AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
Peer Reviewed, Referred & Indexed Journal
E-ISSN:3069-0102
VOL.6, NO. 2(2026)

AI-Based Fraud Detection Using Graph Neural Networks

Narayana Setti Siva Krishna
Reg. No. 24Q71F0036
nsivakrishna01@gmail.com
Department of Master of Computer Applications
Avanthi Institute of Engineering and Technology (Autonomous)
Vizianagaram, Andhra Pradesh, India
Under the guidance of Mrs. A. Anitha, M.Tech., Assistant Professor
anithaadireddy96@gmail.com

***Abstract*—Fraud detection is an important problem in today's digital world due to the rapid increase in online transactions. Traditional methods often fail to identify complex patterns of fraud. This project proposes an AI-based fraud-detection system using Graph Neural Networks (GNNs) to improve accuracy. In this system, transaction data is represented as a graph where nodes represent users or accounts and edges represent transactions between them, and a GraphSAGE-based GNN analyses these relationships to detect unusual patterns and identify fraudulent activities effectively. The system is implemented in Python using PyTorch and PyTorch Geometric for the GNN, with NetworkX for graph construction, scikit-learn for preprocessing, and a Streamlit interface for visualisation and prediction. It models a user–merchant transaction graph for edge-level fraud classification, incorporates realistic fraud indicators such as unusual transaction timing, device changes, and abnormal behaviour, supports uploading real UPI transaction CSVs or using built-in synthetic data, and displays risk scores together with explanations. The approach is designed to handle large amounts of data and provide faster, more reliable fraud detection, reducing false alerts while strengthening decision-making in real-time financial systems. The prototype was validated through functional test cases for dataset upload, missing-value handling, model training, fraud prediction, and visualisation, all of which passed. Overall, the system demonstrates how graph-based deep learning can significantly enhance fraud detection, making it more robust, scalable, and suitable for real-world financial applications.*

***Keywords*—Fraud Detection; Graph Neural Network; GraphSAGE; UPI Transactions; Edge-Level Classification; PyTorch Geometric; Streamlit; Financial Security.**

I. INTRODUCTION

In today's digital era, the rapid growth of online payment systems—especially the Unified Payments Interface (UPI)—has significantly transformed financial transactions. However, this growth has also led to a sharp increase in fraudulent activities, making fraud detection a critical challenge for financial institutions and digital platforms. Traditional fraud-detection methods, which rely on rule-based systems and simple machine-learning models, often struggle to identify complex and evolving fraud patterns.

This project develops an AI-based fraud-detection system using Graph Neural Networks. Unlike conventional approaches, GNNs are capable of capturing relationships and interactions between entities such as users, merchants, and transactions. Transaction data is modelled as a graph in which nodes represent

users or accounts and edges represent transactions between them; by analysing the structure and connectivity of this graph, the model can detect suspicious patterns that are otherwise difficult to identify. The system uses feature engineering, graph construction, and deep-learning-based classification to improve detection accuracy, and is designed to handle large-scale transaction data efficiently and provide faster, more reliable predictions while reducing false positives.

The implementation integrates Streamlit for visualisation, PyTorch for model development, and a GraphSAGE-based GNN architecture for edge-level fraud classification, and incorporates realistic fraud indicators such as unusual transaction timing, device changes, and abnormal transaction behaviour. Overall, the project demonstrates how graph-based deep learning can significantly enhance fraud detection systems, making them more robust, scalable, and suitable for real-world financial applications.

II. LITERATURE SURVEY

Fraud detection in financial transactions has been an active research area for many years due to the increasing number of digital payments and cybercrimes. Researchers have proposed various techniques ranging from traditional statistical methods to advanced artificial intelligence and deep-learning approaches, and the literature shows a clear evolution from rule-based and shallow models to graph-based deep learning. Early fraud-detection systems were based on predefined rules such as transaction limits, unusual login behaviour, or geographic mismatches; these systems were simple to implement but lacked adaptability, failed to detect new or sophisticated fraud patterns, and produced many false positives.

