HAL
archives-ouvertes.fr

Neural Network Rule Extraction to Detect Credit Card Fraud

Nick Ryman-Tubb, Paul Krause

▶ To cite this version:

Nick Ryman-Tubb, Paul Krause. Neural Network Rule Extraction to Detect Credit Card Fraud. 12th Engineering Applications of Neural Networks (EANN 2011) and 7th Artificial Intelligence Applications and Innovations (AIAI), Sep 2011, Corfu, Greece. pp.101-110, 10.1007/978-3-642-23957-1_12 . hal-01571371

HAL Id: hal-01571371
https://hal.inria.fr/hal-01571371

Submitted on 2 Aug 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution| 4.0 International License

Neural Network Rule Extraction to Detect Credit Card Fraud

Nick F Ryman-Tubb[1], Paul Krause[1]

[1] University of Surrey, Department of Computing, Guildford, Surrey, GU2 7XH, UK.
email: n.ryman-tubb@surrey.ac.uk, p.krause@surrey.ac.uk

Abstract. Neural networks have represented a serious barrier-to-entry in their application in automated fraud detection due to their black box and often proprietary nature which is overcome here by combining them with symbolic rule extraction. A Sparse Oracle-based Adaptive Rule extraction algorithm is used to produce comprehensible rules from a neural network to aid the detection of credit card fraud. In this paper, a method to improve this extraction algorithm is presented along with results from a large real-world European credit card data set. Through this application it is shown that neural networks can assist in mission-critical areas of business and are an important tool in the transparent detection of fraud.

Keywords: fraud detection, credit card, neural applications, rule extraction, neuro-symbolic, data mining.

1 Introduction

The Oxford dictionary defines fraud as, *"wrongful or criminal deception intended to result in financial or personal gain"*. Fraud can involve the use of stolen credit, debit, fuel or gift cards, opening a bank account through identity theft, concealing the source of illegally or criminally received money, making a wrongful or exaggerated claim for medical expenses through insurance or social security claims, etc. Fraud can be found in on-line shopping, computer gaming, telecommunications, banking, social security, tax evasion, customs fraud, insurance claims, payments and financial services, to the illegal sale of endangered species. The list of crimes and their targets is disappointingly long.

Public perceptions of fraud are often tempered by a belief that it is a "white-collar" crime which targets the wealthy, government and big business and is of less concern as the effects are cushioned for the victim [1]. However, violent criminals are increasingly moving into fraud [2] so that fraud can involve the threat of violence including murder. In the USA, the fear of fraud supersedes that of terrorism, computer and health viruses and personal safety [3]. In the UK the Attorney General described fraud as, "*second only to drug trafficking in causing harm to the economy and society*." [4]. Today, the proceeds from fraud are paying for organised crime, drug smuggling and terrorism [5]. It was estimated that in the UK fraud cost £30bn in 2010, that is 1.3% of the entire UK economic output [6]. In the US, the fraud cost was

estimated at $994bn in 2008 [7]. Whatever country, it seems that fraud is large and there is a human cost for each individual act of fraud [8].

The detection of fraud is a complex scientific and business challenge. The datasets are large and therefore need to be properly sampled as it becomes computationally impractical to use the natural population. Real world transactional data is noisy, unbalanced, computationally expensive to maintain and highly dimensional. Such data is skewed, has uneven distributions and contains a mixture of symbolic and continuous variables. The dataset is sparse - there is a large quantity of transactional data which contain only a small number of example frauds. Often researchers are unable to report on real fraud datasets due to their sensitivity with many published papers using either synthetic or small datasets; where there is no reason to believe that the conclusions drawn will hold true when they are scaled-up to large, real-world applications [9], [10].

