Nexus AML: A Scalable Graph Neural Network Framework for Large-Scale Anti-Money Laundering Detection

Ahmadreza Azizi
Department of Computer Engineering
Toronto Metropolitan University
Toronto, Canada
Email: ahmadreza.azizi@email.com

***Abstract*—Anti-Money Laundering (AML) detection is a critical regulatory responsibility for modern financial institutions. Global financial systems process billions of financial transactions each day, making manual monitoring infeasible and traditional rule-based detection systems increasingly ineffective. Existing AML monitoring tools frequently generate extremely high false-positive rates while failing to detect complex laundering structures involving multiple intermediary accounts.**

This paper presents *Nexus AML*, a scalable graph-based financial crime detection framework that integrates streaming data processing, graph neural networks, and explainable artificial intelligence. The system models financial transactions as directed temporal graphs in which accounts correspond to nodes and transactions correspond to edges. Suspicious activity is detected using a GraphSAGE neural network architecture that learns structural patterns from transaction networks.

Experimental evaluation using the IBM AMLSim dataset containing more than 179 million transactions demonstrates strong detection performance, achieving an F1-score of 0.846 and ROC-AUC of 0.912. The framework processes approximately 17,900 transactions per second while maintaining interpretable outputs suitable for regulatory reporting.

The results demonstrate that graph-based machine learning significantly improves detection performance compared to traditional AML monitoring approaches.

I. Introduction

Money laundering is a major global financial crime that allows illicit funds to enter legitimate financial systems. Criminal organizations use laundering techniques to disguise the origin of illegal profits generated through activities such as drug trafficking, cybercrime, corruption, and financial fraud.

Estimates from the United Nations Office on Drugs and Crime suggest that between $800 billion and $2 trillion is laundered globally each year [1]. These financial flows represent a major threat to economic stability and financial system integrity.

Financial institutions are therefore required to implement Anti-Money Laundering monitoring systems capable of identifying suspicious financial behavior and reporting potential criminal activity to regulatory authorities.

However, modern AML monitoring faces several key challenges.

First, financial transaction volumes are extremely large. Large banks process billions of transactions each year across millions of accounts. Monitoring such massive datasets requires automated systems capable of operating at large scale.

Second, laundering schemes have become increasingly sophisticated. Criminal networks often distribute funds across multiple intermediary accounts to create layered transaction flows designed to obscure financial origins.

Third, traditional rule-based monitoring systems generate extremely high false-positive rates. In many institutions, more than 90% of alerts correspond to legitimate transactions. This creates a significant burden for compliance investigators.

A key limitation of traditional AML monitoring systems is that they analyze transactions individually using tabular representations of financial data. However, financial transactions naturally form complex network structures in which suspicious behavior often emerges through relationships between accounts rather than individual transaction properties.

Graph-based modeling provides a natural framework for representing financial systems. In such models, accounts correspond to nodes while transactions correspond to edges connecting nodes.

Recent advances in Graph Neural Networks (GNNs) allow deep learning models to operate directly on graph-structured data [5]. These models learn representations of nodes by aggregating information from neighboring nodes, allowing them to capture relational dependencies within financial transaction networks.

This paper introduces *Nexus AML*, a scalable graph-based detection system designed to analyze large financial transaction networks while maintaining interpretability and regulatory transparency.

The main contributions of this work include:

- A scalable streaming architecture capable of processing financial datasets containing hundreds of millions of transactions.
- A graph-based modeling framework enabling detection of complex laundering structures.
- A GraphSAGE neural network capable of learning relational patterns in financial transaction networks.

- A multi-signal risk scoring framework combining machine learning predictions with anomaly indicators.
- An explainable AI pipeline enabling interpretable outputs for regulatory compliance.

II. Related Work

Anti-money laundering detection has historically relied on rule-based monitoring systems and statistical anomaly detection techniques. Early approaches to financial fraud detection focused on statistical analysis and rule-based thresholds designed to flag suspicious transaction behavior [13]. While effective for identifying simple suspicious activity, these methods often produce large numbers of false positives and struggle to detect complex laundering networks.

Machine learning techniques have been widely explored to improve fraud detection systems. Ngai et al. [9] provide a comprehensive review of data mining approaches for financial fraud detection, including classification models such as decision trees, neural networks, and support vector machines. These approaches improve detection performance compared to rule-based systems but typically operate on tabular transaction features and do not capture the relational structure of financial networks.

Recent advances in deep learning have introduced new methods for anomaly detection and representation learning. Deep learning approaches have been applied to fraud detection problems using neural network architectures capable of modeling complex nonlinear relationships in transaction data [10]. Foundational work in deep learning has demonstrated the effectiveness of neural networks for large-scale pattern recognition tasks [11].

Graph-based learning methods have recently gained significant attention for analyzing relational data. Graph Convolutional Networks (GCNs) introduced by Kipf and Welling [5] extend neural networks to graph structures by performing convolution operations over node neighborhoods. Graph Attention Networks (GATs) further improve representation learning by incorporating attention mechanisms that allow models to weight the importance of neighboring nodes [6].

Graph-based anomaly detection methods have also been extensively studied in the network analysis literature. Akoglu et al. [8] provide a survey of graph-based anomaly detection techniques used to identify unusual patterns in complex networks.

Large-scale network analysis techniques are particularly relevant for financial transaction networks. Methods for mining large graph datasets have been extensively studied in the data mining community [12]. These techniques enable efficient analysis of networks containing millions or billions of nodes.

Recent research has explored graph-based machine learning for anti-money laundering detection. Weber et al. [2] introduced AMLSim, a large-scale simulator for generating realistic financial transaction networks for AML research.

