*
low-risk	- low-risk			*X*
relative	proportion of high-risk nodes, weighted in time			*X*
Triangles	number of closed triples in the neighborhood that contain			
high-risk	- at least one high-risk node	*X*		
low-risk	- no high-risk nodes	*X*		
relative	proportion of triples that contain at least one high-risk node	*X*		
Quadrangles	number of quadrangles in the extended neighborhood that contain			
high-risk	- at least one high-risk company node		*X*	*X*
time-weighted	- at least one high-risk company node, weighted in time			*X*
low-risk	- no high-risk company nodes		*X*	*X*
time-weighted	- no high-risk company nodes, weighted in time			*X*
relative	proportion of quadrangles that contain at least one high-risk company node		*X*	*X*
time-weighted	- weighted in time			*X*
Quadrangle Frequency	quadrangles in the extended neighborhood that contain the same two first-order neighbors, and have			
mean (high-risk)	- at least one high-risk company node, averaged		*X*	*X*
time-weighted	- at least one high-risk company node, averaged and weighted in time			*X*
max (high-risk)	- at least one high-risk company node, maximum		*X*	*X*
time-weighted	- at least one high-risk company node, maximum and weighted in time			*X*
mean (low-risk)	- no high-risk company nodes, averaged		*X*	*X*
time-weighted	- no high-risk company nodes, averaged and weighted in time			*X*
max (low-risk)	- no high-risk company nodes, maximum		*X*	*X*
time-weighted	- no high-risk company nodes, averaged and weighted in time			*X*
Neighborhood Similarity	count of similar neighbors	*X*	*X*	*X*
INDIRECT FEATURES				
Exposure Score	node's own exposure score	*X*	*X*	*X*
Neighborhood Exposure	first-order neighbors' exposure score			
mean	- averaged	*X*	*X*	*X*
weighted mean	- time-weighted			*X*
maximum	- maximum	*X*	*X*	*X*

Table 2 Network-based feature extraction.

3.4. Network Feature Extraction

Given all legitimate companies at time t, we want to rank those companies according to their *fraud risk* – i.e., the probability that they will commit fraud in the near future. As this risk depends on a combination of intrinsic and network-based variables, we need to transform network information to a set of promising network-based features for each active company (Eliassi-Rad and Henderson 2011). We infer two types of network-based features: direct and indirect features. The direct network features are derived from each company's direct neighborhood. Given the bipartite structure of our network, for each company we take into account all nodes that are one and two hops removed from the center (i.e., a company's associated resources and companies). Figure 9 illustrates the direct neighborhood of a company with varying neighborhood size. The indirect network features are derived from the exposure scores which use the whole network rather than a node's neighborhood. Table 2 gives an overview of the features derived from the network.

Our approach *GOTCHA!* is evaluated against three baselines: (1) a model without network features, (2) a model with unipartite features, and (3) a model with bipartite features not time-weighted. In (2), companies are directly linked to each other. The link weight expresses the number of shared resources between both companies. Here, the direct features are derived from the first-order neighborhood as this explicitly comprises the associated companies. In (3), the network has a bipartite structure, but the links are not weighted in time. For each company, the unipartite model (2) extracts the following direct features: degree, triangles, neighborhood similarity. The degree counts the number of neighbors. Since the impact of high-risk neighbors is an important indicator of fraud, we distinguish between the number of first-order high-risk and low-risk neighbors, and the ratio hereof. Remark that a node is classified as high-risk if the node is a fraudulent company or if the node has a sufficient large exposure score as explained in Section 3.3. A triangle is defined as three nodes that are all connected to each other. We say that a triangle has a high-risk if at least one of the associated nodes is classified as high-risk. Neighborhood similarity measures the extent to which the characteristics of the neighbors are similar to the node of interest. Here, we compare companies based on location and sector-specific information, guided by expert expectations.

The indirect features include the company's own exposure score and the exposure scores of the first-order neighborhood aggregated by the mean and maximum. The exposure score is computed according to Equation 3 where the restart vector incorporates fraud (Requirement 2). The bipartite model (3) derives the same set of features as the unipartite model, with the exception of triangles. In our bipartite network structure where companies (resources) are exclusively connected to resources (companies), no triangles exist. However, a shift of many resources from one company to another might indicate the existence of a spider construction. Hence, we count the number of quadrangles

Feature	Summary Statistics			
	Fraud		Non-Fraud	
	μ	σ	μ	σ
DIRECT FEATURES				
Neighborhood Degree				
high-risk	31.37	39.09	13.13	48.12
low-risk	2.00	8.43	5.55	18.44
relative	0.91	0.23	0.55	0.39
Time-Weighted Degree				
high-risk	19.15	22.01	6.70	17.86
low-risk	0.56	4.10	2.64	7.20
relative	0.93	0.23	0.56	0.45
Quadrangles				
high-risk	56.71	198.83	0.62	37.04
time-weighted	45.34	155.41	0.17	13.16
low-risk	131.30	294.40	375	108729.60
time-weighted	55.88	162.35	134	36197.67
relative	0.19	0.32	0.0040	0.0048
time-weighted	0.23	0.36	0.0036	0.0046
Quadrangle Frequency				
mean (high-risk)	0.64	0.87	0.03	0.23
time-weighted	0.28	0.40	0.0075	0.0641
max (high-risk)	1.52	2.46	0.0398	0.3328
time-weighted	0.62	0.98	0.0104	0.0956
mean (low-risk)	0.89	0.57	0.45	0.56
time-weighted	0.47	0.38	0.19	0.27
max (low-risk)	1.75	1.59	0.64	1.22
time-weighted	0.95	0.85	0.31	0.50
Neighborhood Similarity				
Sector	0.60	0.49	0.73	0.45
Location I	0.55	0.50	0.45	0.50
Location II	0.01	0.11	0.01	0.12
INDIRECT FEATURES				
Exposure Score	0.0027	0.0046	3.565e-5	2.569e-4
Neighborhood Exposure				
mean	0.0106	0.01756	3.381e-4	2.024e-3
weighted mean	8.49e-3	0.0156	1.93e-4	1.668e-3
maximum	0.0353	0.0467	0.0028	0.0119

Table 3 Network-based feature extraction.

