Supplementary Material

A Realism of Data Generation

As described in Section 3.5, we used *AMLworld* to create several synthetic AML datasets. The datasets are published on Kaggle [50] under a Community Data License Agreement. As noted previously, data is divided into two high-level groups: **HI** and **LI**. The **HI** datasets have slightly higher illicit ratios (more laundering) than **LI**. Both **HI** and **LI** are further divided into *small, medium*, and *large* datasets, with *large* having $175M - 180M$ transactions. Table 4 provides a number of basic statistics about all six Kaggle datasets. We include more detailed analyses below.

Table 4: Public Synthetic Data Statistics. **HI** = Higher Illicit (more laundering). **LI** = Lower Illicit.

	Small		**Medium**		**Large**	
Statistic	**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

Figure 7 histograms annualized transactions per account. It gives a micro view of how different entities drive the overall transaction counts found in Table 4. Figure 7 also provides a touch point to real data: numbers roughly align with U.S. Federal Reserve data [51]. Table 5 shows the distribution of transaction formats used in the dataset, such as ACH, wire, and cheque. Again numbers correspond roughly with Federal Reserve statistics [41].

Table 8 breaks down laundering transactions between (a) transactions following one of 8 standard patterns discussed in Section 3.2 – in particular Figure 2; and (b) the *integration* transactions disguised as other activities (e.g. employee payroll or company supplies).

As with other data, we base salary and pension amounts on real data, in this case from the US Internal Revenue Service [54]. Figure 8 shows the number of returns filed for the 2018 tax year for each amount of salary income and pension income. Having proper distribution of salaries and pensions in turn helps drive accurate modeling and statistics for transaction sizes and frequency, as just discussed. The \$0 top bin in Figure 8 reflects the fact that some people have no salary income or no pension income. Beyond salary and pension people can have income from other sources such as interest or dividends.

Table 5: Distribution of transaction counts by format in **LI**-*Large*.

Format	**# Transactions**	**Format**	**# Transactions**
Cheque	69,720,485	Reinvestment	7,258,251
Credit Card	49,923,366	Wire	6,346,402
ACH	21,650,558	Bitcoin	3,601,817
Cash	18,069,965		

Table 6: Histogram of # of nodes (accounts) in **LI**-*Large* laundering patterns. **Gather Scatter** has 2 counts: (a) # of nodes from which initial funds come; (b) # of nodes to which funds ultimately go.

# of Nodes		**Fan-out**	**Fan-in**	**Cycle**	**Random**	**Bipartite**	**Stack**	**Scatter-Gather**	**Gather-Scatter**	
Min	Max								(1)	(2)
1	2	50	47	72	70	72	87	54	67	67
2	4	46	54	50	44	30	31	47	43	38
4	8	53	54	84	79	62	56	64	61	55
8	12	57	62	76	62	51	48	48	42	52
12	18	71	62	16	23	62	37	63	71	72
18	∞	0	0	0	0	0	0	0	0	0

Table 7: Occurrence count for each laundering pattern in **LI**-*Large*.

Pattern	**Pattern Count**	**# Trans in Pattern**	**Pattern**	**Pattern Count**	**# Trans in Pattern**
Fan-out	277	2,014	Stack	259	3,239
Fan-in	279	2,003	Random	278	1,831
Cycle	298	2,326	*Scatter-gather*	276	4,202
Bipartite	277	1,858	*Gather-scatter*	284	4,010

Table 8: Laundering rates in **LI**-*Large*. **Ratio** is the total transaction count divided by the laundering transaction count.

Description	**# of Trans**	**Ratio**
Laundering Trans. - Patterns	19,461	9,047
Laundering Trans. - Other	81,143	2,170
Laundering Trans. - Total	100,604	1,750

Figure 8 indicates that there are about 3.4× as many returns showing salary income as pension income, although returns can show both [3]. We assume about 62.5% of people in our models have salaries – matching the value from the U.S. Department of Labor for labor force participation of the adult workforce [42]. Following the IRS ratios, about 18.3% of the population in our datasets has pension income, with around half of those pensioners also receiving a salary.

[3]For income tax purposes in the U.S., salary income also includes total wages for hourly workers.

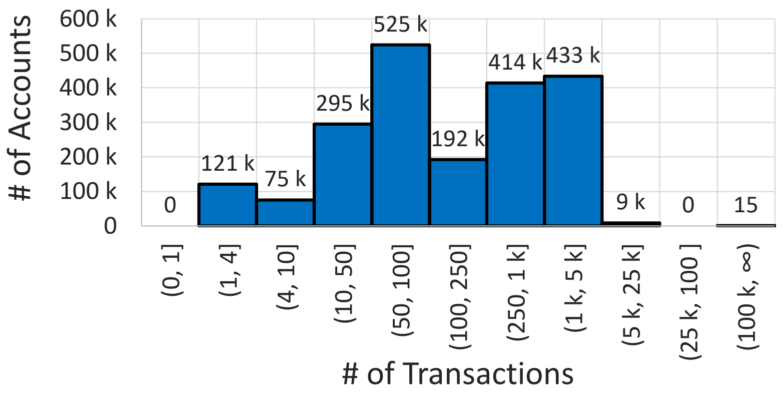


Figure 7: Annualized transaction rate across accounts in **LI**-*Large*.

