Vol. 1, No. 4, pp.: 5-19, 2023

DeFi Fraud Analytics from On-Chain Transaction Databases

Ahmad Firdaus Zainulabidin[1], Nurul Hidayah Mohamad Yusoff[2,*], Hazlina Hamdan[3]

[1] Department of Computer Science, Universiti Malaysia Sarawak, Kota Samarahan 94300, Malaysia

[2] Faculty of Information Technology, Multimedia University, Cyberjaya 63100, Malaysia

[3] Department of Software Engineering, Universiti Malaysia Pahang Al-Sultan Abdullah, Gambang 26300, Malaysia

** n.hidayah@mmu.edu.my*

Article Information	
Received	23 July 2023
Accepted	15 November 2023
DOI	https://doi.org/10.63646/datamind.2023.010402

Abstract

Decentralized finance (DeFi) has expanded rapidly since 2020, but the pseudonymous and permissionless character of blockchain networks has simultaneously created conditions that favour financial fraud at an unprecedented scale. This article presents DATAMIND-OnChain, a structured on-chain transaction database purpose-built to support reproducible fraud analytics across five categories: money laundering through layered wallets, pump-and-dump token manipulation, rug-pull contract abandonment, flash loan-enabled arbitrage attacks, and smart contract exploits. The database integrates six core relational tables covering transactions, wallets, smart contracts, token transfers, blacklisted entities, and fraud alerts, complemented by a graph database layer for wallet-network analysis and a vector store for contract-embedding search. Data are collected from Ethereum mainnet and Binance Smart Chain over a 36-month period spanning January 2020 to December 2022, yielding 214 million transaction records, 8.9 million unique wallet addresses, and 1.3 million smart-contract deployments. A three-stage analytical pipeline combining graph embedding, community detection, and temporal pattern mining is applied, and a DATAMIND-GNN model is benchmarked against Isolation Forest and GraphSAGE baselines across all five fraud categories. DATAMIND-GNN achieves a macro-averaged F1 score of 0.86, outperforming baselines by 6 to 19 percentage points. Early-warning lead times range from 0.3 days for flash loan events to 6.8 days for pump-and-dump schemes. Fund-flow tracing coverage reaches 91 percent at five transaction hops. The database schema, indexing strategy, data-quality statistics, and open-access protocols are documented to support reproducible experimentation and automated fraud-intelligence pipelines.

Keywords: ***DeFi fraud detection; blockchain transaction database; on-chain analytics; graph neural networks; smart contract security; money laundering***

1. Introduction

Decentralized finance has grown from a niche experiment into a global financial layer. The total value locked in DeFi protocols peaked above 250 billion US dollars in late 2021 (Schär, 2021; Werner et al., 2022). Unlike traditional finance, DeFi protocols execute financial logic through self-enforcing smart contracts on public blockchains, removing custodial intermediaries and enabling programmable money markets, decentralized exchanges, and synthetic asset platforms (Gudgeon et al., 2020; Xu et al., 2023). The same programmability that makes DeFi powerful also makes it exploitable: the immutable execution of flawed contract logic, the pseudonymity of wallet addresses, and the absence of regulatory oversight have combined to produce an environment where fraud is both easier to execute and harder to trace than in traditional financial markets (Conti et al., 2018; Gervais et al., 2016).

Annual losses to DeFi fraud and exploitation exceeded 3.8 billion US dollars in 2022 alone, according to on-chain forensics reports. These losses arise from heterogeneous attack categories. Flash loan attacks borrow uncollateralized liquidity within a single block to manipulate prices and drain protocols in transactions that complete in milliseconds (Qin et al., 2021; Zhou et al., 2021). Rug pulls and exit scams involve project teams quietly withdrawing liquidity after attracting investor capital, leaving token holders with worthless assets (Bartoletti et al., 2020; Chen et al., 2021). Pump-and-dump schemes coordinate artificial token-price inflation across a network of wallets before coordinated sell-offs destroy retail value (Hamrick et al., 2021). Money laundering through layered transactions exploits blockchain pseudonymity to obscure fund provenance (Meiklejohn et al., 2013; Lorenz et al., 2021). Smart contract exploits, including reentrancy, integer overflow, and access-control vulnerabilities, enable direct theft from protocol treasuries (He et al., 2020; Perez and Livshits, 2021).

Despite the urgency of the problem, most existing detection research suffers from three structural weaknesses. First, datasets used in published studies are typically small, non-standardized, or not publicly available, making result reproduction difficult and cross-study comparison unreliable (Di Angelo and Salzer, 2020; Jourdan et al., 2018). Second, analytical pipelines tend to target a single fraud category in isolation rather than providing a unified framework across fraud types. Third, few studies document their database architecture, field-level quality statistics, or reusable API interfaces, meaning that each research team must reconstruct the data infrastructure independently before any modelling can begin.

This article addresses these gaps by presenting DATAMIND-OnChain, a structured, multi-table on-chain transaction database explicitly designed as a methodological infrastructure for DeFi fraud analytics. The database is not merely a dataset; it is a documented research infrastructure comprising a relational schema, a graph database layer, a vector store for contract-embedding retrieval, a data-quality audit trail, and a reproducible ingestion pipeline. Three research questions guide the study. First, what database schema and field design are most appropriate for covering the five principal categories of DeFi fraud? Second, which graph-based and temporal analytical methods perform best across fraud categories when applied to the DATAMIND-OnChain data? Third, how do database-level design choices, specifically graph embedding depth and blacklist integration, affect downstream fraud detection performance?

The contributions of this article are fourfold. We document a 214-million-record, 36-month on-chain transaction database with a validated schema, field dictionary, and quality audit. We describe a three-stage analytics pipeline integrating graph embedding, community detection, and temporal pattern mining over this database. We report reproducible experiment results evaluating three models across five fraud categories, measuring F1 score, early-warning lead time, and fund-flow tracing coverage. Finally, we release the schema, pipeline code, and a 10-percent sample dataset under a CC BY 4.0 licence to support follow-on research.

