



SLID Scam Detection under Temporal Drift: A Practical Framework for DeFi Security [†]

Minh Trung Tran ^{1,*}, Nasrin Sohrabi ², Brayden Killeen ¹, Zahir Tari ¹ and Tony McGrath ³

¹ School of Computing Technologies, RMIT University, Melbourne, VIC 3000, Australia

² School of Information Technology, Deakin University, Melbourne, VIC 3125, Australia

³ Novatti Group Ltd, Melbourne, VIC 3000, Australia

* Correspondence: minh.trung.tran@student.rmit.edu.au

[†] Extended from: Adaptive Detection of DeFi SLID Scams: A Data-Driven and Industry-Oriented Framework for Large-Scale DeFi Security. In Proceedings of the 17th Australasian Information Security Conference (AISC 2026), Melbourne, VIC, Australia, 11–12 February 2026.

How To Cite: Tran, M.T.; Sohrabi, N.; Killeen, B.; et al. SLID Scam Detection under Temporal Drift: A Practical Framework for DeFi Security. *Pragmatic Cybersecurity* **2026**, *1*(3), 18. <https://doi.org/10.53941/pc.2026.100018>

Received: 10 April 2026
 Revised: 7 August 2026
 Accepted: 13 August 2026
 Published: 1 October 2026

Abstract: The Slow Liquidity Drain (SLID) scam has recently become a significant problem for decentralized exchanges (DEXs) and the decentralized finance (DeFi) environment in general. As SLID evolved from rug-pull scams toward cryptocurrency transaction fraud, several studies have implemented and published methods of detecting SLID scams, including heuristic-based and machine-learning (ML) based techniques. However, the applications of those detection methods to the real running DeFi application on blockchain are limited due to several constraints. Specifically, the newly arrived datasets that were collected from recent DEX activities have revealed that SLID behavior could evolve, which leads to the obsolescence and ineffectiveness of the previous methods when applying static thresholds to the detection in a real-time system. In this paper, we present a data-driven revalidation of SLID detection under various DeFi conditions and discover a robust method for threshold and feature-importance updating as the SLID behavior evolves over time with new data emerging. From the findings, we propose two practical and applicable models: A slow-adaptive solution that leans toward stable detection over a long period of time, and a fast-adaptive solution for more robust and real-time-sensitive detection. Those models together transform the theoretical detection methods to adapt to the real-world system that requires robustness and adaptability to various SLID evolutions.

Keywords: decentralized finance; ethereum; anomaly detection; blockchain; DEX; machine learning

1. Introduction

The rise of DeFi (decentralized finance) has benefited the whole blockchain environment by turning the networks into a financial ecosystem with various finance-focused permissionless applications to cryptocurrency, including asset creation/management/trading through Decentralized Exchanges (DEXs) [1–4]. With simple operations to create a cryptocurrency and a liquidity pool (LP) for trading, making assets available to a large number of users, the environment has lowered the barrier to entry and become an ideal place for malicious actors to launch their fraud on a large scale [5,6]. In recent years, we observed numerous reports of fraudulent activities around cryptocurrency on DEX, from simple liquidity pool manipulation in rug-pull to higher smart contract abuse scams [7,8]. Nevertheless, with the constant growth of the user population on blockchain and DeFi applications and DEXs, the number of scam variations and their complexity has grown correspondingly, turning it into a vital problem for product owners, researchers, and users.

One of the most recent variations of cryptocurrency scams is known as Slow Liquidity Drain (SLID) [9] that has several similarities to rug-pull. Unlike the rug-pull methods of asset exploitation that are identifiable by the



characteristic of abruptly collapsing liquidity pools (LPs), the SLID scam owner gradually withdraws the pool value over long periods under small-sized profit-taking activities (of withdrawal/sell). With the slow and gradual development of these exploitations, the SLID liquidity pool can maintain a legitimate appearance to the user (through data, price chart and online alerts) and maintain a longer community engagement while continuing their fraudulent actions. Although SLID gained the attention of the community with the detection methods proposed, those detection methods are still inapplicable to real-world blockchain decentralized applications (dApps). Since the proposed detections heavily rely on fixed feature threshold assumptions and offline verification on data, they face several challenges, including maintaining the accuracy with the updated SLID behavior without retraining the model and maintaining a balance between robustness and alert confidence.

