Graph Representation Learning for Fraud Prediction: A Nearest Neighbour Approach

Rafaël Van Belle, Sandra Mitrović, Jochen De Weerdt
Research Centre for Information Systems Engineering
KU Leuven
Leuven, Belgium
rafael.vanbelle@kuleuven.be

Abstract

The rise of graph representation learning has introduced a vast collection of techniques for the automated featurization of network data. With new algorithms empirically evaluated on a set of publicly available data sets, this study takes a different approach: the social graph resulting from a large, real-life credit card transaction data set is scrutinized. The social mechanisms at work in credit card fraud are exploited by means of node embeddings from the transaction network. Node embedding algorithms automatically map each node in a graph into a low-dimensional, latent vector representation. The credit card fraud application introduces a number of challenges for representation learning, such as: volume of transactions, on-line requirements and imbalanced data. As a result, we put forward a fast and efficient inductive extension based on a nearest neighbour search which overcomes these challenges. The extension is empirically evaluated against state-of-the art inductive representation learning algorithms.

1 Introduction

Recent years have seen a tremendous interest in representation learning applied to network data [3; 10; 19; 7]. Without proper preprocessing, raw network data is laborious to leverage in downstream machine learning or statistical analysis tasks. The usual approach to incorporating graph data has been to create a set of features based on structural properties of the network, such as node degree, betweenness centrality, etc. Despite promising results obtained with these handcrafted feature sets, they require substantial domain-expertise and tedious feature engineering while failing to comprehensively capture the structural graph properties [2]. Graph representation learning, on the contrary, automatically represents the entire graph or its elements in a low dimensional, latent vector representation that can be optimized to holistically capture and conserve relational and structural aspects of the network.

The performance of said algorithms is often displayed on a fixed set of publicly available datasets [13]. As a result, comparison of techniques becomes more or less feasible. Nevertheless, performance of these algorithms is only rarely measured on recent, voluminous, real world datasets.

In this work we assess the feasibility of network representation learning for fraud prediction. Previous research has already established the added value of relational information in fraud prediction [17]. Though, this particular application domain presents a number of new challenges for network representation learning. For instance, the extreme class imbalance might inhibit an unsupervised preprocessing step like node embeddings. In addition, fraud detection in production settings has stringent time requirements that might exclude time demanding algorithms from consideration. Finally, the volume of transactions will translate into a huge network of transactions, which will make it considerably more difficult to include many historical observations into the training dataset.

To this end an inductive extension based on a nearest neighbour search is designed with the business application and associated constraints in mind. We extensively evaluate the combination of a transductive node embedding algorithm and inductive extension against a well-known inductive node embedding framework.

The main contributions of this work are twofold. First, we gauge the capacity of representation learning to operate in real-world conditions, in particular credit card fraud prediction. Second, an efficient and fast inductive extension based on nearest neighbour search is described and its performance is juxtaposed with the results from an existing inductive representation learning framework. Third, extensive experimentation highlights the added-value of relational information for fraud prediction, while the comparison of techniques shows that the inductive extension outperforms state-of-the-art inductive node embedding algorithms.

2 Related work

This work focuses on node embedding algorithms, which can be categorized into (1) matrix factorization [4; 14], (2) random walk [15; 8] , and (3) deep learning-based methods, such as graph autoencoders [18; 5] and convolutional embedding algorithms [9; 12].

Most of the aforementioned techniques are designed to work on static graphs. Hence, no changes to either edge or node sets are allowed without expensive retraining. The transductive nature impedes the application in fast-evolving settings, such as credit card fraud. The exception to this are the convolutional embedding algorithms, like [9; 12] that have the ability to generalize to unseen nodes. GraphSAGE [9], for instance, iteratively aggregates attribute information from neighbouring nodes to generate embeddings. Structural Identity Preserved Inductive Network Embedding (SPINE) is similar to GraphSAGE in that the technique refrains from learning node embeddings independently. Instead, an embedding generator is created based on the combination of Rooted Pagerank for structural feature generation, a multilayer perceptron and finally a biased skipgram model with negative sampling. In the same vein, DeepGL [16] creates a set of base features that capture the structural properties of the network and subsequently processes said features through a number of successive layers of relational feature operators. Inductive matrix factorization approaches have been proposed as well. Fast Inductive Graph Representation Learning (FI-GRL) [11] starts from the random walk normalized laplacian to transform it in two separate stages: first a matrix sketch is obtained by means of the Johnson Lindenstrauss projection and secondly, a low rank approximation is applied. Finally, Role2Vec [1] generalizes random walk-based approaches to inductive tasks.

