IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151184

Anti-Money Laundering Detection in Financial Transaction Networks using Graph Convolutional Networks (GCN)

Pranav Joshi[1], Tushar Talekar[2], Dnyanesh Patil[3], Subodh Patil[4], P.A. Jain[5]

Student, Computer Engineering, SCTR's Pune Institute of Computer Technology, Pune, India[1-4]

Assistant Professor, Computer Engineering, SCTR's Pune Institute of Computer Technology, Pune, India[5]

Abstract: Money Laundering is an illegal activity that hides the source and movement of illicitly obtained funds. AML or Anti - Money Laundering system cannot flag such activity easily. These systems usually depend on set rules. Look at each transaction, by itself, which isn't enough when the activities are carefully hidden. Such approaches may overlook patterns such as layering, smurfing and circular movement of funds that emerge across interconnected financial entities. This paper presents a Graph Convolutional Network (GCN) framework. It uses transaction features and the links between connected entities. Then it normalizes transaction features, builds the financial transaction graph and handles class imbalance to spot illicit transactions. GCN-based detection is compared with Logistic Regression and Random Forest as baselines and with GraphSAGE and Graph Attention Network (GAT) as alternative graph-based approaches. The models are evaluated using Precision, Recall, F1-score and Area Under the Precision-Recall Curve (AUC-PR) with emphasis on detection of the minority class. In addition, suspicious transaction subgraphs are visualized to better interpret and investigate flagged activities. The framework provides an approach to evaluate whether graph-based learning can capture hidden relational patterns in financial transaction networks more effectively than transaction-level approaches while providing interpretable network-level views, for AML investigation.

Keywords: Anti-Money Laundering, Graph Convolutional Networks, Graph Neural Networks, Financial Transaction Networks, Illicit Transaction Detection, Class Imbalance, Cryptocurrency, Elliptic Dataset

I. INTRODUCTION

Money laundering is a very serious financial crime in which the illicitly acquired funds are transferred or transformed through multiple transactions to bury its origin and show them as legitimate. For detection, it may turn out hard as laundering methods are designed in such a way to bypass standard monitoring mechanisms. The layering, smurfing, rapid movement of funds, and circular transfers of funds that takes place during money laundering has more than one entity and transaction, so individual observation isn't sufficient. The detection of such hidden patterns is therefore an important problem for financial institutions and Anti-Money Laundering (AML) systems.

Traditional techniques mainly depend on rules. It can be fixed limits and observations of transactions. When traditional ways flag known suspicious behaviour, it does not help much if new or complex patterns of relationships appear. Traditional machine learning models such as Logistic Regression and Random Forest can learn useful information from transaction-level features, but they generally represent transactions as independent observations. Also, relationships and connectivity among accounts, wallets, and transactions are not taken into account in the learning process.

Financial transaction data can be shown in a way as a graph. In that graph nodes stand for entities and edges are the money transfers between them. This graph keeps the structure of the transaction network clear and gives us clues about how money moves between connected nodes. It helps see patterns that can be hidden otherwise. Graph Neural Networks or GNNs use graph information that learns and understands both from the node features and the graph structure. In particular Graph Convolutional Networks, also called GCNs pull together data from neighbouring nodes. That lets the GCN model use the network context when it labels suspicious entities or suspicious transactions.

In AML, the recent study shows its relevance. Previous work has looked at GCNs, GraphSAGE, Graph Attention Networks (GATs), self-supervised graph representation, subgraph representation, graph transformers and heterogeneous GNNs. These approaches have helped make it simpler to see the structure and relationships in financial transaction networks. There are still problems, dealing with AML data that is very imbalanced, complicated transaction setups, scaling up the models, and finding and understanding suspicious parts of the transaction graph. The progression of recent

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151184

work also motivates systematic comparison between graph-based approaches and conventional transaction-level machine learning methods.

This paper looks at the challenge of finding transactions by creating a framework based on Graph Convolutional Networks or GCNs for financial transaction networks. First build a transaction graph using data then clean and normalize the features. Also handle class imbalance, an important part as illegal transactions are few compared to normal ones. Logistic Regression and Random Forest form traditional ways. For comparison it also tests two graph-based methods: GraphSAGE and Graph Attention Network or GAT. These are chosen to see its performance when analyzing the network structure. This comparative design enables the effect of incorporating graph structure to be examined against both transaction-level and alternative graph-based learning approaches.

