CS313.023 - Final Project Report: Money Laundering Detection using Graph Neural Network

Nguyen Tran Viet Anh 21520006
Le Thanh Minh 21520063
Vo Tran Thu Ngan 21520069
Nguyen Truong Thinh 21520110

***Abstract*—Money laundering poses significant challenges to financial institutions and regulatory bodies due to its complex and covert nature. This project evaluates a novel approach for detecting money laundering transactions using advanced Graph Neural Networks (GNNs) tailored for directed multi-graphs incorporating multi-graph port numbering, ego IDs, and reverse message passing to enhance the detection capabilities of standard message-passing GNNs. We apply this enhanced GNN framework to the IBM Transactions for Anti Money Laundering dataset and compares its performance against traditional machine learning models. The method demonstrates substantial improvements in detecting suspicious transactions, evidenced by a significant increase in the minority-class F1 score.**

***Index Terms*—Money Laundering Detection, Graph Neural Network**

Our code is available here: synthetic pattern detection AML results (main results), XGboost and LightGBM.

I. Introduction

Money laundering poses a significant challenge to financial institutions and regulatory bodies worldwide. The complexity and sophistication of laundering schemes necessitate advanced techniques for effective detection and prevention. Traditional methods, often rule-based, struggle to adapt to the evolving nature of these illicit activities. As a result, there is a growing interest in leveraging machine learning and, more specifically, Graph Neural Networks (GNNs) to address this problem [1], [2] [3]. [4] Graph Neural Networks have emerged as a powerful tool for analyzing graph-structured data, which is particularly relevant for financial transactions that can naturally be represented as graphs. In a recent study by *Egressy et al* [5], the authors propose a novel approach to enhance the expressiveness of GNNs when dealing with directed multigraphs. This is crucial for financial transaction networks where edges can have multiple types (e.g., different transaction types) and directions (e.g., money flow direction), adding layers of complexity to the graph structure.

In parallel, recent research [6] addresses the scarcity of publicly available datasets for training and evaluating machine learning models in this domain. This paper introduced a method for generating realistic synthetic financial transaction data that maintains the statistical properties of real-world data while incorporating patterns indicative of money laundering. This synthetic data provides a valuable resource for developing and testing GNN-based models without the constraints associated with sensitive financial information.

Combining these two advancements, this project aims to develop a GNN-based model for money laundering detection. By leveraging the enhanced capabilities of GNNs for directed multigraphs and utilizing realistic synthetic transaction data, we seek to create a model that not only performs well in identifying suspicious transactions but also generalizes effectively to real-world scenarios.

II. Methods

To tackle this problem, we utilize the method from this paper [5], which presents adaptations to standard Message Passing Neural Networks (MPNNs) to enhance their capabilities in detecting subgraph patterns in directed multigraphs. The methods include reverse message passing, directed multigraph port numbering, and ego IDs.

A. Reverse Message Passing

The first adaptation involves reverse message passing, which addresses the limitation of standard MPNNs that only aggregate messages from incoming neighbors. By incorporating messages from outgoing neighbors, nodes can count their outgoing edges, enhancing the network's ability to detect certain patterns.

$$a_{in}^{(t)}(v) = \text{AGG}_{in}\left(\{h^{(t-1)}(u) \mid u \in N_{in}(v)\}\right), \quad (1)$$

$$a_{out}^{(t)}(v) = \text{AGG}_{out}\left(\{h^{(t-1)}(u) \mid u \in N_{out}(v)\}\right), \quad (2)$$

$$h^{(t)}(v) = \text{UPDATE}\left(h^{(t-1)}(v), a_{in}^{(t)}(v), a_{out}^{(t)}(v)\right), \quad (3)$$

Here, a_{in} aggregates messages from incoming neighbors, and a_{out} aggregates messages from outgoing neighbors, allowing the network to differentiate between in-degrees and out-degrees.