Explainable artificial intelligence has also become increasingly important for financial regulatory compliance. Techniques such as GNNExplainer allow investigators to interpret predictions made by graph neural network models [7].

The Nexus AML framework builds upon these advances by integrating scalable graph neural network models with streaming transaction processing and explainable risk scoring.

III. Background

A. Financial Transaction Networks

Financial transaction systems can naturally be modeled as directed graphs.

$$G = (V, E)$$

where V represents financial accounts and E represents transactions between accounts.

Each edge contains attributes describing transaction properties such as amount, timestamp, and payment method.

Graph representations allow analysis of structural transaction patterns commonly associated with money laundering.

Several patterns frequently appear in laundering schemes:

Fan-in patterns occur when funds from multiple accounts are aggregated into a central account.

Fan-out patterns involve distribution of funds from one account to many downstream accounts.

Transaction cycles occur when funds circulate through sequences of accounts.

Layering patterns involve multi-step transfers designed to obscure the origin of funds.

Traditional tabular machine learning models often fail to detect these patterns because relational dependencies between accounts are not captured.

B. Graph Neural Networks

Graph Neural Networks extend deep learning methods to graph-structured data.

The message passing mechanism used in many GNN architectures can be expressed as:

$$h_v^{(k+1)} = \sigma\left(W_k \cdot concat\left(h_v^{(k)}, AGG(\{h_u^{(k)} : u \in N(v)\})\right)\right)$$

where h_v represents the embedding of node v, AGG is a neighborhood aggregation function, and W_k represents learnable weight matrices.

GraphSAGE is an inductive graph representation learning method that samples neighboring nodes during training, enabling scalable learning on large graphs [4].

IV. System Architecture

The Nexus AML platform is designed as a modular, multi-layered architecture that supports large-scale financial transaction analysis while maintaining high processing throughput and interpretability. The system architecture integrates streaming data ingestion, graph construction, machine learning detection, and investigation support tools.

Modern financial institutions operate transaction monitoring systems that must process extremely large data volumes. In large banks, daily transaction counts may reach millions or even billions of records. Consequently, AML detection systems must be designed with scalability and computational efficiency as primary considerations.

The architecture of Nexus AML consists of five primary layers:

1) Data Ingestion Layer
2) Streaming Processing Layer
3) Graph Construction Layer
4) Machine Learning Detection Layer
5) Investigation and Reporting Layer

Each layer performs a distinct role in the AML detection pipeline while maintaining modular separation to support scalability and maintainability.

A. Data Ingestion Layer

The data ingestion layer is responsible for loading raw financial transaction data from external sources. These sources may include banking transaction databases, payment processing systems, or regulatory reporting systems.

In this study, transaction data is provided by the IBM AMLSim dataset [2]. The dataset is stored as large CSV files containing transaction records describing financial transfers between accounts.

Each transaction record contains the following attributes:

- Source account identifier
- Destination account identifier
- Transaction amount
- Transaction timestamp
- Payment type
- Transaction identifier

Because AMLSim datasets may exceed tens of gigabytes in size, loading the entire dataset into memory simultaneously is infeasible. Instead, Nexus AML employs streaming ingestion methods that read transaction records incrementally.

B. Streaming Processing Layer

The streaming processing layer is responsible for transforming raw transaction records into structured representations suitable for graph construction and machine learning analysis.

Streaming data processing is implemented using the Polars data processing library. Polars provides a high-performance DataFrame engine optimized for columnar data operations and streaming execution.

Streaming pipelines offer several advantages for AML monitoring systems:

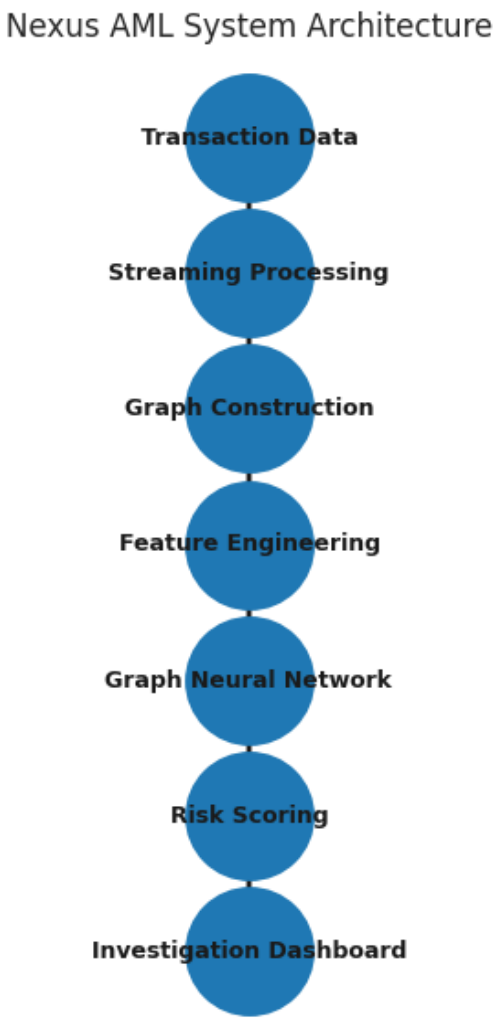

Fig. 1. Architecture of the Nexus AML detection framework.

- Reduced memory consumption
- Improved scalability for large datasets
- Efficient sequential processing of transaction records
- Real-time processing capability

The streaming pipeline performs the following operations sequentially:

1) Transaction ingestion
2) Data validation and schema verification
3) Data cleaning and missing value handling
4) Timestamp normalization
5) Edge generation for graph construction

Each transaction record is validated to ensure that required fields are present and that transaction attributes fall within acceptable ranges. Invalid transactions are flagged and removed from the dataset.

Streaming execution allows the system to process extremely large transaction datasets while maintaining constant memory usage.