Current solutions for the application of this study, fraud prediction, are based on customer profiles. For each (group of) cardholder(s) a profile is created based on historical spending patterns. A neglected source of information are the relations between individuals in the network of transactions, mainly because of stringent time constraints and efficiency requirements in production settings. Nevertheless, the value of network information to detect fraudsters has already been noticed [17].

3 Inductive nearest neighbour-based representation learning

This paper proposes a fast and efficient inductive embedding generator based on Nearest Neighbours (NN) search, specifically designed for the requirements of fraud prediction. Given an undirected graph $G = (V, E)$ with node set V and edge set E, the required input for the inductive NN extension are a set of pre-trained node embeddings for nodes $v \in V$.

To this end, Deepwalk introduced in [15] is applied. Instead of the skipgram architecture, we opt for Continuous-Bag-of-Words (CBOW). CBOW takes neighborhood information h as input to predict a single node i as output, which resembles closely the mechanism involved in the inductive extension outlined below.[1] In addition, the truncated random walks in DeepWalk are non-biased (in contrast to walks in Node2Vec [8]), yielding substantial efficiencies in terms of computing resources. Details regarding the transductive step are discussed in Supplements.

The intermediate result is a look-up function that maps each node to its associated embedding ($V \mapsto \mathbb{R}^d$). As a result of the transductive nature of [15] the domain of the embedding function is restricted to the initial node set V. Over time, however, a dynamic network will evolve to

[1] not only is CBOW a better theoretical fit, it also outperformed skipgram empirically on our data

$G' = (V + V', E + E')$ with new nodes V' and edges E'. To avoid lengthy retraining, an inductive extension based on a nearest neighbour search is applied. The mechanism is outlined below, while Supplements contains the algorithm in pseudocode.

An unseen node $v' \in V'$ is either isolated or has at least one link with a node in the original graph $G = (V, E)$. The existence of nodes in the egonet of v' are a crucial requirement for the Nearest Neighbour extension. In case of isolated nodes ($\texttt{egonet} = \emptyset$), an average embedding is assigned to the isolated node. In case of at least one neighbour, the NN extension will search for the Nearest Neighbour in terms of euclidean distance of the normalized attribute vectors. This approach can easily be expanded with additional constraints $(c_1, c_2 \cdots c_n)$ on the egonet. For instance, one could limit the search for a nearest neighbour node to the set of nodes that fall within certain attribute ranges. Optional sorting of the considered egonet helps to decide which node is chosen as nearest neighbour in case of equidistant nodes. Finally, the pre-trained embedding of the nearest neighbour is retrieved as the new embedding for node v'.

Despite the deceiving simplicity, the approach will demonstrate excellent results. In addition, the main objective is obtaining a fast and efficient approach that can scale beyond the set of benchmark datasets, towards real-life production settings. The next section will illustrate the empirical results achieved on a credit card fraud dataset.

4 Empirical evaluation

Experiments The dataset contains 3,240,339 credit card transactions, with a fraud rate of 0.32% . A rolling window approach is used to create consecutive splits of training/validation and test sets. A delimited testing period of five days is covered with a number of replications that depends on the testing set size. The transactions are arranged in a tripartite network, with the involved parties and the transaction represented as nodes.

In addition, the impact of infusing attribute information from the nodes into the network is assessed by means of two artificial nodes, labeled 'fraud' and 'non-fraud'. All historical transactions are connected to these artificial nodes based on their fraud labels. Thanks to these artificial nodes, the random walking algorithm has more freedom of movement.

An overview of the hyperparameter ranges considered during hyperparameter tuning is given in Supplements. The remaining hyperparameters are fixed to the defaults provided in their respective implementations. XGBoost is used for label classification [6]. The performance from the optimal parameter set obtained through grid search hyperparameter optimization is compared against the inductive representation learning framework GraphSAGE [9] and maxpooling. The maxpooling method combines the existing vectors of parties involved in the transaction via maxpooling and can also be considered an inductive extension.

Results Table 1 contains the hyperparameter combinations that produced the highest Area Under the Receiver Operating Curve (AUC) scores. For graphical comparison see Figure 1. Adding artificial nodes to the network seems to change the preferred hyperparameter values. For instance, a network with artificial nodes performs better with longer training periods and hence larger training set sizes, while the opposite holds in case of no artificial nodes. Overall, smaller test set sizes seem preferred, with a more outspoken preference for networks with artificial nodes. The variability in AUC scores increases considerably when adding artificial nodes. To a certain extent this is true, but part of it can be explained by the larger differences in performance between hyperparameter combinations. Hence, when adding artificial nodes to the network, giving consideration to the right hyperparameter combination is crucial for predictive performance. Finally, all oversampling algorithms seem adequate, undersampling performs markedly worse.