– i.e., a closed path of four nodes – in the extended neighborhood where we both include the first- and second-order neighborhood. We say that a quadrangle is of high risk if at least one high-risk company node is associated with the quadrangle. For each pair of resources, the quadrangle frequency measures how many times they are both included in a quadrangle. At company level, we summarize this feature by the mean and maximum amongst all pairs of resources associated with that company. The features in *GOTCHA* differ from those of (3) as they are time-weighted by the edges. For high-risk degree for example, this means that we sum the edge weight of the associated high-risk resources. The value of a weighted quadrangle is determined by the arithmetic mean of the link weights (Opsahl and Panzarasa 2009). We also derive the weighted mean of the first-order neighbors' exposure scores, weighing the impact of each node's exposure score by the edge weight.

We construct the features for each timestamp $t \in \{t_0, t_1, t_2, t_3\}$ and hence take into account the time-evolving property of fraud. Together with the intrinsic features, these network-based features are fed into a learning algorithm. An overview of the features' summary statistics for year t_3 can be found in Table 3.

4. Modeling Approach

The social security institution keeps track of fraudulent companies and labels them fraudulent as soon as suspicious activities are discovered. Having an extensive database containing time-related records, we are able to evaluate time-consistent models at different timestamps and time windows. In our analysis, we define four timestamps $t \in \{t_0, t_1, t_2, t_3\}$. For each timestamp, we specify within which time window the learning algorithm has to predict whether a company will be fraudulent or not. We evaluate the models on their detection of short-, medium- and long-term frauds. For instance, a short-term model estimates the probability of short-term fraud. The time windows are set to 6, 12 and 24 months, by experts' agreement.

A key challenge in predicting social security fraud is making the right trade-off between a small time window that accurately reflects current types of fraud, and a larger time window which provides more confirmed evidence of fraud and anticipates new fraudulent structures.

As mentioned, in order to evaluate the relevance of relational information in fraud prediction, we compare the *GOTCHA!* model with three baselines. The same instances are used in the training and test sets for the baselines and *GOTCHA!* network model. By doing so, we are able to determine the added value of incorporating relational knowledge (in terms of network-based features) on the performance of the prediction models. We discuss each of the models in more detail below.

Baseline - Intrinsic – is trained and tested with intrinsic-only variables. Relationships with other companies and resources are neglected in the analysis.

Baseline - Unipartite – integrates intrinsic and network-based variables into one model (see *Challenge II* in Section 2.2). The network only consists of companies that are linked to each other by means of resources. Link weight is defined as the number of resources that both companies share.

Baseline - Bipartite – integrates intrinsic and network-based variables into one model (see *Challenge II* in Section 2.2). The network includes both companies and resources. A binary link weight is imposed, defining whether a link exists between a company and a resource.

Proposed GOTCHA! model – enriches the bipartite model with time-weighted network features.

4.1. Rebalancing the data set

To address the extremely skewed data distribution (see *Challenge I* in Section 2.2), we use the *SMOTE* approach (Chawla et al. 2011) to rebalance the data set. *Synthetic Minority Oversampling Technique (SMOTE)* is a combination of oversampling the minority class and undersampling the majority class (Chawla et al. 2011). Based on the experimental results of Chawla et al. (2011), we choose an oversampling and undersampling percentage of 400% and 200% respectively.

4.2. Learning algorithm

Random Logistic Forests and *Random Forests* are implemented to train the models. We opt for ensemble methods as individual logistic regression or decision trees often fail to appropriately weigh features based on their predictiveness (Gallagher et al. 2008), which our data set confirmed (see Section 5). Breiman (2001) proposed Random Forests, an ensemble of trees. Random Logistic Forests, as proposed by (Gallagher et al. 2008), is an ensemble of plain vanilla logistic regressions, where each classifier is fed with $||log(N)+1||$ random features, with N the total number of features. The final label assigned to an instance is based on the majority vote of each individual model. We estimate an ensemble of 500 individual models, each with 6 random features.

Using ten-fold stratified cross-validation, we enforce the learning algorithms to use each instance once in the test set. Stratified sampling ensures that each sample represents the real fraud distribution. As such, we can average the results, obtaining more stable performance measures of each of the models and resulting in a better impression of the significance of the different types of variables.

In summary, our experiments are designed to answer the following questions: 1) Do network-based variables yield better performance over intrinsic-only variables? If so, by how much? 2) Is the incorporation of a bipartite, time-weighted network structure essential? 3) Are network models able to capture changes in the environment? Do they statistically perform better as the baselines over the different timestamps? 4) Are the network models able to identify companies that will perpetrate fraud in the near future and also on long term?

AUC Performance	Year t_0			Year t_1			Year t_2			Year t_3		
	ST	MT	LT	ST	MT	LT	ST	MT	LT	ST	MT	LT
(1) Baseline - Intrinsic												
Random Log. Forest	0.8438	0.8868	0.8232	0.8604	0.8310	0.7802	0.8473	0.8074	0.7540	0.7343	0.7381	0.7288
Random Forest	0.8619	0.8782	0.8183	0.8841	0.8514	0.7988	0.8247	0.8272	0.7938	0.7805	0.7792	0.7619
(2) Baseline - Unipartite												
Random Log. Forest	0.8962	0.9151	0.8650	0.9167	0.8715	0.8277	0.8953	0.8679	0.8076	0.7854	0.7702	0.7721
Random Forest	0.9056	0.9104	0.8691	0.9300	0.8924	0.8436	0.8816	0.8742	0.8159	0.8267	0.8125	0.8126
(3) Baseline - Bipartite												
Random Log. Forest	0.8749	0.8893	0.8517	0.8910	0.8652	0.8101	0.8698	0.8262	0.7826	0.7798	0.7652	0.7357
Random Forest	0.8907	0.8867	0.8726	0.9075	0.8910	0.8325	0.8670	0.8543	0.8095	0.8221	0.8250	0.7897
(4) GOTCHA!												
Random Log. Forest	**0.9233**	0.9281	0.9066	**0.9534**	0.9380	0.8943	0.9053	0.8953	0.8707	0.9035	0.8877	0.8567
Random Forest	0.9173	**0.9312**	**0.9246**	0.9507	**0.9409**	**0.9074**	**0.9069**	**0.9044**	**0.8755**	**0.9176**	**0.9114**	**0.8953**

Table 4 AUC scores of the baseline and GOTCHA! models.