The evaluation is done using Precision, Recall, F1-score and AUC-PR with utmost importance on minority illicit class. The Elliptic Bitcoin Transaction dataset is considered and asks whether graph-based approach dominates conventional ways. Along with classification, the proposed system also visualizes suspicious transaction subgraphs thus providing a network-level view.

The main research question in this study is: **Does adding graph structure with a Graph Convolutional Network make it easier to spot transactions than regular transaction-level machine learning models and other graph-based methods?** It is answered by running experiments that use chosen performance metrics and various transaction network representations.

II. LITERATURE SURVEY

Recent research on Anti-Money Laundering (AML) detection has increasingly explored machine learning and graph-based approaches to overcome the limitations of conventional rule-based systems. Since financial transactions form interconnected networks involving accounts, entities, and money transfers, graph-based methods can capture relational and structural patterns that may not be evident from individual transaction features. Recent studies have therefore investigated Graph Neural Networks (GNNs), including Graph Convolutional Networks (GCNs), GraphSAGE, Graph Attention Networks (GATs), graph transformers, and other graph-based architectures for detecting illicit financial activities. These studies focus on improving the detection of hidden transaction patterns while addressing challenges such as highly imbalanced datasets, evolving laundering strategies, complex network structures, and the interpretability of detected suspicious activities. The following literature survey reviews recent approaches for graph-based AML detection and compares their methodologies, datasets, and key findings to identify the research gap addressed by the proposed GCN-based framework.

TABLE I LITERATURE SURVEY

Paper	Best Classification Technique	Other Techniques Tested	Preprocessing / Feature Techniques	Dataset	Performance Parameter / Key Result
[1]	EvolveGCN	SkipGCN, FastGCN, GraphSAGE, Node2Vec, Attri2Vec, DGT + XGBoost, RF, LightGBM, XGBoost	Node embeddings; local/global topological features; temporal information; feature combination	Elliptic, AMLSim, IBM AML, SynthAML	Minority-class F1, micro-F1
[2]	HMPNN-ct	HGraphSAGE, Regular Neural Network, XGBoost, HMPNN-sum	Heterogeneous graph construction; individual/corporate node types; transaction and business-role features	Large real-world DNB bank transaction dataset	PR-AUC, ROC-AUC, Precision at fixed Recall

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151184

Paper	Best Classification Technique	Other Techniques Tested	Preprocessing / Feature Techniques	Dataset	Performance Parameter / Key Result
[3]	ChebNet / GATv2 with linear layer and skip connections	GCN, GAT, GraphSAGE	Transaction + aggregated features; feature subset analysis; graph convolution and skip connections	Elliptic Bitcoin Dataset	Recall and F1-score
[4]	RevClassify / RevTrack	Sub2Vec, GNN-SEG, GNN-PLAIN, GLASS	Sender/receiver tracking; subgraph filtering; node-feature incorporation	Elliptic2 Dataset	F1, PR-AUC, preprocessing/ training/infere nce cost
[5]	ChebNet / GraphSAGE / GATv2	GCN, GAT, GraphSAGE	Transaction and aggregated feature extraction; graph construction; feature subsets	Elliptic Bitcoin Dataset	Recall and F1-score
[6]	GAMLNet	GraphSAGE, GIN, XGBoost, Random Forest	Structurally aware features; statistically significant graph features; directed graph construction	Synthetic financial transaction datasets containing seven AML topologies	F1-score, Accuracy
[7]	FraudGT/ Multi-FraudGT	GAT, MLP, PNA and other GNN baselines	Edge-based message passing; edge-attribute attention bias; directed multigraph representation	Large-scale synthetic financial transaction datasets	F1-score, throughput, latency
[8]	MLaD2	Existing semi-supervised GNN methods, XGBoost and other baselines	Transaction Relationship Network based on Node Similarity (TRNNS); contrastive self-supervised pretraining; decoupled training	Real-world money-laundering dataset	Accuracy, F1-score, False Positive Rate
[9]	Hierarchical Clustering + Graph Network Analysis	K-Means clustering	Graph construction; clustering of account representations; outlier/suspicious-account identification	Real-world banking transaction dataset	Precision, Recall, TPR, FPR, TNR, FNR, ROC-AUC
[10]	Inspection-L (DGI + GIN + RF)	Logistic Regression, Random Forest, GCN-based embeddings, supervised approaches	Self-supervised node representation learning; DGI; GIN-based node embeddings	Elliptic Dataset	Precision, Recall, F1, AUC
[11]	DIGA	GNN-based AML methods, graph-learning baselines	Graph recovery; diffusion-based graph generation; handling sparse and imbalanced transaction graphs	Large-scale real-world banking transaction data / AML data	F1-score and detection performance