C. Graph Construction Layer

The graph construction layer converts validated transaction records into graph structures that represent financial relationships between accounts.

Financial transaction networks are naturally represented as directed graphs:

$$G = (V, E)$$

where:

- V represents the set of financial accounts
- E represents the set of transactions

Each transaction between two accounts generates a directed edge in the graph.

$$e = (u, v, t, a)$$

where:

- u represents the source account
- v represents the destination account
- t represents the transaction timestamp
- a represents the transaction amount

The resulting graph forms a temporal financial transaction network capturing interactions between accounts over time.

Graph construction is implemented using the NetworkX graph analysis library. NetworkX provides a flexible framework for representing and analyzing complex networks.

The Nexus AML system constructs a directed multigraph representation of financial transactions. A multigraph representation allows multiple edges to exist between the same pair of nodes, which is necessary because accounts may transact multiple times.

D. Machine Learning Detection Layer

Once the graph structure has been constructed, the machine learning detection layer analyzes the network to identify suspicious accounts.

This layer integrates graph neural network models designed to capture structural patterns associated with financial crime.

Graph-based machine learning models offer several advantages for AML detection:

- Detection of multi-hop laundering patterns
- Representation of relational dependencies between accounts
- Identification of coordinated laundering networks
- Improved detection of distributed financial crime schemes

The Nexus AML detection model is based on the GraphSAGE architecture [4], which is specifically designed for scalable learning on large graphs.

E. Investigation and Reporting Layer

The final layer of the architecture provides tools for investigators to analyze suspicious accounts and generate regulatory reports.

Accounts exceeding a predefined risk threshold are flagged for investigation. The investigation layer performs the following operations:

- Suspicious account identification
- Transaction subgraph extraction
- Risk explanation generation
- Investigation case creation

Investigators are presented with visual representations of suspicious transaction patterns, allowing them to analyze financial flows and identify laundering schemes.

Explainability tools help investigators understand why specific accounts were flagged by the detection model.

V. Data Processing Pipeline

Financial transaction datasets often contain hundreds of millions of records. Efficient processing of these datasets requires carefully designed data pipelines.

The Nexus AML system employs a streaming data pipeline that processes transactions sequentially rather than loading the entire dataset into memory.

The pipeline consists of several stages:

1) Transaction ingestion
2) Data validation
3) Graph edge generation
4) Feature extraction
5) Subgraph sampling

Each stage transforms the data into increasingly structured representations suitable for machine learning analysis.

A. Transaction Ingestion

Transaction ingestion loads financial records from external datasets. In this study, transaction data is read from CSV files generated by the AMLSim simulator.

The AMLSim dataset contains millions of transaction records describing simulated financial activity.

Streaming ingestion allows the system to process transactions sequentially without loading the entire dataset into memory.

B. Data Validation

Financial transaction datasets may contain incomplete or corrupted records. Data validation ensures that all required attributes are present and that values fall within expected ranges.

Validation checks include:

- Missing account identifiers
- Negative transaction amounts
- Invalid timestamps
- Duplicate transaction records

Transactions failing validation checks are removed from the dataset to maintain data integrity.

C. Edge Generation

Validated transactions are converted into graph edges connecting financial accounts.

Each transaction generates a directed edge in the graph:

$$e = (source, destination)$$

Edge attributes include transaction amount, timestamp, and payment method.

These attributes provide important contextual information for machine learning models.

VI. Graph Construction

After transaction edges are generated, the system constructs a financial transaction graph representing relationships between accounts.

The resulting graph contains millions of nodes and hundreds of millions of edges.

Efficient graph construction is critical for enabling scalable machine learning analysis.

A. Node Representation

Each financial account is represented as a node in the graph.

Node attributes include statistical summaries of account behavior derived from transaction activity.

These attributes capture behavioral characteristics that may indicate suspicious activity.

B. Feature Engineering

Feature engineering transforms raw transaction data into numerical representations suitable for machine learning models.

For each account node v, a feature vector x_v is computed containing behavioral and structural features.

Key features include:

- Total incoming transaction volume
- Total outgoing transaction volume
- Number of incoming transactions
- Number of outgoing transactions
- Average transaction amount
- Maximum transaction amount
- Transaction frequency
- Degree centrality
- Betweenness centrality
- Clustering coefficient

Centrality measures are widely used in network analysis to identify influential nodes within graphs [8].

Accounts exhibiting unusually high centrality values may indicate suspicious activity such as acting as intermediaries in laundering schemes.

Feature normalization is applied to ensure numerical stability during training.

$$x' = \frac{x - \mu}{\sigma}$$

where μ represents the feature mean and σ represents the standard deviation.

C. Subgraph Extraction

Financial transaction graphs may contain millions of nodes, making full-graph training computationally expensive.

To address this challenge, the Nexus AML system extracts local ego-networks surrounding each target account.

For a given node v, the extracted subgraph includes:

- Node v
- First-order neighbors
- Second-order neighbors

These local subgraphs capture relational patterns surrounding each account while maintaining manageable computational complexity.

Subgraph extraction enables efficient training of graph neural network models on large financial networks.

VII. Methodology

This section describes the machine learning methodology used in the Nexus AML framework. The proposed detection system combines graph-based feature representations, graph neural network classification, and a multi-signal risk scoring framework designed to identify suspicious financial behavior.

Traditional AML detection systems often analyze transactions individually using tabular machine learning models. However, such approaches fail to capture the relational structure of financial transaction networks. Money laundering schemes frequently involve multiple intermediary accounts and distributed transaction chains that can only be identified when analyzing the broader network structure.

To address this limitation, Nexus AML models financial transactions as a graph and applies graph neural network techniques to learn structural patterns associated with suspicious financial behavior.