5. Results

In this section, we discuss the results of our *GOTCHA!* network model compared to the baselines. All models are evaluated in terms of the AUC score (Area Under ROC Curve), precision and recall. We use an extensive time-dependent data set obtained from the Belgian Social Security Institution. For each timestamp, approximately 220,000 active companies and more than 5 million resources are registered. Our goal is to find companies that exhibit a high risk of perpetrating fraud. We extract intrinsic features that describe the current characteristics of a company, and network-based features that take into account the present and past relationships to the resources. We train and test models based on fraudulent companies found and confirmed by experts. We analyze the difference in performance between the baselines and the *GOTCHA!* model, as well as the difference in performance for the various time windows (i.e., short, medium and long term).

Do network-based features boost the performance of traditional models that only use intrinsic features? That is, does the *GOTCHA!* model significantly outperform the baselines? As opposed to existing methods (Chau et al. 2006, Šubelj et al. 2011) which bootstrap the network propagation algorithm with the output of an intrinsic model, we opt to include both intrinsic and domain-driven network-based features in the final model. There are two reasons. First, our approach indicates which variables (including intrinsic variables) contribute to fraudulent behavior, and as a consequence, experts will gain insights in the current fraud process. Second, we start from a set of confirmed fraudulent companies to initialize the propagation algorithm which other methods lack.

Table 4 outlines the average AUC score for the different estimated models, based on 10-fold cross validation. The results show that the intrinsic baseline can be improved by including network-based variables. The unipartite baseline (1) performs significantly better than the intrinsic baseline (2) at a significance level of 0.05 (except for year t_0 on short term and t_2 on long term for the

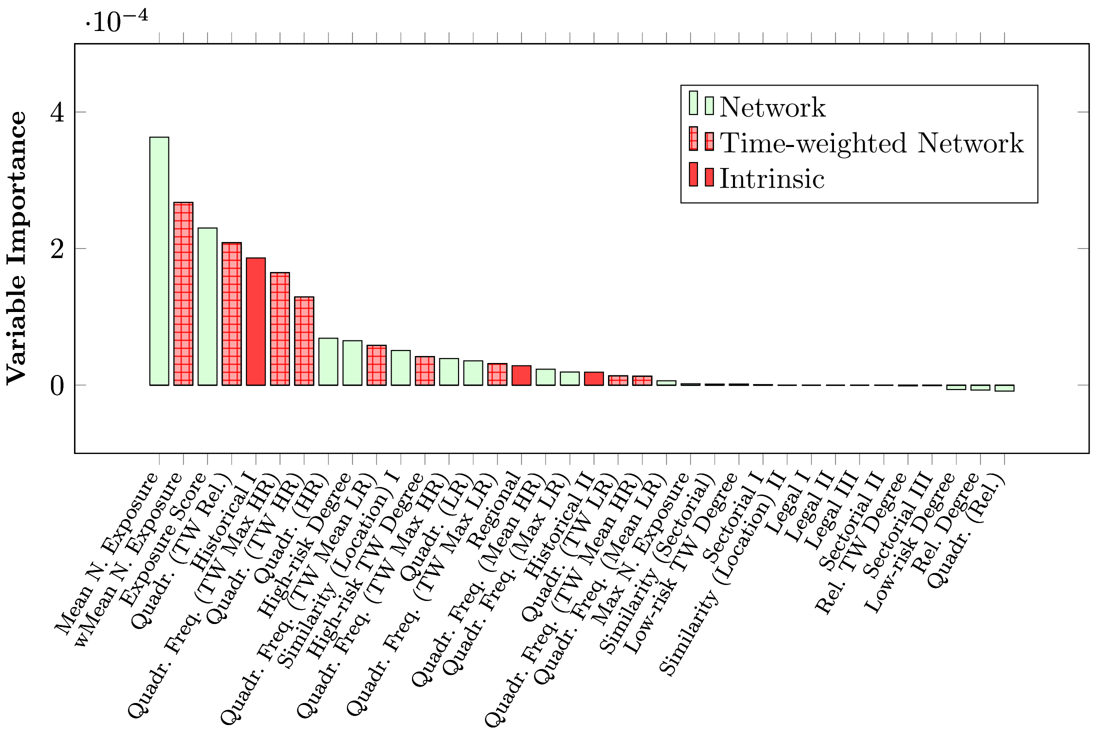


Figure 10 **Variable Importance of Random Forest for timestamp t_3. (TW = time-weighted; LR = low-risk; HR = high-risk; N = neighborhood)**

Random Forest model). We conclude that network-based variables boost the performance of the fraud detection models. Remember, link weight in a unipartite network represents the number of shared resources between two companies. Including resources as a separate entity in the network, allows us to integrate time in the bipartite network by the link weight between a resource and a company. We find that the bipartite baseline (3) without time-weighted edges does improve the intrinsic baseline (1), but does not outperform the unipartite baseline (2). The *GOTCHA!* model (4) significantly surpasses all baselines (1)-(3) in terms of AUC score from which we can conclude that features derived from a time-weighted bipartite network are an important enrichment for fraud detection models.

Ensemble methods perform better than the individual models. We compare a Decision Tree model to Random Forests, and Logistic Regression to Random Logistic Forests, and find that the highest performance in terms of AUC score is achieved with ensemble models. For brevity, we omit the model details.

Which variables (or variable categories) are mainly responsible for the performance of our *GOTCHA!* models? Figure 10 depicts the variable importance of Random Forests in year t_3 when we are testing long-term fraud. Variable importance of each feature is measured by permutating

Feature	**Year t_0**			**Year t_1**			**Year t_2**			**Year t_3**		
	ST	MT	LT	ST	MT	LT	ST	MT	LT	ST	MT	LT
DIRECT FEATURES												
Neighborhood Degree												
high-risk				+		+	+	+	+	+		+
low-risk		+	-	-	-	-	-	-	-	-		+
relative	-	+	+	+		-		-	-			+
Time-Weighted Degree												
high-risk		-		-		-		-		-	-	-
low-risk	-	-		+		+	+	+	+		-	-
relative	+	-	-	-	-	+		+	+		-	-
Quadrangles												
high-risk		-	-					+		-		
time-weighted	-	+	+	+	+	+	+	+	+	+	+	
low-risk	+			-	-	-	-		-	-	-	-
time-weighted	-			+			+			+	+	+
relative		+	+	-		-	+		+	+	-	
time-weighted	+		+	+							+	+
Quadrangle Frequency												
mean (high-risk)			-			-			-			
time-weighted	+			+	+	+	+		+	-		-
max (high-risk)			+	+				-	+		+	
time-weighted	-		+	-		-	-	+	-	+	+	+
mean (low-risk)			+		+	-		-	-	-	-	-
time-weighted		-	-		-	+	-	+	+	+	+	+
max (low-risk)	-	-	-					-	-	+		
time-weighted	+	+	+	+	+	+	+	+	+			+
Res. Similarity												
Sector	-		+			-	+		-	-		
Location (1)			-		-	-	-		-	-	-	-
Location (2)				+	+		+		-			
INDIRECT FEATURES												
Exposure Score	+			+	+		+					
Neighborhood Exposure												
mean		-	-		+	+	-	+				+
weighted mean		+	+	+			+		+	+	+	
maximum		+	+	-		-				-		