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151184

III. METHODOLOGY

A. *Overview of the Proposed Framework*

The proposed framework identifies potentially illicit activities within financial transaction networks by integrating individual transaction characteristics with the underlying relationships and connections among transactions. It is an end-to-end pipeline of seven stages (Figure. 1):

- Data collection
- Data preprocessing
- Feature selection
- Graph construction
- Data splitting
- Model training (traditional machine learning and graph-based models)
- Model evaluation and an interpretable output layer for compliance analysts

Conventional machine learning techniques, such as Logistic Regression and Random Forest, generally analyse each transaction as a separate and independent data point. The graph-based models instead use the neighbourhood of each transaction, which is where layering, smurfing, and circular fund movement appear. The pipeline is designed so that the effect of graph structure can be compared fairly against transaction-level baselines and alternative graph architectures, in line with the research question in Section 1.

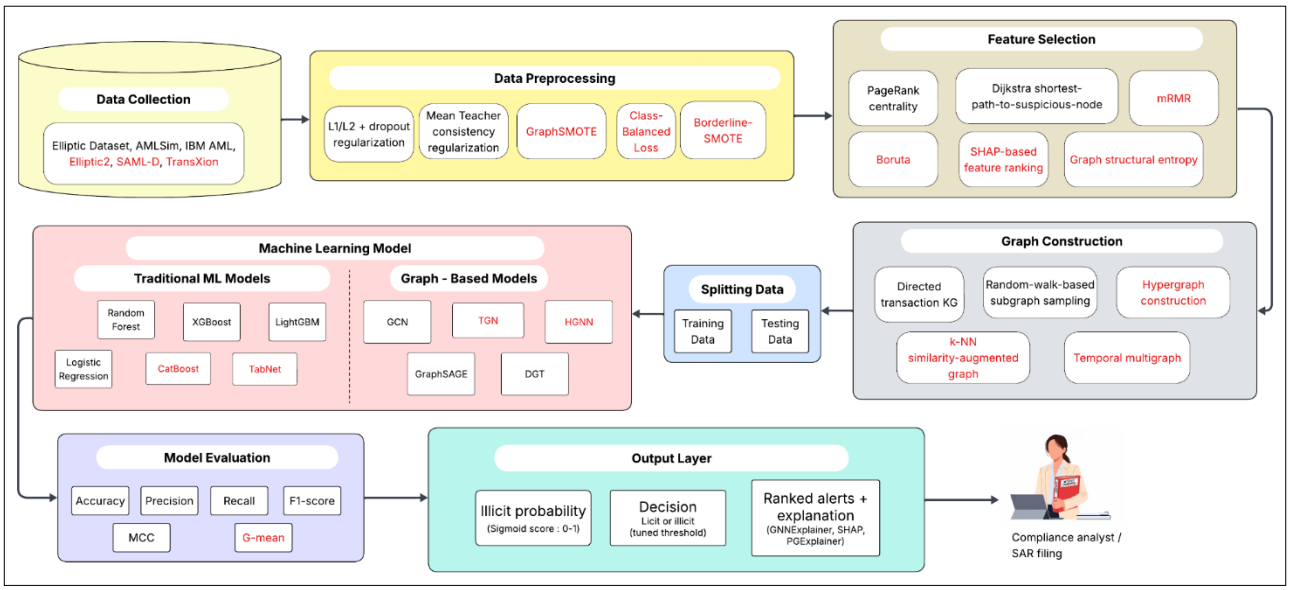


Fig. 1 Architecture of Anti-Money Laundering Detection System

B. *Data Collection*

The primary dataset is the Elliptic Bitcoin Transaction Dataset. It is a directed graph of about 203,769 transactions (nodes) connected by 234,355 payment flows (edges) over 49 discrete time steps. Each node is represented by 166 features, including 94 features that capture information about the transaction itself and 72 aggregated features that describe its one-hop neighbouring nodes. The nodes are classified as illicit (approximately 2%), licit (around 21%), or unknown (the remaining nodes). The significant class imbalance, together with the high proportion of unlabelled nodes, makes this dataset a practical and challenging benchmark for AML research.

The framework is dataset-agnostic, and the following datasets can be added to test generalisation:

- AMLSim and IBM AML, synthetic account-level transaction datasets with known laundering typologies.
- Elliptic2, a subgraph-level dataset in which whole laundering subgraphs are labelled.
- SAML-D and TransXion, further synthetic AML benchmarks.

C. *Data Preprocessing*

Cleaning and normalisation. Only labelled nodes (illicit or licit) contribute to the supervised loss. Unknown nodes stay

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151184

in the graph so that message passing can use their structural context. Missing values are imputed and features are standardised using statistics from the training partition only, to avoid information leakage.

Class-imbalance handling. Illicit transactions are rare, so a model that predicts "licit" for everything can reach high accuracy while detecting no laundering. The framework supports complementary strategies at the data and loss levels:

- Class-balanced loss: each class is weighted by the inverse of its effective number of samples,

$$w_c = \frac{1-\beta}{1-\beta^{n_c}} \qquad (1)$$