When compared to graphSAGE and maxpooling the NN-based extension yields on average higher AUC scores (Figure 2a) and more efficient results (300 ms/transaction for embedding generation, 2x faster than GraphSAGE). The addition of artificial nodes seems to substantially increase the classification performance based on embeddings obtained through the NN-extension. This result is confirmed in Figure 2b, which presents the ROC curves for five independent replications. Most striking is the ability of our proposed extension to deliver substantially higher True Positive Rates for low values of False Positive Rates, which is crucial in production settings.

Table 1: Parameter settings with highest average AUC score. Lift is reported at 10%. Each scenario is replicated 30 times (which covers the five days testing period). Notation: A: training set size, B: test set size, C: dimension, D: artificial nodes, E: sampling.

A (days)	B (hours)	C	D	E	AUC avg ± std	Lift avg ± std
8	4	32	True	SMOTE	0.862 ± 0.065	5.880 ± 2.118
8	4	32	True	OS	0.860 ± 0.070	6.026 ± 2.109
8	4	32	True	ADASYN	0.859 ± 0.066	5.974 ± 2.118
10	4	32	True	ADASYN	0.858 ± 0.069	6.215 ± 1.837
10	4	32	True	SMOTE	0.857 ± 0.073	6.141 ± 1.783
6	4	32	True	ADASYN	0.853 ± 0.078	5.555 ± 1.940

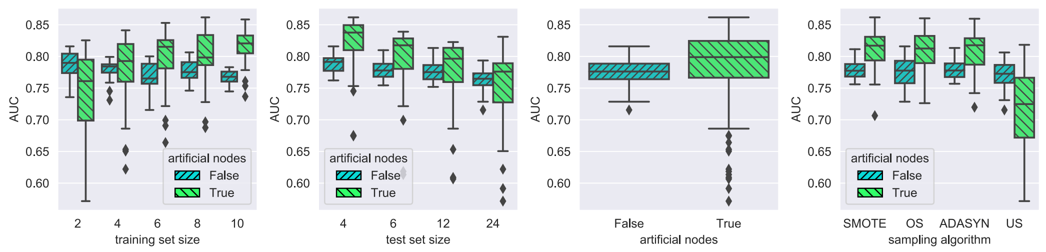
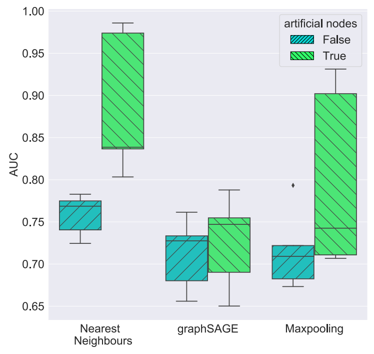


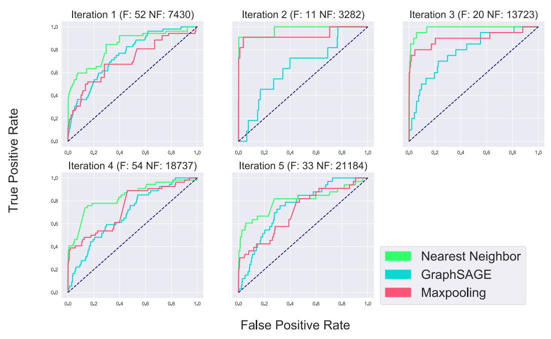
Figure 1: AUC Performance of NN inductive extension. Each subgraph presents the results for different experiment parameter values.

5 Conclusion

This work addresses the capability of graph representation learning to operate in a credit card fraud setting. For this an efficient and fast inductive extension based on Nearest Neighbours (NN) was presented. The NN-based inductive extension surpasses state-of-the-art inductive algorithms, that require considerably more resources and time (2x slower). In addition, comprehensive empirical evaluation of hyperparameters provided valuable insights regarding the optimal parameter combinations. Finally, our research has underlined the importance of relational information for fraud prediction.



(a) Comparison of inductive algorithms (five replications)



(b) Receiver Operating Curves (ROC) of five independent replications (with artificial nodes). F: fraud cases, NF: non-fraud cases.

Figure 2

References

[1] AHMED, N. K., ROSSI, R. A., ZHOU, R., LEE, J. B., KONG, X., WILLKE, T. L., AND ELDARDIRY, H. Inductive representation learning in large attributed graphs. *arXiv preprint arXiv:1710.09471* (2017).

[2] BENGIO, Y., COURVILLE, A., AND VINCENT, P. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence 35*, 8 (2013), 1798–1828.