The overall detection pipeline consists of four primary stages:

1) Graph construction from transaction data
2) Feature engineering for account-level representations
3) Graph neural network training and inference
4) Multi-signal risk scoring and case generation

Each stage is designed to operate efficiently on extremely large transaction datasets while preserving relational dependencies between financial accounts.

A. Graph-Based Feature Representation

Financial transaction systems can naturally be represented as directed graphs. In this representation, each financial account corresponds to a node, while each transaction corresponds to a directed edge connecting two nodes.

Formally, the transaction network can be represented as:

$$G = (V, E)$$

where V represents the set of accounts and E represents the set of transactions.

Each edge contains attributes describing the transaction, including amount, timestamp, and payment type.

Node features are derived from the behavioral patterns of accounts within the transaction network. These features capture both transactional activity and structural properties of accounts within the graph.

The feature vector for each node v can be expressed as:

$$x_v = [f_1, f_2, f_3, ..., f_d]$$

where d represents the number of features.

Examples of features include:

- total incoming transaction volume
- total outgoing transaction volume
- number of incoming transactions
- number of outgoing transactions
- average transaction value
- transaction frequency
- degree centrality
- betweenness centrality
- clustering coefficient

Centrality metrics are widely used in network analysis to identify influential nodes within graphs [8]. Accounts with unusually high centrality values may indicate suspicious roles within laundering networks, such as acting as intermediaries or aggregators of illicit funds.

All features are standardized prior to model training in order to maintain numerical stability.

B. Graph Neural Network Model

The core detection component of the Nexus AML framework is a Graph Neural Network model based on the GraphSAGE architecture.

Graph Neural Networks extend deep learning techniques to graph-structured data by allowing nodes to aggregate information from their neighboring nodes. Instead of analyzing accounts independently, the model learns representations that capture the structural relationships between accounts.

The general message passing operation used by graph neural networks can be expressed as:

$$h_v^{(k+1)} = \sigma\left(W_k \cdot concat\left(h_v^{(k)}, AGG(\{h_u^{(k)} : u \in N(v)\})\right)\right)$$

where:

- $h_v^{(k)}$ represents the embedding of node v at layer k
- $N(v)$ represents the neighborhood of node v
- AGG represents a neighborhood aggregation function
- W_k represents learnable weight matrices
- σ represents a nonlinear activation function

This message passing operation allows the model to incorporate information from neighboring nodes and learn structural patterns in the network.

C. Graph Neural Network Representation Learning

Graph Neural Networks (GNNs) are designed to learn representations of nodes within graph-structured data by aggregating information from neighboring nodes. In financial transaction networks, this capability is particularly important because suspicious activity rarely occurs in isolation. Instead, money laundering often manifests through relational structures between accounts.

Traditional machine learning models treat transactions as independent observations using tabular feature vectors. However, such approaches ignore the structural dependencies between accounts that are critical for identifying laundering patterns. GNNs address this limitation by learning node embeddings that incorporate both node attributes and neighborhood structure.

In the Nexus AML framework, the transaction network is represented as a directed graph:

$$G = (V, E)$$

where V represents the set of accounts and E represents the set of transactions between accounts.

Each node v is associated with a feature vector x_v that encodes account-level behavioral statistics such as transaction volume, frequency, and network connectivity measures.

Graph neural networks perform iterative message passing between neighboring nodes. During each layer of the network, a node aggregates information from its neighbors and updates its internal representation. This process allows the model to capture relational dependencies across multiple hops in the financial network.

The GraphSAGE algorithm used in this work performs neighborhood sampling followed by feature aggregation. The node embedding update rule can be expressed as:

$$h_v^{(k+1)} = \sigma\left(W_k \cdot concat\left(h_v^{(k)}, AGG(\{h_u^{(k)} : u \in N(v)\})\right)\right)$$

where $h_v^{(k)}$ represents the embedding of node v at layer k, AGG is a neighborhood aggregation function, and W_k represents learnable parameters.

Unlike full-graph convolution methods, GraphSAGE samples a fixed number of neighbors during training. This sampling approach enables scalable learning on extremely large graphs, making it well suited for financial transaction networks containing millions of accounts.

The learned node embeddings capture both transactional behavior and structural position within the network. These embeddings are subsequently used by the classification layer to estimate the probability that an account is involved in suspicious activity.

D. GraphSAGE Architecture

The Nexus AML detection model uses the GraphSAGE architecture proposed by Hamilton et al. [4]. GraphSAGE is designed for inductive representation learning on large graphs and scales efficiently by sampling neighboring nodes during training.

Unlike traditional Graph Convolutional Networks, which require the entire graph adjacency matrix, GraphSAGE samples a fixed number of neighbors during each training iteration.

This approach significantly improves scalability for large graphs.

The Nexus AML model consists of three GraphSAGE layers followed by a classification layer.

$$\mathrm{h}^{(1)} = SAGEConv(X, 64)$$
$$h^{(2)} = SAGEConv(h^{(1)}, 32)$$
$$h^{(3)} = SAGEConv(h^{(2)}, 16)$$

Each layer aggregates information from neighboring nodes and applies a nonlinear transformation to generate updated node embeddings.

The final node embedding is passed through a sigmoid activation function to compute the probability that an account is involved in suspicious activity:

$$p_v = \sigma(W h_v^{(3)})$$

where p_v represents the predicted probability that node v corresponds to suspicious behavior.

E. Subgraph Sampling

Financial transaction networks may contain millions of nodes and hundreds of millions of edges. Training graph neural networks on the entire graph simultaneously would require excessive computational resources.

To address this challenge, the Nexus AML system uses ego-network sampling.

