Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 195 (2009) 595–612
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

O.R. Applications

On the communal analysis suspicion scoring for identity crime in streaming credit applications

Clifton Phua [a,*], Ross Gayler [c], Vincent Lee [b], Kate Smith-Miles [d]

[a] Institute of Infocomm Research, 21, Heng Mui Keng Terrace, 119613, Singapore
[b] Clayton School of Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, Australia
[c] Veda Advantage, Level 12, 628 Bourke Street, Melbourne, VIC 3000, Australia
[d] School of Engineering and Information Technology, Deakin University, 221 Burwood Highway, Burwood, Victoria, Australia

Received 3 March 2006; accepted 13 February 2008
Available online 21 February 2008

Abstract

This paper describes a rapid technique: communal analysis suspicion scoring (CASS), for generating numeric suspicion scores on streaming credit applications based on implicit links to each other, over both time and space. CASS includes pair-wise communal scoring of identifier attributes for applications, definition of categories of suspiciousness for application-pairs, the incorporation of temporal and spatial weights, and smoothed *k*-wise scoring of multiple linked application-pairs. Results on mining several hundred thousand real credit applications demonstrate that CASS reduces false alarm rates while maintaining reasonable hit rates. CASS is scalable for this large data sample, and can rapidly detect early symptoms of identity crime. In addition, new insights have been observed from the relationships between applications.
© 2008 Elsevier B.V. All rights reserved.

Keywords: Risk analysis; Credit application fraud detection; Communal scoring; Multi-attribute directed graph; Dynamic application data streams; Anomaly detection

1. Introduction

Annually, credit bureaus collect millions of enquiries from financial institutions (subscribers) relating to credit applications. In Australia, credit card and personal loan applications have increased significantly, and currently, close to half a million credit bureau enquiries are made per month (Baycorp, 2005). Each credit application contains a large number of identity attributes such as personal names, address(es), telephone number(s), driver licence number (or social security number), date-of-birth, and other personal identifiers which are potentially available to the credit bureau (if local privacy laws permit it). Therefore, from a commercial perspective, it is uneconomical to physically validate and approve each attribute in every credit application.

Application fraud, a manifestation of identity crime, is present when application form(s) contain plausible and synthetic (identity fraud), and/or real, but stolen identity information (identity theft). In developed countries, the monetary cost of application fraud and identity crime is often estimated to be in the billions of dollars; and this is strongly correlated with the large volume of widely available personal information. By performing better once-off assessments in the first stage of the credit life cycle, credit scoring processes could be improved and some transactional fraud can be prevented.

* Corresponding author. Tel.: +65 93296236.
E-mail addresses: cwphua@i2r.a-star.edu.sg (C. Phua), ross.gayler@vedaadvantage.com (R. Gayler), vincent.lee@infotech.monash.edu (V. Lee), katesm@deakin.edu (K. Smith-Miles).

0377-2217/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2008.02.015

Typical commercial techniques for the identification of such fraud involve the use of attribute verification rules using reference tables, and pair-wise matching rules between credit application and credit history data. However, the success rate of rule-based approaches can be weak when faced with increasingly common fraudster-tampered applications (Oscherwitz, 2005) which have valid attributes and no credit history. Other techniques being used include known fraud matching using blacklists (list of applications previously submitted by fraudsters) and supervised modelling/classification using labelled data. Often, these labelled data approaches alone are operationally inefficient and ineffective (Phua et al., 2005). Our work focuses on credit application data only, with no checks carried out against credit history.

As it is simulated to run in real-time, communal analysis suspicion scoring (CASS) does not take class labels into account when scoring applications. It only uses class labels to determine the effectiveness of its approach. Its purpose is to generate numeric suspicion scores on streaming credit applications based on implicit links to each other, over both time and space. CASS includes pair-wise communal scoring of identifier attributes for applications, definition of categories of suspiciousness for application-pairs, the incorporation of temporal and spatial weights, and smoothed k-wise scoring of multiple linked application-pairs.

In Section 2, this paper explores and compares semi-related literature and contrasts the research findings with this paper's contributions. Section 3 describes a rapid technique known as CASS for scoring and generating links on incoming current/new application streams on demand and focuses at the level of each pair of linked applications (application-pair). Section 4, firstly, lays out the retrospective and stream processing experimental results on ethically-approved data and subsequently visualises and discusses the unique patterns of identity crime in credit applications. Section 5 concludes the paper.

2. Related work

There is no academic research, to the best of our knowledge (Phua et al., 2005) into the scoring of dynamic credit applications which accounts for its sparse-identifiers, communal, temporal, and spatial aspects. However, there are other related and established application fields. Section 2.1 summarises multi-attribute pair-wise matching (for example, record linkage/de-duplication detection and an interesting effort to discover cheating amongst teachers in exams by altering their students' answers to multiple-choice questions). Section 2.2 reviews single attribute scoring/directed graphs with explicit links (for example, telecommunications fraud detection, terrorist detection, social network analysis, and webpage ranking). Section 2.3 clarifies and describes the contribution of our work.

2.1. Multi-attribute pair-wise matching

Bilenko et al. (2003) applies a series of character-based and token-based string similarity metrics to identifying approximately duplicate database records from multiple sources. They propose treating each record as a set of fields and then measuring the average similarity across these fields. By doing so, the similarity in the record is represented with a feature vector (represented by our attribute vector y in Section 3.2). They also propose adapting the string similarity metrics' edit operation costs to each attribute (not implemented as it is computationally too expensive for large data sets). The authors applied their techniques to comparatively small data sets and have good results on some data sets.

In a later study, Levitt and Dubner (2005, pp. 25–37) applies an algorithm to find interesting patterns in corrected students' answers for multiple-choice questions which can be indicative of teachers cheating by changing their students' answers in the state-wide exams. Although Levitt and Dubner's algorithm is not elaborated in their book, CASS works in a similar way. It seems that each student's answer string is pair-wise matched against the other students within each class and a high score is given if the answers are highly identical. In comparison to other classes for the same exam or same class in different exams, a high average score for each class indicates suspicious activity on the part of teachers and is worth investigating. The cheating algorithm has been used on a hundred million individual student answers from the Chicago Public School system which have been grouped by class and exam.

2.2. Single attribute link analysis

Cortes et al. (2003) illustrate computational methods which construct large dynamic graphs of entity-pair interactions for telecommunications fraud detection. To prevent the management and storage of a large number of graphs from different time steps, this method exponentially smooths the previous and current graphs (in Section 3.5, our approach is to smooth linked nodes/applications) and uses a continually updated top-k link set for each node/telephone account to monitor suspicious calling patterns (after incoming applications are scored, they are not updated unless they have been flagged as fraudulent). Cortes et al. dealt with hundreds of millions of nodes and billions of links per day.

Macskassy and Provost (2005) also use suspicion scoring to detect malicious individuals and their associates. Their relational classification and collective inference estimates suspicion as the weighted sum of connected individuals (very similar

to our smoothed scoring function in Section 3.5 although our total suspicion score for an application is not scaled to between 0 and 1, and our approach requires no algorithm iteration as it takes into account the temporal sequence of applications). Macskassy and Provost's algorithm was tested on data sets created by a terrorist-world simulator and concludes that good rankings are generated even with a small number of known labels and moderate noise.

Graphs and distance measures for predicting future links between friends was examined by Kubica et al. (2003). This method uses temporal weights to exponentially smooth the effects of older links (different from our temporal weights which reflect the importance for the difference in days for each application-pair, and also different from our exponential smoothing of current and previous scores), and typical weights to reflect the quality/importance of link types (similar to our $w_{\text{communal}_{ij}}$ in Section 3.3). Their approach was evaluated on five link data sets together with five other competing algorithms, and their proposed algorithm was either the best or close to the best performer.

Both Page et al. (1998) and Kleinberg (1999) utilise the web link structure to determine the importance of webpages. *PageRank* (Page et al., 1998) use a hyperlink to a webpage as a popularity vote. This is defined recursively (however, this is not possible if applications need to be processed in real-time), and depends on quantity and *PageRank* webpages' quality of incoming links. In relation to this aspect, our approaches are very similar except that we measure suspiciousness of the current/most recent application based on outgoing links. *PageRank* is an essential part of Google's search engine. Hypertext Induced Topic Selection (*HITS*) (Kleinberg, 1999) recursively seeks to find static webpages which are authorities (provides good information and so many webpages link to it) and hubs (links to many authorities) to locate similar topic groups (in Section 3.5, our approach also penalises current application scores heavily if the incoming links threshold T_{E_1} is exceeded for any previous applications linked to the current one). *HITS* is used by the AltaVista search engine.

2.3. Contributions

There are four novel technical contributions vis-à-vis the related work which used a few hundred thousand credit applications. These contributions are listed below in a descending order of innovativeness and are addressed, in depth, within various sub-sections of this paper:

- Reduction of false alarm rates and achievement of consistent hit rates: a whitelist (list of most common relationships between real people) accounts for the possible communal relationships between real identities (see Sections 3.3.2 and 3.3.4).
- Improvement of scalability: least suspicious applications are removed at regular fixed intervals (see Section 3.2) and key attributes are used to filter the number of pair-wise comparisons (see Section 4.3.1).
- Attainment of rapid detection time: pair-wise matching is designed to score the incoming current application stream (see Section 3.2).
- Measurement of extreme rarity of the minority class, a variety of appropriate performance measures are used (see Section 4.3.2).