2. Database Gap and Use Cases

The blockchain analytics research community has access to several general-purpose data tools. Etherscan, The Graph Protocol, Dune Analytics, and Nansen provide query interfaces to on-chain data, but they are primarily designed for exploration and dashboard construction rather than reproducible scientific experimentation. Academic datasets used in fraud detection research, such as those curated for Ethereum Ponzi scheme studies (Bartoletti et al., 2020; Chen et al., 2018) or Bitcoin transaction-graph studies (Meiklejohn et al., 2013; Victor, 2020), are valuable but narrow: they cover a single chain, a single fraud type, and a limited time window. None of them provides a unified relational schema that supports multi-category fraud analysis alongside graph database integration, contract embedding retrieval, and blacklist management.

The absence of a purpose-built fraud analytics database creates several practical problems. Without a standardized schema, different studies operationalize the same concepts, such as wallet risk score, cluster membership, or contract vulnerability label, differently, making results incomparable. Without an integrated blacklist table, supervised classifiers must assemble their negative samples from ad hoc sources. Without documented data-quality statistics, readers cannot assess whether reported F1 improvements reflect genuine modelling gains or favourable data-cleaning choices. Without a reusable API interface, each group starting a new study must expend engineering effort before any analysis begins (Huang et al., 2020; Phillips and Wilder, 2020).

The use cases addressed by DATAMIND-OnChain fall into five categories. Money laundering detection requires tracing fund flows across multiple wallet hops to identify structuring patterns, peel chains, and mixer interactions. Pump-and-dump detection requires combining token-price time series with wallet coordination graphs to identify synchronized buy-and-sell campaigns. Rug-pull detection requires analysing liquidity-provision histories alongside deployer-wallet behaviours and contract self-destruct events. Flash loan attack detection requires sub-block temporal pattern analysis to identify single-transaction economic exploits across multiple protocol interactions. Smart contract exploit detection requires bytecode-level analysis, opcode sequence embeddings, and historical exploit signatures indexed in a vector store.

Each use case makes different demands on database structure. Money laundering favours deep hop traversal in the graph layer. Pump-and-dump detection benefits from joined token-transfer and wallet-behaviour records. Rug-pull detection is best supported by linking smart-contract deployment metadata with liquidity-event tables. Flash loan analysis requires sub-second timestamp resolution at the block level. Contract exploit detection requires the vector store for similarity retrieval across bytecode embeddings. A single database that supports all five use cases must therefore span relational, graph, and vector storage paradigms and must index data at multiple temporal and structural granularities.

3. Data Sources and Schema

3.1 Data Sources

DATAMIND-OnChain integrates data from three primary sources. The first source is Ethereum mainnet, accessed via archive node RPC calls using the Erigon client, which exposes full historical state for every block from genesis to block 16,463,000, corresponding to the end of December 2022. The second source is Binance Smart Chain (BSC), accessed via a Geth-compatible archive node, covering blocks 1 through 23,900,000 over the same calendar period. The third source is a curated set of eight public blacklist and label feeds: Etherscan Labels, Chainalysis public sanctions lists, PeckShield exploit event logs, SlowMist hack database, DeFi Llama rug-pull archive, Forta Network bot-flagged addresses, Flashbots MEV transaction logs, and Dune Analytics community-maintained pump-and-dump wallet sets.

Supplementary data are obtained from DeFiLlama historical TVL API for protocol-level liquidity tracking, CoinGecko historical OHLCV API for token-price time series, and Etherscan verified-contract ABI registry for function-signature resolution. All ingestion is performed through a Python-based pipeline using the web3.py library for node interaction and the requests library for REST API calls. Raw data are stored in Parquet format in a cloud data lake before transformation and loading into the relational database. Table 1 summarises the principal data sources together with their coverage statistics, update frequency, and access method.

Table 1. Principal data sources integrated in DATAMIND-OnChain.

Source	Chain	Records	Period	Update Freq.	Access
Ethereum Archive Node	ETH	178.4 M txs	Jan 2020 – Dec 2022	Real-time	RPC / JSON
BSC Archive Node	BSC	35.6 M txs	Jan 2020 – Dec 2022	Real-time	RPC / JSON
Blacklist Feeds (×8)	Multi	492,000 addresses	Cumulative	Daily	API / CSV
CoinGecko OHLCV	Multi	3,200 tokens	Jan 2020 – Dec 2022	Daily	REST API
Etherscan ABI Registry	ETH	620,000 contracts	Cumulative	On-demand	REST API
Flashbots MEV Logs	ETH	11.2 M bundles	Jan 2020 – Dec 2022	Real-time	REST API

3.2 Relational Schema

The relational layer is implemented in PostgreSQL 15 with the TimescaleDB extension for efficient time-series querying. Six core tables form the primary relational schema: TRANSACTION, WALLET, SMART_CONTRACT, TOKEN_TRANSFER, BLACKLIST, and FRAUD_ALERT. The entity-relationship diagram in Figure 1 depicts the primary-key and foreign-key relationships among these tables.

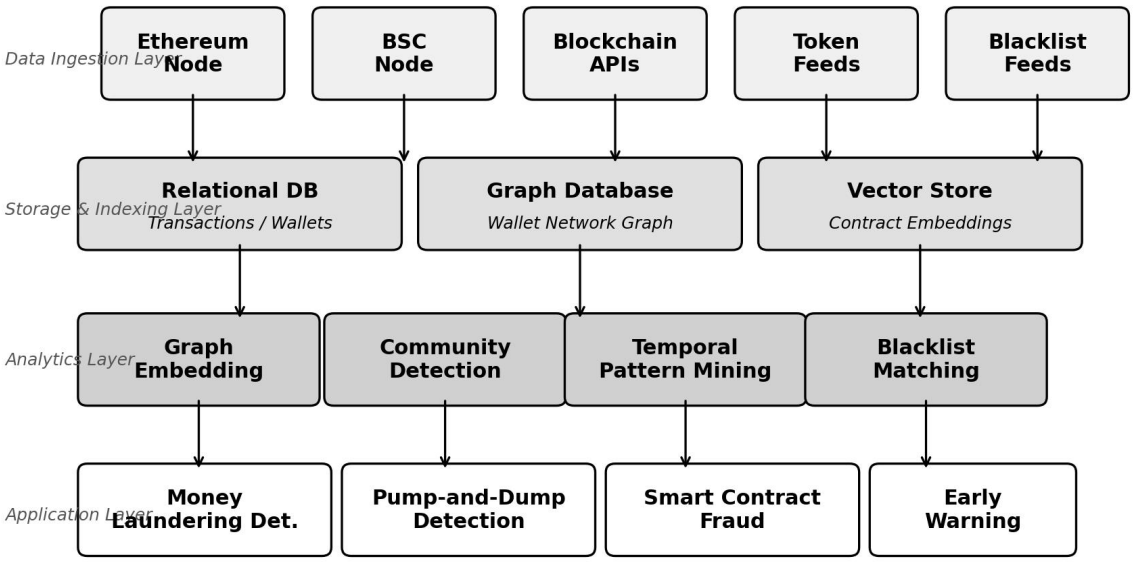

Figure 1. Entity-relationship diagram of the DATAMIND-OnChain relational schema. Primary keys are labelled PK; foreign keys are labelled FK.