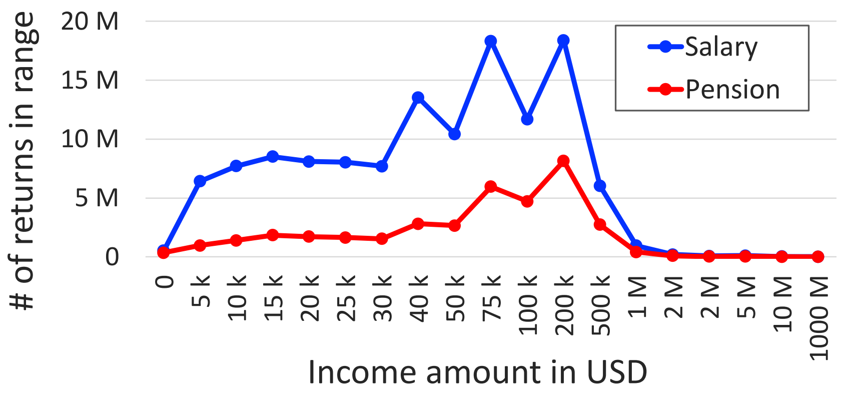


Figure 8: Distribution of U.S. tax returns for 2018 by salary income and pension income.

Table 9: Successive halving parameter configurations used for hyperparameter tuning of shared models trained using all banks of a dataset (*Multi-bank models*) and private models trained using the data of a single bank (*Single-bank models*).

	Multi-bank models			**Single-bank models**	
Datasets	*Small*	*Medium*	*Large*	*Medium*	*Large*
x_0	1000	100	16	1000	100
η	2	2	2	1.5	2
r_0	0.1	0.1	0.2	0.3	0.1

Additional statistics about our virtual world were provided in Section 3.5.

B Ethical Use of the Data

We addressed a number ethics issues in Section 7. We offer a few additional observations here.

Ethical use of the data includes using it for benchmarking and improving models for the detection of money laundering activity. We foresee significant positive societal impacts from such uses of our data. Money laundering has a huge cost for society in itself, but more importantly, money laundering enables a whole range of criminal activities to continue, ranging from phishing attacks to human trafficking. Detecting money laundering transactions can help authorities uncover such activities and identify the criminals behind them. Additionally, the data could be used for pretraining detection models before fine-tuning on real data.

Given that the dataset is synthetic, there is no risk of it containing personally identifiable information or offensive content. Moreover, researchers do not need to take special care with its use, but they should bear in mind that performance might not translate one-to-one to real data.

C Hyperparameter Tuning of Machine Learning Models

GBT Baselines. We use successive halving [26] for the hyperparameter tuning of the LightGBM and XGBoost models. Successive halving starts by randomly sampling x_0 model parameter configurations. Each configuration of model parameters is evaluated using a fraction $r_0 \leq 1$ of the initial training set. The algorithm then finds the best x_0/η configurations, with $\eta > 1$. These best configurations are used in the next round of successive halving using a fraction $\eta \times r_0$ of the train set. This process continues until the fraction of the training set used for the evaluation reaches 1. We use different successive halving configurations for different datasets, as shown in Table 9. Furthermore, the parameter ranges from which x_0 initial parameter configurations are sampled are shown in Table 10.

GNN Baselines. We used random sampling to identify a good range of GNN hyperparameters. A second round of random sampling was conducted with this narrower range to pick our final set of hyperparameters. We varied the following hyperparameters: the number of GNN layers, hidden embedding size, learning rate, dropout, and minority class weight (for the weighted loss function).

Table 10: Model parameter ranges used at for the hyperparameter tuning of GBT models. The small parameter range, indicated with *Range-small*, is only used for hyperparameter tuning of LightGBM models related to **HI**-*Large* and **LI**-*Large* datasets. The large parameter range shown in column *Range-large* is used for other datasets.

LightGBM			XGBoost	
Parameter	**Range-small**	**Range-large**	**Parameter**	**Range**
num_round	$(32, 512)$	$(10, 1000)$	num_round	$(10, 1000)$
num_leaves	$(32, 256)$	$(1, 16384)$	max_depth	$(1, 15)$
learning_rate	$(0.001, 0.01)$	$10^{(-2.5,-1)}$	learning_rate	$10^{(-2.5,-1)}$
lambda_l2	$(0.01, 0.5)$	$10^{(-2,2)}$	lambda	$10^{(-2,2)}$
scale_pos_weight	$(1, 10)$	$(1, 10)$	scale_pos_weight	$(1, 10)$
lambda_l1	$10^{(0.01,0.5)}$	$10^{(0.01,0.5)}$	colsample_bytree	$(0.5, 1.0)$
			subsample	$(0.5, 1.0)$

The exact ranges we used are listed in 11. The number of random samples was set to between 10 to 50, depending on the training time of the model on a particular dataset. To get our final results, we use the hyperparameters with the best validation score to train four models initialized with different random seeds.