B. Directed Multigraph Port Numbering

The second adaptation is the utilization of port numbering to direct multigraphs to distinguish between multiple edges from the same neighbor. This involves assigning unique local IDs to each neighbor at a node, enabling the network to identify messages from the same neighbor in consecutive rounds.

$$a_{in}^{(t)}(v) = \text{AGG}_{in}\left(\{(h^{(t-1)}(u), \text{port}_{in}) \mid u \in N_{in}(v)\}\right), \quad (4)$$

$$a_{out}^{(t)}(v) = \text{AGG}_{out}\left(\{(h^{(t-1)}(u), \text{port}_{out}) \mid u \in N_{out}(v)\}\right), \tag{5}$$

Port numbers are precomputed, ensuring that training and inference times remain efficient.

C. Ego IDs

Ego IDs mark a central node with a distinct binary feature, allowing it to recognize cycles and other patterns that include the marked node. When combined with reverse message passing and port numbering, ego IDs significantly enhance the network's ability to detect complex subgraph patterns.

D. Principal Neighbourhood Aggregation (PNA) for Graph Nets

Graph Neural Networks (GNNs) have limitations when dealing with continuous features. Theoretical analysis shows the need for multiple aggregators, especially for continuous feature spaces. However, most work in the literature uses only a single aggregation method, with mean, sum and max aggregators being the most used in the state-of-the-art models. To address that issue, G. Corso et al. [7] propose Principal Neighbourhood Aggregation (PNA) which combines muliple aggregators with degree-scalers to improve expressive power. Degree-scalers are functions that adjust the aggregated information from neighboring nodes based on the degree (number of neighbors) of the central node. They can amplify or attenuate signals, allowing the network to better capture the influence of different neighborhoods. One common degree-scaler is the logarithmic function. It scales the aggregated information based on the logarithm of the node degree.

Since the PNA model has outperformed many popular models like GIN, GAT, GCN, and MPNN through empirical results from synthetic and real-world domains, we use it as the main base model to experiment with the three adaptations proposed above.

III. Dataset

Anti-Money Laundering (AML). Given the strict privacy regulations around financial data, real-world datasets are not readily available. Instead, we use simulated money laundering data generated by an agent-based simulator called AMLworld [8]. The simulator generates a financial transaction network by modeling agents (banks, companies, and individuals) in a virtual world. The generator uses well-established laundering patterns to add realistic money laundering (illicit) transactions. They are eight commonly used patterns in money laundering, which were introduced by Suzumura and Kanezashi [9]. These eigth patterns are illustrated in Figure 1.

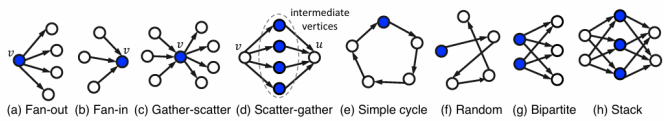

Fig. 1. Laundering Patterns Modelled

The AML data is currently publicly available on Kaggle [10]. The data is divided into two groups: **HI** and **LI**, with higher and lower illicit (laundering) ratios, respectively. Both **HI** and **LI** are further divided into small, medium, and large datasets, with the large datasets containing between 175 million and 180 million transactions, resulting in a total of six datasets. The statistics of them are shown in Table I.

TABLE I
PUBLIC SYNTHETIC DATA STATISTICS. **HI** = HIGHER ILLICIT (MORE LAUNDERING). **LI** = LOWER ILLICIT.

Statistic	Small		Medium		Large	
	HI	**LI**	**HI**	**LI**	**HI**	**LI**
# of Days Spanned	10	10	16	16	97	97
# of Bank Accounts	515K	705K	2077K	2028K	2116K	2064K
# of Transactions	5M	7M	32M	31M	180M	176M
# of Laundering Transactions	3.6K	4.0K	35K	16K	223K	100K
Laundering Rate (1 per N Trans)	981	1942	905	1948	807	1750

Due to the limitation of computation resources, we only use small datasets in this project.