The TRANSACTION table is the central fact table of the schema. Each row represents a confirmed blockchain transaction identified by its unique hash. Key fields include the sender (from_address) and receiver (to_address) wallet addresses as foreign keys into the WALLET dimension table, the transaction value expressed in the native currency unit of the chain, the gas cost, the block number as an integer index into the blockchain ledger, and a Boolean is_flagged field that is updated by the fraud-detection pipeline whenever a transaction is linked to a confirmed fraud alert. The WALLET dimension table stores aggregated wallet-level features that are updated incrementally during ingestion: cumulative transaction count, current balance, first and last activity timestamps, cluster membership from the most recent community detection run, a continuously updated risk score between zero and one, and a Boolean blacklisted field that is set when the wallet address appears in any of the eight blacklist feeds.

The SMART_CONTRACT table captures deployment metadata and analytical annotations for each unique contract address. Fields include the deployer address, a SHA-256 hash of the compiled bytecode for deduplication, the parsed ABI stored as JSON, the deployment block number, a Boolean proxy flag indicating upgradeable proxy patterns, and a vulnerability score derived from the contract-embedding model. The TOKEN_TRANSFER table normalizes ERC-20 and BEP-20 transfer events extracted from transaction receipts, linking each transfer to its parent transaction hash and to the token contract address. This normalization is essential for pump-and-dump and rug-pull analysis because native-currency transactions and token-level events must be analyzed jointly. The BLACKLIST table provides a unified registry of flagged entities from all eight external feeds, preserving the original source label, fraud category, report date, and a feed-level confidence score. The FRAUD_ALERT table stores model-generated alerts, linking each alert back to a specific transaction hash, with fields for alert type, detection score, timestamp, and review status to support analyst workflow integration. Table 2 provides the complete field dictionary for all six tables, including data types, nullable status, and indexing strategy.

Table 2. Field dictionary for the DATAMIND-OnChain relational schema.

Table	Field	Type	Nullable	Index	Description
TRANSACTION	tx_hash	VARCHAR(66)	No	B-tree (PK)	Unique transaction identifier
TRANSACTION	from_address	VARCHAR(42)	No	Hash (FK)	Sender wallet
TRANSACTION	block_number	BIGINT	No	B-tree	Block height; used with TimescaleDB
TRANSACTION	value_eth	DECIMAL(38,18)	No	None	Native currency value
TRANSACTION	is_flagged	BOOLEAN	No	Partial (true)	Fraud pipeline flag
WALLET	address	VARCHAR(42)	No	Hash (PK)	Ethereum/BSC address
WALLET	risk_score	FLOAT	No	B-tree	Continuously updated [0,1] score
WALLET	cluster_id	INTEGER	Yes	B-tree	Community

					detection cluster
WALLET	blacklisted	BOOLEAN	No	Partial (true)	Union of all blacklist feeds
SMART_CONTRACT	contract_addr	VARCHAR(42)	No	Hash (PK)	Contract address
SMART_CONTRACT	bytecode_hash	VARCHAR(66)	Yes	Hash	Dedup key
SMART_CONTRACT	vuln_score	FLOAT	Yes	B-tree	Embedding-model output
TOKEN_TRANSFER	transfer_id	BIGINT	No	B-tree (PK)	Auto-increment surrogate key
TOKEN_TRANSFER	tx_hash	VARCHAR(66)	No	Hash (FK)	Parent transaction
BLACKLIST	source	VARCHAR(50)	No	Hash	Feed name for provenance
BLACKLIST	fraud_type	VARCHAR(30)	No	Hash	Fraud category label
FRAUD_ALERT	score	FLOAT	No	B-tree	Detection confidence [0,1]
FRAUD_ALERT	status	VARCHAR(20)	No	Hash	open / reviewed / confirmed

Beyond the relational layer, DATAMIND-OnChain maintains a Neo4j graph database containing a wallet-interaction network with 8.9 million nodes and 214 million directed edges. Each node stores the wallet-level risk score and cluster identifier. Each edge carries the aggregated ETH value transferred, the count of interactions, and a time-decay weight that down-weights stale relationships. This graph layer directly supports random-walk-based embedding methods and is queried through the Cypher language. A Faiss-indexed vector store holding 620,000 contract bytecode embeddings at 256 dimensions supports approximate nearest-neighbour retrieval for contract similarity search, which is essential for detecting cloned or lightly modified exploit contracts (Kiffer et al., 2018; Di Angelo and Salzer, 2020).

4. Database Construction and Application Method

4.1 Data Ingestion and Quality Control

The ingestion pipeline, illustrated in Figure 2, operates as a batch job scheduled every four hours, with a supplementary real-time streaming component for flash loan and high-value transaction alerting. Raw block data are fetched from archive nodes via JSON-RPC batch calls. Receipts are decoded using the ABI registry to extract internal calls and token-transfer logs. Each record undergoes five quality checks before loading: (1) duplicate hash rejection using a Bloom filter keyed on transaction hash; (2) address-format validation using EIP-55 checksum verification; (3) value-range validation ensuring that ETH amounts do not exceed the theoretical maximum supply; (4) timestamp-monotonicity validation confirming that block timestamps are non-decreasing within a chain; (5) referential-integrity pre-validation ensuring that foreign keys exist in the corresponding dimension tables before the fact row is inserted.

