Using Machine Learning for Anti Money Laundering

Gregor Kržmanc
gregor.krzmanc@ijs.si
Jožef Stefan Institute
Ljubljana, Slovenia

Filip Koprivec
filip.koprivec@ijs.si
Jožef Stefan Institute
IMFM
Ljubljana, Slovenia

Maja Škrjanc
maja.skrjanc@ijs.si
Jožef Stefan Institute
Ljubljana, Slovenia

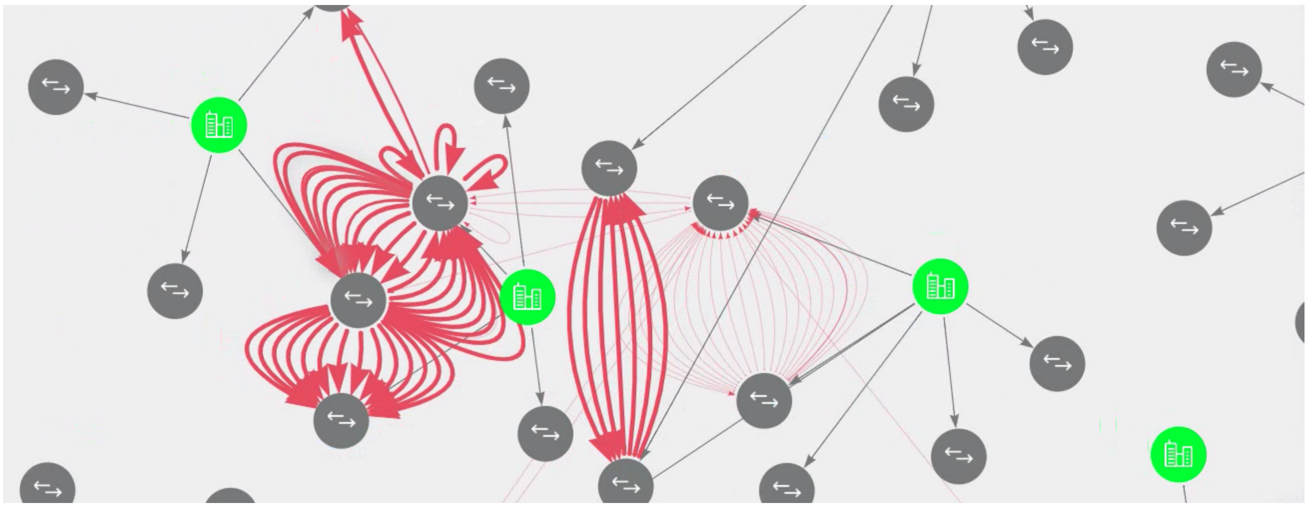

Figure 1: Example transaction network visualization

ABSTRACT

Here we present early results of a network component for anomaly detection in an attributed heterogeneous financial network. Utilizing both externally provided features and generated topological features, we train different models for a simple link prediction task. We then evaluate the models using initial dataset corruption. We show that gradient boosting and multi-layer perceptron generally have the best anomaly detection performance, despite graph neural network models initially showing better results in the link prediction task.

KEYWORDS

Anti Money Laundering (AML), machine learning, networks, link prediction

1 INTRODUCTION

Observing complex real-world graphs, be it a social, financial, biochemical, or physics-related network, is an interesting task. Given a time-evolving network and rich information about the nodes and edges, can we assume that there are some regular dynamics in the network?

Fraud and financial crime are important issues of our time. According to the United Nations Office on Drugs and Crime, an estimated 2-5 % of the world GDP is laundered each year. To keep pace with evolving trends, the European Union has decided to strengthen its anti money laundering and terrorist financing regulatory framework and expects the same from financial institutions and supervisory authorities.

Given a pseudonymized dataset of financial transactions, can we use machine learning to detect interesting, perhaps novel, patterns that should be inspected manually? In this paper, we try to answer this question.

2 RELATED WORK

Both supervised [7, 6, 12] and unsupervised or self-supervised [2, 14] learning approaches have been proposed to deal with the task of detecting money laundering. Due to the lack of labelled data and the closed nature of financial data and, therefore, the lack of standardised datasets, approach evaluation can be difficult. Despite that, cryptocurrency datasets such as [13] have been published, explored, and labelled to some extent.