Table 5 Variable importance and sign of the GOTCHA! model for social security fraud detection. A positive sign indicates a positive contribution of that variable to fraud. A negative sign means that the variable negatively impacts fraud.

the feature values among all out-of-bag observations – i.e., the observations left out when training one tree. The difference between the number of correct votes with and without permutation is the variable importance. Figure 10 shows that network-based variables are important indicators in fraud detection models. The most predictive features comprise features derived from the exposure scores and quadrangles. The exposure score captures the extent to which a node is influenced by fraud. According to Figure 10, aggregated features derived from the neighborhood exposure scores are more meaningful than the company's own exposure score. Quadrangles measure whether a pair of resources has been transferred between multiple companies before. The relative number of high-risk quadrangles plays an important role in the detection of fraud. Quadrangle frequency measures how many times a transfer between the same pair of resources occurred. The more companies the two resources have in common, the more suspicious the transfer is. This is in line with the process of a spider construction, where resources are continuously moved from one fraudulent company to another.

Table 5 summarizes the signs of the coefficients for the network parameters, based on Random Logistic Forests. Note that, in general, features aggregating high-risk characteristics are positively related with fraud, which complies with expert's intuition. One exception is the high-risk time-weighted degree which is overall negatively related with fraud. Remark that low-risk quadrangle frequency (maximum and time-weighted) positively impacts the suspiciousness of a company. It appears that two resources that frequently move together even between many legitimate companies are anomalous. In other words, the frequent hopping behavior in itself might thus be an important indicator of suspiciousness. This might indicate that the *GOTCHA!* model is able to find new spider constructions, and does not completely rely on high-risk influences from the surrounding environment. Based on the large parameter value, we find that the weighted mean neighborhood exposure score is a crucial element in the prediction of fraud, which is in accordance with Figure 10. We conclude that network-based features remain relevant to estimate fraud over time, irrespective of the timestamp and the time window.

Does the impact of network-based variables depend on the intrinsic variables of a company or are they independent of other intrinsic features (e.g., are network effects more pronounced for companies that operate in a high-risk sector or legal category)? We do not find significant interaction effects between intrinsic and network-based features. Network-based features play an important role, irrespective of the intrinsic characteristics of the company.

The companies outputted by our models are passed on to experts for further inspection. As experts' resources are limited, they require models that generate a short list (high precision) with as many possible fraud cases in the near future (high recall). In practice, however, we often need to make a trade-off between precision and recall. Figure 11 depicts the precision and recall for the baselines and the *GOTCHA!* model over various timestamps and time windows. Error bars indicate the minimum and maximum results achieved over the folds. Although the *GOTCHA!* model does not achieve a higher precision than the network models, it performs on average better than the intrinsic-only model. A pairwise t-test confirms that these results are significant at ($\alpha = 0.1$), with the exception of the medium-term model for the intrinsic baseline in year t_0. Although subtle, notice the stepwise increase in precision over the different time windows for almost all models within each timestamp. Overall we can say that shorter-term models achieve a slightly lower precision than models estimated on a longer time window. This can be explained by the lack of confirmed fraudulent cases to learn from.