1.1 Payment Card Fraud

One common type of fraud is payment card fraud – this is the criminal act of deception through the use of a physical plastic card or card information without the knowledge of the cardholder. When a transaction takes place, the details of that transaction are processed by the acquiring bank for authorisation. It is reported that in 2009, the USA total card fraud losses cost banks and merchants $8.6 billion [11] and in the UK £609.9 million [12]. To detect this fraud, organisations use a range of methods; manual methods and some form of automated Fraud Management System (FMS) [13]. The FMS is often a rule-based system that stores and uses knowledge in a transparent way and is easy for a Fraud-Analyst to modify and interpret. Rules provide a convenient mechanism for explaining decisions. However, the generation of comprehensible rules is an expensive and time-consuming task, requiring a high degree of skill both in terms of the developers and the Fraud-Analysts concerned. The performance of the FMS is dependent upon the skill of the Fraud-Analyst and how past data and events are interpreted by them.

An alternative to rules is the use of a learning approach that does not require expert knowledge but learns from examples given in transaction data. A model is formed that can then be used on new transactions to make a decision. The ability of such a model to generalise is fundamental. One such method is to use supervised neural networks that have been widely used to learn from fraud data [9], [14], [15].

Supervised neural networks are essentially a collection of large numbers of real-valued parameters (weights, etc.) with no obvious method to determine their meaning. The knowledge is represented by the distributed weights between the connections, the threshold values and activation functions. The uptake and use of neural networks has been hampered by their “black box” nature and the requirement for often proprietary software to implement the neural network model to be deployed on a server within a mission-critical part of the business – a serious barrier-to-entry in the automated use of FMS.

1.2 Mission Critical Application

Once a transaction is made a decision has to be taken to accept or refer/decline the payment. This authorisation process is part of a real-time payments system, which means that the FMS is mission critical – it's failure will cause damage to the business in terms of both money and reputation. It is for this reason that many businesses are reluctant to deploy a FMS that is based on a neural network, where the exact method of fraud detection is kept hidden and they are reliant on their supplier. Many businesses require a transparent approach to fraud detection so that both Fraud-Analysts and management teams can understand exactly why a decision to refer/decline is being made.

A practical solution to promote the widespread use of neural networks within FMS is to use them off-line. One approach is to use the neural network to create evolving models that automatically learn from patterns in transaction data of payment cards that can then be used to extract knowledge in the form of symbolic rules. The rules can be formatted into human-readable ANSI SQL statements that can then be deployed within an existing live Fraud Server environment. This will allow existing FMS systems (that support SQL) to be used and reduce the need for expensive and time consuming creation of manual rules and reliance on Fraud-Analysts.

1.3 Rule Generation

There are a number of rule induction algorithms that can be used to generate rules from a dataset. One common method is the decision tree - based on information gain [16]. In applications where the dataset is large and contains noise, such as fraud detection, this approach and others have been found to generate a large number of rules where each rule has many conditions and are unfortunately difficult to understand. A key objective for a FMS is to use as few rules as possible – as a "global" view of the fundamental fraud factors is generally preferred over actual accuracy. Neural networks have shown considerable promise in terms of accuracy with generalisation for fraud detection. There are two key approaches to rule extraction from a neural network: (1) Decompositional, (2) Pedagogical. A decompositional approach is where rules are created using heuristic searches that guide the process of rule extraction [17], [18] by decomposing the neural network architecture and therefore produce rules that represent the internal architecture. Since there is no reason for individual neurons to represent a recognisable "concept", the extracted rules are often not sufficiently comprehensible. A pedagogical approach is where a set of global rules are extracted from the neural network in terms of the relationships between only the inputs and the outputs [19], [20]. These individual rules are then combined into a set of rules that describes the neural network as a whole. The main difficulty with this approach is that the size of the search space is large so that a straightforward search is not practical in real-world problems, where even a small number of input neurons (fields) mean an unrealistic level of computing power is required.