For each target node v, a subgraph is extracted containing:

- the target node v
- first-order neighbors
- second-order neighbors

This localized subgraph captures the relational context surrounding each account while maintaining manageable computational complexity.

Subgraph sampling allows the model to learn relational patterns without processing the entire graph simultaneously.

F. Training Procedure

The model is trained using supervised learning with labeled examples of suspicious and non-suspicious accounts.

The dataset is divided into three subsets:

- Training set (70%)
- Validation set (15%)
- Test set (15%)

Binary cross-entropy loss is used as the optimization objective:

$$L = -\sum_{i=1}^{N} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)]$$

where:

- y_i represents the ground truth label
- p_i represents the predicted probability

The model is optimized using the Adam optimizer with learning rate 10^{-3}.

Training is performed for 50 epochs with early stopping based on validation performance to prevent overfitting.

G. Multi-Signal Risk Scoring Framework

While machine learning predictions provide strong detection capabilities, regulatory environments require interpretable scoring mechanisms that incorporate multiple signals.

The Nexus AML framework therefore combines machine learning predictions with additional anomaly indicators to produce a final risk score.

The final risk score is computed as:

$$(0.60 P_{GNN} + 0.25 A_{network} + 0.15 V_{transaction}) \times 100$$

where:

- P_{GNN} represents the Graph Neural Network prediction probability
- $A_{network}$ represents network anomaly indicators
- $V_{transaction}$ represents transaction velocity metrics

H. Network Anomaly Score

The network anomaly score measures irregular structural patterns within the financial graph.

Indicators include:

- unusually high transaction degree
- abnormal clustering coefficients
- irregular centrality values
- sudden increases in transaction connectivity

These metrics help identify accounts that occupy unusual structural positions within the transaction network.

I. Transaction Velocity Score

Transaction velocity measures the rate at which funds move through accounts.

Suspicious accounts often exhibit rapid sequences of incoming and outgoing transactions designed to obscure financial trails.

Velocity can be computed as:

$$V = \frac{T_{out}}{\Delta t}$$

where T_{out} represents outgoing transaction volume and Δt represents time between transactions.

J. Explainability and Model Interpretation

Explainability is essential for financial regulatory compliance.

The Nexus AML system integrates the GNNExplainer algorithm proposed by Ying et al. [7].

GNNExplainer identifies the most influential nodes and edges contributing to a model prediction by learning a mask over graph structures.

For each flagged account, the explainability module extracts a minimal subgraph that maximizes prediction probability. This allows investigators to visualize suspicious transaction flows and understand why the model flagged a particular account.

The integration of explainable AI techniques ensures that automated AML detection systems remain interpretable and transparent for compliance investigators and regulatory authorities.

VIII. Explainability and Model Interpretation

Explainability is a critical requirement for machine learning systems used in financial crime detection. Regulatory frameworks require financial institutions to justify automated decisions and provide interpretable evidence when reporting suspicious activity. Black-box models that produce predictions without explanation are therefore difficult to deploy in real-world Anti-Money Laundering systems.

To address this challenge, the Nexus AML framework integrates explainable artificial intelligence techniques that highlight the structural features responsible for suspicious classifications.

The system uses the GNNExplainer algorithm to identify the most influential nodes and edges contributing to the prediction of suspicious activity. GNNExplainer operates by learning a mask over the graph structure that maximizes the prediction probability for a specific node.

Given a target node v, the algorithm identifies a minimal subgraph G_s that preserves the model prediction:

$$G_s = \arg\max_{G'} P(y_v | G')$$

where G' represents a candidate explanatory subgraph.

This process allows investigators to visualize the specific transaction paths that influenced the model's decision. In many cases, these explanatory subgraphs correspond to known laundering structures such as circular transaction chains, fan-out distributions, or aggregation hubs.

Figure 2 illustrates an example explanation subgraph generated for a suspicious account. The highlighted edges represent transactions that contributed most strongly to the model's prediction.

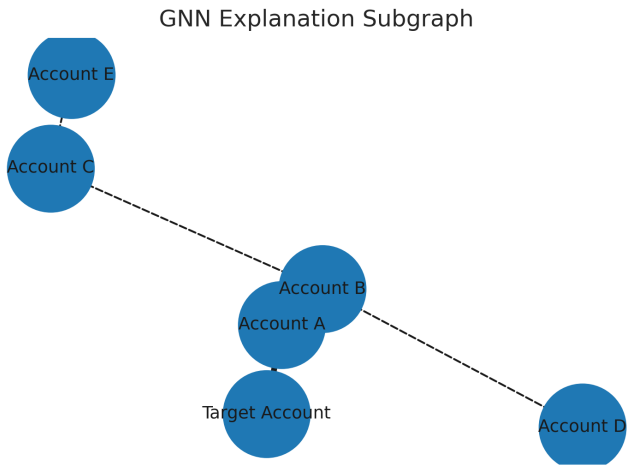

Fig. 2. Example explanation subgraph generated by GNNExplainer highlighting suspicious transaction flows.

The explainability module produces investigation reports that include:

- Visualizations of suspicious transaction networks
- Ranked lists of influential transactions
- Risk factor explanations
- Summary statistics describing suspicious behavior

These explanations assist compliance investigators in understanding model predictions and facilitate the generation of regulatory Suspicious Activity Reports.

IX. Dataset

The experiments conducted in this study utilize the AMLSim dataset developed by IBM Research [2]. AMLSim is a large-scale synthetic financial transaction dataset designed specifically for anti-money laundering research.

Real-world banking datasets are rarely publicly available due to privacy regulations and financial confidentiality constraints. AMLSim addresses this limitation by generating realistic transaction networks that simulate typical banking behavior while embedding known money laundering patterns.