With the advancement of machine learning, models such as Logistic Regression, Decision Trees, Random Forest, and Support Vector Machines were widely used and improved detection accuracy by learning patterns from historical data, but they treated transactions independently and did not capture relationships between users and transactions. Deep-learning models such as ANN, CNN, and RNN were later introduced for large and complex datasets but still struggled with relational and graph-based dependencies. Recent research focuses on graph-based approaches in which transactions are represented as networks. Foundational work in graph-based learning by Kipf and Welling (semi-supervised classification with Graph Convolutional Networks) and Hamilton, Ying, and Leskovec (GraphSAGE: inductive representation learning on large graphs) provides the basis for modern GNN-based fraud detection. Surveys by Wu et al. on Graph Neural Networks and by Akoglu, Tong, and Koutra on graph-based anomaly detection further establish the field, motivating the GraphSAGE-based approach used in this project.

TABLE I. EVOLUTION OF FRAUD-DETECTION APPROACHES

S.No	Approach	Technique	Limitation / Note
1	Rule-based systems	Predefined rules / limits	No adaptation; high false positives
2	Classical ML	LogReg, DT, RF, SVM	Treats transactions independently
3	Deep learning	ANN, CNN, RNN	Weak on relational/graph data

S.No	Approach	Technique	Limitation / Note
4	Graph Convolutional Networks	Kipf & Welling (GCN)	Foundational graph learning
5	Inductive graph learning	GraphSAGE (Hamilton et al.)	Scales to unseen nodes
6	Graph anomaly detection	Akoglu et al., survey	Graph-based fraud framing

III. EXISTING SYSTEM AND PROPOSED SYSTEM

A. Existing System

Existing fraud-detection systems mainly rely on rule-based engines and classical machine-learning models such as Logistic Regression, Decision Trees, Random Forest, and SVM. Rule-based systems use predefined thresholds and patterns and are easy to implement but unable to adapt to new fraud strategies, while classical machine-learning models learn patterns from historical labelled data but treat each transaction independently and ignore relationships between users, merchants, devices, and locations. As a result, complex, multi-hop fraud schemes that span several entities are difficult to identify, and detection often produces a large number of false positives.

Limitations of the existing system:

- Cannot detect complex and evolving fraud patterns.
- High false-positive rate from rule-based engines.
- Treats each transaction independently; ignores relationships.
- Limited scalability and adaptability to evolving fraud.
- Misses multi-hop fraud across multiple users and merchants.

B. Proposed System

The proposed system introduces a Graph Neural Network-based approach that overcomes these limitations by modelling transactions as a graph and classifying edges (transactions) as fraudulent or legitimate. Key features include representation of transactions as a user–merchant graph, a GraphSAGE-based GNN for inductive learning over the graph, feature engineering for fraud indicators (unusual timing, device changes, abnormal behaviour), reduction of false positives, scalability to large datasets, real-time or near-real-time prediction, and an interactive Streamlit dashboard for visualisation and explanation.

Advantages of the proposed system:

- Captures hidden relationships in transaction data.
- Detects complex, multi-hop fraud schemes.
- Reduces false positives compared with traditional methods.
- Scales to large user-merchant graphs.
- Real-time / near-real-time prediction with risk scores.
- Interactive visualisation and explanation through a dashboard.