Table 11: Model parameter ranges used for hyperparameter tuning of GNN models. The hyperparameters were optimized on the small datasets and used for all GNN models.

Parameter Ranges	GIN (+EU)	PNA
hidden embedding size	(16,72)	(16,64)
learning rate	(0.005,0.05)	(0.0001, 0.002)
number of GNN layers	(2,4)	(2,4)
dropout	(0,0.5)	(0, 0.2)
minority class weight	(6,8)	(6,8)

D Graph Feature Preprocessor Configuration

The Graph Feature Preprocessor (GFP) processes transactions represented as temporal edges in a streaming fashion. The input of this preprocessor is a batch of temporal edges that the preprocessor inserts into its in-memory dynamic graph and extracts various graph features from this graph, such as scatter-gather patterns, simple cycles, and vertex statistics. The output of the preprocessor is the same batch of edges with these additional graph-based features. This library is available as part of the Snap ML library [49], and the experiments in this paper were performed using version 1.14 of Snap ML. More details on GFP are available in the documentation [48].

The graph-based features for our experiments are extracted using the batch size of 128. The GFP is configured to generate features based on scatter-gather patterns, temporal cycles, simple cycles of length up to 10, and vertex statistics. We set the GFP to use a time window of six hours for scatter-gather patterns and a time window of one day for the rest of the graph-based features. The vertex statistic features are computed using the "Amount" and "Timestamp" fields of the basic transaction features (see Figure 1a). This configuration is used for all datasets and experiments that use GFP in this paper.

For each dataset, GFP processes transactions in the increasing order of their timestamps. This ordering ensures that the graph-based features for each transaction are extracted using the past data. As a result, transactions from the training set will not contain graph-based features computed using the validation or test sets. Processing transactions in such a way prevents data leakage.

E Additional GNN Experiments

Our GNN code is included with the supplementary material and available publicly on GitHub [4] under an Apache License. The GNNs are implemented using PyTorch Geometric version 2.3.1 [21] and PyTorch version 2.0.1.

All the baseline GNN experiments were run on an internal cluster on Nvidia Tesla V100 GPUs. Table 12 shows the runtimes when training the different GNN baselines on the AML Small and Medium datasets. The size of the GNN models was kept the same: 2 GNN layers and a hidden embedding size of 64. The total GPU time, including initial experiments and hyperparameter optimization, is estimated to have been around 1000 GPU hours.

Table 12: Total training times (TTT) and inference performance in Transaction per Second (TPS) for all GNN baselines on the AML Small and Medium datasets using an Nvidia Tesla V100 GPU.

Model	**HI**-*Small*		**LI**-*Small*		**HI**-*Medium*		**LI**-*Medium*	
	TTT (s)	TPS	TTT (s)	TPS	TTT (s)	TPS	TTT (s)	TPS
GIN	22703	17210	29655	15784	85652	7101	85994	7316
GIN+EU	26046	11844	36625	11640	85753	4981	85887	5097
PNA	27745	16557	39648	14994	85654	6725	85827	6819

In addition to the minority class F1 scores in Table 2, we include more fine-grained results detailing the precision and recall rates of the GNN-based models. Precision evaluates the accuracy of laundering predictions, recall measures the model's ability to identify all laundering instances, while the F1 score is the harmonic mean of precision and recall. Table 13 shows the laundering recall rates and Table 14 shows the corresponding precision scores. In Figure 9, we give example precision-recall curves, which visually capture the trade-off between precision and recall across different decision thresholds. The examples are taken from the best-performing seed from the best-performing model (PNA) on all small and medium AML datasets.

Table 13: Minority class recall rate (%) for the GNN-based models. HI indicates a higher illicit ratio. LI indicates a lower illicit ratio.

Model	**HI**-*Small*	**LI**-*Small*	**HI**-*Medium*	**LI**-*Medium*
GIN [65, 25]	38.16 ± 5.92	14.59 ± 2.37	39.86 ± 3.61	8.07 ± 9.32
GIN+EU [11, 15]	55.41 ± 5.96	23.26 ± 2.87	48.06 ± 6.45	5.51 ± 6.82
PNA [60]	53.15 ± 2.26	16.43 ± 2.62	47.42 ± 4.30	20.44 ± 0.66

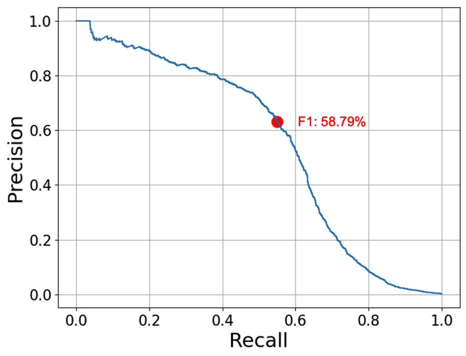
Table 14: Minority class precision (%) for the GNN-based models. HI indicates a higher illicit ratio. LI indicates a lower illicit ratio.