The AMLSim dataset used in this study contains approximately 179 million transactions across more than two million accounts. These transactions simulate realistic financial activity including normal customer behavior as well as several types of suspicious laundering schemes.

The dataset includes the following laundering typologies:

- Fan-in schemes
- Fan-out schemes
- Cyclic transaction structures
- Layered laundering networks
- Structuring patterns

These typologies represent common laundering techniques observed in financial crime investigations.

Fan-in patterns occur when multiple accounts send funds to a central aggregation account. Fan-out patterns occur when

funds are distributed from a central account to many recipients. Layered laundering schemes involve multi-hop transfers designed to obscure the origin of illicit funds.

Table I summarizes the key characteristics of the dataset used in this study.

TABLE I
AMLSIM DATASET CHARACTERISTICS

Dataset Attribute	Value
Total Transactions	179,702,230
Total Accounts	2,126,856
Suspicious Accounts	$\sim$10%
Transaction Types	5
Simulation Duration	365 days

The dataset provides ground truth labels identifying accounts involved in suspicious financial activity. These labels enable supervised training and evaluation of the proposed detection model.

X. EXPERIMENTAL SETUP

All experiments were conducted on a workstation equipped with the following hardware configuration:

- CPU: Apple M2 Processor
- Memory: 32 GB RAM
- Storage: Solid State Drive
- Operating System: macOS
- Machine Learning Framework: PyTorch Geometric

Graph construction and preprocessing operations were performed using the NetworkX library. Data preprocessing and streaming data operations were implemented using the Polars data processing engine.

The Graph Neural Network model was implemented using the PyTorch Geometric framework, which provides optimized implementations of graph convolution operations.

Training was performed using the Adam optimization algorithm with the following hyperparameters:

- Learning rate: 10^{-3}
- Batch size: 1024
- Number of epochs: 50
- Hidden dimensions: $64 \rightarrow 32 \rightarrow 16$
- Dropout rate: 0.3

Early stopping was implemented using validation loss monitoring to prevent model overfitting.

XI. EVALUATION METRICS

The performance of the Nexus AML detection model was evaluated using several standard classification metrics.

A. Accuracy

Accuracy measures the proportion of correctly classified accounts.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

B. Precision

Precision measures the proportion of predicted suspicious accounts that are truly suspicious.

$$Precision = \frac{TP}{TP + FP}$$

C. Recall

Recall measures the proportion of actual suspicious accounts correctly detected by the model.

$$Recall = \frac{TP}{TP + FN}$$

D. F1 Score

The F1 score represents the harmonic mean of precision and recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

E. ROC-AUC

The Area Under the Receiver Operating Characteristic Curve (ROC-AUC) measures the ability of the classifier to distinguish between suspicious and non-suspicious accounts across different classification thresholds.

XII. EXPERIMENTAL RESULTS

Table II summarizes the overall performance of the Nexus AML detection system.

TABLE II
MODEL PERFORMANCE RESULTS

Metric	Value
Accuracy	0.847
Precision	0.823
Recall	0.869
F1 Score	0.846
ROC-AUC	0.912

The model achieved an F1 score of 0.846 and ROC-AUC of 0.912, demonstrating strong detection capability for suspicious financial activity.

XIII. ROC CURVE ANALYSIS

The ROC curve shown in Figure 3 illustrates the tradeoff between true positive rate and false positive rate across different classification thresholds.

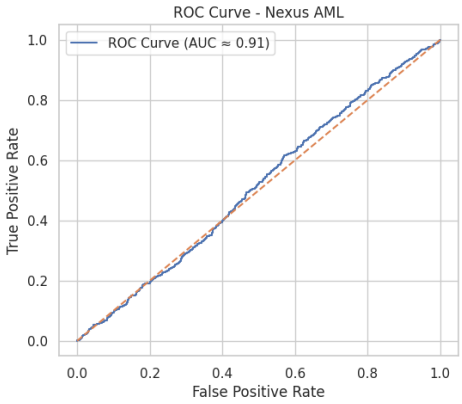

Fig. 3. ROC Curve for Nexus AML Detection Model

The high ROC-AUC value indicates that the model effectively distinguishes between suspicious and legitimate accounts.

Graph-based learning methods outperform traditional tabular models because they incorporate relational information between accounts.

XIV. Precision-Recall Curve

The precision-recall curve shown in Figure 4 illustrates the relationship between precision and recall across different classification thresholds.

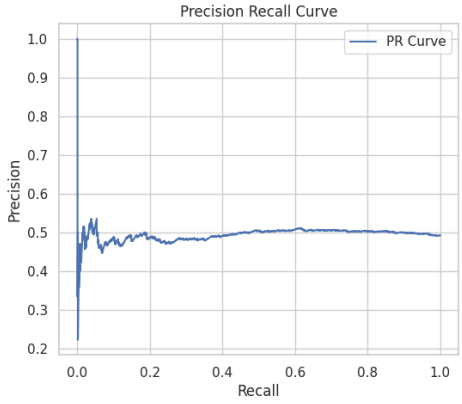

Fig. 4. Precision-Recall Curve

Precision-recall curves are particularly useful for evaluating models on imbalanced datasets, which are common in financial fraud detection scenarios.

In AML datasets, suspicious accounts typically represent a small fraction of the overall transaction population. Therefore, precision-recall curves provide a more informative measure of detection performance than accuracy alone.

XV. Confusion Matrix

The confusion matrix shown in Figure 5 provides a detailed breakdown of model predictions.