IV. SYSTEM DESIGN AND METHODOLOGY

A. Graph Construction

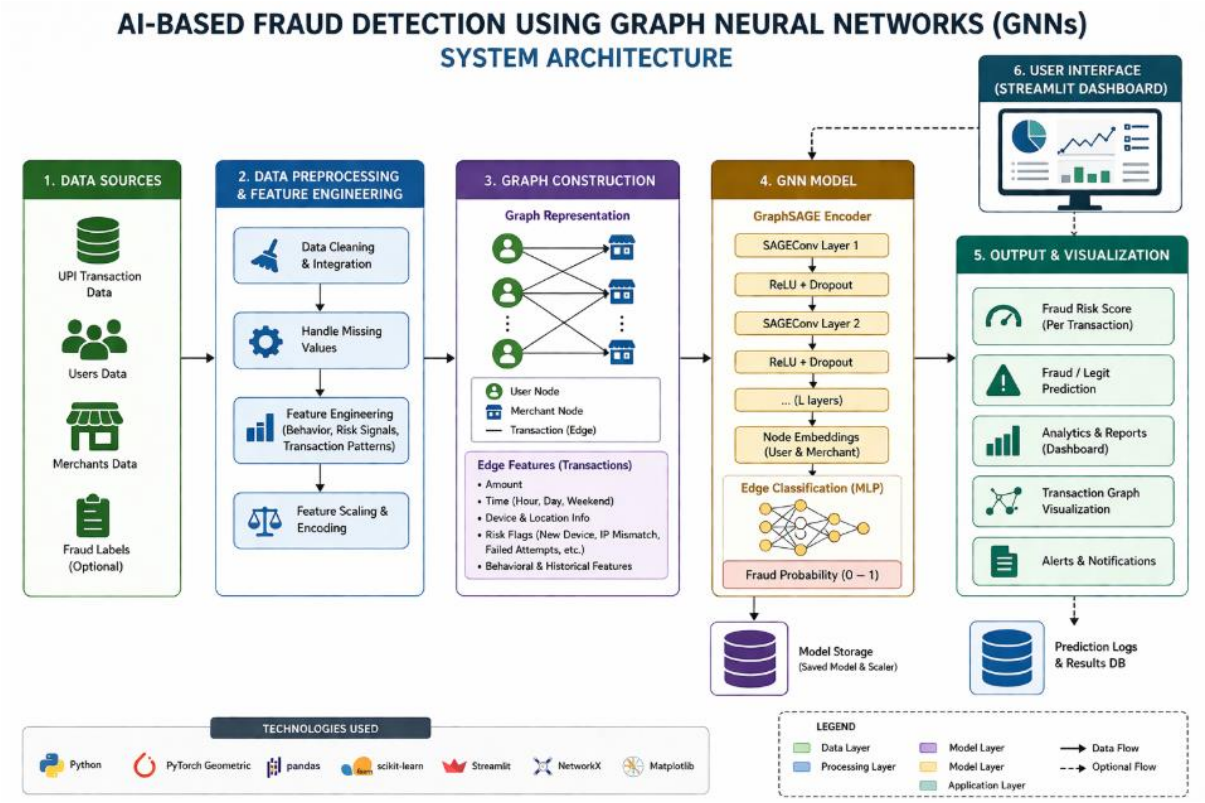
Transaction data is modelled as a graph in which nodes represent users or accounts (and, in the user–merchant variant, also merchants) and edges represent transactions between them. Node features include user-level attributes and aggregated behavioural statistics, and edge features include transaction-level attributes such as amount, timestamp, device, and location. Realistic fraud indicators such as unusual transaction timing, device changes, and abnormal transaction behaviour are computed during feature engineering and attached to the corresponding edges, ensuring that the graph captures both structural relationships and behavioural signals.

B. GNN Model (GraphSAGE)

The system uses a GraphSAGE-based Graph Neural Network for edge-level fraud classification. Multiple GraphSAGE layers compute node embeddings by aggregating information from each node's neighbourhood, so that a node's representation reflects the structure and features of its local subgraph. For an edge connecting two nodes, the embeddings of the endpoints are combined with edge-level features to predict the probability that the transaction is fraudulent. GraphSAGE is chosen because it is inductive—it generalises to unseen nodes—and scales to large graphs, which is essential for real-world payment networks.

C. Workflow

The user uploads a real UPI transaction CSV or selects the built-in synthetic dataset through the Streamlit interface; the system preprocesses the data, handles missing values, and builds the user–merchant transaction graph; the GraphSAGE GNN is trained on the graph for edge-level fraud classification; new transactions are passed through the trained model to obtain a fraud risk score and explanation; and the transaction graph is visualised together with predictions. Invalid file formats or missing columns are detected and handled with clear error messages.