Model	**HI**-*Small*	**LI**-*Small*	**HI**-*Medium*	**LI**-*Medium*
GIN [65, 25]	27.40 ± 7.98	5.14 ± 3.42	47.78 ± 5.20	5.23 ± 3.60
GIN+EU [11, 15]	42.29 ± 10.69	21.60 ± 10.83	52.56 ± 8.27	12.13 ± 19.35
PNA [60]	58.48 ± 10.67	17.37 ± 5.80	69.12 ± 4.26	36.50 ± 10.51

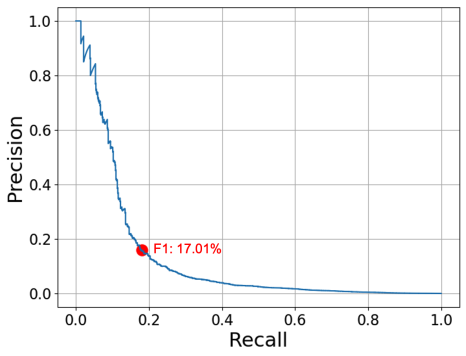
F Additional GBT Experiments

Figure 11 shows the precision-recall curves for XGBoost models trained on all AML datasets using graph-based features created with Graph Feature Preprocessor. These curves correspond to the results from the row "GFP+XGBoost" shown in Table 2. The steep slope of each curve after the red dot, which represent a data point obtained using the prediction threshold of 0.5, indicates that it is challenging to obtain higher recall without significantly degrading the precision. Note that it is possible to achieve higher precision of these XGBoost models by simply reducing the recall by a few percent.

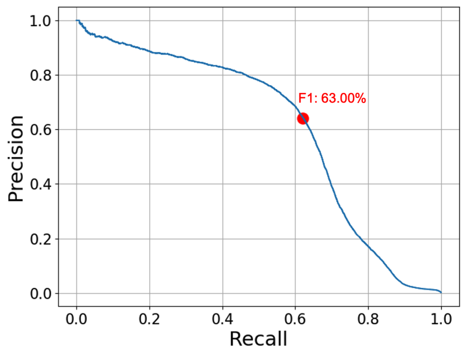
[4]https://github.com/IBM/Multi-GNN



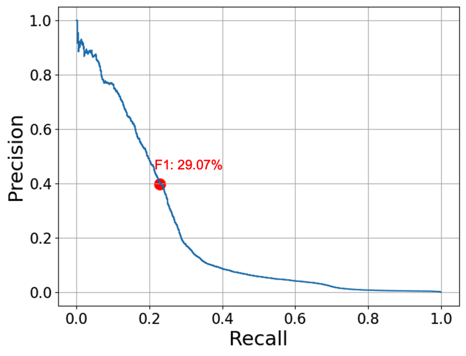
(a) **HI**-*Small*



(b) **LI**-*Small*



(c) **HI**-*Medium*



(d) **LI**-*Medium*

Figure 9: Precision-Recall Curves for the best-performing PNA model for all AML datasets. The red dot indicates the F1 score obtained using the prediction threshold of 0.5.

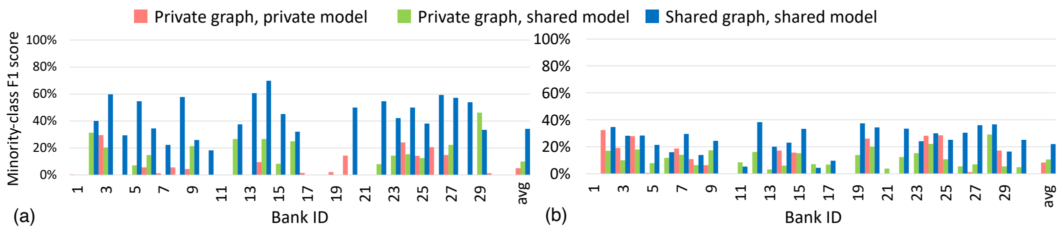
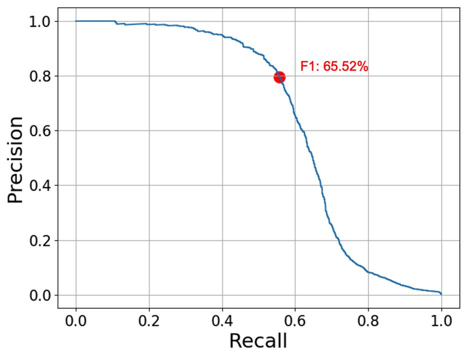
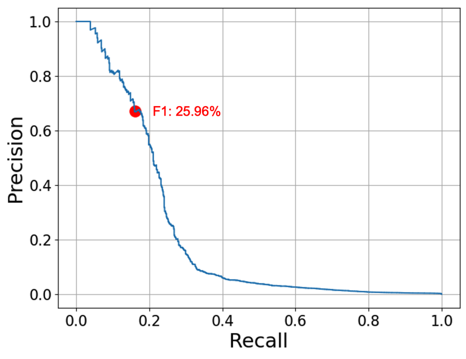