2 SOAR Extraction

The Sparse Oracle-based Adaptive Rule (SOAR) extraction algorithm is detailed in [21] and uses sensitivity analysis to avoid the exhaustive decision boundary searches of other neural rule extraction algorithms e.g. [22], [23], [24]. The SOAR algorithm shown in Figure 1, is independent of the neural structure and here uses a standard Multi-Layer Perceptron (MLP) as a Neural "Oracle" which was chosen as a classifier to produce good generalisation and noise-tolerance. Sparse fraud examples are used as the initial search space "seeds" – since (randomly) locating fraud points on the decision boundary in a large search space would be otherwise inefficient. SOAR has the following steps:

1. Continuous valued fields are converted into discreet literals using a well known clustering algorithm called ART2 [25] that is applied to group together similar fraud examples. The algorithm creates a new cluster when the input does not belong to the cluster that was determined as most probable, based on a user defined parameter. The Euclidean distance measure is used for measuring similarities between the input pattern and the exemplars in each cluster.
2. The trained MLP neural system is used as an oracle, to interrogate.
3. Prototypes are formed by grouping similar examples from the fraud dataset class to reduce complexity.
4. The prototype generated is "expanded" to cover the largest area on the decision boundary that continues to represent a single class e.g. *{genuine, fraud}*. This is accomplished using the neural network as an oracle where the binary digits in each literal in the prototype are sequentially activated e.g., {1,0,…0}, {0,1,…0} until {0,0,…1} and the class membership determined by the oracle. In the case of a discretised numeric type each b_i represents a contiguous real range and so the search is simplified by only having to activate each in turn, $(\{b_1,b_2,b_m\}, \oplus)=1$. Only fields that represent continuous values are expanded since it would make no sense to expand an unordered nominal field. The aim is to capture the generalisation of the neural model while covering as much of the space contiguously as is possible. This is then used to create a propositional rule as a list of antecedents as a combination of the expanded input fields.
5. The rules are optimised to avoid overlap, duplication and redundancy. This step includes rule pruning to remove rules that have a large estimated error and those which never "fire" (have zero coverage) for the training set.
6. A false-positive ratio is calculated for each rule using the Training Dataset. Those rules that have a large ratio (i.e. produce a large number of false-positives compared to the correct number of frauds detected) are removed.

Since the fraud data examples are sparse, the above procedure is computationally efficient. This approach will approximate the classification capability of the neural network by creating a set of rules. The optimisation steps 5 and 6 above aim to improve comprehensibility by reducing the number of rules and approximation errors.

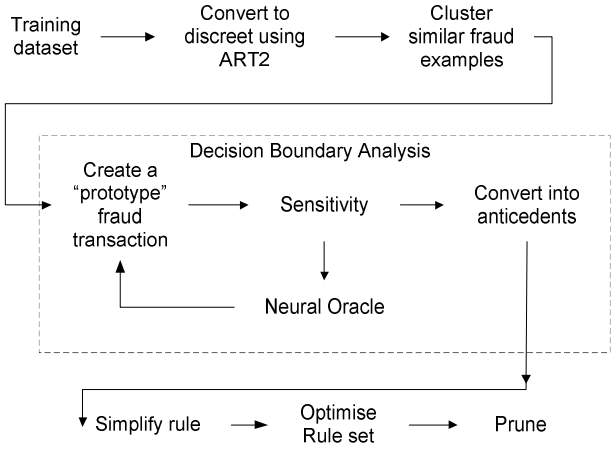

Figure 1. SOAR Extraction Framework.

2.1 Extended SOAR Extraction

In [21] the algorithm only extracted rules for the fraud class. The algorithm has been extended here, called SOAR II, so that steps 3 and 4 are repeated to extract rules for a sample of the genuine class. For those points that overlap/intersect within the two sets of rules, false positives will be generated and the fraud rule is "trimmed" so as to remove the overlap. To compare the performance of the two versions of SOAR extraction experiments were completed using the same dataset with both approaches.

3 Experimental Results

A dataset was provided by a large European company that processes a high volume of transactions per day. The fraud class priors are highly skewed. It has been shown that such a class imbalance may not allow learning algorithms to create good models [26] so sampling has been used. This sampled dataset was created from a natural population of transactions over a fixed time period so as to be representative of the natural population containing all of the known fraud, shown in Table 1.

Table 1. European Credit Card Fraud Dataset for Natural Population.