3. Communal analysis suspicion scoring (CASS) process

In this section, CASS compresses multiple identifier attributes to a single attribute vector representation of each link/non-link (Section 3.2). The approach distinguishes between three different categories of links: black, white, and anomalous, which will result in different weights and scores for every application-pair (Section 3.3). It accounts for the temporal and spatial effects by applying weights to each linked application-pair's communal score computation (Section 3.4). Also, this technique scores multiple linked applications, gradually diminishes the impact of prior linked applications, and presents the different decision thresholds (Section 3.5).

3.1. Mathematical notation

Graph theory is the established academic field which studies graph properties. Connections between credit applications can be denoted in a graph-theoretic notation (Wasserman and Faust, 1994). The simple directed graph (digraph), G_{d} can be described mathematically by two sets as $G_{\text{d}} = \langle V, E \rangle$. Let V represent a set of g vertices or nodes (applications for credit), where $V = \{v_1, v_2, \ldots, v_g\}$. Let Erepresent a set of h directed links or edges (relationships between applications based on shared identity information), where $E = \{e_1, e_2, \ldots, e_h\}$ with $g(g-1)/2$ maximum directed links. Let v_i, v_j each represent a vertex labelled with a set of N attributes, where $v_i = \{a_{i1}, a_{i2}, \ldots, a_{iN}\}$. In this paper, the analysis is on the directional link between every ordered application-pair, where $e_{ij} = \langle v_i, v_j \rangle$ with $v_i \rightarrow v_j \ \forall \ i, j$. The links are directed to maintain temporal sequence of applications, where v_i is the current unscored application and v_j is any previous scored application of interest which has arrived after v_i.

3.2. Pair-wise matching

The purpose is to derive an accurate suspicion score for all incoming current/new applications as rapidly as possible. Given that the design of pair-wise matching is focused on every current application v_i which arrives in the application fraud system. v_i can be pair-wise matched against all previous/existing scored applications v_j within a window W. In other words, a suspicion score is returned at real-time. Here, W denotes an applications window (such as previous twenty thousand applications). Alternatively W can also be a time window (within the last twenty days). This is to set a time (computational steps) constraint on CASS by limiting the running time to $O(gw)$ instead of $O(g^2)$.

In Table 1, all applications are primarily sorted in descending order by *datetime_received* to capture the applications' arrival sequence. In the training/initialisation phase, a fixed/tuned set of parameters is used to set up the whitelist which is represented by $\Re$ (see Section 3.3.2) and to ensure that the scoring/testing phase will be effective (applications in scoring phase are dependent on previous scores and decision thresholds). *Ceteris paribus*, scoring results from the immediate previous time period such as month-year represented by $m1$ can be used to set up the whitelist.

After the training phase, unscored applications continuously extracted from the next time period (represented by $m2$) are merged on top of initialised applications (with suspicion scores) extracted from both $m1$ and $m2$. The initialised applications are also secondarily sorted in descending order by suspicion score threshold which is represented by s, where the least suspicious applications can be removed in the scoring phase to maximise efficiency and increase scalability (application arrival rate is especially high).

In addition, within the scoring phase, a "divide-and-conquer" approach is taken for experimental purposes. g has p batches of b with u applications each to increase effectiveness (for comparison against more recent and risky applications) and efficiency (random access memory is limited).

For example in Table 1 (credit application "cliff-10000" to be used as illustration in subsequent sections), there are $g = 30{,}000$ unscored applications with $p = 3$ batches of $u = 10{,}000$ applications each to be processed and $s = 0.001$. For $b = 1$, "legal-1" is compared to the previously trained applications below it which have non-zero suspicion scores. When "cliff-10000" arrives into the system, it is compared to "legal-9999" until "legal-1" and perhaps applications below "legal-1" if $W > 10000$. For the next b, unscored and non-zero suspicion data are continuously extracted and scored until $b = p$. This is to set a space (memory resource) constraint on CASS.

The attribute vector y, between v_i and v_j, which represents the relationship for an application-pair is defined as:

$$y[v_i, v_j] = \{y_1[a_{i1}, a_{j1}], y_2[a_{i2}, a_{j2}], \ldots, y_N[a_{iN}, a_{jN}]\} \quad \forall i, j \tag{1}$$

where N is the number of attributes.

y_k is case sensitive and can also be determined by $y_k[a_{ik}, a_{jl}]$, where k is cross-matched with l (another similar attribute). For example, "cliff-10000's" home phone number can be matched against "legal-9999's" work phone number. All of "cliff-10000's" identity attributes are correspondingly matched against "legal-9999's". The Boolean suspicion score of each pair-wise attribute y_k is determined by:

Table 1
Pair-wise matching design from the table point-of-view

	rec_id	datetime_received	...	suspicion_score	
Before Training	legal-30000	31/5/2004	...		
	:	:	...		**b=p=3**
	legal-20001	1/5/2004	...		
	legal-20000	30/4/2004	...		
	:	:	...		**b=2**
	legal-10001	1/4/2004	...		
	cliff-10000	31/3/2004	...	0.2	
	:	:	...	:	**b=1**
	~~legal-1~~	~~1/3/2004~~	~~...~~	~~0~~	
After Training	:	:	:	:	
	legal-iv	14/02/2004	...	0.055	
	~~legal-iii~~	~~6/02/2004~~	~~...~~	~~0~~	
	fraud-i	26/01/2004	...	1.55	
	legal-ii	26/01/2004	...	0.9	
	~~legal-i~~	~~25/01/2004~~	~~...~~	~~0~~	

$$S(y_k) = \begin{cases} 1 & \text{if similarity}(a_{ik}, a_{jk}) \geqslant T_{\text{similarity}} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where the function similarity($\cdot$) can be any string similarity metric (Chapman, 2005) (such as *Levenshtein, Jaro-Winkler*, or *Monge-Elkan*) which measures the similarity of v_i's attribute value to v_j's. $T_{\text{similarity}}$ is the threshold for fuzzy matching where a threshold of 1 requires an exact match.

The maximum possible score of an application-pair in this case is $\sum_{k=1}^{N} S(y_k) = N$.

To account for different types of identifier attributes (IDAnalytics, 2004), those with higher fraud risk and/or missing values, each pair-wise attribute weight $w_{\text{attribute}_k}$ is heuristically assigned according to permanent attributes (for example, driver licence number), followed by stable attributes (for example, personal name), transient attributes (for example, referee phone number), and attribute ranking algorithms (Witten and Frank, 2005) so that the maximum possible score of an application-pair is now $\sum_{k=1}^{N} [S(y_k) * w_{\text{attribute}_k}] = 1$.

3.3. Approximate pair-wise score

This section defines the categories of suspiciousness/risk for the dynamic and complex nature of application-pairs. In the following order, a linked application-pair is first associated with either a blacklist or whitelist or as anomalous, and then the communal suspicion score for an application-pair is computed, accordingly.

3.3.1. Blacklist

A blacklist is a global list of un-desirable items to avoid. In our case, it is made up of actual identity examples reported as fraudulent by various subscribers. And the credit applications similar to the known fraudulent identity examples should not be approved for credit card issuance.

$$w_{\text{communal}_{ij}} = \begin{cases} 1 \text{ and } v_i \xrightarrow{\text{fraud}} v_j & \text{if } v_j \text{ is a known fraud} \\ & \text{and } \sum_{k=1}^{N} S(y_k) \geqslant T_{\text{attribute}} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $w_{\text{communal}_{ij}}$ is the communal link weight derived from pair-wise matching of identifier/string attributes (numerical attributes are treated as identifier/string attributes) and $0 \leqslant w_{\text{communal}_{ij}} \leqslant 1$. In this situation, v_j is only considered a known fraud if its fraud-status entry date and time is earlier than v_i's. $T_{\text{attribute}}$ is the threshold for the minimum number of matched attributes required to link v_i to v_j for a blacklist score and $N \geqslant T_{\text{attribute}}$. For example, if $T_{\text{attribute}} = 3$ and "cliff-10000" has 3 or more exact/similar identity attributes to "fraud-i", "cliff-10000" is assigned the largest possible pair-wise score. This is assuming that "cliff-10000's" arrival date and time happens after "fraud-i's" has been reflected in the database as fraudulent.

3.3.2. Whitelist

A whitelist $\Re$ is a global list of accepted items within a set (opposite of a blacklist) which allows the limited explanation and measurement of human relationships. Also, a non-whitelist item can be regarded as one which is so different from the whitelist items that it was produced artificially.