  where n_c is the number of training samples of class c and $\beta \in (0,1)$ is a hyperparameter.
- Oversampling: Borderline-SMOTE synthesises minority samples near the decision boundary for the tabular models. GraphSMOTE does the same for graph models by generating synthetic minority nodes in the embedding space and predicting their edges.
- Leakage control: all oversampling is applied to the training partition only, after splitting.

Regularisation. To limit overfitting on the small labelled set, the graph models use L1/L2 weight penalties and dropout between layers. Where the large unlabelled portion should be exploited, Mean Teacher consistency regularisation is added. A teacher network, whose weights are an exponential moving average of the student's,

$$\theta'_t = \alpha\,\theta'_{t-1} + (1-\alpha)\,\theta_t \qquad (2)$$

produces target predictions on unlabelled nodes. A consistency loss penalises disagreement between student and teacher under input perturbation, so the unlabelled nodes also contribute a training signal.

D. *Feature Selection*

Feature selection reduces redundancy among the 166 features and adds graph-derived signals that raw transaction attributes do not capture. Two groups of methods are used.

Graph-structural features capture a node's position in the network:

- PageRank centrality measures how much flow passes through a node, which helps identify aggregation or pass-through accounts.

Shortest-path distance to a node. Dijkstra's algorithm calculates the shortest-path distance, for each node to the known illicit node showing how close each node is to confirmed illicit activity.

- Graph structural entropy measures the diversity of a node's local connectivity (extended module).

Statistical and model-based selection is applied to the combined feature set:

- mRMR (minimum redundancy, maximum relevance) selects features strongly related to the label but weakly related to each other.
- Boruta keeps features that outperform randomised "shadow" features in a Random Forest.
- SHAP-based ranking orders features by mean absolute Shapley value from a tree-based model.

Selection is fitted on training data only. The selected subset is then used by all models, so any performance difference reflects the model rather than the inputs.

E. *Graph Construction*

The transaction network is a directed graph G = (V, E, X), where V is the set of transactions,
E ⊆ V × V is the set of payment flows (an edge u → v meaning an output of transaction u is spent as an input of v), and

$X \in R^{|V|\times d}$ is the node feature matrix.

Graph definition:

$$G = (V, E, X), \quad E \subseteq V \times V, \quad X \in R^{|V|\times d} \qquad (3)$$

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.151184

Directionality is preserved because it indicates fund flow, which matters for layering and circular-flow patterns. The core construction is the directed transaction graph. Because full-graph training is costly at scale, random-walk-based subgraph sampling is used to draw mini-batches of local neighbourhoods.