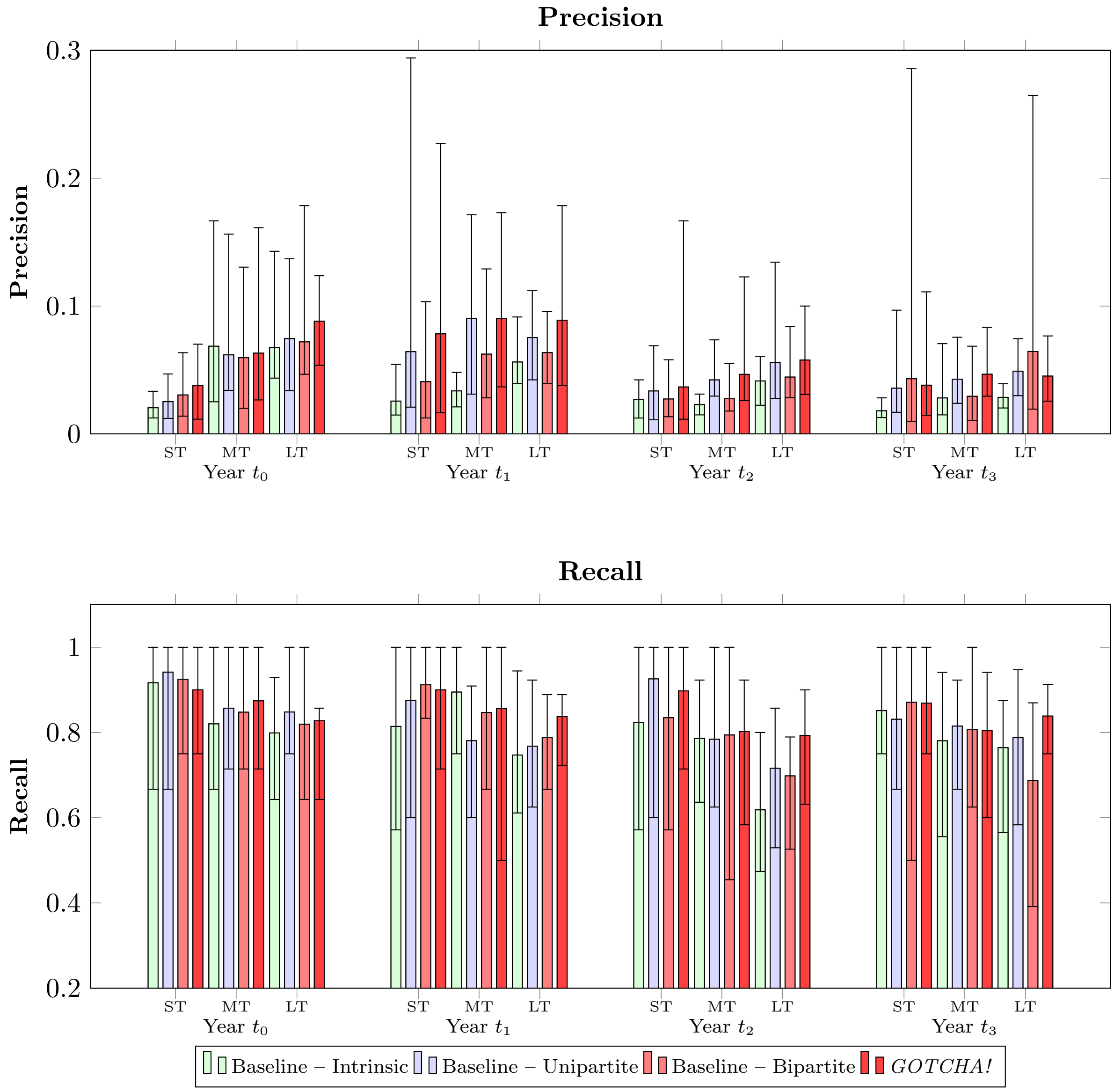


Figure 11 Precision and recall for the various models. The network-based models outperform the intrinsic baseline. While the Gotcha! models generate a high precision, the baselines and the Gotcha! model perform equally good in terms of recall. As such, the Gotcha! models can detect almost all fraud cases (high recall), like the intrinsic baseline, but are able to dramatically reduce the list that experts need to investigate (high precision).

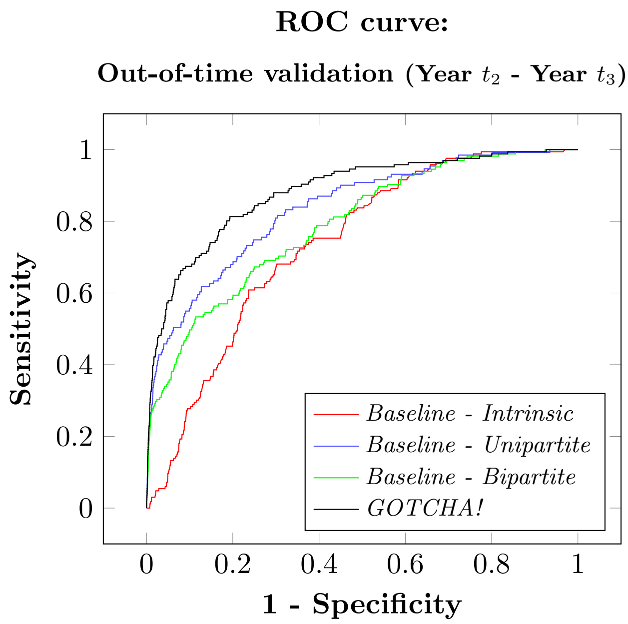


Figure 12 **ROC analysis of the proposed approach applied in practice. All models are estimated on Year t_2 and tested on Year t_3.**

In terms of recall, the *GOTCHA!* model and baselines follow a similar pattern: the ratio of detected companies decreases when the time window is extended. On short term, every model succeeds to identify all the fraudulent companies which is shown by the maximum error bars of one. This assesses the trade-off between recall and precision. Long-term models are more precise, which is penalized by a lower recall. Short-term models are able to identify all fraudulent companies at the expense of a lower precision.

5.1. Out-of-time Validation

Up until now, models were trained and validated on the same timestamp. Results prove the superiority of our proposed model compared to the baselines. However, in practice, models are trained on a previous timestamp and used in real-time. This section discusses our findings when implementing the models in this way. This is called *out-of-time validation*. The models are trained on year $t-1$ and tested on year t.

Figure 12 represents a ROC analysis of an out-of-time validation on medium-term period between year t_2 and year t_3 (other timestamps perform similarly). The figure shows that the baselines already perform well. However, including network-based variables has a positive effect on the predictive power of the network models. In particular, when a network model gives a company a high score, it has a higher probability of begin truly fraudulent. This is shown in the steep increase in the beginning of the curve that represents the performance of the network models. Remark that the unipartite model outperforms the bipartite model. The most probable reason is that every

		Total	ST Fraud	MT Fraud	LT Fraud	Fraud after analysis	Total Fraud	Bankrupt	Non-Active	Active	% detected
t_1	***Baseline - Intrinsic***	100	4	1	5	1	**11%**	11	8	70	**22%**
	Baseline - Unipartite	100	15	7	2	7	**31%**	20	19	30	**51%**
	Baseline - Bipartite	100	16	9	4	7	**36%**	17	10	37	**53%**
	GOTCHA!	100	20	6	6	10	**42%**	29	10	19	**71%**
t_2	***Baseline - Intrinsic***	100	4	1	1	1	**7%**	8	7	78	**15%**
	Baseline - Unipartite	100	7	5	4	1	**17%**	26	12	45	**43%**
	Baseline - Bipartite	100	14	7	10	3	**34%**	24	7	35	**58%**
	GOTCHA!	100	17	4	12	7	**40%**	30	6	24	**70%**
t_3	***Baseline - Intrinsic***	100	2	0	1	0	**3%**	14	1	82	**17%**
	Baseline - Unipartite	100	15	3	3	0	**21%**	19	3	57	**40%**
	Baseline - Bipartite	100	24	6	3	0	**33%**	12	5	50	**45%**
	GOTCHA!	100	16	12	8	0	**36%**	20	4	40	**56%**

Table 6 Future lifecycle of the top 100 most suspicious companies. The number of confirmed fraudulent companies is reported, together with the number of bankrupt and non-active companies. These results show that GOTCHA! outperforms the other baselines by detecting up to 33%, 23% and 6% more fraud than the intrinsic, unipartite and bipartite baseline respectively (column total fraud).

resource has an equal impact on a company in the bipartite model, independent of the recency of the relationship. This means that old relationships are equally treated as new relationships in the analysis which can deteriorate the performance. The unipartite network connects companies to companies by means of shared resources. A higher link weight indicates a more important connection. As such, unipartite networks are able to make a better distinction between more important links – in terms of the number of resources associated to that company – than bipartite networks. However, when we include time-weighted features, we achieve the best performance. This is shown by the *GOTCHA!* model.