V. SYSTEM IMPLEMENTATION

A. Technology Stack

TABLE II. TECHNOLOGY STACK

Component	Technology / Tool
Programming Language	Python
Deep-Learning Framework	PyTorch
Graph Library	PyTorch Geometric
Graph Construction	NetworkX
Classical ML / Preprocessing	scikit-learn
Data Handling	pandas, NumPy
Visualisation	Matplotlib, Seaborn
Frontend / UI	Streamlit (‘streamlit run app.py’)
Model Architecture	Multi-layer GraphSAGE (edge-level classification)

B. Implementation Pipeline

The implementation is carried out using Python-based technologies and deep-learning frameworks, organised as a modular pipeline that includes data preprocessing, graph construction, model training, and

prediction with a user-friendly interface. Data preprocessing handles missing values, encodes categorical features, and engineers fraud indicators; graph construction builds the user–merchant network with node and edge features using NetworkX; the modelling stage builds and trains the GraphSAGE GNN with PyTorch Geometric and PyTorch; and the prediction stage uses the trained model to score new transactions. The Streamlit application ties the pipeline together: a user uploads transaction data or uses synthetic data, trains the GNN, visualises the transaction graph, and predicts fraud on new transactions with risk scores and explanations.

C. Model Details and Outputs

The model uses multiple GraphSAGE layers to compute node embeddings; for each transaction edge, the source and destination embeddings together with edge-level features are combined and passed through a classification head that outputs a probability of fraud. The system also produces explanatory signals based on the strongest fraud indicators that contributed to the prediction—such as unusual timing, device changes, or abnormal behaviour—so that analysts can interpret why a transaction was flagged. Outputs include the fraud risk score, the explanation, and the visualised user–merchant graph showing flagged edges.

VI. SYSTEM TESTING AND RESULTS

The system was validated through functional testing of dataset upload, missing-value handling, model training, fraud prediction, and graph visualisation. All defined test cases passed successfully, behaving as expected. The reported results state that the system successfully detects fraudulent transactions with high accuracy compared with traditional methods, reduces false positives, performs well even with large datasets, and achieves real-time prediction with minimal delay; the source describes these outcomes qualitatively, so no specific numeric accuracy values are claimed here.

TABLE III. FUNCTIONAL TEST CASES

ID	Test Case	Input	Expected Output	Result
TC01	Upload dataset	Valid CSV file	Data loaded successfully	**Pass**
TC02	Missing values	Incomplete dataset	Handled without error	**Pass**
TC03	Model training	Training data	Model trains without crash	**Pass**
TC04	Fraud prediction	New transaction	Fraud score generated	**Pass**
TC05	Visualisation	Graph data	Graph displayed correctly	**Pass**

A. Observed Results and Error Handling

The implemented system handles fraud-related challenges through layered error handling: invalid file formats are detected and rejected; missing columns are handled with default values or warnings; and model-related errors are minimised through proper validation. Compared with rule-based engines and classical ML, the GraphSAGE-based GNN improves detection of complex, multi-hop fraud and lowers false

positives, while the Streamlit dashboard makes the workflow accessible to analysts who can train, predict, and visualise within one interface. Real-world performance depends on the quality and balance of the underlying transaction data and on the evolving nature of fraud.

Representative screenshots from the prototype implementation:

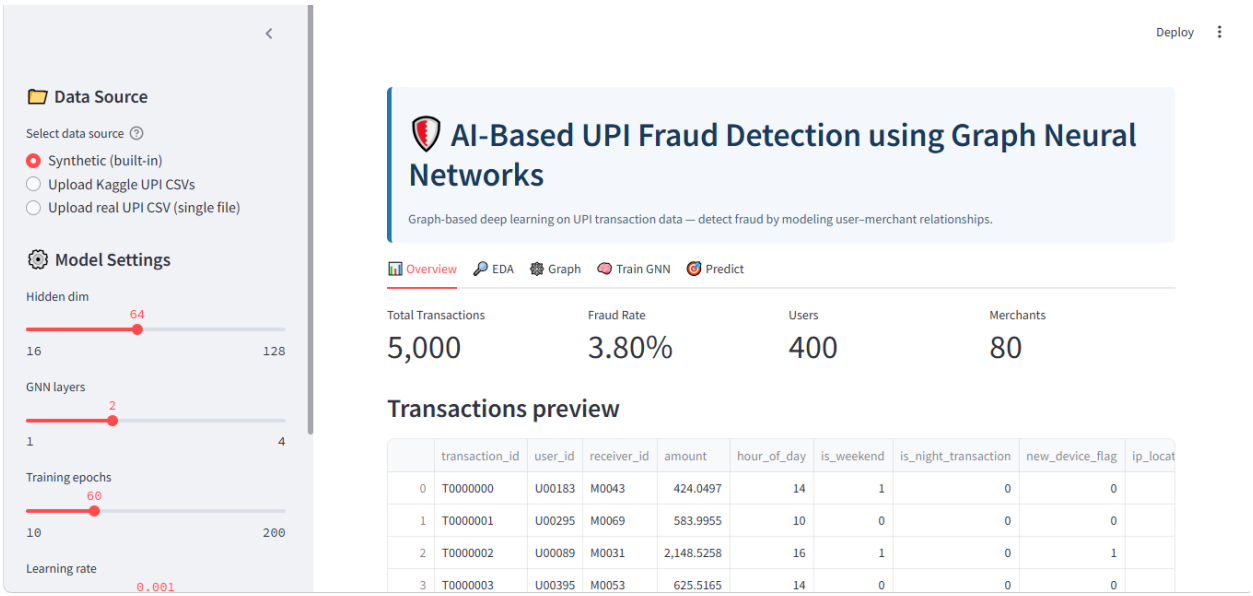


Fig. 1. Dataset upload and overview.