[3] CAI, H., ZHENG, V. W., AND CHANG, K. C.-C. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering 30*, 9 (2018), 1616–1637.

[4] CAO, S., LU, W., AND XU, Q. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM international on conference on information and knowledge management* (2015), ACM, pp. 891–900.

[5] CAO, S., LU, W., AND XU, Q. Deep neural networks for learning graph representations. In *Thirtieth AAAI Conference on Artificial Intelligence* (2016).

[6] CHEN, T., AND GUESTRIN, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (2016), ACM, pp. 785–794.

[7] GOYAL, P., AND FERRARA, E. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems 151* (2018), 78–94.

[8] GROVER, A., AND LESKOVEC, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (2016), ACM, pp. 855–864.

[9] HAMILTON, W., YING, Z., AND LESKOVEC, J. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems* (2017), pp. 1024–1034.

[10] HAMILTON, W. L., YING, R., AND LESKOVEC, J. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584* (2017).

[11] JIANG, F., ZHENG, L., XU, J., AND YU, P. Fi-grl: Fast inductive graph representation learning via projection-cost preservation. In *2018 IEEE International Conference on Data Mining (ICDM)* (2018), IEEE, pp. 1067–1072.

[12] KIPF, T. N., AND WELLING, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[13] LESKOVEC, J., AND KREVL, A. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, 2014.

[14] OU, M., CUI, P., PEI, J., ZHANG, Z., AND ZHU, W. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (2016), ACM, pp. 1105–1114.

[15] PEROZZI, B., AL-RFOU, R., AND SKIENA, S. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (2014), ACM, pp. 701–710.

[16] ROSSI, R. A., ZHOU, R., AND AHMED, N. K. Deep inductive network representation learning. In *Companion of the The Web Conference 2018 on The Web Conference 2018* (2018), International World Wide Web Conferences Steering Committee, pp. 953–960.

[17] VAN VLASSELAER, V., BRAVO, C., CAELEN, O., ELIASSI-RAD, T., AKOGLU, L., SNOECK, M., AND BAESENS, B. Apate: A novel approach for automated credit card transaction fraud detection using network-based extensions. *Decision Support Systems 75* (2015), 38–48.

[18] WANG, D., CUI, P., AND ZHU, W. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (2016), ACM, pp. 1225–1234.

[19] ZHANG, D., YIN, J., ZHU, X., AND ZHANG, C. Network representation learning: A survey. *IEEE transactions on Big Data* (2018).

Supplements

Supplement 1: Transductive Formulas

Given a set of truncated random walks, the next step is to optimize each individual node embedding $e(i)$ according to the learning objective in Equation 1. The objective is to maximize the posterior probability of observing node i in a random walk given the set of previous nodes in the walk h. C_i is the collection of nodes in the context window of node i. Optimizing for the entire training dataset involves maximizing the log-likelihood (see Equation 2).

$$P(i|h) = \frac{\exp \sum_{j \in C_i} (e(i) \cdot e(j))}{\sum_{v \in V} \exp \sum_{k \in C_i} (e(v) \cdot e(k))} \tag{1}$$

$$\max_e \sum_{v \in V} \sum_{j \in C_i} e(i) \cdot e(j) - \log \left(\sum_{v \in V} \exp \sum_{k \in C_i} \Big(e(v) \cdot e(k) \Big) \right) \tag{2}$$

Supplement 2: NN Extension Pseudocode

Algorithm 1: Nearest Neighbour Inductive Extension

```
Input      : graph $G' = (V + V', E + E')$, new node $n$
Initialize : $d_{min}, nn$

for neighbour ∈ egonet($n, G'$) do
    $d$ = distance($n, neighbour$)
    if $d < d_{min}$ then
        $d_{min} := d$                              // update shortest distance
        $nn := n$                                   // update nearest neighbour

    end
end
$e(n) := e(nn)$

Function egonet($n, G(V, E)$):
    Z := $\{v | v \in V \wedge (v, n) \in E\}$
    Z := $\{v | v \in Z \wedge c_1 \wedge c_2 \wedge \ldots c_n\}$
    Z := sort(Z)
return Z

Output     : embedding $e(n)$ for new node $n$
```

Supplement 3: Hyperparameter Value Ranges

Parameters and corresponding value ranges. Each parameter combination gives rise to a separate experiment. The parameter space is searched exhaustively by means of grid search.

Parameter	Values	Unit
Training set size	2, 4, 6, 8, 10	days
Test set size	4, 6, 12, 24	hours
Embedding dimension	32, 64	
Artificial nodes	True, False	
Sampling algorithm	undersampling (US), oversampling (OS), SMOTE, ADASYN	