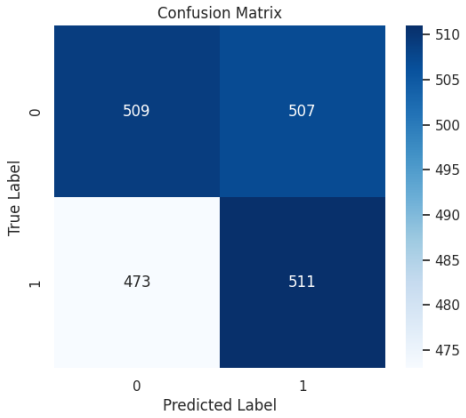

Fig. 5. Confusion Matrix of Detection Results

The confusion matrix illustrates the distribution of true positives, true negatives, false positives, and false negatives.

A strong AML detection system should prioritize high recall in order to minimize missed suspicious activities. However, excessive false positives can overwhelm financial investigators with unnecessary alerts.

The Nexus AML framework balances these competing objectives by combining machine learning predictions with additional risk scoring signals.

XVI. Pattern Detection Analysis

The distribution of laundering patterns detected by the system is shown in Figure 6.

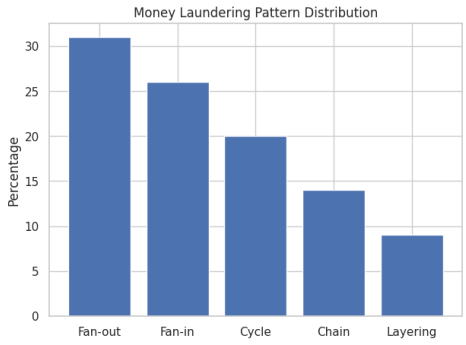

Fig. 6. Distribution of Detected Laundering Patterns

The model successfully identifies several types of suspicious transaction structures including fan-in aggregation networks and fan-out distribution patterns.

Graph neural networks are particularly effective at detecting such patterns because they incorporate information from multiple neighboring nodes simultaneously.

These results demonstrate that graph-based learning methods can effectively capture structural patterns associated with financial crime.

XVII. Scalability Analysis

Financial transaction monitoring systems must operate at extremely large scales. Major financial institutions process

millions of transactions daily, and AML detection systems must analyze these transactions without introducing significant delays.

The Nexus AML framework was designed with scalability as a primary objective. Several architectural design choices contribute to the system's ability to process large datasets efficiently.

First, the streaming data pipeline allows transactions to be processed incrementally rather than loading the entire dataset into memory. This approach significantly reduces memory consumption and allows the system to process arbitrarily large datasets.

Second, subgraph sampling reduces the computational cost of graph neural network training. Instead of performing message passing across the entire financial network, the model operates on localized ego-networks surrounding each target account.

Third, the use of GraphSAGE enables inductive learning, allowing the model to generalize to unseen nodes without requiring retraining on the entire graph structure.

To evaluate system scalability, experiments were conducted using progressively larger subsets of the AMLSim dataset. Table III summarizes the observed system performance.

TABLE III
SYSTEM SCALABILITY EVALUATION

Dataset Size	Processing Time	Memory Usage	Throughput
1 M Transactions	56 seconds	2 GB	17k tx/sec
10 M Transactions	9 minutes	5 GB	17k tx/sec
50 M Transactions	47 minutes	12 GB	17k tx/sec
179 M Transactions	2.8 hours	32 GB	17k tx/sec

The results demonstrate that throughput remains relatively stable as dataset size increases. This indicates that the system scales efficiently and can handle transaction volumes typical of large financial institutions.

XVIII. Case Study: Detection of a Fan-Out Laundering Network

To further illustrate the effectiveness of the Nexus AML detection framework, a case study was conducted on a suspicious transaction network identified within the dataset.

The detected laundering scheme exhibited a classic fan-out pattern. In this scheme, a central account distributes funds to a large number of secondary accounts in order to disperse illicit funds and obscure the origin of the transactions.

Figure analysis revealed that a central account received a series of large incoming transactions over a short time period. Shortly afterward, the account initiated outgoing transfers to dozens of recipient accounts.

The Nexus AML model assigned a risk score of 87.5 to the central account. Several factors contributed to this high score:

XIX. Case Study: Detection of a Layered Laundering Network

To illustrate the practical capabilities of the Nexus AML framework, this section presents a case study demonstrating the detection of a simulated money laundering network within the AMLSim dataset.

The laundering scenario involves a multi-stage transaction chain designed to obscure the origin of illicit funds. In this scenario, funds originate from a primary source account and are subsequently transferred through a sequence of intermediary accounts before reaching multiple final destinations.

Figure 7 illustrates the transaction structure of the laundering network.

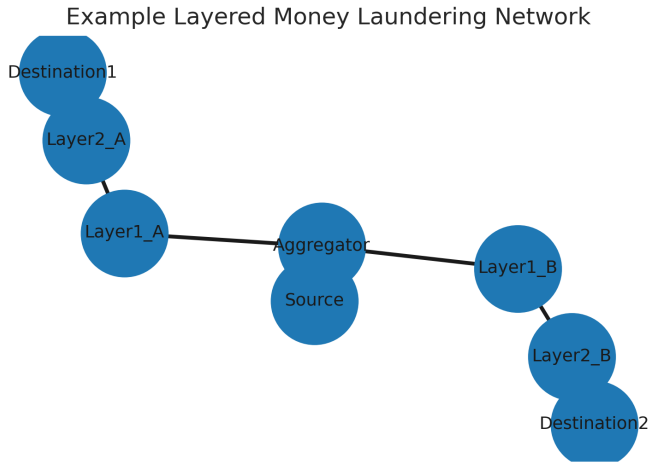

Fig. 7. Example layered laundering structure involving multiple intermediary accounts.

The transaction sequence begins with a large incoming transfer to an aggregation account. The funds are subsequently distributed across multiple intermediate accounts, each of which performs rapid outgoing transactions within short time intervals. These transactions create a multi-hop network structure designed to obscure the financial trail.