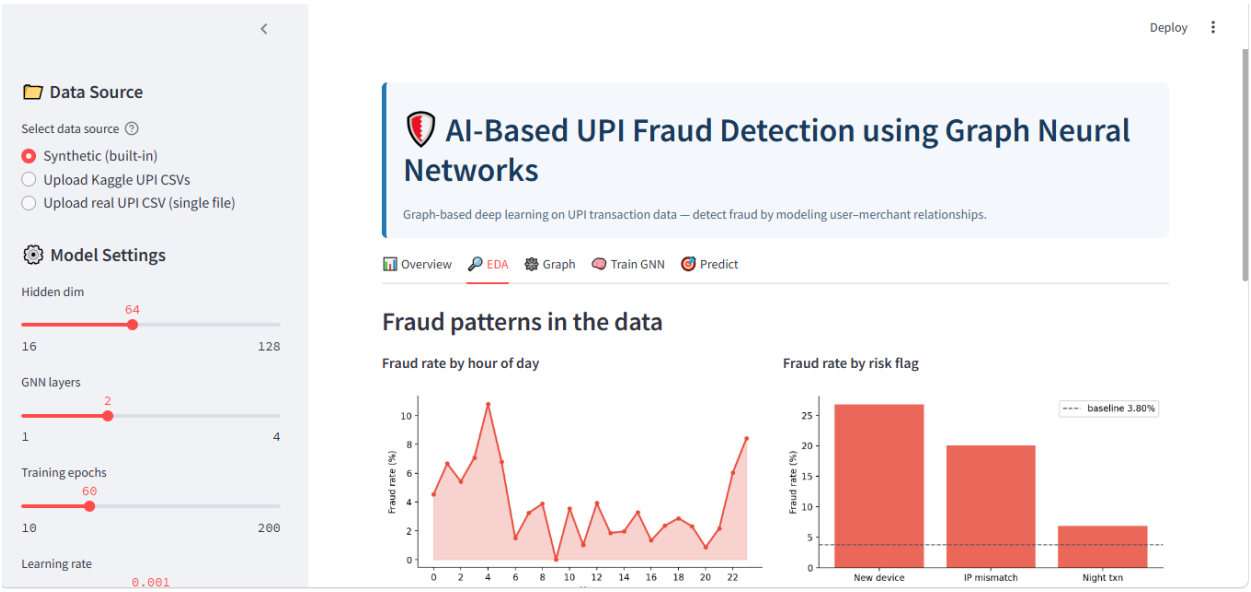


Fig. 2. User–merchant transaction-graph visualisation

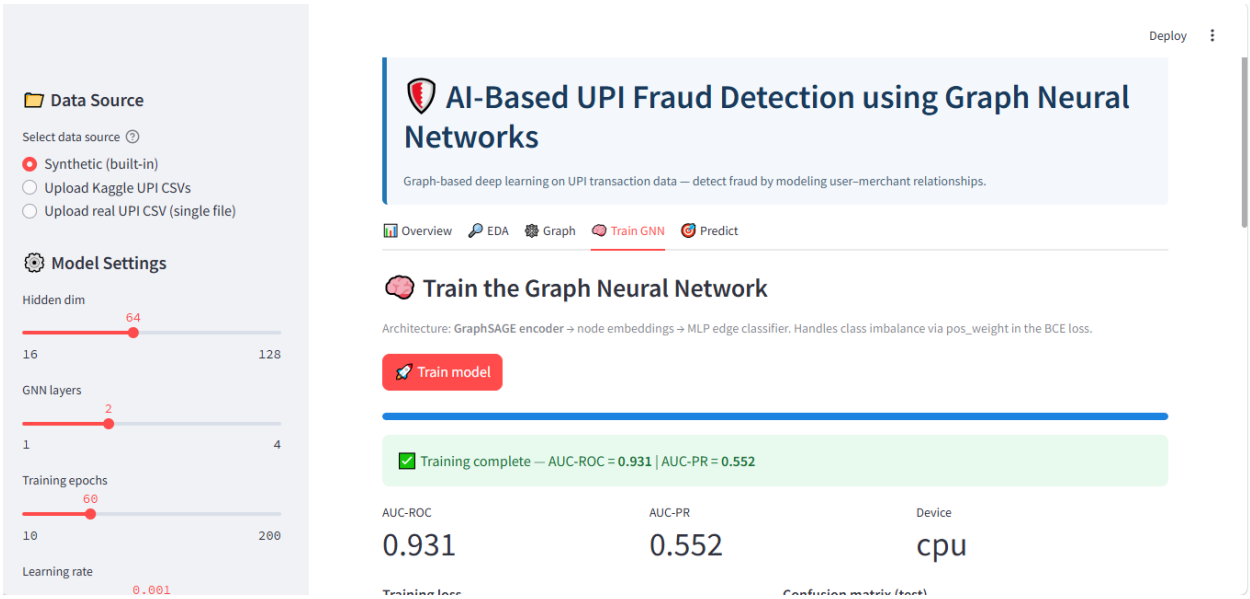


Fig. 3. GNN training progress and metrics.