Usually, synthetic oversampling or other strategies of sampling need to be employed in cases where labelled entities are used for evaluation [12, 13].

3 DATA

In this study, we use a snapshot of the transaction data processed through the international payment system *Target2-Slovenija* [11]. The dataset spans from November 2007 to December 2017, containing around 8 million financial transactions. No live data was used when performing this research - only archived datasets were used.

For some nodes, the data about the sending or receiving party is additionally linked to data from the Slovenian Business Register (ePRS) [1] and the Slovenian Transaction Account Registry (eRTR) [3] in order to provide additional context about each transaction.

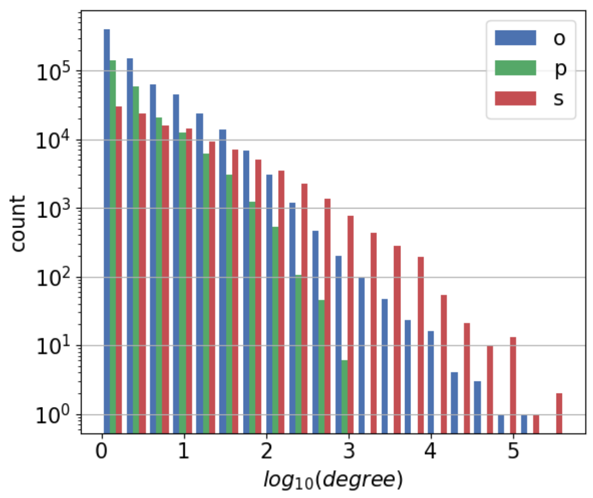

Figure 2: Degree distribution by node type.

Due to the sensitive nature of the data, all personal and confidential data about individuals and legal entities provided to JSI is pseudonymized.

4 DATA REPRESENTATION AS A HETEROGENEOUS GRAPH

There are large differences in the availability of data across different entities performing the transactions. In order to fully utilize all available features, we model the network as a heterogeneous temporal graph. Here, we treat the snapshot of the transaction graph from t_0 to t_1 $G = G(t_0, t_1)$ as a heterogeneous graph consisting of 3 discrete node types representing each entity's legal status. The types of accounts are those belonging to companies (node type s), natural persons (node type p), and all other accounts (node type o). Each transaction is represented as a directed edge from its source account to its destination account.

4.1 Network statistics

Due to different legislative bases for different types of entities, inherent differences regarding data availability are expected. Naturally, it is also expected that different categories usually act differently in a network - for example, companies usually transact more than individuals. While the degree distribution (Figure 2) closely resembles the power law, significant differences in distributions between different node types can be observed, which can be attributed to varying amounts of data available for our specific data source across account profiles.

It can be seen from Figure 2 that companies (node type s) perform most of the transactions.

4.2 Feature generation

Categorical features are one-hot encoded. Rare categories with $< 2\%$ incidence are marked as *other.* Additionally, node features encoding the role of a node in the network (Table 1) are generated. The node-level features for each node are computed on the whole network as well as for the subgraph induced by the node's own type.

feature	level
degree $\deg(A) = \lvert N(A)\rvert$	node-level
PageRank [9] $PR(A) = \frac{1-d}{N} + d\sum_{J\in N_{in}(A)} \frac{PR(J)}{\lvert N_{out}(J)\rvert}; d = 0.85$	node-level
Jaccard coefficient $J(A, B) = \frac{\lvert N(A)\cap N(B)\rvert}{\lvert N(A)\cup N(B)\rvert}$	edge-level
Adamic-Adar Index $A(x, y) = \sum_{u\in N(x)\cap N(y)} \frac{1}{\log \lvert N(u)\rvert}$	edge-level

Table 1: The structural features used for the link prediction task. $N(\cdot)$ represents the set of neighbours of the given node. N_{in} and N_{out} represent the sets of the nodes from which there is an edge to the given node (*in*), or to which there is an edge from the given node (*out*). $|\cdot|$ represents cardinality of the given set.

5 ANOMALY DETECTION PROBLEM DEFINITION

We corrupt the original graph by rewiring the total of $p = 1\%$ randomly picked edges of each edge type.