Instead of an absolute whitelist, which can be made up of the actual identity examples, our link whitelist consists of a user-defined, finite N number of ranked relationship-types between identity examples (obtained from the previous time period's credit applications). Its main purpose is to reduce the suspicion scores and incidence of false positives, while maintaining true positives.

$$w_{\text{communal}_{ij}} = \begin{cases} \left[w_{R_x} * \sum_{k=1}^{N} S(y_k * w_{\text{attribute}_k}) \right] \text{ and } v_i \xrightarrow{\text{normal}} v_j & \text{if } y \in \Re \\ & \text{and } \sum_{k=1}^{N} S(y_k) \geqslant T_{\text{attribute}} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where $\Re = [R_1, R_2, \ldots, R_N]$ is a set of one or more selected relationships/items defined as normal where each normal relationship R_x is ranked in descending order by (or filtered according to) volume derived from training phase. $w_{\text{normal}} \equiv [w_{R_1}, w_{R_2}, \ldots, w_{R_N}]$ where w_{R_x} is the corresponding weight of $R_x$and $0.5 \leqslant w_{R_x} \leqslant 1$. w_{normal} is sorted in ascending order. Assuming all parameters are constant and no significant changes in external environment, the higher number of R_x in a previous time period implies a higher possibility that it is less suspicious in the next. The other alternative is to manually

craft the whitelist. However, this is an *ad hoc*, tedious, and difficult procedure best avoided because each subscriber contributes a different number and type of attributes, and the whitelist items' rank changes frequently.

Each $y = y_1 \& y_2 \& \cdots \& y_N$ and $y_k = [\text{“}O\text{”}, \text{“}Y\text{”}, \text{“}N\text{”}, \text{“?”}]$ where "O" means attribute is deliberately omitted, "Y" represents a pair-wise attribute match, "N" a non-match, and "?" indicates one/both missing attribute value(s). Therefore, the maximum$N$ in $\Re$ is number of possible combinations in y_k to the power of number of chosen identity attributes. For example, given 12 identity attributes, there are $(4 \wedge 12) - 1 = 16$ million possible relationships. However, the actual ranked whitelist will only contain a very small fraction of these possible pair-wise relationships, retaining only those with high volume after the completion of the training phase.

For example, if "cliff-10000's" identity attributes are very similar to "legal-1" and its relationship-type is very common, and therefore reflected in the whitelist, its pair-wise score will be reduced according to the rank of the relationship-type in the whitelist.

3.3.3. Anomalous application-pairs

This paper's definition of anomalies refers to linked applications which are not in the blacklist and whitelists (as opposed to the widely accepted definition of anomalies as deviants from the whitelist).

$$w_{\text{communal}_{ij}} = \begin{cases} \sum_{k=1}^{N} S(y_k * w_{\text{attribute}_k}) \text{ and } v_i \overset{\text{anomalous}}{\rightarrow} v_j & \text{if } y \notin \Re \\ & \text{and } \sum_{k=1}^{N} S(y_k) \geqslant T_{\text{attribute}} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

For example, if "cliff-10000's" identity attributes are very similar to "legal-1" but its relationship-type is quite rare and therefore not in the whitelist, its pair-wise score will remain the same.

3.3.4. Summary of application-pair categories

Table 2 describes the score range, advantage, disadvantage, and secondary use of each application-pair category. The main advantage of a blacklist is the feedback of fraudulent applications to the system to stop subsequent similar applications from being approved. However, there is usually a significant time delay to flag particular applications as fraudulent during which similar applications go unnoticed (Phua et al., 2005).

The whitelist accounts for linked applications submitted by real identities. It defines the rational applicant behaviour: for example, application-pair submitted by the same identity with address changes, and the normal social relationships: for example, application paired with another family member's, housemate's, colleague's, neighbour's, or friend's application (IDAnalytics, 2004). The whitelist also filters out redundant relationships which are caused by matching on denser attributes with a low $T_{\text{attribute}}$. In short, the whitelist differentiates similar/linked applications as normal or anomalous. However, the fraudster can also create masqueraded "normal" applications to delay the detection time. Assuming minimal fraudster activity and high social interaction, it seems reasonable that $\Re$ can be used to exploit pre-existing social networks for the marketing of credit products by targeting customers with the strongest influence.

Linked pairs of anomalous applications will have lower scores than those in the blacklist and generally higher scores than those in the whitelist. This reveals abnormal relationships between applications which could be indicative of fraud, but also of data entry errors either by customers or employees.

Unlinked application-pairs are the result of too few (below $T_{\text{attribute}}$) or no attribute matches at all. There is a strong possibility that there could be many fraudulent applications which seem to be standalone that CASS would not be able to detect.

CASS can also work without a blacklist (with no known frauds), so it is an anomaly detection system.

Table 2
Properties of blacklist and whitelists, anomalous and unlinked applications

Application-pairs	Blacklist	Whitelist	Anomalous links	Unlinked
Score range	Highest	Lowest	Medium to high	None
Main advantage	Reasonably accurate	Filters normal and redundant relations	Finds irregular relations	Contains weak or no relations
Main disadvantage	Time delay	Prone to manipulation	Reliant on attribute weights	Unlinked (once-off) fraud
Secondary use other than real-time fraud detection	Update attribute weights	Viral marketing	Error detection	–

3.4. Communal, temporal, and spatial weights

The overall current link weight w_{ij} between v_i and v_j (to incorporate time and space into the scoring function) is defined as:

$$w_{ij} = \begin{cases} w_{\text{communal}_{ij}} * w_{\text{temporal}_{ij}} * w_{\text{spatial}_{ij}} & \text{if } w_{\text{communal}_{ij}} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $w_{\text{temporal}_{ij}} \forall i, j$ is the link weight derived from pair-wise time difference, in days, of *datetime_received* and $0.5 \leqslant w_{\text{temporal}_{ij}} \leqslant 1$; $w_{\text{spatial}_{ij}} \forall i, j$ is the link weight derived from pair-wise geographical distance, in kilometres, of *postcode* and $0.5 \leqslant w_{\text{spatial}_{ij}} \leqslant 1$. Therefore, w_{ij} roughly captures the strength of the communal links over time and space and $0 \leqslant w_{ij} \leqslant 1$.

A small time difference means that v_i is new, relative to v_j, but more importantly it implies that there is not enough time to get feedback or reveal the true state of v_j. A large geographical distance means that v_i is far off relative to v_j, and it suggests that the sphere of influence of v_j has changed. Given that identity attributes are being matched, the hypothesis is that a linked application-pair is most suspicious if there is no time difference and there is the maximum geographic distance between v_i and v_j. Therefore, $w_{\text{temporal}_{ij}} = w_{\text{spatial}_{ij}} = 1$. On the other hand, a linked application-pair is the least suspicious if it has the largest time difference specified and there is no geographic distance between v_i and v_j. Therefore, $w_{\text{temporal}_{ij}} = w_{\text{spatial}_{ij}} = 0.5$. Within the credit application context, *datetime_received*, being system-generated, and *postcode*, being the essential geographical area of contact (for example, card destination or card collection), are less prone to significant external manipulation than other identifier attributes.

For example, let us assume that "cliff-10000's" identity attributes are very similar to "legal-1". If these two applications are submitted within a same/short time period and across a large physical space, "cliff-10000's" pair-wise score will remain the same or be slightly reduced. If the opposite is true (a long time period and same/small physical space), "cliff-10000's" pair-wise score will be significantly reduced.

3.5. Smoothed k-wise scoring function

The average suspicion score of each previous application (to incorporate previous scores into the scoring function) is defined as:

$$S(v_j)/E_{\text{O}}(v_j) = \begin{cases} 1 & \text{if } w_{ij} > 0 \text{ and } \left\{ \left[v_i \overset{\text{fraud}}{\rightarrow} v_j \right]. \right. \\ & \text{or } \left(v_i \overset{\text{normal}}{\rightarrow} v_j \text{ or } v_i \overset{\text{anomalous}}{\rightarrow} v_j \right). \\ & \left. \text{and } E_{\text{I}}(v_j) \geqslant T_{E_{\text{I}}} \right] \right\} \\ S(v_j)E_{\text{O}}(v_j) & \text{if } w_{ij} > 0 \\ & \text{and } S(v_j) > 0 \text{ and } E_{\text{O}}(v_j) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$S(v_j)$ is the total suspicion score of a previous application which $v_i$links to. $E_{\text{O}}(v_j)$ is the number of outgoing links from v_j. Therefore, $0 \leqslant S(v_j)E_{\text{O}}(v_j) \leqslant 1$. $E_{\text{I}}(v_j)$ is the number of incoming links into v_j and $T_{E_{\text{I}}}$is the threshold for the acceptable number of incoming links.

For example, let us again assume that "cliff-10000's" identity attributes are very similar to "legal-1". Since "legal-1" has already been scored, it is important to factor the "legal-1's" average suspicion score into "cliff-10000's" pair-wise score.

Our scoring approach for v_i with k-pairs (multiple-pairs) of linked v_js is as follows:

$$S(v_i) = \sum_{v_j \in M(v_i)} [w_{ij} + S(v_j)/E_{\text{O}}(v_j)] \tag{8}$$

where $S(v_i)$ is the total suspicion score of the current application and $S(v_i) \geqslant 0$. $M(v_i)$ is the set of k applications which v_i links to. Therefore, a high suspicion score $S(v_i)$ indicates three possibilities: strong connection between v_i and v_j (high link weight – w_{ij}) which is dependent on number of non-missing attribute values; and/or the high quality of v_j(high average suspicion scores – $S(v_j)/E_{\text{O}}(v_j)$); and/or the large quantity of v_j (many connected applications – k).

For example, since "cliff-10000" can have multiple pair-wise scores (e.g. similar to "legal-1" and "fraud-i"), its overall suspicion score is to add them all up.