In this work, our experiment results demonstrate that SLID is highly sensitive to the data collection window length (of data used to generate ML-features and run mathematical heuristic from proposed detection), and the optimal time-range window for generating detection-required features will shift depending on how the scam behavior emerges over time. We also discovered that the feature importance also changes over time, which will affect the effectiveness of the previous detection methods, with the static features driving the detection. These 2 new findings make the previously proposed method appear unstable and unreliable for real-time deployment in industrial dApps. Moreover, the proposed methods lack the flexibility for real-world applications where different enterprises might have different requirements for detection and resource tolerance (hardware power and data availability).

This paper is an extension of one of our earlier works on SLID scam evaluation [10], which first recognized the instability and unreliability of static threshold and proposed a more robust framework. In this paper, we explore beyond threshold and time-range window adaptability for detection and focus on SLID detection deployment in a more practical landscape (of real-world robust systems). We support multiple deployment modes to satisfy various distinct use cases and resource preferences by proposing an adaptive framework that tunes time-range selection as a system-controllable parameter. We have conducted empirical/scam evolution analysis and ML-metrics characterization and feature-weight re-evaluation in order to identify the most reliable time-range window parameter and the most suitable model for any given condition, making the framework integrable into real-world DeFi applications.

2. Related Work

DeFi (Decentralized Finance) has been widely recognized as a main factor for the development of the blockchain environment and plays a vital role in attracting a huge number of users to use their provided diversity of financial products [1], including DEXs—where users can create, provide liquidity and buy/sell cryptocurrencies [11,12]. Various studies have been published to study the optimal financial design that combines the strength of a decentralized system (from blockchain) and finance characteristic to propose a more useful DeFi protocol that benefits the financial activity on blockchain, including automated market maker (AMM) studies and liquidity pools price dynamics that create standards for creating/trading DEX assets [6,13].

With rapid development and huge attraction to users, DeFi and DEX easily become a primary target for scammers to try to launch their attack for financial exploitation. One of the most popular classes of DEX scams is to exploit the financial value of cryptocurrency by abrupt liquidity withdrawal/selling (rug-pull) [7,14,15] and smart contract malicious modification to create traps (honeypot) [8,16,17]. These scams are designed for fast exploitation and are short-lived, leading to numerous evolutions later that try to enlarge the size and length of the scam by slower-paced action of malicious activity, including SLID [9].

3. Research Motivation

The challenges in SLID scam detection on decentralized exchanges come from the nature of the slow and gradual exploitation by the pool owner: any attempt at static threshold assumptions in heuristic/ML-feature extraction can reduce the robustness of the detection in real-world scenarios. In the proposed work, the ML-based detection could flag a pool as SLID from its data of a static 57-day time-range window. However, our evaluation result using the same pipeline and detection method with newly updated data suggests otherwise. We reproduced the experiment and detection systems (both heuristic and ML-based) used for SLID detection and applied them to the updated DEX dataset until January 2025 and discovered that the static threshold has led to reduced detection accuracy. From the new empirical results that we conducted, the reliable time-range window for data collection and feature extraction to achieve the same accuracy (same precision as the previous paper claimed) is now 64 days minimum (demonstrated in Section 4). This finding proves that the static threshold cannot maintain high and stable SLID detection over time and motivates us to explore the dynamic options that fit with contemporary DeFi conditions.

Besides the detection instability caused by the fixed minimum time-range window in data collection and feature engineering, SLID detection performance also relies on temporal drift in the pool's feature behavior. From our analysis in Section 4.5, SLID scammers have the tendency to, over time, reduce the owner profit extraction frequency, lengthen order intervals to a longer period of time, and create changes in feature signal that were once detected and analyzed in previous SLID research. We propose the adaptive frameworks in Section 5 to decouple ML inference from observation window selection and support configurable time-range windows across different detection modes. These modifications will be later proven to balance the accuracy, latency of detection, alert volume and users' requirements (on data/hardware power) without depending on any single static configuration as stated in the original SLID detection methods.

4. Empirical Re-Evaluation with Updated Data

For the first step to implement an optimal SLID detector that is integrable into an industry-standard application, this section presents an empirical re-evaluation of SLID detection using updated real-world data. Following the motivation in Section 3, we extend data collection to January 2025 and re-run existing ML-based detectors to examine how detection accuracy varies with different time-range windows. Our primary goal in this section is not to improve SLID detection accuracy or compare modeling techniques, but to isolate how fixed observation window assumptions behave as datasets evolve over time.