IV. Experiments Setup

Base GNNs and Baselines. PNA [7] is used as our base model with the adaptions on top, and we refer to its adapted versions as Multi-PNA. Furthermore, we also use GIN [11] as a base model and refer its adapted version as Multi-GIN. Since AML is an edge classification problem, we also include a baseline using edge updates, denoted GIN+EU. This approach is similar to replacing edges with nodes and running a GNN on said line graph, which recently achieved state-of-the-art (SOTA) results in self-supervised money laundering detection. We include a baseline representing the parallel line of work in financial crime detection that uses pre-calculated graph-based features (GFs) and tree-based classifiers to classify nodes or edges individually. We train XGBoost model on the individual edges (or nodes) using the original raw features combined with additional graph-based features.

Data split. We use a 60-20-20 temporal train-validation-test split. Data split is defined by two timestamps, t1 and t2. Train indices correspond to transactions before time t1, validation indices to transactions between times t1 and t2 and test indices after t2. We also construct train, validation and test graphs as a dynamic graph by taking three snapshots at time t1, t2, t3=t_max. The train graph contains only the training transactions (and corresponding nodes). The validation graph contains the training and validation transactions, but only the validation indices are used for evaluation. The test graph contains all the transactions, but only the test indices are used for evaluation.

Scoring. Since AML dataset is very imbalanced, accuracy and other popular metrics are not suitable. Instead, we use the minority class F1 score. This aligns well with what banks and regulators use in real-world scenarios.

V. Results

Model	AML Small HI	AML Small LI
XGBoost	1.35	0
LightGBM	4.50	0
GIN (Xu et al. 2018; Hu et al. 2019)	32.05	9.10
PNA (Velickovic et al. 2019)	52.49	15.91
GIN+EU (Battaglia et al. 2018)	33.79	10.8
GIN+ReverseMP+Ports+EgoIDs (Multi GIN)	60.75	27.34
PNA+ReverseMP+Ports+EgoIDs (Multi PNA)	49.56*	

TABLE II
PERFORMANCE COMPARISON OF VARIOUS MODELS ON DIFFERENT DATASETS. METRIC: MINORITY CLASS F1.

Our results are saved on WandB or Drive. If you need more information, please contact us.

Evaluation of GBT and GNN Baselines: (Table II)

GBT Models (XGBoost and LightGBM): Both models show limited effectiveness on these datasets. This indicates that traditional gradient boosting techniques might struggle with the specific characteristics of money laundering datasets, particularly in capturing complex patterns in transactions.

GNN Baselines (GIN, PNA, GIN+EU): GNN models generally outperform GBT models significantly. This suggests that exploiting the connectivity between accounts using message-passing GNNs is crucial for detecting laundering transactions. PNA: Stands out with the best overall performance, especially on the AML Small HI dataset. This can be attributed to its advanced message aggregation mechanisms. GIN+EU: Also shows strong performance, indicating the benefits of including edge update mechanisms.

Multi GNN Models (Multi GIN, Multi PNA): These models demonstrate the highest performance across the board, highlighting the effectiveness of incorporating multiple advanced features (ReverseMP, Ports, EgoIDs). The Multi GIN model, in particular, achieves the highest F1 score on both datasets. However, it is important to note that Multi PNA is computationally intensive, requiring significantly more resources, and can only be run for one-fifth the number of epochs compared to normal. The results in original paper for multi PNA are 64.59 and 30.65 respectively.

Additionally, we provided more information by conducting checks on the detection rate of high-risk laundering patterns in the Hi_Small_Pattern.txt and Li_Small_Pattern.txt datasets from the same Kaggle file [10]. These checks were performed using the previously trained methods, GIN and PNA. The results indicate that out of 3209 transactions, approximately 0.16% involved laundering.