Exponential smoothing is needed to gradually discount the effects of previous suspicion scores, that is

$$S(v_i) = \sum_{v_j \in M(v_i)} [(1 - \alpha) * w_{ij} + \alpha * S(v_j)/E_{\text{O}}(v_j)] \tag{9}$$

where α is the smoothing factor and $0 \leqslant \alpha \leqslant 1$.

Another possible un-implemented scoring approach similar to but much more conservative (produce fewer and lower suspicion scores) than (9) is as follows:

$$S(v_i) = \sum_{v_j \in M(v_i)} \{w_{ij}^{(1-\alpha)} * [S(v_j)/E_O(v_j)]^{\alpha}\} \tag{10}$$

The decision thresholds requiring different precautionary actions can be defined in the following manner:

- If $0 < S(v_i) \leqslant T_{\text{lower}}$, then current application v_i is unusual (action: filter out for comparison with subsequent applications).
- If $T_{\text{lower}} < S(v_i) < T_{\text{upper}}$, then v_i is suspicious (action: make a phone call).
- If $S(v_i) \geqslant T_{\text{upper}}$, then v_i should be investigated (actions: check against other internal and external databases. For example, credit history, telephone directory, and/or electoral roll. And/or hand over entire case to a human investigator).
- If the investigated v_i is fraudulent, feedback should be provided to the fraud detection system.

4. Experiments

All experiments were performed on a single Pentium IV 3.0 GHz, 2 Gb RAM workstation, running on Windows XP platform. CASS itself is written in Visual Basic and works together with C#.NET libraries, and the credit application data is stored in Microsoft Access. In this section, the ethically-approved data sample, privacy and confidentiality issues, and the attributes are explained (Section 4.1). The analysis of individual non-identity attributes highlights some general non-compliant behavioural characteristics (Section 4.2). CASS is applied on multiple identity attributes with different parameters and key results are presented (Section 4.3).

4.1. Credit application input data

In Australia, as part of the process to obtain credit, applicants sign an agreement which allows their credit history file to be accessed and checked (permission has been granted by applicants themselves). There are two main data sources which offer an opportunity to study the *modus operandi* of identity criminals. These are the subscriber-contributed blacklist: Shared Fraud Database (SFD) and subscriber enquiries made against the blacklist: Fraud Inquiry Log (FIL). In this paper, CASS is performed against two recent and consecutive monthly samples $m1$ and $m2$, each with several hundred thousand applications extracted from SFD and FIL. Let each day in m1 be represented by $m1d1, m1d2, \ldots, m1d31$ and the same is true for $m2$.

Privacy and confidentiality is enforced by hashing key personal identifiers, although it also means that fuzzy matching cannot be performed on the hashed attributes. With this safeguard, the actual identity in each application cannot be ascertained. In total, there are 30 extracted attributes and 24 are needed by CASS.

4.2. Non-identity attributes analysis output

This is a top-level retrospective analysis, basic statistics of the data enabled by spreadsheet software. The purpose of this section is to show that most of the knowledge observed in this sub-section, using basic statistics, has been reported before in data mining-based fraud detection literature (Phua et al., 2005), and that CASS can find more sophisticated fraud. The first figure relates to the scalability problem and the subsequent six figures reveal the broad indicators of application fraud. (The y-axis in Figs. 1 and 2 which represent credit application volume has been deliberately removed for confidentiality purposes.)

From $m1$ to $m2$, there is a 7% slump of applications. There are obvious day-of-week patterns: weekends have significantly fewer applications. The high volume of applications, especially on weekdays, present scalability issues for CASS (see Fig. 1).

From $m1$ to $m2$, known frauds fell by 28% as fraudulent applications takes time to be discovered and reported. There is a fraud trend in $m1$: proportion of fraud increases sharply on weekends (see Fawcett and Provost, 1997). On average, less than 1 in 200 applications were subsequently flagged as fraudulent (see Fig. 2).

For both months' applications, fraud rate is constant during office hours (9 am–5 pm) and after office hours (6 pm–8 am). Although there are many fewer applications in the early mornings, the rates of hourly fraud from 2 am (six times higher than average fraud rate), 4 am (nine times higher than average fraud rate), and 5 am (five times higher than average fraud rate) are much higher in comparison to the other hours in the day (see Fawcett and Provost, 1997; Cortes and Pregibon, 1999) (see Fig. 3).

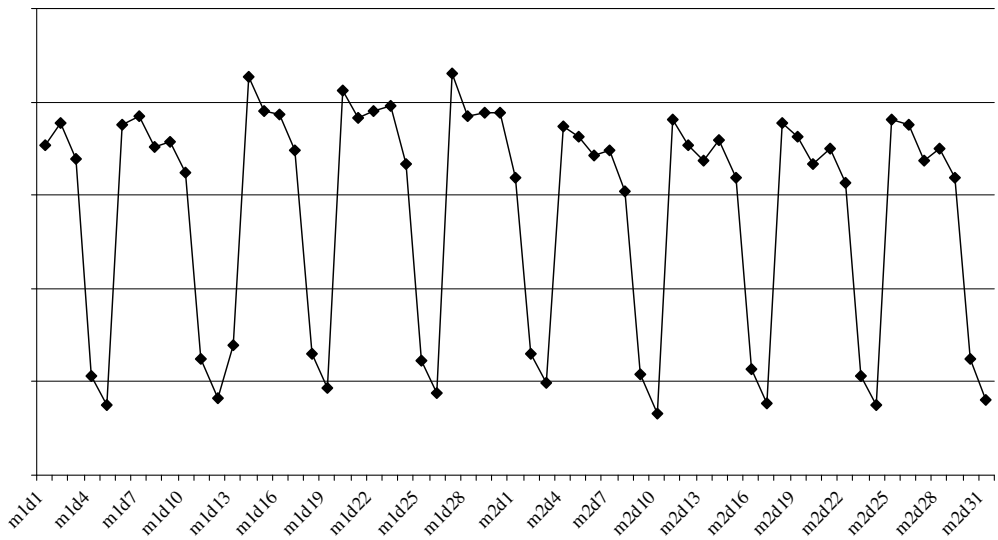

Fig. 1. Daily number of applications received in $m1$ and $m2$.

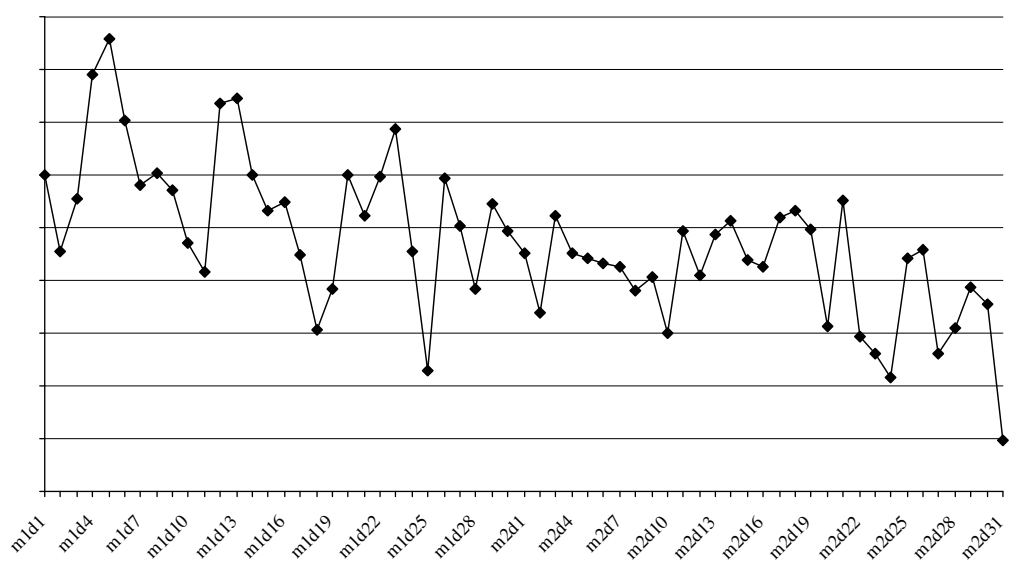

Fig. 2. Daily proportion of frauds in $m1$ and $m2$.

Rates of frauds are higher than the average fraud rate when the applications do not indicate their state of origin or when they come from New South Wales (NSW). Also, in $m2$, there is a more than sevenfold spike in the fraud rate from the state of Northern Territory (NT) (see Fig. 4).

Certain postcodes are more likely to be targeted by fraudsters: six of the top 10 most risky postcodes, each with at least 10 fraudulent applications per month, can be found in both $m1$and $m2$ (see Fawcett and Provost, 1997).

Higher risk postcodes are in closer proximity to each other, based on geographical centroid proximity of less than 15 km. In particular, postcodes 1–6 are near each other and to postcodes 7–19, 28–31; postcodes 8–18 are near each other and to postcodes 26–32. There is a strong indication here that, to avoid easy detection, the fraudster does not act alone and purposely submit applications with addresses that have different postcodes but are nearby to facilitate collection (see Fig. 5).

There are 31 subscribers of which 11 can be considered large (with at least 10,000 enquiries per month). Subscriber E (a large subscriber) and T (a smaller subscriber) have a significant rate of fraud in both months. Perhaps they have recently introduced a new credit product (fraudsters are notorious for probing the (in)vulnerability of the latest credit product) (see Fig. 6).