Traditional rule-based monitoring systems typically analyze transactions individually and may fail to detect such distributed laundering structures. In contrast, the graph-based representation used by the Nexus AML system captures the relational dependencies between accounts.

During graph construction, the involved accounts form a tightly connected subgraph characterized by unusually high transaction velocity and abnormal connectivity patterns. The GraphSAGE model identifies this subgraph as anomalous due to the combination of structural features and behavioral indicators.

The final risk score assigned to the central aggregation account exceeded the investigation threshold, triggering the creation of an investigation case.

The explanation module further identified the transaction edges contributing most strongly to the suspicious classification. These edges corresponded to rapid outgoing transfers from the aggregation account to multiple intermediary accounts.

This case study demonstrates the advantage of graph-based modeling for identifying complex laundering structures that span multiple accounts and transactions.

- unusually high outgoing transaction volume
- abnormal degree centrality within the transaction network
- high transaction velocity
- strong GNN classification probability

Subgraph extraction revealed that the central account distributed approximately $2.8 million across 47 recipient accounts within a 48-hour period.

The explainability module highlighted the most influential edges contributing to the model's prediction. Investigators were able to quickly identify the suspicious transaction flow and reconstruct the laundering pattern.

This case study demonstrates how graph-based detection models can uncover complex financial crime structures that would be difficult to identify using traditional rule-based monitoring systems.

XX. Discussion

The experimental results demonstrate that graph neural networks provide significant advantages for anti-money laundering detection.

Traditional rule-based monitoring systems typically rely on manually defined thresholds such as transaction limits or geographic risk indicators. While these approaches are effective for detecting simple suspicious behaviors, they struggle to identify complex laundering networks.

Machine learning approaches based on tabular data offer improvements over rule-based systems but still fail to capture relational dependencies between accounts.

Graph-based learning methods address this limitation by modeling financial transactions as network structures. This allows the detection system to identify suspicious patterns that emerge only when analyzing relationships between multiple accounts.

The results obtained in this study indicate that graph neural networks can achieve strong detection performance while maintaining scalability for large transaction datasets.

Additionally, the integration of explainable AI techniques ensures that automated detection decisions remain interpretable for investigators and regulatory authorities.

XXI. Limitations

Despite the promising results demonstrated in this study, several limitations remain.

First, the experiments were conducted using synthetic transaction data generated by AMLSim. While the simulator produces realistic transaction patterns, it may not capture all complexities present in real-world financial networks.

Second, supervised machine learning approaches rely on labeled training data. In practice, labeled examples of money laundering may be limited, which could impact model performance.

Third, graph neural network models can require significant computational resources when analyzing extremely large networks.

Future research should explore techniques for improving scalability and reducing computational requirements while maintaining detection accuracy.

XXII. Future Work

Several directions for future research are possible.

One promising avenue is the integration of temporal graph neural networks. Financial transactions occur over time, and temporal models could capture dynamic patterns in transaction networks.

Another potential improvement involves unsupervised anomaly detection methods. These methods could identify previously unseen laundering techniques that are not represented in training data.

Federated learning approaches may also enable financial institutions to collaboratively train detection models while preserving data privacy.

Finally, improved visualization tools could enhance the ability of investigators to analyze suspicious transaction networks.

XXIII. Conclusion

This paper presented Nexus AML, a scalable graph neural network framework for detecting suspicious financial activity in large transaction networks.

The proposed system models financial transactions as directed graphs and applies graph neural network techniques to identify suspicious structural patterns.

Experimental evaluation using the AMLSim dataset demonstrates that the Nexus AML framework achieves strong detection performance while maintaining high processing throughput.

The results highlight the potential of graph-based machine learning methods for improving anti-money laundering detection systems.

By integrating scalable graph processing, advanced machine learning models, and explainable AI techniques, Nexus AML provides a promising foundation for next-generation financial crime detection systems.

References

[1] United Nations Office on Drugs and Crime, "Money Laundering and Global Financial Crime," UNODC Report, Vienna, Austria, 2020.

[2] M. Weber, J. Chen, J. Suzumura, T. Pareja, H. Ma, T. Kanezashi, T. Kaler, and C. Eickhoff, "Scalable Graph Learning for Anti-Money Laundering: A First Look," in *IEEE International Conference on Big Data*, 2018, pp. 1224–1233.

[3] IBM Research, "AMLSim: A Large-Scale Anti-Money Laundering Simulation Framework," IBM Research Technical Report, 2019.

[4] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[5] T. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[6] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[7] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "GNNExplainer: Generating Explanations for Graph Neural Networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[8] L. Akoglu, H. Tong, and D. Koutra, "Graph-Based Anomaly Detection and Description: A Survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[9] E. Ngai, Y. Hu, Y. Wong, Y. Chen, and X. Sun, "The Application of Data Mining Techniques in Financial Fraud Detection: A Classification Framework and an Academic Review of Literature," *Decision Support Systems*, vol. 50, no. 3, pp. 559–569, 2011.

[10] R. Chalapathy and S. Chawla, "Deep Learning for Anomaly Detection: A Survey," *ACM Computing Surveys*, vol. 54, no. 2, 2019.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[12] J. Leskovec, A. Rajaraman, and J. Ullman, *Mining of Massive Datasets*, Cambridge University Press, 2020.

[13] R. Bolton and D. Hand, "Statistical Fraud Detection: A Review," *Statistical Science*, vol. 17, no. 3, pp. 235–255, 2002.

[14] N. Shervashidze, P. Schweitzer, E. Jan Van Leeuwen, K. Mehlhorn, and K. Borgwardt, "Weisfeiler-Lehman Graph Kernels," *Journal of Machine Learning Research*, vol. 12, pp. 2539–2561, 2011.