VI. LIMITATIONS AND CONSTRAINTS

Dataset Challenges: AML Small LI Dataset: Presents more significant predictive challenges compared to AML Small HI. This is due to the lower illicit ratios and longer time spans of laundering patterns, which make it harder to detect illicit activities. To address these challenges, models are adjusted by weighting the predictions of the minority class higher in the loss function. Additionally, strategies like reusing or fine-tuning models pretrained on HI datasets for LI datasets are suggested to improve performance.

Hyperparameter Settings and Computational Resource Constraints: Firstly, there is a lack of public code documentation and comprehensive hyperparameter settings for model tuning. Additionally, this paper leverages computational resources from Colab and Kaggle. On Colab, a T4 GPU is used, providing approximately 73 hours of computation time for GNN baselines. On Kaggle, T4x2 GPUs are utilized, offering around 8 hours of computation for GBT models. The computational time required is excessive. Complex models like Multi PNA, GPU RAM overflow can occur. Consequently, reproducing the original models presents significant challenges.

VII. CONCLUSION

This work investigates a series of adaptations that transform conventional message-passing GNNs into powerful directed multigraph learners. Our contributions are threefold. Firstly, our theoretical analysis addresses a gap in the existing literature by proving that the combination of ego IDs, port numbering, and reverse message passing allows GNNs baseline like GIN, PNA to detect edge laundering and subgraph patterns. Secondly, we apply our adaptations to financial crime detection using synthetic laundering data, achieving impressive results in identifying money laundering transactions. Reverse message passing and port numbering are crucial, while ego IDs offer limited additional benefit. Lastly, we present some limitations and constraints of this project.

Although focused on financial crime, our methods have broader applications. Future work should explore their use in other directed multigraph problems and examine the relationship between subgraph detection complexity and GNN performance.

REFERENCES

[1] T. N. Kipf and M. Welling, *Semi-supervised classification with graph convolutional networks*, 2016. eprint: arXiv:1609.02907.

[2] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov, and A. Smola, *Deep sets*, 2017. eprint: arXiv:1703.06114.

[3] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, *Graph attention networks*, 2017. eprint: arXiv:1710.10903.

[4] A. Gupta, S. Narayan, K. J. Joseph, S. H. Khan, F. S. Khan, and M. Shah, "OW-DETR: open-world detection transformer," *CoRR*, vol. abs/2112.01513, 2021. arXiv: 2112.01513. [Online]. Available: https://arxiv.org/abs/2112.01513.

[5] B. Egressy, L. von Niederhäusern, J. Blanusa, E. Altman, R. Wattenhofer, and K. Atasu, *Provably powerful graph neural networks for directed multigraphs*, 2023. eprint: arXiv:2306.11586.

[6] E. Altman, J. Blanuša, L. von Niederhäusern, B. Egressy, A. Anghel, and K. Atasu, *Realistic synthetic financial transactions for anti-money laundering models*, 2023. eprint: arXiv:2306.16424.

[7] G. Corso, L. Cavalleri, D. Beaini, P. Liò, and P. Veličković, "Principal neighbourhood aggregation for graph nets," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 260–13 271, 2020.

[8] E. Altman, J. Blanuša, L. V. Niederhäusern, B. Egressy, A. Anghel, and K. Atasu, "Realistic synthetic financial transactions for anti-money laundering models," in *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. [Online]. Available: https : / / openreview . net / forum?id=XZf2bnMBag.

[9] T. Suzumura and H. Kanezashi, *Anti-money laundering datasets: Inpluslab anti-money laundering datadatasets*, 2021.

[10] IBM Research, *Ibm transactions for anti money laundering (aml)*, 2023. [Online]. Available: https://www.kaggle.com/datasets/ealtman2019/ibm-transactions-for-anti-money-laundering-aml (visited on 05/10/2024).

[11] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, *How powerful are graph neural networks?* 2019. arXiv: 1810.00826 [cs.LG].