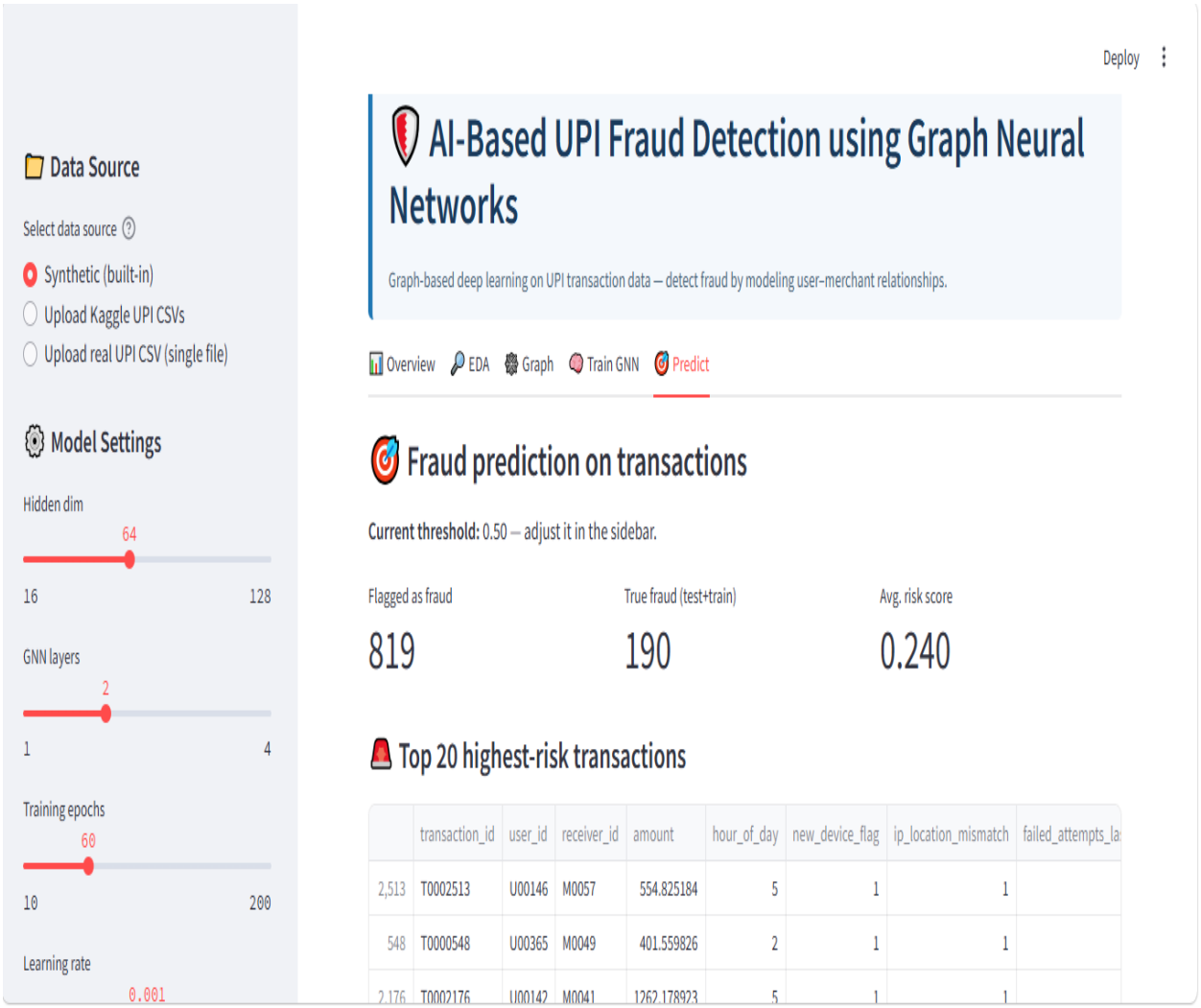


Fig. 4. Fraud prediction with risk score and explanation.

VII. CONCLUSION AND FUTURE WORK

The AI-Based Fraud Detection using Graph Neural Networks project demonstrates an effective and modern approach to detecting fraudulent activities in digital payment systems. By representing transactions as a graph and analysing relationships between users and merchants, the system overcomes the limitations of traditional fraud-detection methods, and the GraphSAGE-based GNN model captures complex interaction patterns and hidden fraud behaviours that are often missed by conventional machine-learning techniques. The integration of feature engineering, graph construction, and deep learning improves detection accuracy while reducing false positives, and the system provides efficient handling of large-scale transaction data, real-time or near-real-time prediction, an interactive visualisation dashboard, and a scalable architecture suitable for real-world applications. Overall, the project shows that graph-based deep learning is a powerful and reliable solution for enhancing security in financial transactions.

Although the system performs effectively, several areas remain for improvement. Real-time deployment with live payment gateways and streaming technologies such as Apache Kafka would enable

continuous data processing. Larger and more diverse datasets, adversarial training, and continual learning would harden the model against evolving fraud strategies. Richer graph structures incorporating devices, locations, and time windows, together with more advanced GNN variants and explainability tools, would further improve detection and analyst trust. Cloud-native deployment, integration with hospital or bank information systems, and compliance with regulatory and privacy requirements are also key directions for production use.

REFERENCES

[1] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in Proc. Int. Conf. Learning Representations (ICLR), 2017.

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

[3] Z. Wu et al., "A Comprehensive Survey on Graph Neural Networks," IEEE Transactions on Neural Networks and Learning Systems, 2020.

[4] L. Akoglu, H. Tong, and D. Koutra, "Graph-Based Anomaly Detection and Description: A Survey," Data Mining and Knowledge Discovery, 2015.

[5] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016.

[6] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. Sebastopol, CA, USA: O'Reilly Media, 2019.

[7] PyTorch Documentation. [Online]. Available: https://pytorch.org/docs/

[8] PyTorch Geometric Documentation. [Online]. Available: https://pytorch-geometric.readthedocs.io/

[9] Streamlit Documentation. [Online]. Available: https://docs.streamlit.io/

[10] Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/