Figure 10: Effect of sharing the data and LightGBM models across banks for the (a) **LI**-*Medium* and (b) **LI**-*Large* datasets.

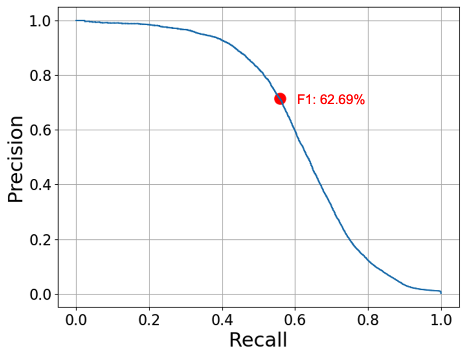
Figure 10 shows performance on a per-bank basis using the **LI** datasets. The experiment is explained in Section 4 and the plot is analogous to Figure 6, but here we are using the **LI**-*Medium* and **LI**-*Large* datasets. These dataset contain fewer illicit transactions compared to the **HI**-*Medium* and **HI**-*Large* datasets (see Table 4). Therefore, it is more challenging to build a machine learning model for a bank using only local data of these **LI** datasets compared to the banks of **HI** datasets. In this case, the average minority-class F1 score across 30 banks shown in Figure 10 is only 4.9% for **LI**-*Medium* and 8.7% for **LI**-*Large*. Nevertheless, the improvement in minority-class F1 score of the *shared graph, shared model* case compared to the *private graph, private model* case is still significant. Sharing the transaction graph and the global model across the banks increases the average minority-class F1 score to 20.8% for **LI**-*Medium* and to 22.1% for **LI**-*Large*.



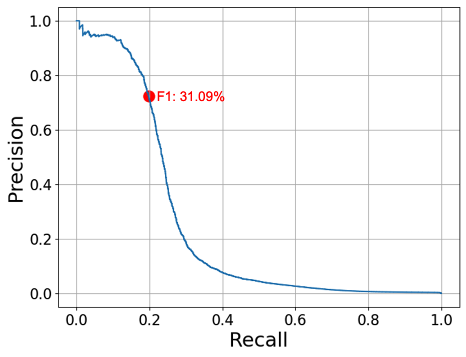
(a) **HI**-*Small*



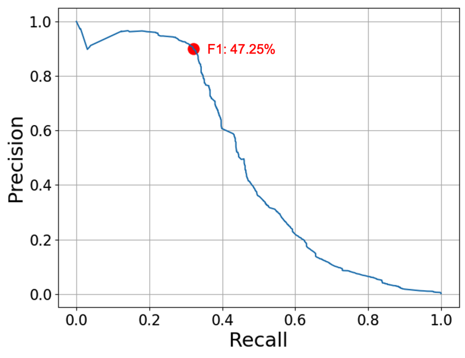
(b) **LI**-*Small*



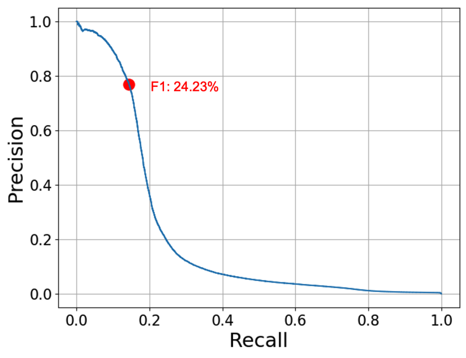
(c) **HI**-*Medium*



(d) **LI**-*Medium*



(e) **HI**-*Large*



(f) **LI**-*Large*

Figure 11: Precision-Recall Curves for the XGBoost models for all AML datasets trained using graph-based features of GFP. The red dot indicates the F1 score obtained using the prediction threshold of 0.5.

G Datasheet

We include a datasheet based on the framework set forward by Gebru et al. [22]. Some parts of the proposed datasheet are omitted since our datasets are synthetic.

G.1 Motivation

For what purpose was the dataset created? The dataset was created to test, develop, and improve machine-learning models for financial crime detection. In particular, the datasets focus on identifying money laundering transactions.

Who created the dataset and on behalf of which entity? Erik Altman on behalf of IBM.