The previous section illustrated that the medium- and long-term models perform better in terms of precision. It appears, however, that many companies detected by the short-term model will have solvency problems sometime in the future; this is shown in Table 6. The table represents the results of an out-of-time validation over the different timestamps, when the models are estimated on short-term fraud. We analyze the top 100 most suspicious companies, as experts can only investigate maximum 100 companies during each time period which thus reflects model usage in practice. Remark that the results in Figure 11 are based on the optimal sensitivity-specificity cutoff, whereas in Table 6 we report the top 100 most suspicious companies. Table 6 indicates how many companies in the list commit fraud on short (*ST Fraud*), medium (*MT Fraud*) and long (*LT Fraud*) term. The model even identifies companies that will commit fraud after the time window of analysis (*Fraud after analysis*). Note that this effect diminishes over time due to the recency

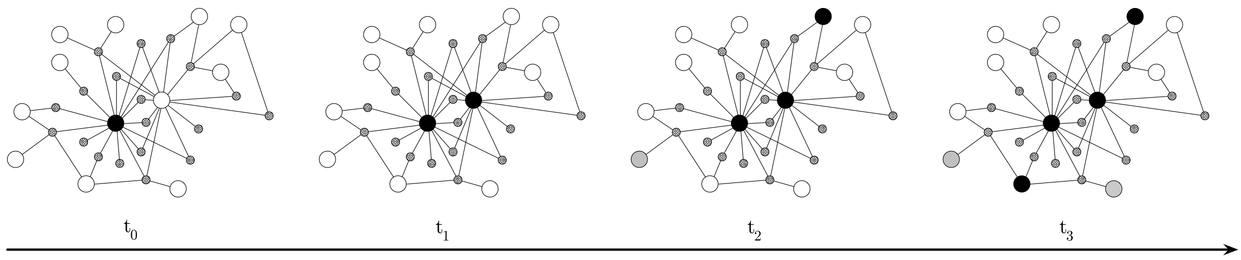


Figure 13 Evolution of a spider construction over time. The network represents the nodes and connections as observed at time t_0. Only one company is fraudulent, but passes many resources to other companies. Future data shows that three extra companies will commit fraud, and two companies will go bankrupt. If we had applied GOTCHA!, our results show that we could have avoided the development of this spider construction at time t_0.

of the data used. *GOTCHA!* improves the intrinsic baseline by detecting 31%, 33% and 33% more fraudulent and high-risk cases for the respective timestamps, resulting in a higher precision and recall. The unipartite baseline is improved by 11%, 23% and 15%, respectively. The bipartite baseline is outperformed by an increase of 6%, 6% and 3% respectively. Recall that the ROC curve of the bipartite model (see Figure 12) did not achieve a better performance than the unipartite model. However, when analyzing the results by a limited set of the top 100 suspicious companies, we find that the bipartite model is more precise than the unipartite model. These results are consistent over all timestamps.

What happens to the other companies in the list? Some are still active (*Active*). Others are normally suspended (*Non-active*), and redeemed all their outstanding debts. Surprisingly, we see that 29%, 30% and 20% of these companies go bankrupt in the future. Although there is a lack of hard evidence and the time passed, experts are convinced that those companies are missed fraudsters. Assuming the expert is right in his/her expectation, this would mean that the detection model is able to reach higher levels of precision up to 71%, improving the intrinsic baseline by detecting up to 55% additional fraudsters over time (year t_2). There are thus reasons to believe that *GOTCHA!* is suitable for *retrospective* fraud detection. To summarize, by using *GOTCHA!*, experts can identify fraudulent companies much faster and more accurately, and potentially are still able to recover some of the losses occurring with fraud.

5.2. Curtailing newly originated spider constructions

Rather than detecting far-evolved spider constructions, experts are enticed to identify newly originated spider constructions as well, i.e., new fraudulent setups with only few fraudulent companies in it. In Figure 13, we illustrate that *GOTCHA!* is also able to find such constructions. The figure shows a subgraph for timestamp *year* t_0. One company has committed fraud during this timestamp. Note that almost all its resources flow towards another company. Indeed, this and two other companies commit fraud in the future, as well as two companies that went bankrupt. Our results show that if we had applied *GOTCHA!* during timestamp t_0, we could have avoided the development of this spider construction as those companies would have appeared in the list generated on t_0. It can be questioned whether the two bankruptcies are purely coincidental or that they are part of the fraud construction.

6. Conclusions

In this paper, we improve the performance of traditional classification techniques for social security fraud detection by including domain-driven network information using *GOTCHA!*, a new fraud detection approach. We start by identifying the challenges that concur with fraud and design *GOTCHA!* such that it addresses each of these challenges to detect future fraud. In particular, we represent the network as a time-weighted bipartite graph, including two node types: companies and their resources. Starting from a limited set of confirmed fraudulent companies, we spread fraudulent influences of one node type through the network and infer an initial exposure score for both node types, i.e., the unlabeled companies and resources. Our propagation algorithm inherits concepts from the Personalized PageRank algorithm as proposed by (Page et al. 1998), and is extended by making the following domain-dependent adjustments: (1) propagation for bipartite graphs (i.e., scoring both companies and resources), (2) emphasizing fraud, (3) dynamical behavior: use of a time-dependent weight to represent relationships between companies and resources, ánd to weigh the impact of fraud, (4) degree-independent propagation. The time-dependent weight allows to both anticipate and forgive the riskiness of the resources. For each company, we aggregate the properties of the direct and indirect neighborhood, and combine them with intrinsic features.

The Social Security Institution benefits from our developed approach in multiple ways: (1) *Guided search for fraud.* Instead of randomly investigating companies, the *GOTCHA!* algorithm produces an accurate list of companies that are worthwhile to investigate by experts. Our experiments show that our *GOTCHA!* network model exploits essential information for predicting future fraud more efficiently. Our model is compared to three baselines. The first one is an intrinsic-only baseline and uses only intrinsic features. The second one is a unipartite baseline, linking the companies directly to each other and aggregating resource information in the link weight. The third one extends the network representation to a bipartite graph but does not include time in the link weights. Results show that *GOTCHA!* produces more accurate results than the baselines in terms of their AUC score. We find that network models achieve a higher precision, although the recall is approximately the same. Hence, network-driven models reduce the set of high-risk companies passed on to the experts for further screening. (2) *Faster fraud detection.* The predictability of short-term models is surprising. Short-term models are not only able to accurately predict which companies will commit fraud in the near future, but also identify companies that perpetrate fraud many years later. This results in a higher overall precision compared to medium- and long-term models, favoring the short-term models in the fraud detection process. This also indicates that, so far, many fraudulent companies already radiate fraudulent behavior, which used to take several months, or even years, before they were actually captured. (3) *Immediate feedback loop.* Findings of experts are immediately implemented in the models. The models update their detection process accordingly. Consequently, changes in the fraud environment are captured by the models. Our results show that models indeed use different sets of variables over time. Our future work will elaborate more on active learning, by updating the model using both correctly and incorrectly classified instances.