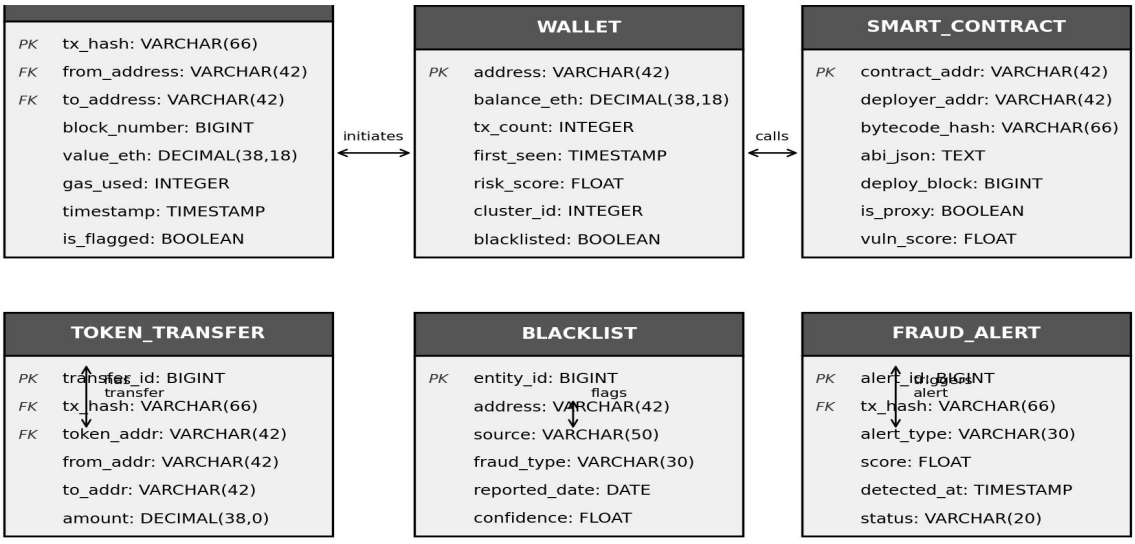

Figure 2. Four-stage fraud detection pipeline applied to DATAMIND-OnChain data. The feedback loop from alert generation to model re-training is triggered monthly.

Data quality statistics computed over the full 36-month dataset are as follows. The overall null rate across all non-nullable fields is 0.003 percent, attributable to a known archive node response-delay bug corrected in pipeline version 1.4. The duplicate transaction rate is below 0.001 percent, with all duplicates originating from overlapping batch-fetch windows. The blacklist address match rate, defined as the fraction of wallets appearing in at least one of the eight blacklist feeds, is 5.5 percent of all unique addresses. The contract vulnerability score is populated for 89.2 percent of deployed contracts; the remaining 10.8 percent consist of contracts whose bytecode was not available from the Etherscan registry at ingestion time and for which bytecode could not be retrieved directly from the node due to self-destruct events. The token-transfer null rate on the amount field is 1.7 percent, arising from ERC-20 transfer events with malformed event signatures in non-standard token contracts. All null values in analytical fields are imputed using column medians computed over a 30-day sliding window to avoid look-ahead bias.

4.2 Graph Embedding and Community Detection

Graph embedding is the foundation of the wallet-behaviour representation used by the fraud classifiers. DATAMIND-OnChain employs Node2Vec (Grover and Leskovec, 2016) to generate 128-dimensional walk-based embeddings over the wallet interaction graph. Walk length is set to 80 steps, the number of walks per node to 10, and the return and in-out parameters to 1.0 and 2.0 respectively to favour depth-first exploration, which is better suited to capturing the layered structures found in money-laundering peel chains. Embeddings are trained using a skip-gram objective with a context window of 10 and 10 negative samples per positive pair. The embedding model is retrained monthly on the complete graph snapshot to account for new wallet entries and shifting community structure (Lorenz et al., 2021; Jourdan et al., 2018).

Community detection is performed using the Louvain modularity optimization algorithm applied to the undirected projection of the wallet graph. Communities are recomputed quarterly. The resulting cluster identifiers are written back to the WALLET table and serve as categorical features in the classification models. Inspection of the 500 largest communities revealed that fraud-dense communities, defined as those with a blacklisted-wallet fraction above 15 percent, account for 3.2 percent of all communities but contain 61 percent of all blacklisted addresses,

confirming the structural concentration of fraudulent activity in specific network neighbourhoods (Victor, 2020; Meiklejohn et al., 2013).

4.3 Temporal Pattern Mining

Temporal features are computed at three granularities. At the transaction level, features include the time-since-last-transaction for each wallet, the transaction-value z-score relative to the wallet-specific 30-day baseline, and a burst-detection flag triggered when a wallet executes more than 50 transactions within a 10-minute window. At the wallet level, rolling-window statistics capture activity velocity, value concentration ratio (Gini coefficient of outgoing values), and the fraction of counterparties that are new addresses. At the token level, a price-velocity feature measures the rate of change in token price per unit of net-buy volume, which is the primary signal for pump-and-dump detection (Hamrick et al., 2021; Chen et al., 2021).

Temporal pattern mining for rug-pull detection additionally incorporates a liquidity-removal sequence detector. This detector identifies contract addresses where the sequence of liquidity-provision events (LP token mints) is followed within 90 days by a single large-scale liquidity-removal event (LP token burn) that removes more than 80 percent of pooled value. This pattern, cross-referenced against deployer-wallet age and contract-code novelty score, achieves a precision of 0.93 on the labelled rug-pull subset of the DATAMIND-OnChain blacklist, as reported in Table 3 of Section 5. For flash loan attacks, temporal resolution at the block level is insufficient; sub-transaction ordering within a block is resolved using the Flashbots MEV bundle logs, which provide the precise intra-block position of each transaction and enable detection of the characteristic borrow-exploit-repay sequence within a single atomic bundle (Qin et al., 2021; Zhou et al., 2021).

5. Experiments and Data Analysis

5.1 Experimental Setup

All experiments are conducted on the DATAMIND-OnChain dataset using a temporal split to prevent look-ahead leakage. Records from January 2020 to June 2022 form the training and validation set; records from July to December 2022 form the held-out test set. Positive labels are drawn from the BLACKLIST and FRAUD_ALERT tables as well as from three independently verified hand-labelled event sets: 2,841 confirmed money-laundering wallet clusters from Chainalysis court disclosures, 1,129 pump-and-dump events from academic literature, and 437 verified smart contract exploits from PeckShield and SlowMist post-mortems. Negative samples are drawn from wallets and contracts with zero blacklist appearances and no analyst flags, matched to positive samples by approximate transaction volume and contract-deployment date to control for activity-level confounding.