Who funded the creation of the dataset? IBM

G.2 Composition

What do the instances that comprise the dataset represent? The datasets are synthetic financial transaction networks. Each node in the network represents an account/entity and each directed edge represents a transaction from one account to another. Edge features detail the amount, currency, and type of transaction amongst other properties. The datasets also contain transactions that are labeled as money laundering. All the information is simulated. No real account or transaction details were used to create the datasets.

How many instances are there in total? There are 6 datasets. Each dataset consists of one graph. The number of transactions (samples) ranges from 5M to 180M.

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? The datasets are synthetic. Any number of transactions could be generated.

What data does each instance consist of? When used for transaction classification, the edges can be considered instances. Each instance consists of a set of transaction features (incl., amount, currency, date, time, and type). In addition, each transaction is part of a whole network of transactions, and since the network topology plays an important role, the position of the instance in the whole network could be considered a "part" of the instance.

Is there a label or target associated with each instance? Yes.

Is any information missing from individual instances? No.

Are there recommended data splits (e.g., training, development/validation, testing)? Yes.

Are there any errors, sources of noise, or redundancies in the dataset? Not to the knowledge of the authors.

Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? The dataset is self-contained.

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctor–patient confidentiality, data that includes the content of individuals' non-public communications)? No.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? No.

G.3 Collection Process, Uses, Distribution and Maintenance

Please refer to Section 4 and Appendix A for details about the generation process. The current usage is detailed in the paper and potential uses are described in Appendix B. The Kaggle page acts as the single source of distribution [50]. The dataset will be maintained there.

References

[1] [n.d.]. Money Laundering. https://en.wikipedia.org/wiki/Danske_Bank_money_laundering_scandal

[2] [n.d.]. Money laundering. https://en.wikipedia.org/wiki/Money_laundering

[3] 2015. Measuring the Economy: A Primer on GDP and the National Income and Product Accounts. , 2 pages. https://www.bea.gov/sites/default/files/methodologies/nipa_primer.pdf

[4] 2017. Blind Spot in the AML Regime. https://www.finextra.com/blogposting/14298/online-payments-the-blind-spot-in-the-aml-regime

[5] 2017. From Suspicion to Action: Converting financial intelligence into greater operational impact. , 4 pages. https://www.europol.europa.eu/cms/sites/default/files/documents/ql-01-17-932-en-c_pf_final.pdf

[6] 2022. Money Laundering. https://www.unodc.org/unodc/en/money-laundering/overview.html

[7] 2022. National Money Laundering Risk Assessment. , 21 pages. https://home.treasury.gov/system/files/136/2022-National-Money-Laundering-Risk-Assessment.pdf

[8] Erik Altman. 2021. Synthesizing Credit Card Transactions. In *Proceedings of the Second ACM International Conference on AI in Finance (ICAIF'21)*, Anisoara Calinescu and Lukasz Szpruch (Ed.). ACM, Virtual Event. https://doi.org/10.1145/3490354.3494378

[9] Ansh Ankul. 2021. IBM AMLSim Example Dataset. (2021). https://www.kaggle.com/datasets/anshankul/ibm-amlsim-example-dataset.

[10] V K Balakrishnan. 1997. *Graph Theory*. McGraw-Hill Professional, New York, NY.

[11] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261* (2018).

[12] Jovan Blanuša, Paolo Ienne, and Kubilay Atasu. 2022. Scalable Fine-Grained Parallel Cycle Enumeration Algorithms. In *Proceedings of the 34th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*. ACM, Philadelphia PA USA, 247–258. https://doi.org/10.1145/3490148.3538585

[13] Jovan Blanuša, Kubilay Atasu, and Paolo Ienne. 2023. Fast Parallel Algorithms for Enumeration of Simple, Temporal, and Hop-constrained Cycles. *ACM Trans. Parallel Comput.* 10, 3 (Sept. 2023), 1–35. https://doi.org/10.1145/3611642

[14] Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M. Bronstein. 2023. Improving Graph Neural Network Expressivity via Subgraph Isomorphism Counting. *IEEE Trans. Pattern Anal. Mach. Intell.* 45, 1 (Jan. 2023), 657–668. https://doi.org/10.1109/TPAMI.2022.3154319

[15] Mário Cardoso, Pedro Saleiro, and Pedro Bizarro. 2022. LaundroGraph: Self-Supervised Graph Representation Learning for Anti-Money Laundering. In *Proceedings of the Third ACM International Conference on AI in Finance*. 130–138.

[16] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. ACM, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785

[17] Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna. 2020. Can Graph Neural Networks Count Substructures?. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/75877cb75154206c4e65e76b88a12712-Abstract.html

[18] Gabriele Corso, Luca Cavalleri, Dominiique Beaini, Pietro Lio, and Petar Velickovic. 2020. Principal Neighbourhood Aggregation for Graph Nets. *NeurIPS* abs/2111.15367 (2020). https://proceedings.neurips.cc/paper/2020/file/99cad265a1768cc2dd013f0e740300ae-Paper.pdf

[19] Kevin Dolan. 2022. *AML Trends in 2022: What You Need to Know*. Technical Report. IDC. 1–10 pages. https://www.idc.com/getdoc.jsp?containerId=US48661022 Accessed: 2023-01-10.