Optional extensions enrich the structure:

- Hypergraph construction groups transactions that share a common entity or pattern into a single hyperedge, capturing multi-party relationships that pairwise edges cannot.
- k-NN similarity-augmented graph adds edges between nodes with similar features, which helps sparsely connected nodes and follows similarity-based transaction networks such as TRNNS in MLaD².
- Temporal multigraph keeps multiple time-stamped edges between the same pair of nodes so that timing and transaction frequency are preserved.

F. *Data Splitting*

Elliptic is time-ordered, so a random split would let the model train on future information. A temporal split is used instead, following Weber et al. [13]. The earlier time steps (1–34) are used for training and the later ones (35–49) for testing. A validation subset is held out from the training period for hyperparameter tuning and threshold selection. This setup also tests robustness to the concept drift expected in real AML, where laundering behaviour evolves.

G. *Machine Learning Models*

Traditional models (transaction-level baselines). These use node features only and ignore edges:

- Logistic Regression
- Random Forest
- XGBoost
- LightGBM
- CatBoost
- TabNet (attention-based deep tabular model)

Graph-based models. These use features together with graph structure:

- GCN (proposed core model). Each layer aggregates normalised neighbour features, GCN layer propagation rule :

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\,\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right), \quad H^{(0)} = X \qquad (4)$$

where $H^{(l)}$ is the node representation at layer l, $W^{(l)}$ the learnable weights, and σ an activation function. Here $\tilde{A} = A + I$ is the adjacency matrix with self-loops, $\tilde{D}$ is its degree matrix, and σ is a non-linearity (ReLU). A final layer with a sigmoid produces the illicit probability of each node. Stacking L layers lets each node use information from its L-hop neighbourhood.

- GraphSAGE learns an aggregation function over sampled neighbours, enabling inductive learning on unseen nodes.
- DGT (dynamic graph transformer) applies attention over temporal graph snapshots.
- TGN (temporal graph network) keeps memory for each node, so it captures event-based temporal behaviour.
- HGNN (heterogeneous or hypergraph GNN) models different node or relation types or hyperedges.

Training setup. All models are trained with optimizer, e.g. Adam, learning rate, epochs, hidden dimension, layers, and dropout, with early stopping on validation AUC-PR. Hyperparameters are tuned on the validation set only.

H. *Model Evaluation*

Accuracy is misleading on imbalanced data, so evaluation focuses on the illicit (minority) class. With TP, FP, TN, FN taken for the illicit class:

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.151184

- $Precision = \frac{TP}{TP+FP}$ (5)
- $Recall = \frac{TP}{TP+FN}$ (6)
- $F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision+Recall}$ (7)
- $MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$ (8)
- $G - mean = \sqrt{Recall \times Specificity}$ (9)
- $Specificity = \frac{TN}{TN+FP}$ (10)
- AUC-PR:
  The area under the precision–recall curve, which is more informative than ROC-AUC under heavy class imbalance
- $AUC - PR = \int_0^1 Precision(r)\, dr$ (11)
- $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$ (12)

All models are compared on the same temporal test set.

I. *Output Layer and Interpretability*

For each transaction the performing model gives three results:

1. Illicit probability: a score between 0 and 1 calculated using a sigmoid function. This value shows how likely it is that the transaction is illegal.

2. Decision: if the illicit probability is greater than or equal to a threshold τ the transaction is marked as suspicious. The threshold τ is not fixed at 0.5. Instead it is adjusted during validation to get the balance between precision and recall. In -money laundering (AML) , missing a money-laundering case (false negative) and creating extra work for investigators (false positive) come with different costs. That's why we tune τ carefully.

3. Ranked alerts with explanation: transactions flagged as suspicious are sorted by their probability. Each one gets an explanation. We use GNNExplainer or PGExplainer to highlight edges and subgraphs in the network. We also use SHAP to show which features had the impact on the score. Suspicious parts of the network are shown visually so analysts can see the flagged transaction, in its context.