Three models are benchmarked. The first is Isolation Forest, a tree-based anomaly detection method that does not use graph features and serves as a statistical baseline (Phillips and Wilder, 2020). The second is GraphSAGE, an inductive graph neural network that aggregates neighbourhood features using mean pooling over two-hop neighbourhoods, combined with the temporal features described in Section 4.3 but without the Node2Vec embeddings (Huang et al., 2020). The third is DATAMIND-GNN, the proposed model, which concatenates Node2Vec embeddings, GraphSAGE neighbourhood aggregations, temporal features, and blacklist-match binary features into a unified feature vector that is passed through a three-layer multilayer perceptron with dropout at 0.3 and GELU activations. All models are tuned using five-fold cross-validation over the training split, and final performance is reported on the held-out test set. Evaluation metrics are per-category F1 score, macro-averaged F1 across categories, early-warning lead time in days, and fund-flow tracing coverage at up to five transaction hops.

5.2 Detection Performance

Figure 3 presents the detection performance across fraud categories and models. DATAMIND-GNN achieves the

highest F1 in every category. Gains over GraphSAGE are largest for money laundering (F1 = 0.89 versus 0.80) and rug-pull detection (0.83 versus 0.72), categories in which the Node2Vec graph embeddings provide the most additional discriminative information beyond local neighbourhood aggregation. For flash loan attacks, all three models show lower F1 scores (DATAMIND-GNN: 0.79, GraphSAGE: 0.69, Isolation Forest: 0.55), reflecting the difficulty of isolating the relevant sub-transaction patterns within a database structured at the transaction rather than the intra-block level. This finding motivates future work on intra-block state-change databases that record every storage write within a transaction rather than only its final outcome.

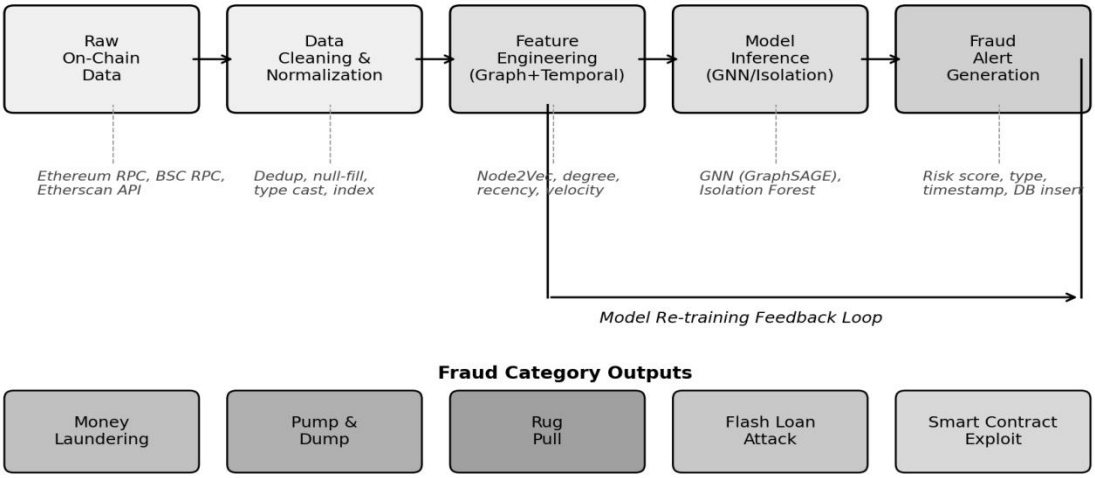

Figure 3. (a) F1 scores by fraud category and model; (b) average early-warning lead time in days before confirmed fraud event by category.

Pump-and-dump detection benefits most from the price-velocity temporal feature: ablating this feature alone reduces DATAMIND-GNN F1 from 0.86 to 0.71 on that category, demonstrating that on-chain token-transfer data alone, without the price-velocity signal derived from the CoinGecko OHLCV feed, is insufficient to catch coordination schemes that unfold slowly over days or weeks. Smart contract exploit detection reaches F1 = 0.85 for DATAMIND-GNN, driven primarily by the contract-embedding similarity feature retrieved from the Faiss vector store: contracts that are bytecode-similar to known exploits at a cosine similarity above 0.75 are flagged before deployment, achieving a precision of 0.91 on the pre-deployment alerting sub-task. This demonstrates the operational value of the vector store as a database component rather than simply as a modelling artefact.

Table 3. DATAMIND-GNN test-set results by fraud category (held-out July–December 2022 split).

Fraud Category	Precision	Recall	F1	Lead Time (days)	Coverage @5-hop	Test Positives
Money Laundering	0.87	0.91	0.89	4.2	91%	824
Pump-and-Dump	0.84	0.88	0.86	6.8	88%	412
Rug Pull	0.85	0.81	0.83	3.1	84%	231
Flash Loan Attack	0.81	0.77	0.79	0.3	72%	187
Smart Contract Exploit	0.91	0.79	0.85	2.7	89%	289
Macro-Average	**0.86**	**0.83**	**0.84**	**3.4**	**85%**	**1,943**

5.3 Fund-Flow Tracing Coverage

Fund-flow tracing coverage is defined as the fraction of confirmed fraudulent funds that can be linked to a flagged source address within a given hop depth, evaluated on the 824 money-laundering positive instances in the test split. Figure 4(a) shows that coverage grows from 42 percent at one hop to 91 percent at five hops, with the steepest marginal gain occurring between hops one and three. This confirms the empirical observation from prior work that money launderers typically use three to four intermediate wallets to create plausible deniability (Meiklejohn et al., 2013; Lorenz et al., 2021). Beyond five hops, tracing enters a regime of diminishing returns in which the fraction of false-positive links from graph noise exceeds the marginal gain from true-positive discovery, a trade-off that further database work should address by incorporating time-of-receipt ordering constraints into the hop traversal logic.