Let $f : V\times V \rightarrow [0, 1]$ be a binary link prediction classifier that is trained to predict the probability that a directed edge between the two given nodes exists.

We define the anomaly score of edge $(i, j) \in E$ as

$$\phi(i, j) = 1 - f(i, j) \qquad (1)$$

The intuition behind equation 1 is that links that are typical to the model would have a smaller anomaly score than links for which the model predicts they would not exist (and are, thus, anomalous).

6 RESULTS

We train several models for the downstream task of link prediction and then use the predictions for anomaly detection.

6.1 Experiment details

The traditional (non-GNN) machine learning approaches are trained to predict whether the given edge exists or not. For each edge, the feature vector fed into the model is constructed by concatenating source node features, destination node features, and edge features. For traditional models, a model for each edge type is constructed separately, while the graph neural network-based models are the same across all edge types.

The GNN (graph neural network) models are constructed of 2 layers of GraphSAGE aggregations [8, 5] using parametric ReLU activations and embedding dimensions of 128 for the first and 64 for the second layer. As messages are passed in the direction of edges, we construct another model to facilitate information diffusion both ways. We do this by adding edges of opposite directionality than existing edges and marking them as a separate edge types. We still, however, only train for the downstream link prediction objective only on the existing (non-transposed) edges. We mark this approach as GNN$^+$.

The traditional ML models used are gradient boosting (GradBoost), decision tree (DecTree), multi-layer perceptron (MLP) and logistic regression (LogReg). The hidden layer sizes of the MLP are 20 and 10, using ReLU activation in all layers except the last one, where softmax activation is used. Different combinations of reasonable hidden layer sizes were tested (32+16, 64+32, 256+128, 128+128, 20+10) and the best one was selected. The training of MLP models was performed with a batch size of 200.

6.2 Link prediction

Traditional ML models for link prediction map concatenated source and destination node features and edge features to the probability that a link between such nodes exists. The models are implemented using scikit-learn [10] and are trained and evaluated using 5-fold cross-validation.

As a preprocessing step, each feature is scaled individually using a standard scaler such that it has a mean of 0 and a standard deviation of 1 across the training set.

When training and evaluating each model, an approximately equal number of positive and negative links is given to the classifier. The provided edge features such as transaction amount are sampled randomly for negative edges.

Additionally, we train a 2-layer graph neural network (GNN) for link prediction. The GNN model is trained jointly for all edge types using weighted binary cross-entropy loss. The model has ReLU activations in all layers except the last one, where it has softmax activation. The hidden layer sizes are 64 and 32. The graph neural network is implemented using PyTorch Geometric [4].

We use a random link split for link prediction and not a temporal one, as our end goal is not to predict future links, but rather to learn what kinds of transactions are typical in the given network.

Table 2 shows the aggregated link prediction results. Bold results highlight the best performance across observed methods. The GNN does slightly improve link prediction performance in some cases. See Appendix A for more detailed non-GNN method results. The data here is computed across multiple year-long time windows.

edge	non-GNN	no str. f.	GNN	GNN$^+$
ss	0.92 ± 0.01	0.89 ± 0.01	0.92 ± 0.02	**0.94 ± 0.01**
oo	**0.80 ± 0.02**	0.57 ± 0.01	0.79 ± 0.02	0.53 ± 0.04
so	0.83 ± 0.01	0.75 ± 0.01	**0.88 ± 0.02**	0.74 ± 0.04
os	0.76 ± 0.01	0.64 ± 0.01	0.81 ± 0.01	**0.83 ± 0.02**
sp	**0.85 ± 0.02**	0.69 ± 0.03	0.78 ± 0.05	0.73 ± 0.02
ps	0.74 ± 0.02	0.67 ± 0.01	**0.87 ± 0.02**	0.75 ± 0.04
po	0.78 ± 0.02	0.66 ± 0.01	**0.84 ± 0.04**	0.54 ± 0.08
op	**0.89 ± 0.01**	0.53 ± 0.01	0.78 ± 0.05	0.50 ± 0.05
all	0.84 ± 0.01	0.72 ± 0.01	0.86 ± 0.02	**0.89 ± 0.01**

Table 2: Link prediction performance comparison measured in area under the receiver operating characteristic curve (AUC) (mean ± standard deviation). Edge types are marked with two letters, representing the source and destination node type in this order. Best non-GNN score, as well as best non-GNN score without using any structural features, are reported next to the GNN results.