These ranked and explained alerts go directly to the compliance analyst. The analyst reviews them. Decides whether to file a Suspicious Activity Report (SAR). The model helps the analyst. It does not take over the decision-making process. Human judgment remains essential.

J. *Mathematical Formulation of the Proposed System*

The complete pipeline described in Sections B–I can be written as a single composite function that maps raw transaction data to an illicit-transaction decision. Let $D = (V, E, X, y)$ denote the raw data, where V is the set of transactions, $E \subseteq V \times V$ the directed payment flows, $X \in R^{|V| \times d}$ the node feature matrix, and y the (partial) labels. The proposed system is then defined as

DOI: 10.17148/IJARCCE.2026.151184

$$\hat{y}_v = 1\left[\sigma(f_{\theta^*}(\tilde{X}, \tilde{A})_v) \geq \tau^*\right], \qquad \tilde{X} = S\big(P(X)\big), \quad \tilde{A} = G\big(E, \tilde{X}\big) \qquad (13)$$

where:

$\hat{y}_v \in \{0,1\}$ is the predicted label of transaction v (1 = illicit, 0 = licit), and $1[\cdot]$ is the indicator function.

$P(\cdot)$ is the preprocessing operator: normalisation and class-imbalance handling (Section C).

$S(\cdot)$ is the feature-selection operator: it keeps the selected features and appends graph-structural features such as PageRank and shortest-path distance (Section D).

$G(\cdot)$ is the graph-construction operator: it produces the (normalised) adjacency structure $\tilde{A}$ from the directed transaction graph, with optional k-NN, hypergraph, or temporal extensions (Section E).

$f_{\theta^*}(\cdot)$ is the trained graph model (GCN as the core model, Section G), and $\sigma(\cdot)$ is the sigmoid function that converts its output score into the illicit probability $p_v = \sigma\left(f_{\theta^*}(\tilde{X}, \tilde{A})_v\right)$.

θ^* and τ^* are the optimal model parameters and decision threshold, obtained as follows.

The model parameters minimise a class-weighted classification loss over the labelled training nodes V_L, plus the consistency and regularisation terms:

$$\theta^* = \underset{\theta}{argmin} \sum_{v \in V_L}^{v} w_{y_v}\, \ell(y_v, p_v) + \lambda_c\, L_{cons}(\theta) + \lambda_r\, \Omega(\theta) \qquad (14)$$

where $\ell(\cdot)$ is the binary cross-entropy loss, w_{y_v} is the class-balanced weight from (1), L_{cons} is the Mean Teacher consistency loss from (3), $\Omega(\theta)$ is the L1/L2 weight penalty, and λ_c and λ_r are weighting coefficients.

The decision threshold is tuned on the validation set V_{val} so that detection quality is maximised, for example using the F1-score:

$$\tau^* = \underset{\tau \in [0,1]}{arg\,max}\ F_1(y_{val},\ 1[p_{val} \geq \tau]) \qquad (15)$$

Finally, the flagged transactions are ranked by p_v and passed to the compliance analyst together with an explanation, where $\Phi(\cdot)$ denotes the explainer (GNNExplainer, PGExplainer, or SHAP):

$$E_v = \Phi\big(f_{\theta^*}, \tilde{X}, \tilde{A}, v\big) \qquad (16)$$

Equations (14)-(17) together define the whole system: data are pre-processed, selected, and turned into a graph; the graph model is trained with an imbalance-aware objective; and its probabilities are thresholded, ranked, and explained for SAR decisions.

CONCLUSION

This paper reviewed recent graph-based approaches to Anti-Money Laundering (AML) detection and proposed a conceptual framework for detecting illicit transactions in financial transaction networks. The reviewed literature shows that Graph Neural Networks, including GCN, GraphSAGE, GAT, heterogeneous GNNs, and graph transformers, can capture relational patterns such as layering, smurfing, and circular fund movement that transaction-level models treat as independent observations. The same studies show recurring challenges: severe class imbalance, scarce labelled data, evolving laundering behaviour, scalability, and limited interpretability of flagged activity. To address these, the proposed framework combines imbalance-aware preprocessing, graph-structural feature selection, flexible graph construction, temporal data splitting, and a comparison of traditional and graph-based models. It ends in an explainable output layer that supports compliance analysts in SAR filing. Because this work is a review and conceptual design, no experimental claims are made about the relative performance of the models. Future work will implement and empirically evaluate the

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.151184

framework, with GCN as the core model, on the Elliptic dataset and additional AML benchmarks, and will report the results in a separate implementation paper that builds on this one.