[20] Beni Egressy, Luc von Niederhäusern, Jovan Blanusa, Erik Altman, Roger Wattenhofer, and Kubilay Atasu. 2023. Provably Powerful Graph Neural Networks for Directed Multigraphs. *arXiv preprint arXiv:2306.11586* (2023). https://arxiv.org/abs/2306.11586

[21] Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. *arXiv preprint arXiv:1903.02428* (2019).

[22] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III au2, and Kate Crawford. 2021. Datasheets for Datasets. arXiv:1803.09010 [cs.DB]

[23] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*.

[24] Daniel A. Harris, Kyla L. Pyndiura, Shelby L. Sturrock, and Rebecca A.G. Christensen. 2021. Using real-world transaction data to identify money laundering: Leveraging traditional regression and machine learning techniques. (Dec. 2021). https://journal.stemfellowship.org/doi/pdf/10.17975/sfj-2021-006.

[25] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2019. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265* (2019).

[26] Kevin Jamieson and Robert Nowak. 2014. Best-arm identification algorithms for multi-armed bandits in the fixed confidence setting. In *2014 48th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, Princeton, NJ, USA, 1–6. https://doi.org/10.1109/CISS.2014.6814096

[27] Donald B. Johnson. 1975. Finding All the Elementary Circuits of a Directed Graph. *SIAM J. Comput.* 4, 1 (March 1975), 77–84. doi: 10.1137/0204007.

[28] Hiroki Kanezashi, Toyotaro Suzumura, Xin Liu, and Takahiro Hirofuchi. 2022. Ethereum Fraud Detection with Heterogeneous Graph Neural Networks. http://arxiv.org/abs/2203.12363 arXiv:2203.12363 [cs].

[29] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017), 3146–3154.

[30] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=SJU4ayYgl

[31] Manish Kumar and et al. 2018. What are some good sources of data for Anti Money Laundering research? https://www.quora.com/What-are-some-good-sources-of-data-for-Anti-Money-Laundering-research

[32] Xuan Li, Kunfeng Wang, Yonglin Tian, Lan Yan, and Fei-Yue Wang. 2017. The ParallelEye Dataset: Constructing Large-Scale Artificial Scenes for Traffic Vision Research. *https://arxiv.org/abs/1712.08394* (December 2017).

[33] Guimei Liu, Kelvin Sim, and Jinyan Li. 2006. Efficient Mining of Large Maximal Bicliques. In *Data Warehousing and Knowledge Discovery*. Vol. 4081. Springer Berlin Heidelberg, Berlin, Heidelberg, 437–448. https://doi.org/10.1007/11823728_42 Series Title: Lecture Notes in Computer Science.

[34] Xiao Fan Liu, Xin-Jian Jiang, Si-Hao Liu, and Chi Kong Tse. 2021. Knowledge Discovery in Cryptocurrency Transactions: A Survey. *IEEE Access* 9 (2021), 37229–37254. https://doi.org/10.1109/ACCESS.2021.3062652

[35] Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. 2021. Pick and Choose: A GNN-Based Imbalanced Learning Approach for Fraud Detection. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 3168–3177. https://doi.org/10.1145/3442381.3449989

[36] Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. 2021. Pick and choose: a GNN-based imbalanced learning approach for fraud detection. In *Proceedings of the web conference 2021*. 3168–3177.

[37] Edgar Alonso Lopez-Rojas, Ahmad Elmir, and Stefan Axelsson. 2016. Paysim: A Financial Mobile Money Simulator for Fraud Detection. In *28th European Modeling and Simulation Conference (EMSS)*, (Ed.). , Larnaca, Cyprus.

[38] Maryam Mahootiha. 2020. money laundering data. (2020). https://www.kaggle.com/datasets/maryam1212/money-laundering-data.

[39] Maryam Mahootiha. 2020. money laundering data production. (2020). https://web.archive.org/web/20200916200819/https://github.com/mahootiha-maryam/moneylaundering-data-production.

[40] Neuromation. 2019. https://neuromation.io/marketplace. (2019).

[41] Board of Governors of the Federal Reserve System. 2018. Changes in U.S. Payments Fraud from 2012 to 2016: Evidence from the Federal Reserve Payments Study. (2018). https://www.federalreserve.gov/publications/files/changes-in-us-payments-fraud-from-2012-to-2016-20181016.pdf.

[42] U.S. Bureau of Labor Statistics. 2023. Civilian labor force participation rate. (2023). https://www.bls.gov/charts/employment-situation/civilian-labor-force-participation-rate.htm.