These are subscribers with at least 50 frauds within the two months. At opposite ends of the scale, subscriber E's frauds are all flagged by other subscribers and almost all of subscriber T's frauds are flagged by itself. The fact that fraudsters attack a single subscriber repeatedly (white), and/or target multiple organisations simultaneously (black), is not new (see Fig. 7).

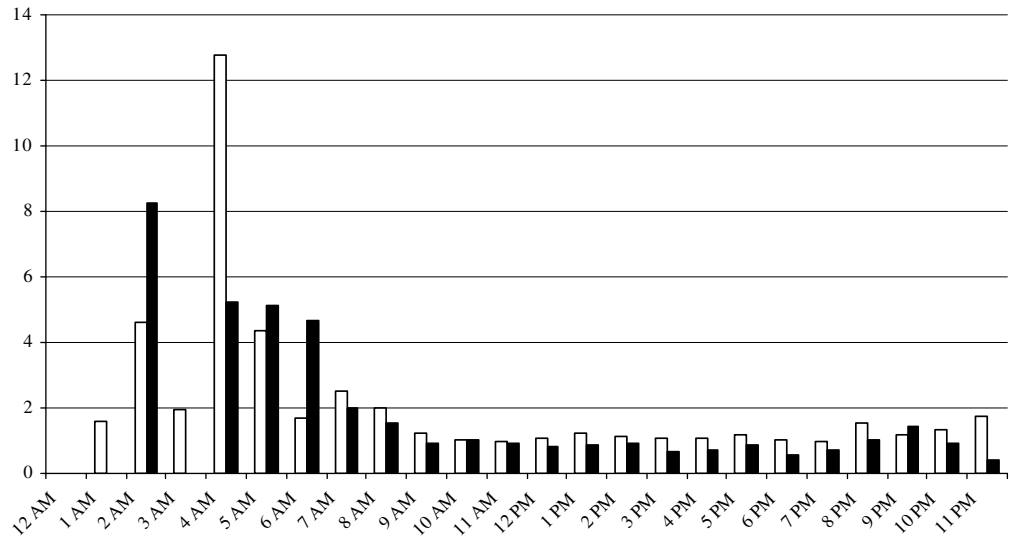

Fig. 3. Hourly proportion of frauds divided by average fraud rate in *m1* (white bars) and *m2* (black bars).

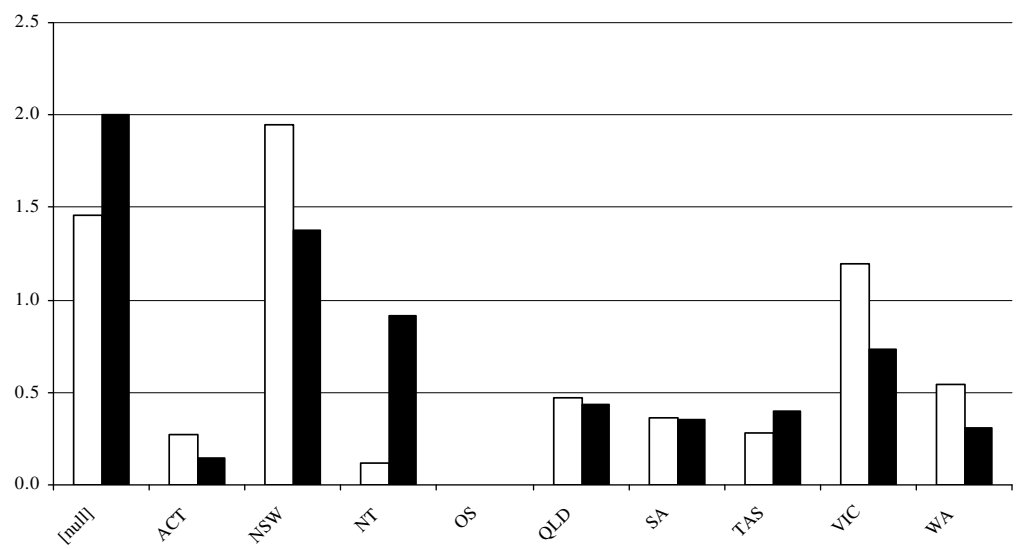

Fig. 4. Proportion of frauds by Australian state divided by average fraud rate in *m*1 (white bars) and *m*2 (black bars).

4.3. Identity attributes analysis output

The identity attributes analysis is a more sophisticated experimentation facilitated mainly by CASS than the previous sub-section on non-identity attributes analysis. All the experiments process the data in a stream-like fashion (justification and results for the experiments are provided in Sections 4.3.1 and 4.3.2, respectively), except for the second experiment which examines known fraud applications from a retrospective point-of-view (Section 4.3.3).

4.3.1. Experimental plan

The CASS parameters are classified by group A parameters – g, u, W, s, N and blocking; group B parameters – attribute weights, cross-matching and fuzzy matching, temporal and spatial weights (currently static, but should be dynamic); and group C parameters – $T_{\text{similarity}}, T_{\text{attribute}}, T_{E_1}, \alpha$. In all, group A parameters are the most important scalability factors because increasing g, W, N, decreasing u, s, and having no blocking will cause very long computation times. This is followed by those in groups B and C, although the parameters in all three groups have the potential to increase hit rates.

This exercise, of varying the groups A and B parameters and holding group C parameters constant, is important for two reasons:

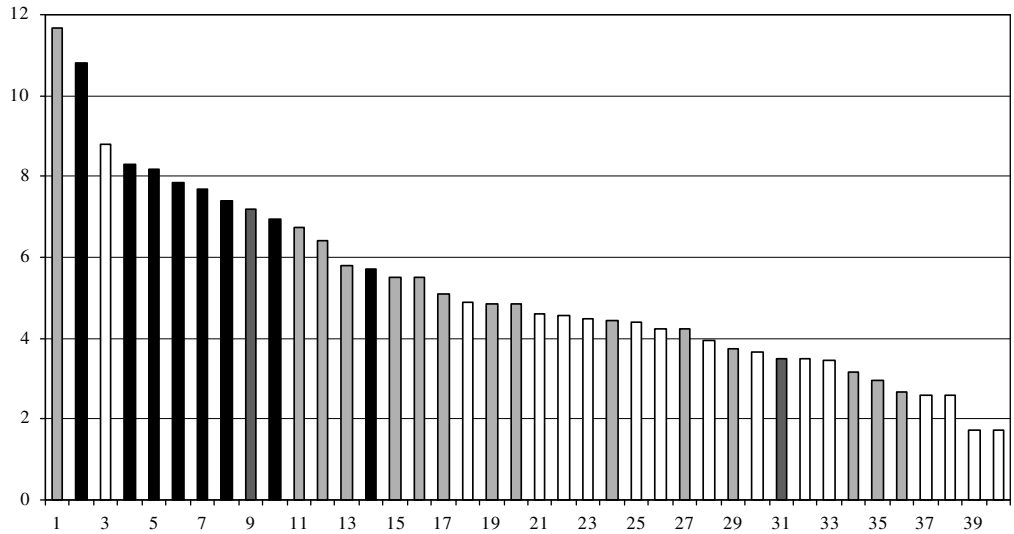

Fig. 5. Top 40 postcodes with highest proportion of frauds in both $m1$ (light grey bars) and $m2$ (dark grey bars), both months (black) and none (white).

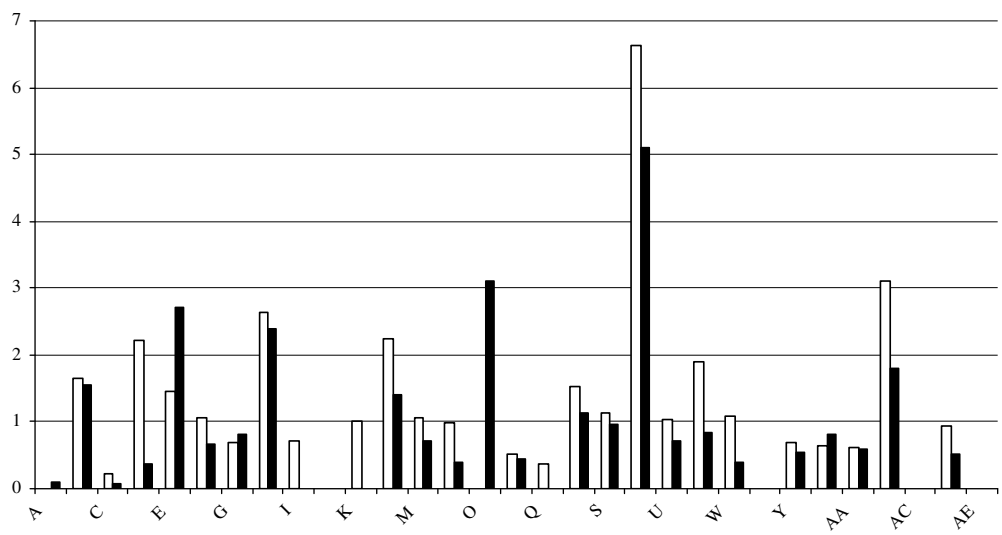

Fig. 6. Proportion of frauds according to each subscriber divided by average fraud rate in $m1$ (white bars) and $m2$ (black bars).