Although we applied our approach to social security fraud detection, we have promising results that our proposed framework can be employed for the detection of other fraud types where the network can be represented as a higher order graph (n-partite graph). In (Van Vlasselaer et al. 2015), we demonstrate the benefits of a similar approach on credit card fraud where merchants are explicitly connected to buyers through their transactions. Other examples might include money laundering linking people to cash transactions, etc. This work focuses on finding individual companies. Another topic for future research is community detection which may find groups of suspicious companies. Community detection allows experts to gain a thorough understanding in the creation and development of spider constructions.

Acknowledgments

This work was performed by Véronique Van Vlasselaer, Monique Snoeck and Bart Baesens through support of FWO grant G.0551.15, by Tina Eliassi-Rad through support from NSF CNS-1314603, DTRA HDTRA1-10-1-0120, DARPA under SMISC Program Agreement No. W911NF-12-C-0028, and under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, and by Leman Akoglu through support from National Science Foundation CAREER 1452425, IIS 1408287, DARPA Transparent Computing Program under Contract No. FA8650-15-C-7561, a Facebook Faculty Gift, and an R&D grant from Northrop Grumman Aerospace Systems 1452425.

References

Akoglu, Leman, Rishi Chandy, Christos Faloutsos. 2013. Opinion fraud detection in online reviews by network effects. *Proceedings of the Seventh International Conference on Weblogs and Social Media*. AAAI, 1–11.

Akoglu, Leman, Mary McGlohon, Christos Faloutsos. 2010. Oddball: Spotting anomalies in weighted graphs. *Advances in Knowledge Discovery and Data Mining*. Springer, 410–421.

Aral, Sinan, Lev Muchnik, Arun Sundararajan. 2009. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proceedings of the National Academy of Sciences* **106**(51) 21544–21549.

Baesens, Bart, Rudy Setiono, Christophe Mues, Jan Vanthienen. 2003a. Using neural network rule extraction and decision tables for credit-risk evaluation. *Management Science* **49**(3) 312–329.

Baesens, Bart, Tony Van Gestel, Stijn Viaene, Maria Stepanova, Johan Suykens, Jan Vanthienen. 2003b. Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society* **54**(6) 627–635.

Bapna, Ravi, Akhmed Umyarov. 2012. Do your online friends make you pay? A randomized field experiment in an online music social network. *NBER Working Paper* `http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2498080`.

Boccaletti, Stefano, Vito Latora, Yamir Moreno, Martin Chavez, D-U Hwang. 2006. Complex networks: Structure and dynamics. *Physics reports* **424**(4) 175–308.

Bolton, Richard J, David J Hand. 2002. Statistical fraud detection: A review. *Statistical Science* **17**(3) 235–249.

Breiman, Leo. 2001. Random forests. *Machine Learning* **45**(1) 5–32.

Carrizosa, Emilio, Belén Martín-Barragán, Dolores Romero Morales. 2014. A nested heuristic for parameter tuning in support vector machines. *Computers & Operations Research* **43** 328–334.

Chan, Philip K, Salvatore J Stolfo. 1998. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. *Knowledge Discovery and Data Mining*, vol. 1998. 164–168.

Chandola, Varun, Arindam Banerjee, Vipin Kumar. 2009. Anomaly detection: A survey. *ACM Computing Surveys* **41**(3) 15.

Chau, Duen Horng, Shashank Pandit, Christos Faloutsos. 2006. Detecting fraudulent personalities in networks of online auctioneers. *Proceedings of the Tenth European Conference on Principles and Practice of Knowledge Discovery in Databases*. Springer, 103–114.

Chawla, Nitesh V., Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer. 2011. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* **16** 321–357.

Chen, Hsinchun, Wingyan Chung, Jennifer Jie Xu, Gang Wang, Yi Qin, Michael Chau. 2004. Crime data mining: A general framework and some examples. *Computer* **37**(4) 50–56.

Chiu, Chaochang, Yungchang Ku, Ting Lie, Yuchi Chen. 2011. Internet auction fraud detection using social network analysis and classification tree approaches. *International Journal of Electronic Commerce* **15**(3) 123–147.

Cortes, Corinna, Daryl Pregibon, Chris Volinsky. 2001. Communities of interest. *Proceedings of the Fourth International Conferrence on Advances in Intelligent Data Analysis*. Springer, 105–114.

Dionne, Georges, Florence Giuliano, Pierre Picard. 2009. Optimal auditing with scoring: Theory and application to insurance fraud. *Management Science* **55**(1) 58–70.

Easley, David, Jon Kleinberg. 2010. *Networks, Crowds, and Markets*. Cambridge University Press.

Eliassi-Rad, Tina, Keith Henderson. 2011. Ranking information in networks. *Social Computing, Behavioral-Cultural Modeling and Prediction*. Springer, 268–275.

Estévez, Pablo A, Claudio M Held, Claudio A Perez. 2006. Subscription fraud prevention in telecommunications using fuzzy rules and neural networks. *Expert Systems with Applications* **31**(2) 337–344.

Fast, Andrew, Lisa Friedland, Marc Maier, Brian Taylor, David Jensen, Henry G Goldberg, John Komoroske. 2007. Relational data pre-processing techniques for improved securities fraud detection. *Proceedings of the Thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 941–949.

Fawcett, Tom, Foster Provost. 1997. Adaptive fraud detection. *Data Mining and Knowledge Discovery* **1**(3) 291–316.

Furlan, Štefan, Marko Bajec. 2008. Holistic approach to fraud management in health insurance. *Journal of Information and Organizational Sciences* **32**(2) 99–114.

Gallagher, Brian, Hanghang Tong, Tina Eliassi-Rad, Christos Faloutsos. 2008. Using ghost edges for classification in sparsely labeled networks. *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 256–264.