Feature	Total
Transactions	60m
Cards	8m
Number of Fraud Transactions	4,000
Value of Fraud Transactions	€1m
A priory transaction fraud	1:15,000

3.1 Pre-Processing and Sampling

The pre-processing was designed to enhance the modelling characteristics of the provided dataset through the following steps:

- All records checked for incomplete or missing data. These records are ignored if found.
- Additional feature fields (such as average spend in previous 30-days) are calculated and added to each transaction record.
- Attempts are made to identify outliers in the data and to remove them.
- Assessing the relevance of each field (to ensure maximum entropy).
- Identifying redundancy in the data.
- Relevance determination.
- Redundancy removal.
- Frequency histograms of numeric data are created to ascertain which variables could benefit from transformation.
- Frequency analysis of symbolic data is undertaken to ascertain which low frequency variables could benefit from being grouped (into SQL tables).
- Re-weighting and splitting.

The pre-processing splits the transaction data into two datasets, shown in Table 2.

Table 2. Sampled Datasets.

Training Dataset	Consists of a randomly selected proportion of the transaction data where the class is known. This information is used by the training algorithm to train the neural network to learn the relationships within the data.
Validation Dataset	Consists of a sample of data where the outcome is known and is used only once the model has been completed to validate the accuracy of the model.

3.2 Neural Network Training

A three layer MLP architecture was selected as is was the common choice in the previously mentioned research and so is representative of the range of classifiers that could be used. All neuron values were in the range [0,1] using a standard sigmoidal activation function. The training of the neural network is an iterative process that requires considerable processing power to complete where the more transactions

presented the longer the algorithm will take. For this reason a Training Dataset was created as a sub-sample shown in Table 3. The Training Dataset used a small sub-sample of 40,000 transaction records with 2,800 example fraud transactions. The remaining 1,200 fraud transactions were reserved as "unseen" in a Validation Dataset to produce performance metrics given in Table 4. The MLP was then trained using this Training Dataset by interleaving and repeating the fraud examples with the genuine examples so that they were re-balanced. The Conjugate Gradient Descent (CGD) [27] training algorithm was then used. Using a Constructive Algorithm [28] the number of hidden neurons in the hidden layer was automatically determined by starting at a single neuron and then dynamically growing the layer by adding additional neurons until the stopping criteria is reached; in this case, the final MLP had 60 discreet inputs, 10 hidden and 1 output neuron.

Table 3. Dataset Distribution.

	Total Sample	**Training Dataset**		**Validation Dataset**	
Total Transaction Records	1,100,000	40,000	4%	1,060,000	96%
Containing Fraud Records	4,000	2,800	70%	1,200	30%
Value of Fraud Records	€1,000,000	€703,000		€297,000	

3.3 Rule Extraction

Once the neural network was trained, the SOAR and SOAR II extraction algorithms were applied to the Training Dataset and two rule sets created. A confusion matrix was used to evaluate the performance of both the trained neural network and the extracted rule set using the Validation Dataset. The metrics used for calculating accuracy and precision are given in equations (1), (2) and (3).

$$accuracy = \frac{TP + TN}{TP + FN + FP + TN}. \tag{1}$$

$$precision_{genuine} = \frac{TP}{TP + FP}. \tag{2}$$

$$precision_{fraud} = \frac{FP}{TP + FP}. \tag{3}$$

While the standard evaluation metrics are important, in the practical application of the approach these need to be projected into the real world. Businesses that deploy a FMS generally have a case management system to review the fraud alerts generated by the rules and have a maximum number of alerts that they can reasonably review in a day. Therefore, performance needs to be projected to the natural population with reference to the number of alerts per day:

$$Sampling\ Rate\ S_g = \frac{\#Transactions\ Natural}{\#Transactions\ Sample}. \tag{4}$$

$$Segmentation\ Rate\ R_t = \frac{\#Transactions\ Training}{\#Transactions\ Validation}. \tag{5}$$

$$Transaction\ \ FP\ Ratio = \frac{\left(\mathrm{FP}.S_g.R_t\right)}{\mathrm{TP}}. \tag{6}$$

$$Transaction\ \ alerts\ per\ day\ T_d = \frac{\left(\mathrm{FP}.S_g.R_t\right)+\left(\mathrm{TP}.R_t\right)}{Days\ in\ period}. \tag{7}$$

$$Transaction\ \ correctly\ alerted\ per\ day\ T_{dc} = \frac{T_d}{FP}. \tag{8}$$

Here, S_g is calculated as 53 (4), R_t is 0.04 (5) and the resulting performance metrics are given in Table 4, where a "card" is a sequence of transactions linked by a single card number. The number of antecedents in each rule is a good measure on human comprehensibility; shorter and simpler rules are more straightforward to understand than complex rules.