- First, it allows the assessment of the types of relationships in the whitelists. It also allows the understanding of which group A and/or B parameters under which CASS will scale and perform well on $m2$ data via score and graph measurements (Section 4.3.2).
- Second, is to shed light on dynamic relationships between applications and types of relationships, both fraudulent and legitimate, which are new, interesting, and actionable and visualise this as a graphical structure (Section 3).

With reference to Table 3 where a series of ten experiments were carried out, a brief description of each experiment is given below:

- 1(i), (ii), and (iii) trains *m1* data with different *W*s. The whitelists can help lower the false alarm rates in $m2$ data and provide insights on the common and rare types of relationships.
- In a retrospective manner for $m1$ and $m2$ data, 2 matches *all* the known fraud applications against *all* the supposedly legal applications. In this way, it is easier to extract insights from blacklists and whitelists.
- From this experiment onwards, scoring is done in real-time. 3 scores $m2$ application streams by turning off group B parameters. This is to show which parameters can provide higher hit rates. 4, 5, 6 switches on cross-matching only, both cross- and fuzzy matching only, and all parameters, respectively.
- With group A parameters, 7(i) and 7(ii) tests out larger us to determine how much it increases computational efficiency. Under experimental conditions, the optimum setting is $u = g$, meaning only 1 access to the database; but this is subject

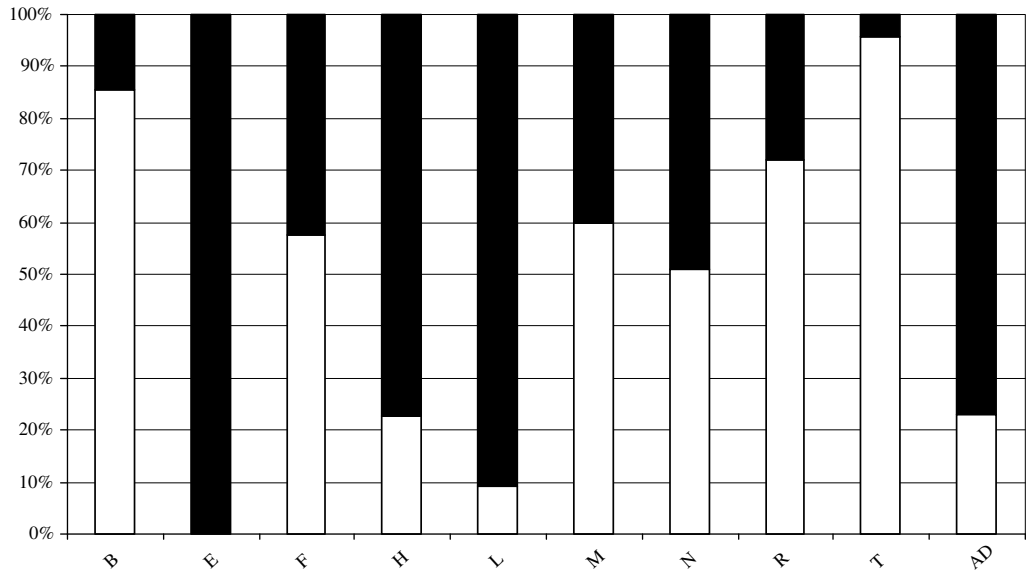

Fig. 7. Top 10 subscribers with most fraudulent applications from Fig. 6: either flagged their applications as frauds themselves (white section of bar) or by another subscriber (black section of bar).

Table 3
User-defined parameters for each experiment

#	$g = u\text{ł} \times p$	W	s	N in $\Re$
1(i)	**444,209**	**500**	0	0
1(ii)	444,209	**10,000**	0	0
1(iii)	444,209	**20,000**	0	0
2	**2082** = 2082 × 1	**885,823**	0	0
3, 4, 5, 6	**410,000** = **1000** × 410	500	0	**233**
7(i)	410,000 = **5125** × 80	500	0	233
7(ii)	410,000 = **10,250** × 40	500	0	233
8	410,000=10,250 × 40	**10,000**	0	**275**
9	410,000 = 10,250 × 40	10,000	**0.001**	275
10(i), (ii), (iii)	410,000 = 10,250 × 40	**20,000**	0.1	**301**

to memory constraints. Under an operational situation, the optimum setting is $u = 1$, where there are g accesses to database; but experiments will be constrained by the excessive database access times.

- 8 increases W by twentyfold to seek an increase in hit rate.
- 9 filters out the least suspicious applications (those with no links) with a different s to improve scalability, so that the majority of the applications in u can be matched against recent applications with higher suspicion scores.
- 10(i), (ii), and (iii) doubles the size of W such that it is now matching against close to a busy weekday's worth of prior applications, and all streaming applications in u are matched against highly suspicious prior applications. In general, W should be as large as possible to maximise hit rate, but should not be too large as the value of data decreases over time (for example, people move house or fraudsters forget previous attempts), and $u = \sqrt[2]{W}$. 10(i) only uses the blacklist, 10(ii) uses the blacklist and anomalous application-pairs (no whitelist), and they act as a comparison to 10(iii). To achieve significant performance speed-up while maintaining reasonable results, these three experiments utilise standard blocking (Baxter et al., 2003) where one or more attributes are employed to reduce the number of pair-wise matches (blocking increases scalability).

For all 10 experiments, group C parameters $T_{\text{similarity}}, T_{\text{attribute}}, T_{E_1}, \alpha$ are fixed at 0.8, 3, 10, and 0.8, respectively.

4.3.2. Experimental results

After training $m1$data, Table 4 contains the top 10 most common whitelist items R_x of each of the three different sizes of W in experiments 1(i)–(iii). There are exact applications submitted by the same identity – 1(v), (viii), (x); 2(i), (ii), (viii); and 3(i), (ii), (vii), (x). Some relationships are redundant – 1(i), (ii); 2(iii), 2(iv), 2(v), 2(vii), 2(ix); 3(iii), 3(iv), 3(v). Some applicants are possibly neighbours – 1(iv), 1(ix) and 3 (vi). The rest could be slightly different applications submitted by relatives, housemates, or friends.

Table 4
Some types of common relationships on training data

#	1. $W = 500$	2. $W = 10{,}000$	3. $W = 20{,}000$
i	N?YNYN??N?NY	Y????Y?YYY?Y	Y?YYYY???YYY
ii	N?NYYN???NNY	Y?YYYY???YYY	Y?YYYY??YY?Y
iii	N?YYYY??N??Y	N?YNYN??N?NY	N?YNYN??N?NY
iv	N?YYYN??NNNY	N?NYYN???NNY	N?NYYN???NNY
v	Y?YYYY???YYY	N?YYYN??N??Y	N?YYYN??N??Y
vi	N?YYYY????NY	N?YYYY??NNNY	N?YYYN??NN?Y
vii	N?YYYY???N?Y	NYYNNNNNNN?Y	Y?YYYY??Y??Y
viii	Y?YYYY??Y??Y	Y?YYYY??YY?Y	N?YYYY??NNNY
ix	N?YYYN??N?NY	NYYNNN??NN?Y	N?YYYY????NY
x	Y????Y?YYY?Y	N?YYYY????NY	Y?YYYY?YYY?Y

In contrast, anomalous relationships occur less frequently and there are certain types of relationships which are worth investigating. For example, applications submitted by the same legitimate identity are unlikely to have pair-wise relationships like: **Y**XXXXXXXX**NY**X, **N**XXXXXXXX**YY**X, **N**XXXXXXXX**YN**X, and **N**XXXXXXXX**Y?**X. And applications submitted by the social network of a legitimate identity are not likely to have relationships such as: XXX**NYY**XXXXXX, XXX**YNY**XXXXXX, XXX**NNY**XXXXXX. These anomalous relationships and their variants have been found in small quantities in $m1$data and they have been, by default, left out of the whitelist.

As the minority class is very rare, four different score output measurements are used. They include average suspicion score $\Omega = \sqrt[g]{\sum_{i=1}^{N} S(v_i)}$, percentage of known frauds in the top 1% of ranked applications NTOP1 – 4100 applications, hit rate = (known frauds/predicted frauds) with thresholds $T_{\text{lower}} = 0.5$ and $T_{\text{upper}} = 0.8$, and cost savings.

Score results in Table 5 show that Ω is highest for 2, 10(ii), and 10(iii) which are mainly caused by a much larger W that directly increases k for every v_i (most are redundant relationships). In addition, for 2, every application-pair is given the maximum $w_{\text{communal}_{ij}}$, and for 10(ii) and 10(iii) due to a non-zero, larger s and a high alpha α, many application-pairs have higher $S(v_j)/E_{\text{O}}(v_j)$. For NTOP1 and hit rates with two different thresholds, 8 and 9 have much better results than the rest and this indicates that frauds have relatively spiky behaviour. Although the W used in 3–7 are very small, 6 and 7 still has comparatively good hit rates when all group B parameters are utilised as false alarm rates are lower. But in comparison with 10(ii), 10(iii) does show that a whitelist reduces false alarm rates.

In Fig. 8, with $T_{\text{lower}} = 0.5$ and $T_{\text{upper}} = 0.8$, 8 has the highest hit rate of 14.4% (63 out of 439 applications) and 18.1% (62 out of 343 applications) respectively, and closely followed by 7(i), 7(ii) and 9. These are considerable improvements over randomly rejecting applications at baseline of 0.21% (only one out of 439 and 343 applications). To put it in a monetary context, if all the applications in 8 and 9 above T_{upper} were investigated right after they were scored in real-time, the industry would have significant cost savings of (AUD\$2000 × 62=)AUD\$124,000 in $m2$. The cost for each credit application fraud is based on the assumption that the average credit card limit and debt (Liberty Financial, 2004) is close to AUD\$2000.