[43] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B. Schardl, and Charles E. Leiserson. 2020. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. (2020). https://arxiv.org/abs/1902.10191.

[44] Patki_et_al. 2016. The Synthetic Data Vault. *IEEE International Conference on Data Science and Advanced Analytics* (October 2016).

[45] Peng_et_al. 2015. Learning Deep Object Detectors from 3D Models. *https://arxiv.org/abs/1412.7122* (October 2015).

[46] Erich Prisner. 2000. Bicliques in Graphs I: Bounds on Their Number. *Combinatorica* 20, 1 (Jan. 2000), 109–117. https://doi.org/10.1007/s004930070035

[47] Susie Xi Rao, Shuai Zhang, Zhichao Han, Zitao Zhang, Wei Min, Zhiyao Chen, Yinan Shan, Yang Zhao, and Ce Zhang. 2021. xFraud: explainable fraud transaction detection. *PVLDB* 15, 3 (Nov. 2021), 427–436. https://doi.org/10.14778/3494124.3494128

[48] IBM Research. 2022. Graph Feature Preprocessor PyPI Documentation. https://snapml.readthedocs.io/en/latest/graph_preprocessor.html Accessed: 2023-01-10.

[49] IBM Research. 2022. Snap ML PyPI package. https://pypi.org/project/snapml/ Accessed: 2023-01-10.

[50] IBM Research. 2023. IBM Transactions for Anti Money Laundering (AML). https://www.kaggle.com/datasets/ealtman2019/ibm-transactions-for-anti-money-laundering-aml Accessed: 2023-02-28.

[51] Atlanta Federal Reserve. 2019. 2019 Survey of Consumer Payment Choice. (2019).

[52] Ben Roshan. 2021. Transaction Fraud Detection: Automating money laundering alerts. https://medium.com/analytics-vidhya/transaction-fraud-detection-%EF%B8%8F-%EF%B8%8F-automating-money-laundering-alerts-8d7d265befa9

[53] Donald Rubin. 1993. Discussion: Statistical Disclosure Limitation. *Journal of Official Statistics* 9, 2 (January 1993).

[54] United States Internal Revenue Service. 2018. 2018 Salary, Pension, and Other Data. (2018). https://www.irs.gov/pub/irs-soi/18in14ar.xls.

[55] Michele Starnini, Charalampos E. Tsourakakis, Maryam Zamanipour, André Panisson, Walter Allasia, Marco Fornasiero, Laura Li Puma, Valeria Ricci, Silvia Ronchiadin, Angela Ugrinoska, Marco Varetto, and Dario Moncalvo. 2021. Smurf-Based Anti-money Laundering in Time-Evolving Transaction Networks. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track*. Vol. 12978. Springer International Publishing, Cham, 171–186. https://doi.org/10.1007/978-3-030-86514-6_11

[56] Michele Starnini, Charaalampos E. Tsourakakis, Maryam Zamanipour, André Panisson, Walter Allasia, Marco Fornasiero, Laura Li Puma, Valeria Ricci, Silvia Ronchiadin, Angela Ugrinoska, Marco Varetto, and Dario Moncalvo. 2021. Smurf-based Anti-Money Laundering in Time-Evolving Transaction Networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, (Ed.). ACM, Bilbao, Spain, 171–186. https://doi.org/10.1007/978-3-030-86514-6_11

[57] Toyotaro Suzumura and Hiroki Kanezashi. 2021. Anti-Money Laundering Datasets. (2021). https://github.com/IBM/AMLSim.

[58] Toyotaro Suzumura and Hiroki Kanezashi. 2021. Anti-Money Laundering Datasets: InPlusLab Anti-Money Laundering DataDatasets. http://github.com/IBM/AMLSim/.

[59] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations* (2018). https://openreview.net/forum?id=rJXMpikCZ

[60] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. *ICLR (Poster)* 2, 3 (2019), 4.

[61] Austin Walters. 2018. Why You Don't Necessarily Need Data for Data Science. (2018). https://medium.com/capitalonetech/whyyoudontnecessarilyneeddatafordatascience-48d7bf503074.

[62] Jianian Wang, Sheng Zhang, Yanghua Xiao, and Rui Song. 2021. A Review on Graph Neural Network Methods in Financial Applications. *CoRR* abs/2111.15367 (2021). arXiv:2111.15367 https://arxiv.org/abs/2111.15367

[63] Mark Weber, Jie Chen, Toyotaro Suzumura, Aldo Pareja, Tengfei Ma, Hiroki Kanezashi, Tim Kaler, Charles E. Leiserson, and Tao B. Schardl. 2018. Scalable Graph Learning for Anti-Money Laundering: A First Look. (2018). https://arxiv.org/abs/1812.00076.

[64] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. 2019. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. *arXiv preprint arXiv:1908.02591* (2019).

[65] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826* (2018).

[66] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. 2021. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM international conference on web search and data mining*. 833–841.