REFERENCES

[1] M. R. Karim, F. Hermsen, S. A. Chala, P. De Perthuis and A. Mandal, "Scalable Semi-Supervised Graph Learning Techniques for Anti Money Laundering," in IEEE Access, vol. 12, pp. 50012-50029, 2024, doi: 10.1109/ACCESS.2024.3383784

[2] F. Johannessen and M. Jullum, “Finding Money Launderers Using Heterogeneous Graph Neural Networks,” The Journal of Finance and Data Science, vol. 11, Art. no. 100175, 2025, doi: 10.1016/j.jfds.2025.100175.

[3] S. Ferretti, G. D’Angelo, and V. Ghini, “Enhancing Anti-Money Laundering Frameworks: An Application of Graph Neural Networks in Cryptocurrency Transaction Classification,” IEEE Access, vol. 13, pp. 50201–50215, 2025, doi: 10.1109/ACCESS.2025.3552240.

[4] Kiwhan Song, Mohamed Ali Dhraief, Muhua Xu, Locke Cai, Xuhao Chen, Arvind Mithal, and Jie Chen. 2024. Identifying Money Laundering Subgraphs on the Blockchain. In Proceedings of the 5th ACM International Conference on AI in Finance (ICAIF '24). Association for Computing Machinery, New York, NY, USA, 195–203. https://doi.org/10.1145/3677052.3698635.

[5] S. Marasi and S. Ferretti, "Anti-Money Laundering in Cryptocurrencies Through Graph Neural Networks: A Comparative Study," 2024 IEEE 21st Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 2024, pp. 272-277, doi: 10.1109/CCNC51664.2024.10454631.

[6] J. Schmidt, D. Pasadakis, M. Sathe and O. Schenk, "GAMLNet: a graph based framework for the detection of money laundering," 2024 11th IEEE Swiss Conference on Data Science (SDS), Zurich, Switzerland, 2024, pp. 241-245, doi: 10.1109/SDS60720.2024.00043.

[7] Junhong Lin, Xiaojie Guo, Yada Zhu, Samuel Mitchell, Erik Altman, and Julian Shun. 2024. FraudGT: A Simple, Effective, and Efficient Graph Transformer for Financial Fraud Detection. In Proceedings of the 5th ACM International Conference on AI in Finance (ICAIF '24). Association for Computing Machinery, New York, NY, USA, 292–300. https://doi.org/10.1145/3677052.3698648.

[8] X. Luo, X. Han, W. Zuo, X. Wu and W. Liu, "MLaD²: A Semi-Supervised Money Laundering Detection Framework Based on Decoupling Training," in IEEE Transactions on Information Forensics and Security, vol. 19, pp. 4518-4533, 2024, doi: 10.1109/TIFS.2024.3380262.

[9] Irshad, Faizan & Alkhalifah, Tamim & Alturise, Fahad & Khan, Yaser. (2024). GCF-MLD: Integrated Approach for Money Laundering Detection Using Machine Learning and Graph Network Analysis. IEEE Access. 12. 183961-183972. 10.1109/ACCESS.2024.3510115.

[10] Wai Weng Lo, Gayan K. Kulatilleke, Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. 2023. Inspection-L: self-supervised GNN node embeddings for money laundering detection in bitcoin. Applied Intelligence 53, 16 (Aug 2023), 19406–19417. https://doi.org/10.1007/s10489-023-04504-9.

[11] Xujia Li, Yuan Li, Xueying Mo, Hebing Xiao, Yanyan Shen, and Lei Chen. 2023. Diga: Guided Diffusion Model for Graph Recovery in Anti-Money Laundering. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23). Association for Computing Machinery, New York, NY, USA, 4404–4413. https://doi.org/10.1145/3580305.3599806.

 This work is licensed under a Creative Commons Attribution 4.0 International License