Table 4. Evaluation using the Validation Dataset only.

	Neural	SOAR I	**SOAR II**
#Rules	-	47	**44**
Average Antecedents	-	9.4	**9.4**
Accuracy (1)	63%	91%	**93%**
$\mathrm{Precision}_{genuine}$ (2)	63%	91%	**93%**
$\mathrm{Precision}_{fraud}$ (3)	88%	87%	**73%**
Transaction False Positive Rate (6)	1:468	1:219	**1:215**
Transactions alerted per day (7)	7,385	1,734	**1,778**
Transactions correctly alerted per day (8)	8	8	**8**
Cards alerted per day	975	238	**201**
Cards correctly alerted per day	1.7	1.7	**1.6**

For reasons of space, just the top rule from the SOAR II extraction algorithm is presented - this single rule captures over 30% of all frauds in the Validation Dataset. Separate SQL tables of values are created during the pre-processing clustering/grouping stage, e.g. L1 could contain {France, Germany, Spain, Italy, Netherlands}, meaning true if any one of the countries listed is in the transaction field.

```
IF RECORDTYPE=(SA) AND COUNTRY=(L1) AND CODE=(CODE1) AND PRDCODE=(10) AND
NETW=(ID1) AND TRANSTYPE=(5) AND IND=(1) THEN FRAUD
```

In SQL:

```
SELECT * FROM [TRANSACTIONS] WHERE RECORDTYPE IN
(SELECT A FROM SA) AND COUNTRY IN (SELECT A FROM L1)
AND CODE IN (SELECT A FROM CODE1) AND PRDCODE=10 AND
NETW IN (SELECT A FROM ID1) AND TRANSTYPE=5 AND IND=1;
```

3.4 Analysis

From Table 4, SOAR II has extracted three fewer rules than the original algorithm with a similar accuracy but at the expense of the fraud precision; those rules that have been omitted in SOAR II were generated from genuine transactions that completely overlap the fraud transactions in the Training Dataset – they are therefore impossible to classify into a single class as they are indistinguishable from each other**.** The rules in both cases outperform the neural network in terms of accuracy. It is suggested that the rules are able to outperform as the boundaries located are crisp boundaries so that the (inaccurate) generalisation of the neural model is not captured in the extraction. This tends to create rules that do not cover the decision boundary correctly as evidenced by the reduced precision which tends to reduce the false-positive rate suffered by the neural network. The false-positive rate is further reduced by pruning those rules that exhibit a high rate, reducing precision but contributing to improved accuracy. There is no such equivalent process for the neural network. The Fraud-Analysts found that the SOAR II rules were easier to understand and identified both already known patterns as well as previously unknown patterns of fraud.

4 Conclusions

Accurate rules can be extracted from a trained neural network that can then be inspected and applied in a live FMS environment. This removes the fear of applying neural networks within a mission critical part of a business. The rules are easy to understand and can be produced rapidly, reducing the workload on Fraud-Analysts. SOAR II extraction used on a regular basis has the promise of enabling a more dynamic FMS that can respond to the changing patterns of fraud. It is anticipated that the SOAR II extraction approach can be applied to many other areas of fraud detection where transparency of the decision is important.

Acknowledgments. This work was supported in part by Retail Decisions Europe Ltd.