Galloway, John, Simeon J Simoff. 2006. Network data mining: Discovering patterns of interaction between attributes. *Advances in Knowledge Discovery and Data Mining*. Springer, 410–414.

Gao, Zengan, Mao Ye. 2007. A framework for data mining-based anti-money laundering research. *Journal of Money Laundering Control* **10**(2) 170–179.

Gleich, David F. 2014. Pagerank beyond the web. *arXiv preprint arXiv:1407.5107* .

Goldberg, Henry G, Ted E Senator. 1995. Restructuring databases for knowledge discovery by consolidation and link formation. *Knowledge Discovery and Data Mining*, vol. 95. 136–141.

Gyöngyi, Zoltán, Hector Garcia-Molina, Jan Pedersen. 2004. Combating web spam with trustrank. *Proceedings of the Thirtieth International Conference on Very Large Databases*. VLDB Endowment, 576–587.

Hastie, Trevor, Robert Tibshirani, Jerome Friedman. 2001. *The Elements of Statistical Learning*, vol. 1. Springer New York.

Hilas, Constantinos S, John N Sahalos. 2005. User profiling for fraud detection in telecommunication networks. *Proceedings of the Fifth International Conference on Technology and Automation*. 382–387.

Jensen, David. 1997. Prospective assessment of AI technologies for fraud detection: A case study. *Proceedings of the AAAI Workshop on AI Approaches to Fraud Detection and Risk Management*. 34–38.

Levin, Daniel Z, Rob Cross. 2004. The strength of weak ties you can trust: The mediating role of trust in effective knowledge transfer. *Management Science* **50**(11) 1477–1490.

Macskassy, Sofus A, Foster Provost. 2007. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research* **8** 935–983.

Maes, Sam, Karl Tuyls, Bram Vanschoenwinkel, Bernard Manderick. 2002. Credit card fraud detection using bayesian and neural networks. *Proceedings of the First International NAISO Congress on Neuro Fuzzy Technologies*. 16–19.

McGlohon, Mary, Stephen Bay, Markus G Anderle, David M Steier, Christos Faloutsos. 2009. Snare: A link analytic system for graph labeling and risk detection. *Proceedings of the Fifteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1265–1274.

Neville, Jennifer, Özgür Şimşek, David Jensen, John Komoroske, Kelly Palmer, Henry Goldberg. 2005. Using relational knowledge discovery to prevent securities fraud. *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. ACM, 449–458.

Opsahl, Tore, Pietro Panzarasa. 2009. Clustering in weighted networks. *Social Networks* **31**(2) 155–163.

Page, Lawrence. 2001. Method for node ranking in a linked database. US Patent 6,285,999.

Page, Lawrence, Sergey Brin, Rajeev Motwani, Terry Winograd. 1998. The pagerank citation ranking: Bringing order to the web. Tech. rep., Stanford Digital Library Technologies Project. `http://ilpubs.stanford.edu:8090/422/`.

Pandit, Shashank, Duen Horng Chau, Samuel Wang, Christos Faloutsos. 2007. NetProbe: A fast and scalable system for fraud detection in online auction networks. *Proceedings of the Sixteenth International Conference on World Wide Web*. ACM, 201–210.

Park, Juyong, Albert-László Barabási. 2007. Distribution of node characteristics in complex networks. *Proceedings of the National Academy of Sciences* **104**(46) 17916–17920.

Phua, Clifton, Damminda Alahakoon, Vincent Lee. 2004. Minority report in fraud detection: classification of skewed data. *ACM SIGKDD Explorations Newsletter* **6**(1) 50–59.

Prakash, B Aditya, Hanghang Tong, Nicholas Valler, Michalis Faloutsos, Christos Faloutsos. 2010. Virus propagation on time-varying networks: Theory and immunization algorithms. *Machine Learning and Knowledge Discovery in Databases*. Springer, 99–114.

Provost, Foster. 2000. Machine learning from imbalanced data sets. *Proceedings of the AAAI Workshop on Imbalanced Data Sets*. 1–3.

Provost, Foster, Brian Dalessandro, Rod Hook, Xiaohan Zhang, Alan Murray. 2009. Audience selection for on-line brand advertising: Privacy-friendly social network targeting. *Proceedings of the Fifteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 707–716.

Quah, Jon TS, M Sriganesh. 2008. Real-time credit card fraud detection using computational intelligence. *Expert Systems with Applications* **35**(4) 1721–1732.

Rossi, Ryan, Jennifer Neville. 2012. Time-evolving relational classification and ensemble methods. *Advances in Knowledge Discovery and Data Mining*. Springer, 1–13.

Sánchez, Daniel, MA Vila, L Cerda, José-María Serrano. 2009. Association rules applied to credit card fraud detection. *Expert Systems with Applications* **36**(2) 3630–3640.

Šubelj, Lovro, Štefan Furlan, Marko Bajec. 2011. An expert system for detecting automobile insurance fraud using social network analysis. *Expert Systems with Applications* **38**(1) 1039–1052.

Tong, Hanghang, Christos Faloutsos, Jia-Yu Pan. 2006. Fast random walk with restart and its applications. *Proceedings of the Sixth International Conference on Data Mining*. IEEE, 613–622.

Tong, Hanghang, Spiros Papadimitriou, S Yu Philip, Christos Faloutsos. 2008. Proximity tracking on time-evolving bipartite graphs. *Proceedings of the SIAM International Conference on Data Mining*. 704–715.

Van Vlasselaer, Véronique, Cristián Bravo, Olivier Caelen, Tina Eliassi-Rad, Leman Akoglu, Monique Snoeck, Bart Baesens. 2015. Apate: A novel approach for automated credit card transaction fraud detection using network-based extensions. *Decision Support Systems* **75** 38–48.

Verbeke, Wouter. 2012. Profit-driven data mining in massive customer networks: New insights and algorithms. Master's thesis, KU Leuven, Belgium. `https://lirias.kuleuven.be/handle/123456789/327225`.

Wang, Jyun-Cheng, Chui-Chen Chiu. 2008. Recommending trusted online auction sellers using social network analysis. *Expert Systems with Applications* **34**(3) 1666–1679.

Yanchun, Zhu, Zhang Wei, Yu Changhai. 2011. Detection of feedback reputation fraud in taobao using social network theory. *Proceedings of the International Joint Conference on Service Sciences*. IEEE, 188–192.