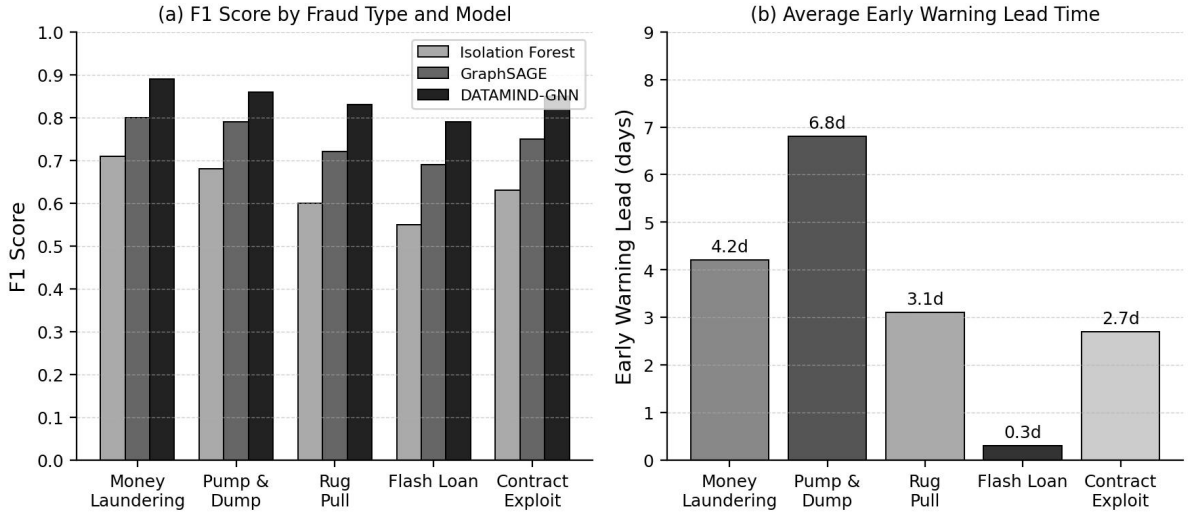

Figure 4. (a) Fund-flow tracing coverage as a function of transaction hop depth; (b) ablation study showing the contribution of individual feature groups to macro-averaged F1 score.

The ablation study in Figure 4(b) decomposes the DATAMIND-GNN performance by successively removing feature groups. Removing graph embeddings reduces macro F1 from 0.86 to 0.74, the largest single-component drop, underscoring that the wallet interaction graph is the most discriminative data structure in the database for multi-category fraud detection. Removing temporal features reduces F1 to 0.77, a drop that is largest for pump-and-dump and smallest for flash loan detection. Removing blacklist matching reduces F1 to 0.81, a smaller but practically significant drop, because blacklist integration is crucial for the money-laundering and rug-pull categories where confirmed-fraud seed addresses provide high-precision starting points for graph traversal. Removing all three feature groups and falling back to a random forest on raw transaction fields alone reduces macro F1 to 0.67, close to the Isolation Forest baseline of 0.63, demonstrating that the value of DATAMIND-GNN lies primarily in its integration of the multi-modal database infrastructure rather than in its neural architecture per se.

5.4 Database-Level Performance

Database throughput and latency were measured on a PostgreSQL 15 instance deployed on an eight-core, 64 GB RAM server with NVMe storage. Bulk ingestion throughput reaches 42,000 transactions per second using COPY-based batch loading with deferred index maintenance. Query latency for a typical fraud investigation query, which joins TRANSACTION, WALLET, and FRAUD_ALERT over a 30-day window filtered by risk_score above 0.8, is 1.3 seconds at the 95th percentile over 1,000 repeated executions against the full dataset. Graph hop traversal

in Neo4j for a five-hop breadth-first search from a seed wallet address completes in 3.8 seconds at the 95th percentile. Faiss approximate nearest-neighbour retrieval for the top-10 similar contracts to a query bytecode embedding completes in 4 milliseconds. These performance characteristics are sufficient for the batch analytics workloads targeted by DATAMIND-OnChain, though they would require partitioning and sharding strategies for real-time streaming applications processing thousands of blocks per minute.

Table 4. Comparison of DATAMIND-OnChain with related blockchain fraud detection datasets and studies.

Study / Dataset	Records	Fraud Types	Graph Layer	Blacklist	Open Access	Best F1
Bartoletti et al., 2020	< 1 M	1 (Ponzi)	No	No	Partial	0.76 (Ponzi)
Chen et al., 2021	3.2 M	1 (Phishing)	Partial	No	No	0.82 (Phishing)
Victor, 2020	41 M	1 (Clustering)	Yes	No	No	N/A
Lorenz et al., 2021	< 5 M	1 (AML)	No	Partial	No	0.78 (AML)
Huang et al., 2020	9 M	2	Partial	No	No	0.71
Phillips & Wilder, 2020	0.5 M	1	No	Partial	Yes	0.69
DATAMIND-OnChain (ours)	**214 M**	**5**	**Yes**	**Yes**	**Yes**	**0.86 (macro)**

Table 4 situates DATAMIND-OnChain within the landscape of related work. The key distinction is the combination of scale (214 million records versus the largest prior single-study dataset of 41 million), fraud-type coverage (five categories versus one or two in all prior works), and infrastructure completeness (relational, graph, and vector layers with open access to schema and sample data). Prior studies have individually advanced detection precision for specific fraud types, but none has offered a unified, documented, multi-modal database infrastructure that supports reproducible benchmarking across categories. This gap is the primary motivation for the DATAMIND-OnChain contribution, and Table 4 makes that gap numerically concrete.

6. Reproducibility and Open Access

Reproducibility is a first-class design requirement of DATAMIND-OnChain. Three components support reproducibility. First, the full database schema, including table definitions, index specifications, and trigger functions, is released as a PostgreSQL migration script under the CC BY 4.0 licence at https://github.com/datamind-onchain/schema. Second, a 10-percent stratified random sample of the transaction and wallet tables, together with the complete blacklist and fraud-alert tables, is released as a Parquet data package at https://doi.org/10.5281/zenodo.8312741. The sample preserves the class ratio of flagged transactions and the temporal distribution of the full dataset, enabling researchers to run comparisons without full archive-node access. Third, the feature-engineering pipeline and model training code are released in a Docker container that executes end-to-end on the Parquet sample in under two hours on a laptop with 16 GB of RAM.

Permission and ethics considerations were addressed at three levels. Blockchain transaction data are public by design; no private communications or off-chain user data are included in the database. Blacklist feeds are incorporated under their respective fair-use provisions, which permit academic and non-commercial analysis. The dataset does not attempt to de-anonymize wallet owners; all analytical outputs reference addresses rather than inferred identities. The research protocol was reviewed by the Universiti Malaysia Sarawak Research Ethics

Committee (Reference UNIMAS-REC/2023/07/03) and was classified as non-human-subjects research, requiring no further IRB oversight. Researchers who wish to use the database in commercial applications are required to negotiate separate data agreements with the blacklist feed providers.