References

1. Transnational Financial Crime Program, The International Centre for Criminal Law Reform & Criminal Justice Policy, Vancouver, Canada: Drawing conclusions about financial fraud: crime, development, and international co-operative strategies in China and the West (2008)
2. IdentityTheft.com Inc., http://www.identitytheft.com/index.php/article/stolen_credit_terroist_attacks
3. Unisys-Corporation: Research shows economic crisis increases Americans' fears about fraud and ID theft (2009)
4. Button, M., Johnston, L., Frimpong, K.: The fraud review and the policing of fraud: Laying the foundations for a centralized fraud police or counter fraud executive? Policing, 241-250 (2008)
5. Everett, C.: Credit card fraud funds terrorism. Computer Fraud & Security, (5) (2003)
6. BBC News, http://news.bbc.co.uk/1/hi/business/8473167.stm

7. Association of Certified Fraud Examiners: Report to the Nation on Occupational Fraud and Abuse (2008)
8. European Healthcare Fraud & Corruption Network: The Human Cost of Fraud (2010)
9. Aleskerov, E., Freisleben, B., Rao, B.: CARDWATCH: a neural network based database mining system for credit card fraud detection. In: Computational Intelligence for Financial Engineering (CIFEr), pp. 220-226. IEEE Press (1997)
10. Rong-Chang, C., Shu-Ting, L., Xun, L.: Personalized Approach Based on SVM and ANN for detecting credit card fraud. In: International Conference on Neural Networks and Brain, pp. 810-815. IEEE Press (2005)
11. Crosman, P.: Card fraud costs U.S. payment providers $8.6 billion per year. Bank Systems & Technology (2010)
12. UK Payments Administration: Card fraud facts and figures (2009)
13. Cybersource: Seventh annual UK online fraud report: trends, key metrics, informed decisions (2011)
14. Brause, R., Langsdorf, T., Hepp, M.: Neural data mining for credit card fraud detection. In: 11th International Conference on Tools with Artificial Intelligence, pp. 103. IEEE Press (1999)
15. Ghosh, S., Reilly, D.L.: Credit card fraud detection with a neural network. In: International Conference on System Sciences, Hawaii pp. 621-630. IEEE Press (1994)
16. Quinlan, J.R.: Induction of decision trees. Machine Learning, 1, 81-106 (1986)
17. Towell, G.G., Shavlik, J.W.: Extracting refined rules from knowledge-based neural networks. Machine Learning, 13, (1), 71-101 (1993)
18. Setiono, R.: Extracting rules from neural networks by pruning and hidden-unit splitting. Neural Computation, 9, (1), 205-225 (1997)
19. Craven, M., Shavlik, J.W.: Using sampling and queries to extract rules from trained neural networks. In: International Conference on Machine Learning, pp. 37-45. Morgan Kaufmann (1994)
20. Thrun, S.:Advances in Neural Information Processing Systems-Extracting rules from artificial neural networks with distributed representations. MIT Press, Cambridge, MA, USA (1995)
21. Ryman-Tubb, N., d'Avila Garcez, A.S.: SOAR - Sparse Oracle-based Adaptive Rule extraction : knowledge extraction from large-scale datasets to detect credit card fraud. In: World Congress on Computational Intelligence, Barcelona, Spain, pp. 1-9. IEEE Press (2010)
22. Barakat, N., Diederich, J.: Learning-based rule-extraction from support vector machines. In: 12th International Conference on Computer Theory and Applications. IEEE Press (2004)
23. Barakat, N., Diederich, J.: Eclectic rule-extraction from support vector machines. International Journal Computational Intelligence, 2, (1), 59-62 (2005)
24. Engelbrecht, A.P., Viktor, H.L.:Engineering applications of bio-inspired artificial neural networks-Rule improvement through decision boundary detection using sensitivity analysis. Springer Berlin / Heidelberg (1999)
25. Carpenter, G.A., Grossberg, S.: ART2 : Self-organization of stable category recognition codes for analog input patterns. Applied Optics, 26, 4919-4930 (1987)
26. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling TEchnique. Journal of Artificial Intelligence Research, 16, 341-378 (2002)
27. Fletcher, R., Powell, M.J.D.: A rapidly convergent descent method for minimization. Computing Journal, 6, 163-168 (1963)
28. Hirose, Y., Yamashita, K., Hijiya, S.: Backpropagation algorithm which varies the number of hidden units. In: International Joint Conference on Neural Networks, pp. 625. Elsevier (1989)