6.3 Anomaly detection

For comparison between different methods, the 2% of edges with the highest anomaly scores are flagged as positive. Precision and recall are calculated by using the corrupted 1% of edges as true positives.

To summarize precision and recall in a single metric, F_1 score (2) is calculated and reported.

edge	non-GNN	no str. f.	GNN	GNN$^+$
ss	**0.19 ± 0.02**	0.16 ± 0.02	0.01 ± 0.00	0.01 ± 0.00
oo	**0.11 ± 0.02**	0.02 ± 0.01	0.05 ± 0.02	0.03 ± 0.02
so	**0.11 ± 0.02**	0.06 ± 0.01	0.01 ± 0.01	0.01 ± 0.01
os	**0.14 ± 0.02**	0.06 ± 0.01	0.01 ± 0.00	0.01 ± 0.01
sp	**0.08 ± 0.04**	0.02 ± 0.02	0.02 ± 0.01	0.02 ± 0.02
ps	**0.05 ± 0.02**	**0.05 ± 0.02**	0.01 ± 0.01	0.01 ± 0.01
po	**0.07 ± 0.04**	0.07 ± 0.05	0.02 ± 0.02	0.01 ± 0.02
op	**0.18 ± 0.04**	0.02 ± 0.01	0.02 ± 0.01	0.03 ± 0.02

Table 3: Anomaly detection performance comparison in F_1 score (mean ± standard deviation). Best non-GNN score, as well as best non-GNN score without using any structural features, are reported next to the GNN results. Bold results highlight the best performance across observed methods.

$$F_1^{-1} = \frac{\text{precision}^{-1} + \text{recall}^{-1}}{2} \tag{2}$$

A naive classifier that assigns the same positive score (recall 1) to each edge has F_1 score of ≈ 0.02. However, the underrepresented edge types typically have higher variance in F_1 score and performance insignificantly different from the naive baseline, as seen from Table 3. The same goes for the GNN-based models. See Appendix A for more detailed non-GNN model results.

7 DISCUSSION AND FUTURE WORK

We have constructed and evaluated a self-supervised approach to anomaly detection in financial networks. Due to the lack of labelled data, this is in most cases the most straightforward approach to tackle the problem with machine learning. There are significant differences in performance across different edge types. Using this approach yields almost comparable results with both raw features and structural features when evaluated on company-to-company transactions only. This may be explained by companies in our dataset having the most insightful features of all node types such as the broader sector and also more precise company industry type classification.

This paper has mainly focused on the use of unsupervised learning for anomaly detection. In the future, we plan to extend our work to supervised and semi-supervised learning approaches to try to utilize the few labelled data points. The following machine learning strategies (or a combination of them) could be tested:

- **Active learning.** Human-assisted active learning approach is a natural way to incorporate domain knowledge into the decision-making process.
- **Synthetic oversampling.** Due to a small number of the positive examples, we could sample new examples that are similar to them and assign them positive labels.
- **Model pretraining and few-shot learning.** Update model parameters with a self-supervised pretraining strategy first, and then optimize it further on the few labeled data points.

ACKNOWLEDGMENTS

The research leading to the results presented in this paper has received funding from the European Union's funded Project INFINITECH under grant agreement no. 856632.

The financial transaction data used in the presented research was collected and pseudonymized by the Bank of Slovenia.

The Bank of Slovenia collaborates with JSI and the Infinitech project in order to research possible efficient and compliant banking system supervision techniques.

We thank Klaudija Jurkošek Seitl for her input on the style of this paper.

REFERENCES

[1] 2022. AJPES - ePRS. (September 2022). https://www.ajpes.si/prs/.

[2] Claudio Alexandre and João Balsa. 2016. Client Profiling for an Anti-Money Laundering System. https://arxiv.org/abs/1510.00878.

[3] 2022. eRTR. (September 2022). https://www.ajpes.si/eRTR/JavniDel/Iskanje.aspx.

[4] Matthias Fey and Jan E. Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

[5] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *arXiv:1706.02216 [cs, stat]*. arXiv: 1706.02216. Retrieved 06/18/2021 from http://arxiv.org/abs/1706.02216.