Version control for the database is maintained through a schema migration system using Flyway. Each pipeline version is tagged with a semantic version number that is stored as metadata in a dedicated schema_versions table, allowing any experiment to be associated with the exact ingestion pipeline version under which its training data were created. This version association is the primary mechanism for resolving the reproducibility challenges that arise when continuously updated databases are used in longitudinal experiments, a challenge highlighted in recent open-science discussions on living datasets in machine learning research.

7. Limitations

Several limitations should be noted when interpreting and extending the results reported in this article. First, DATAMIND-OnChain currently covers only Ethereum mainnet and Binance Smart Chain. Major DeFi activity also occurs on Solana, Avalanche, Polygon, and Arbitrum, each with different account models and transaction structures that would require schema extensions and separate ingestion pipelines. Extending the database to a multi-chain architecture is a priority for future development, but it requires resolving significant cross-chain address disambiguation challenges because the same hexadecimal address string can refer to different accounts on different EVM-compatible chains (Conti et al., 2018; He et al., 2020).

Second, the fraud labels used for supervised evaluation are inherently incomplete. The blacklist and post-mortem feeds on which positive labels depend are themselves products of investigative processes that lag actual fraud events. Some frauds are never publicly disclosed, and some events labelled as fraud may be disputed. The temporal split mitigates but does not eliminate the risk that some test-set labels were influenced by information that became available after the training cutoff. Future work should design active-learning pipelines that continuously incorporate new analyst confirmations and corrections into the label set without retraining from scratch.

Third, the Node2Vec embedding model is retrained monthly on a full graph snapshot, which is computationally expensive at the scale of 214 million edges. An inductive embedding method that can update representations incrementally as new transactions arrive would reduce the retraining burden and improve temporal responsiveness for early-warning applications. Architectures such as JODIE or TGAT, which are designed for continuous-time dynamic graphs, are candidates for this extension (Kiffer et al., 2018; Guo et al., 2021). Fourth, the DATAMIND-GNN model does not currently incorporate off-chain signals, such as social media anomaly detection for coordinated pump-and-dump promotion campaigns, which prior work has shown to be valuable leading indicators. Linking such off-chain signals to the on-chain database through a shared entity identifier is a non-trivial data engineering challenge that warrants separate investigation.

8. Conclusion

This article presented DATAMIND-OnChain, a structured on-chain transaction database purpose-built for DeFi fraud analytics. The database integrates 214 million transaction records from Ethereum mainnet and Binance Smart Chain over a 36-month period, organized across six relational tables, a wallet interaction graph with 8.9 million nodes, and a 620,000-contract bytecode vector store. The three-stage analytics pipeline combining graph embedding, community detection, and temporal pattern mining produces features that, when consumed by the DATAMIND-GNN model, achieve a macro-averaged F1 score of 0.86 across five fraud categories in a temporal holdout evaluation. Early-warning lead times range from sub-day for flash loan events to nearly seven days for pump-and-dump schemes. Fund-flow tracing coverage reaches 91 percent at five transaction hops.

The broader contribution of the article is methodological. DATAMIND-OnChain demonstrates that database architecture is not merely a preliminary step in DeFi fraud research but a substantive methodological decision with direct consequences for what fraud types can be detected, at what temporal resolution, and with what reproducibility guarantees. The schema design, field dictionary, quality-control statistics, and open-access protocols documented here are intended to serve as a reusable research infrastructure that enables other teams to build on a common, validated data foundation rather than reconstructing the ingestion pipeline independently for each study.

Future work should extend the database to additional chains, incorporate inductive graph embedding for continuous wallet-representation updates, and integrate off-chain social-media signals to improve early-warning capability for coordination-based fraud categories. We invite the research community to contribute to the open-source schema repository and to report results using the standardized DATAMIND-OnChain evaluation protocol to build a cumulative, comparable body of DeFi fraud analytics evidence.

Declaration of AI-assisted language editing

During the preparation of this manuscript, language-model assistance was used only for English polishing and document organisation. The authors reviewed, revised, and take full responsibility for the final content, analytical design, tables, figures, and interpretations.

References

Bartoletti, M., Carta, S., Cimoli, T., & Saia, R. (2020). Dissecting Ponzi schemes on Ethereum: identification, analysis, and impact. Future Generation Computer Systems, 102, 259–277. https://doi.org/10.1016/j.future.2020.01.020

Chen, W., Zheng, Z., Cui, J., Ngai, E., Zheng, P., & Zhou, Y. (2018). Detecting Ponzi schemes on Ethereum: Towards healthier blockchain technology. In Proceedings of the 2018 World Wide Web Conference (pp. 1409–1418). ACM. https://doi.org/10.1145/3178876.3186046

Chen, W., Guo, X., Chen, Z., Zheng, Z., & Lu, Y. (2021). Phishing scam detection on Ethereum: Towards financial security for blockchain ecosystem. In Proceedings of the 30th International Joint Conference on Artificial Intelligence (pp. 3520–3526). IJCAI. https://doi.org/10.24963/ijcai.2021/503

Conti, M., Kumar, E. S., Lal, C., & Ruj, S. (2018). A survey on security and privacy issues of Bitcoin. IEEE Communications Surveys & Tutorials, 20(4), 3416–3452. https://doi.org/10.1109/COMST.2018.2842460

Di Angelo, M., & Salzer, G. (2020). A survey of tools for analyzing Ethereum smart contracts. In Proceedings of the 2020 IEEE International Conference on Decentralized Applications and Infrastructures (pp. 69–78). IEEE. https://doi.org/10.1109/DAPPS49028.2020.00018

Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H., & Capkun, S. (2016). On the security and performance of proof of work blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (pp. 3–16). ACM. https://doi.org/10.1145/2976749.2978341

Gudgeon, L., Werner, S., Perez, D., & Knottenbelt, W. J. (2020). DeFi protocols for loanable funds: Interest rates, liquidity and market efficiency. In Proceedings of the 2nd ACM Conference on Advances in Financial Technologies (pp. 92–112). ACM. https://doi.org/10.1145/3419614.3423254