10(i), 10(ii) and 10(iii) did much worse in comparison to the rest at $T_{\text{lower}} = 0.5$ with hit rates of 0.45% (one out of 224 applications), 0.26% (34 out of 13,508 applications) and 0.38% (36 out of 9368 applications) respectively. However, 83.3% (30 out of 36 fraudulent applications) detected from 8 and 9 were unique. For example, with $T_{\text{lower}} = 0.5$, all 13 frauds (100%) in 6 can be found in 8, 47 out of 63 frauds (74.6%) in 8 can be found in 9, but only seven out of 62 frauds (11.3%) in 9 can be found in 10.

Table 5
Score output measurements on scoring data

#	Ω	NTOP1 (%)	$\geqslant T_{\text{lower}}$ (%)	$\geqslant T_{\text{upper}}$ (%)
2	0.745	–	–	–
3	0.006	0.34	5.61	9.68
4	0.006	0.34	5.61	9.68
5	0.007	0.35	6.05	10.16
6	0.005	0.32	11.11	15.85
7(i), (ii)	0.005	0.32	11.11	15.85
8	0.011	1.66	**14.35**	**18.08**
9	0.016	**1.73**	11.27	16.44
10(i)	0.001	0.22	0.45	0.45
10(ii)	0.062	0.39	0.25	0.43
10(iii)	0.051	0.59	0.38	0.60

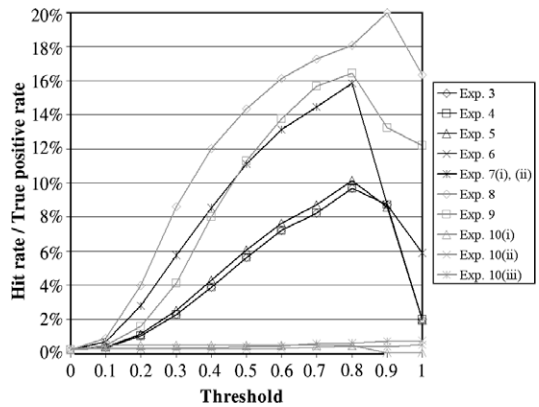

Fig. 8. Hit rate with different thresholds of the stream processing experiments.

Table 6
Graph output and time measurements on scoring data

#	h	$\bar{d}$	Δ	t
2	32,774	15.742	0.007564	185.153
3	44,522	0.109	0	0.157
4	45,579	0.111	0	0.158
5	47,710	0.116	0	0.179
6	47,710	0.116	0	0.181
7(i), (ii)	47,710	0.116	0	0.125, 0.115
8	131,602	0.321	0.000001	2.213
9	172,599	0.421	0.000001	2.211
10(i)	261	0.001	0	0.498
10(ii)	203,187	0.496	0.000001	0.593
10(iii)	207,382	0.506	0.000001	1.311

The three graph output measures consist of the number of links h, mean nodal degree $\bar{d} = \sqrt[g]{h}$, and graph density $\Delta = \sqrt[g(g-1)]{h}$. The number of seconds to process each application is represented by t.

Graph and time measurements in Table 6 reveal that as W and N gets larger, h, $\bar{d}$, Δ, and t correspondingly increases. 7 shows that by finding the appropriate u, t can be reduced without affecting graph measurements; and 10 shows that blocking reduces computation time by around 75% (W has doubled in 10 from 9, but t is halved in 10(iii)). 5, 6, 7 have a higher h and $\bar{d}$, are more effective in detecting fraud than 3, *4*. This is due to more links by fuzzy matching and fewer false alarms by temporal and spatial weights. 9 and 10 shows that by even increasing s a little, h and $\bar{d}$ will increase significantly (resulting in many redundant links).

4.3.3. Retrospective insights

In this micro-section, the discussion using only experiment 2 (all known fraud applications are matched against all supposedly legal applications) is focussed on understanding the changes in frequency, strength, and types of relationships/links, over time, between high-score applications which will most likely be investigated. Also, the discussion will assist in the better understanding of the nature of credit application fraud/identity crime from the point-of-view of hindsight, not from that of a stream processing one. Sections 4.3.3.1 and 4.3.3.2 show the importance of the blacklist and the CASS algorithm, respectively. Here, all graphs are in circular layout format and all links which are based on $w_{\text{communal}_{ij}}$, not w_{ij}.

4.3.3.1. Blacklist patterns. Fig. 9 is based solely on known fraudulent applications inferring that there are professional criminals operating. It also shows the nature of the fraud (some types of fraud styles are more serious and/or obvious than others) and that there are fraud links (discovered by CASS) between these known fraudulent applications. The figure also illustrates that there is evolution over time: the arrows depict temporal sequence. The more arrows pointing to a node indicates that the node is older than the rest. The nodes can be labelled with date/time, and links can be labelled with time difference (but this is not represented in the diagrams as it would clutter the presentation). Also, the user can choose to label the nodes or links with other attributes, e.g. label nodes as postcode, and links as kilometres difference.

In Fig. 9, the fraudulent applications (square nodes) are often connected to each other and recur even *after* they have been tagged fraudulent; and some styles (components) of fraud are more frequent than others. Each fraudulent application

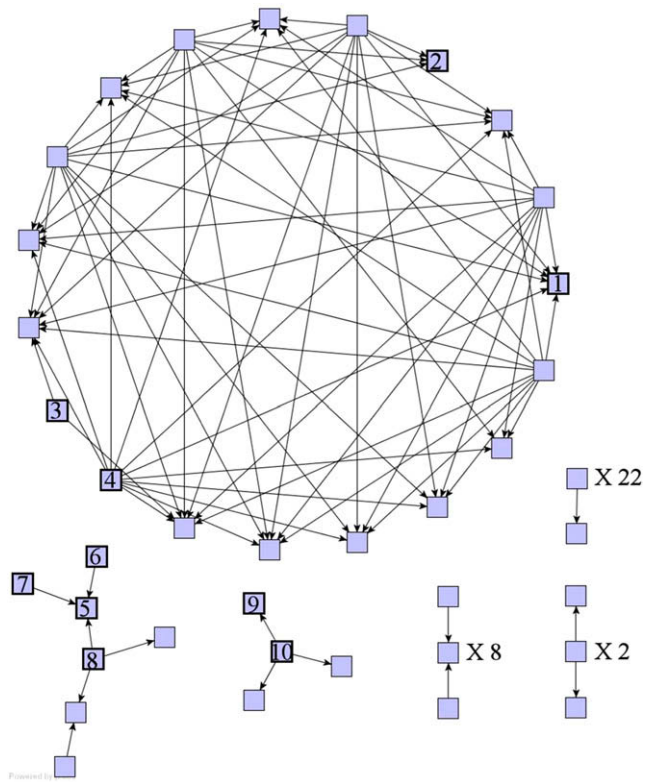

Fig. 9. Using experiment 2, a retrospective view of about a hundred known frauds with blacklist links from $m1$ to $m2$.

is in the blacklist and each link is a connection to at least one known fraud. These known fraudulent applications and/or styles in $m1$ and $m2$ are possibly connected to other known frauds in preceding months.

There are 35 separate styles where each has varying frequency (from as few as two/three to close to twenty applications in each style), although some frauds are likely to be standalone and/or are not obvious. There are also differences in strength and types of links (from changing many, a few, or one attribute(s) in a current from a previous application) within the three different styles of fraud and this is briefly described below.

The top circle in Fig. 9 represents the biggest (19 applications) and longest-lasting (52 days) fraud style which usually occurs during morning and noon. Every application has been submitted by the same fraudster who re-uses a few attributes and submits to several different subscribers. For example, the most recent application 4 is directly linked to the oldest application 1. Applications 3 and 2 are the least-similar applications with the minimum number of inlinks and outlinks respectively, and neither have any direct link to each other within $T_{\text{attribute}}$.

The bottom left structure in Fig. 9, with seven applications submitted over 23 days during all times of the day, is a classic fraud style which reacts to and evolves accordingly to counter-measures. Once their style has been detected, the fraudsters modify their fraud style slightly and submit to different subscribers. Application 5 is submitted on day 1 to subscriber 1, and this is flagged fraudulent on day 13 by subscriber 1. Subsequently, application 6 is received on day 14 by subscriber 2 with changes to one attribute, which will be rejected (as application 5 has already been flagged as fraudulent). On day 23 (9 days later), application 7 followed by application 8 are received by subscribers 3 and 4. And both had changes to only two attributes from application 5.

The bottom middle structure of four applications spans 24 days during all times of day. The fraudster submits application 9 on day 1 to subscriber 1 and this is flagged fraudulent 5 days later by the same subscriber. On day 24, application 10 with major modifications (only current and previous addresses are the same) is submitted to subscriber 2.

Fig. 10 demonstrates that for a certain fraud style, identity attributes (octagon nodes) in subsequent fraudulent applications can undergo a series of various modifications in strength and types of links.