[6] Mikel Joaristi, Edoardo Serra, and Francesca Spezzano. 2019. Detecting suspicious entities in Offshore Leaks networks. *Social Network Analysis and Mining*, 9, 1, 1–15. Publisher: Springer Vienna. ISSN: 1327801906. DOI: 10.1007/s13278-019-0607-5. https://doi.org/10.1007/s13278-019-0607-5.

[7] Martin Jullum, Anders Løland, Ragnar Bang Huseby, Geir Ånonsen, and Johannes Lorentzen. 2020. Detecting money laundering transactions with machine learning. *Journal of Money Laundering Control*, 23, 1. DOI: 10.1108/JMLC-07-2019-0055. https://www.emerald.com/insight/1368-5201.htm.

[8] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907 [cs, stat]*, (February 2017). arXiv: 1609.02907. Retrieved 06/18/2021 from http://arxiv.org/abs/1609.02907.

[9] Larry Page, Sergey Brin, R. Motwani, and T. Winograd. 1998. The PageRank Citation Ranking: Bringing Order to the Web. (1998).

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[11] 2022. TARGET2 in TARGET2-Slovenija. si. (September 2022). Retrieved 09/21/2022 from https://www.bsi.si/placila-in-infrastruktura/placilni-sistemi/target2-in-target2-slovenija.

[12] Dominik Wagner. 2019. Latent representations of transaction network graphs in continuous vector spaces as features for money laundering detection. *Gesellschaft für Informatik*, 1–1.

[13] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel I Karl Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. 2019. Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics. Technical report.

[14] Jiaxuan You, Tianyu Du, Fan-yun Sun, and Jure Leskovec. 2021. Graph Learning in Financial Networks. (September 2021). https://snap.stanford.edu/graphlearning-workshop/slides/stanford_graph_learning_Finance.pdf.

A DETAILED RESULTS

A.1 Link prediction (AUC)

edge	DecTree	GradBoost	LogReg	MLP
ss	0.87 ± 0.01	0.90 ± 0.01	0.79 ± 0.01	**0.92 ± 0.01**
oo	0.80 ± 0.01	**0.80 ± 0.02**	0.51 ± 0.01	0.74 ± 0.01
so	0.82 ± 0.01	**0.83 ± 0.01**	0.65 ± 0.01	0.82 ± 0.01
os	0.75 ± 0.01	**0.76 ± 0.01**	0.58 ± 0.02	0.73 ± 0.01
sp	0.81 ± 0.02	**0.85 ± 0.02**	0.55 ± 0.02	0.83 ± 0.02
ps	0.70 ± 0.02	**0.74 ± 0.02**	0.54 ± 0.02	0.69 ± 0.01
po	0.72 ± 0.02	**0.78 ± 0.02**	0.54 ± 0.02	0.67 ± 0.01
op	0.85 ± 0.01	**0.89 ± 0.01**	0.51 ± 0.03	0.87 ± 0.01
all	0.81 ± 0.01	**0.84 ± 0.01**	0.66 ± 0.02	0.82 ± 0.01

A.2 Anomaly detection (F_1 score)

edge	DecTree	GradBoost	LogReg	MLP
ss	0.12 ± 0.01	0.13 ± 0.02	0.04 ± 0.01	**0.19 ± 0.02**
oo	0.07 ± 0.01	**0.11 ± 0.02**	0.01 ± 0.01	0.10 ± 0.02
so	0.08 ± 0.01	0.10 ± 0.02	0.04 ± 0.01	**0.11 ± 0.02**
os	0.06 ± 0.01	0.12 ± 0.02	0.04 ± 0.01	**0.14 ± 0.02**
sp	0.06 ± 0.01	0.07 ± 0.04	0.02 ± 0.02	**0.08 ± 0.04**
ps	0.04 ± 0.01	**0.05 ± 0.02**	0.01 ± 0.01	**0.05 ± 0.02**
po	0.04 ± 0.01	**0.07 ± 0.04**	0.02 ± 0.03	0.04 ± 0.03
op	0.09 ± 0.01	0.14 ± 0.04	0.01 ± 0.01	**0.18 ± 0.04**