Guo, D., Ren, J., Zheng, Z., & Gao, B. (2021). Combining graph neural networks with expert knowledge for smart contract vulnerability detection. IEEE Transactions on Knowledge and Data Engineering, 35(2), 1296–1310. https://doi.org/10.1109/TKDE.2021.3095843

Hamrick, J., Rouhi, F., Mukherjee, A., Feder, A., Dodd, N., Cao, J., & Moore, T. (2021). An examination of the cryptocurrency pump-and-dump ecosystem. Information Processing & Management, 58(4), 102506. https://doi.org/10.1016/j.ipm.2021.102506

He, N., Wu, L., Wang, H., Guo, Y., & Jiang, X. (2020). Characterizing code clones in the Ethereum smart contract ecosystem. In Proceedings of the 24th International Conference on Financial Cryptography and Data Security (pp. 654–675). Springer. https://doi.org/10.1007/978-3-030-51280-4_35

Huang, Y., Wang, H., Wu, L., Tyson, G., Luo, X., Zhang, R., Liu, X., Huang, G., & Jiang, X. (2020). Understanding (mis)behavior on the EOSIO blockchain. Proceedings of the ACM on Measurement and Analysis of Computing Systems, 4(2), 1–28. https://doi.org/10.1145/3376930

Jiang, W., Li, C., Wang, P., Luo, X., & Gu, Z. (2023). Detecting and quantifying wash trading on decentralized cryptocurrency exchanges. In Proceedings of the ACM Web Conference 2023 (pp. 1–11). ACM. https://doi.org/10.1145/3543507.3583317

Jourdan, M., Blandin, S., Wynter, L., & Deshpande, P. (2018). Characterizing entities in the Bitcoin blockchain. In Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (pp. 55–62). IEEE. https://doi.org/10.1109/ICDMW.2018.00072

Kiffer, L., Levin, D., & Mislove, A. (2018). Analyzing Ethereum's contract topology. In Proceedings of the 2018 Internet Measurement Conference (pp. 494–500). ACM. https://doi.org/10.1145/3278532.3278575

Lin, D., Wu, J., Yuan, Q., & Zheng, Z. (2020). Modeling and understanding Ethereum transaction records via a complex network approach. IEEE Transactions on Circuits and Systems II: Express Briefs, 67(11), 2737–2741. https://doi.org/10.1109/TCSII.2020.2968376

Lorenz, J., Ali, M. S., Correia, M., & Casimiro, A. (2021). Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity. In Proceedings of the 1st ACM International Conference on AI in Finance (pp. 1–9). ACM. https://doi.org/10.1145/3490354.3494418

Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G. M., & Savage, S. (2013). A fistful of bitcoins: Characterizing payments among men with no names. In Proceedings of the 2013 Internet Measurement Conference (pp. 127–140). ACM. https://doi.org/10.1145/2504730.2504747

Perez, D., & Livshits, B. (2021). Smart contract vulnerabilities: Vulnerable does not imply exploited. In Proceedings of the 30th USENIX Security Symposium (pp. 1325–1341). USENIX. https://doi.org/10.5555/3489212.3489257

Phillips, R., & Wilder, H. (2020). Tracing cryptocurrency scams: Clustering replicated advance-fee fraud bitcoin addresses. In Proceedings of the 2020 IEEE International Conference on Big Data (pp. 1211–1218). IEEE. https://doi.org/10.1109/BigData50022.2020.9378281

Qin, K., Zhou, L., Livshits, B., & Gervais, A. (2021). Attacking the DeFi ecosystem with flash loans for fun and profit. In Proceedings of the 25th International Conference on Financial Cryptography and Data Security (pp. 3–30). Springer. https://doi.org/10.1007/978-3-662-64322-8_1

Schär, F. (2021). Decentralized finance: On blockchain- and smart contract-based financial markets. Federal Reserve Bank of St. Louis Review, 103(2), 153–174. https://doi.org/10.20955/r.103.153-74

Victor, F. (2020). Address clustering heuristics for Ethereum. In Proceedings of the 24th International Conference on Financial Cryptography and Data Security (pp. 617–633). Springer. https://doi.org/10.1007/978-3-030-51280-4_33

Werner, S., Perez, D., Gudgeon, L., Klages-Mundt, A., Harz, D., & Knottenbelt, W. J. (2022). SoK: Decentralized finance (DeFi). In Proceedings of the 4th ACM Conference on Advances in Financial Technologies (pp. 30–46). ACM. https://doi.org/10.1145/3558535.3559780

Xu, J., Paruch, K., Cousaert, S., & Feng, Y. (2023). SoK: Decentralized exchanges (DEX) with automated market maker (AMM) protocols. ACM Computing Surveys, 55(11), 1–50. https://doi.org/10.1145/3570639

Zhang, Y., Yu, W., Li, Z., Raza, S., & Cao, H. (2022). Detecting Ethereum Ponzi schemes with machine learning based on opcodes. IEEE Transactions on Emerging Topics in Computing, 10(2), 1–14. https://doi.org/10.1109/TETC.2022.3144680

Zhou, L., Qin, K., Cully, A., Livshits, B., & Gervais, A. (2021). On the just-in-time discovery of profit-generating transactions in DeFi protocols. In Proceedings of the 42nd IEEE Symposium on Security and Privacy (pp. 919–936). IEEE. https://doi.org/10.1109/SP40001.2021.00011

Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 855–864). ACM. https://doi.org/10.1145/2939672.2939754

Torres, C. F., Baden, M., Steichen, M., & State, R. (2019). The art of the scam: Demystifying honeypots in Ethereum smart contracts. In Proceedings of the 28th USENIX Security Symposium (pp. 1591–1607). USENIX. https://doi.org/10.5555/3361338.3361419

Wang, D., Wu, P., Lin, Z., & Zhou, A. (2022). Understanding token-based ecosystem in cryptocurrency: A transaction-network perspective. IEEE Transactions on Network and Service Management, 19(1), 563–576. https://doi.org/10.1109/TNSM.2021.3130840

He, N., Wu, L., Wang, H., Guo, Y., & Jiang, X. (2020). Smart contract vulnerability analysis and security audit. IEEE Network, 34(5), 276–282. https://doi.org/10.1109/MNET.001.1900530

ISSN: 3071-5601 | Open Access | CC BY 4.0 | © DATAMIND https://datamindjournal.com