In chronological order and in reference to application 1 (original application), the only two attribute values are re-used in application 2 (2v and 2x) and in application 3 (3vii is same as 1vi, and 3ix), but 6 attribute values are re-used in application 4 (4iii, 4iv, 4v, 4vi, 4ix, and 4x).

4.3.3.2. Other patterns. Fig. 11 highlights that all legitimate applications (eclipse/circle nodes) here are actually fraudulent, therefore, understating the fraud rate; and there are interesting temporal and spatial characteristics in fraud styles. The

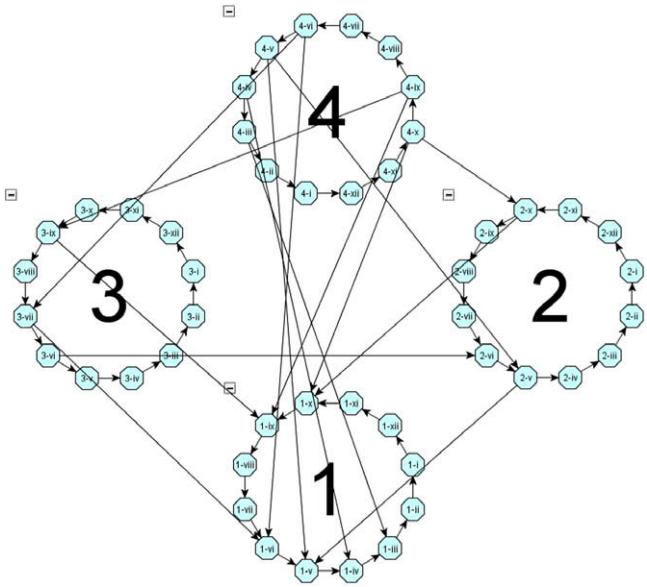

Fig. 10. Using experiment 2, a drill down of four applications in Fig. 9 into attribute level.

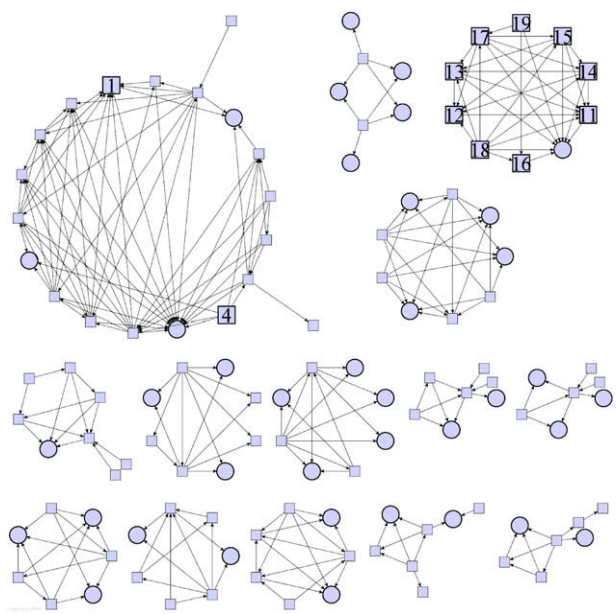

Fig. 11. Using experiment 2, a retrospective view of more than a hundred normal and anomalous applications with high $w_{\text{communal}_{ij}}$ from $m1$ to $m2$.

applications here are different from those in Fig. 10 because the fraudulent applications (square nodes) have no connections to the blacklist *yet* and have high suspicion scores, so these are the ones which CASS has to rapidly detect in application streams.

There are 14 different fraud styles with at least seven applications which are strongly connected to each other. Many applications of the same style basically have the same attributes, but with slightly altered spellings or similar-sounding words.

Fraud is understated because of the time delays for fraud to be revealed and this is not subsequently reflected. Usually if an application has been discovered as fraudulent, only the first enquiry is logged only once in the blacklist and subsequent enquiries are not logged by subscribers (although they will most likely be rejected).

The duration of each sustained fraud style attack in Fig. 11 is an average of 32 days, indicating that these attacks are usually quite well-spaced although sometimes they can be spiky within a single day. The geographical locations, however, do reveal possible collaboration between fraudsters. Within the same style, fraudulent applications 11, 16, and 19 share the exact same identity attributes as the rest, but have different addresses, postcodes, and state details.

4.3.3.3. Lessons and limitations. The main overall benefit of CASS is its ability to mine and score application streams rapidly. Listed below are two pieces of information that were identified in real-time:

- Contrary to the prevailing view, many sophisticated application fraud patterns (high frequency, varying strengths and types in each style) are found during typical office hours and this was the case with many subscribers.
- Within a short period of time and to the same subscriber, it is quite common to find many application-pairs of one particular normal relationship, as well as many exact duplicates which are more likely to be errors than fraud.

In general this axiom applies to most data mining-based detection systems: an attempt to improve one measurement (detection time, hit rate, false alarm rate, or scalability) will usually affect another adversely. This also applies to CASS:

- Cutting down on the false alarm rate with the whitelist is the most novel element in CASS. Without it, fraud patterns will be swarmed and masked by normal activity and data errors. But at the same time, determining the length of the whitelist is a trade-off between efficiency (rapid detection time and high scalability) and effectiveness (high hit and low false alarm rates).
- Scalability is a major limitation of CASS. Given that all parameters are being utilised to maximise hit rate, there will always be a constraint on the window size W.

Listed below are two common compromises of confidentiality and privacy:

- Matching credit applications against an external data source such as public telephone and address directories, semi-public voters' register, and/or private blacklist from another organisation. This can be resolved by first applying a mutually-agreed one-way cryptographic hash function on all records from all sources before sharing/comparing them. This means that personal identifiers are transformed into anonymised alpha–numeric strings.
- Verifying of sensitive personal information by a human investigator such as checking against credit history. This is best regulated by a selective revelation scale (on a need-to-know basis). For example, no suspicion score on an application means only anonymised attributes available to the investigator; and with higher suspicion scores more identity attributes of only related records are accordingly made available.

5. Conclusion

CASS is a new, low false alarm, rapid credit application fraud detection tool and technique which is complementary to those already in existence. It identifies a large number of individuals and keeps scores of the relations between them, detecting normal/abnormal and small/large similarities. The results presented here indicate substantial cost savings by investigating/rejecting only a few hundred of the most suspicious credit applications out of a few hundred thousand. The investigations, however, are limited by use of two relatively small data samples $m1$ and $m2$, a lack of complete understanding of whitelist and data quality issues, and the lack of an academic benchmark in this domain. As fraud is adaptive, an interesting direction is to implement a systematic change of CASS parameters and weights, automatically, without degradation of performance.

Acknowledgements

The first author was financially supported by the Australian Research Council under Linkage Grant Number LP0454077 whilst he was in Monash University as a PhD candidate. Ethics approval has been granted by Monash SCERH under Project Number 2005/694ED. The real credit application data has been provided by Veda Advantage. Special thanks go to the developers of yEd and particular participants in the Credit Scoring and Credit Control (CSCC05) conference for useful comments.

References

Baxter, R., Christen, P., Churches, T., 2003. A Comparison of fast blocking methods for record linkage. In: Proceedings of SIGKDD03 Workshop on Data Cleaning, Record Linkage, and Object Consolidation, pp. 25–27.

Baycorp Advantage, 2005. Zero-Interest Credit Cards Cause Record Growth In Card Applications.

Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., Fienberg, S., 2003. Adaptive name matching in information integration. IEEE Intelligent Systems 18 (5), 16–23.

Chapman, S., 2005. Simmetrics – Open Source Similarity Measure Library. Accessed from: <http://sourceforge.net/projects/simmetrics/> (accessed April 2005).

Cortes, C., Pregibon, D., Volinsky, C., 2003. Computational methods for dynamic graphs. Journal of Computational and Graphical Statistics 12, 950–970.

Cortes, C., Pregibon, D., 1999. Information mining platforms: an infrastructure for KDD rapid deployment. In: Proceedings of SIGKDD99, pp. 327–331.

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

ID Analytics, 2004. Identity 2004: The Identity Risk Management Conference.

Kleinberg, J., 1999. Authoritative sources in a hyperlinked environment. Journal of the ACM 46 (5), 604–632.

Kubica, J., Moore, A., Cohn, D., Schneider, J., 2003. Finding underlying connections: a fast graph-based method for link analysis and collaboration queries. In: Proceedings of the ICML, pp. 392–399.

Levitt, S., Dubner, S., 2005. Freakonomics: A Rogue Economist Explores the Hidden Side of Everything. Penguin Books, Great Britain.

Liberty Financial, 2004. Fears over Holiday Credit Binge.

Macskassy, S., Provost, F., 2005. Suspicion scoring based on guilt-by-association, collective inference, and focused data access. In: Proceedings of the International Conference on Intelligence Analysis.

Oscherwitz, T., 2005. Synthetic identity fraud: unseen identity challenge. Bank Security News 3 (7).

Page, L., Brin, S., Motwani, R., Winograd, T., 1998. The PageRank Citation Ranking: Bringing Order to the Web. Stanford Digital Library Project Technical Report.

Phua, C., Lee, V., Smith, K., Gayler, R., 2005. A comprehensive survey of data mining-based fraud detection research. Clayton School of Information Technology, Monash University.

Wasserman, S., Faust, K., 1994. Social Network Analysis: Methods and Applications. Cambridge University Press, New York.

Witten, I., Frank, E., 2005. Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Francisco.