4.1. Data Collection, Labeling and Preprocessing

We collect an expanded dataset covering decentralized exchange activity up to January 2025. Compared to datasets used in prior SLID studies, this dataset includes a larger number of liquidity pools, longer pool lifetimes, and additional market periods. In total, the dataset contains 358,412 liquidity pools and over 1.02 billion DEX transactions from the six largest DEXs operating on Ethereum [12,18–21] (for PancakeSwap, its Ethereum deployment). This adds approximately 39,000 pools and over 100 million new transactions, collected between mid-2024 and January 2025, on top of the baseline research database. From the new dataset of Ethereum liquidity pools (LPs), we label pools using the SLID baseline study's heuristic [9]. The heuristic flags a pool as SLID when its owner exhibits a sustained pattern of small, repeated value extractions (liquidity withdrawals or sells) over a long horizon that cumulatively drains a substantial share of the pool's value, without the abrupt liquidity collapse that characterizes rug-pulls. The heuristic's reliability was established in the baseline work; since our goal is comparability with the baseline rather than heuristic improvement, we adopt its output as ground-truth labels in the same way as prior work. Running this heuristic on the newly collected data detects 1429 new SLID LPs created within the second half of 2024 (on top of 3117 detected SLID from previous work [9]), making 4546 total SLID LPs involved in our study.

Afterward, the data collection and preprocessing steps proceed as follows, following the same pipeline as prior work to ensure comparability (by rebuilding the same process of the baseline study using their open-source [22]). For every LP, we collect on-chain events, including LP creation details, liquidity mint/burn event details, swap history, and reserve-update events by hosting a Geth (Go Ethereum) node in fast-sync mode, accessed through its HTTP JSON-RPC interface using web3.py (the full collection pipeline is available in the released source code [22]). Each recorded transaction includes: block number and timestamp, the event type, the addresses involved (sender, recipient, and pool contract), and the token amounts moved into or out of the pool. We then reconstruct LP time series of reserves, trading activity, and owner behavior, which form the sole input to the feature extraction step. The full dataset covers LP activity from 2019 to January 2025 across the six DEXs, with SLID pools accounting for 1.27% of all pools (Table 1). For each liquidity pool, we extract features based on owner activity, user activity, pool dynamics, and profit-related signals within a fixed observation window starting from pool deployment (demonstrated in Figure 1).

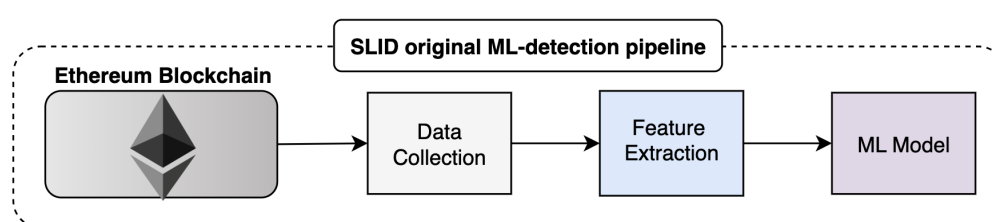


Figure 1. Original proposed SLID detection process.

To provide a more granular understanding of how the expanded dataset differs from the baseline, Table 1 summarizes the distribution of SLID and non-SLID liquidity pools by DEX platform.

We can observe the highly skewed SLID distribution among the DEXs. Uniswap V2 takes a high proportion with 62.6% of all detected SLID, which is reasonable given its dominance in popularity compared with other DEXs (as shown in the total pool count, indicating that most assets have been created there). The dominance of Uniswap V2 remains strong with the newly collected data, with 892 new SLID detected from this DEX, which is notably higher than any other DEX in newly recorded SLID. Curve appears to have become a preferred platform for SLID scammers with the highest SLID rate among all collected liquidity pools, with 1.78%, although having the least total pool count. We also conduct analysis to determine whether pool characteristics have evolved over time. Figure 2 demonstrates the SLID changes in two characteristics of *median pool lifetime* (days between pool deployment and last activity) and *median daily drain rate* (percentage of value from the pool that was exploited by the owner daily on average) across different years.

Table 1. SLID distribution among the six DEXs (until January 2025). New (H2'24): SLID pools newly detected in the second half of 2024; the bold row reports column totals.

DEX Platform	Total	SLID	Rate (%)	New (H2'24)
Uniswap V2	198,412	2847	1.43	892
Uniswap V3	67,234	412	0.61	156
SushiSwap	41,876	589	1.41	178
PancakeSwap	32,145	398	1.24	112
Balancer	12,389	187	1.51	54
Curve	6356	113	1.78	37
Total	358,412	4546	1.27	1429

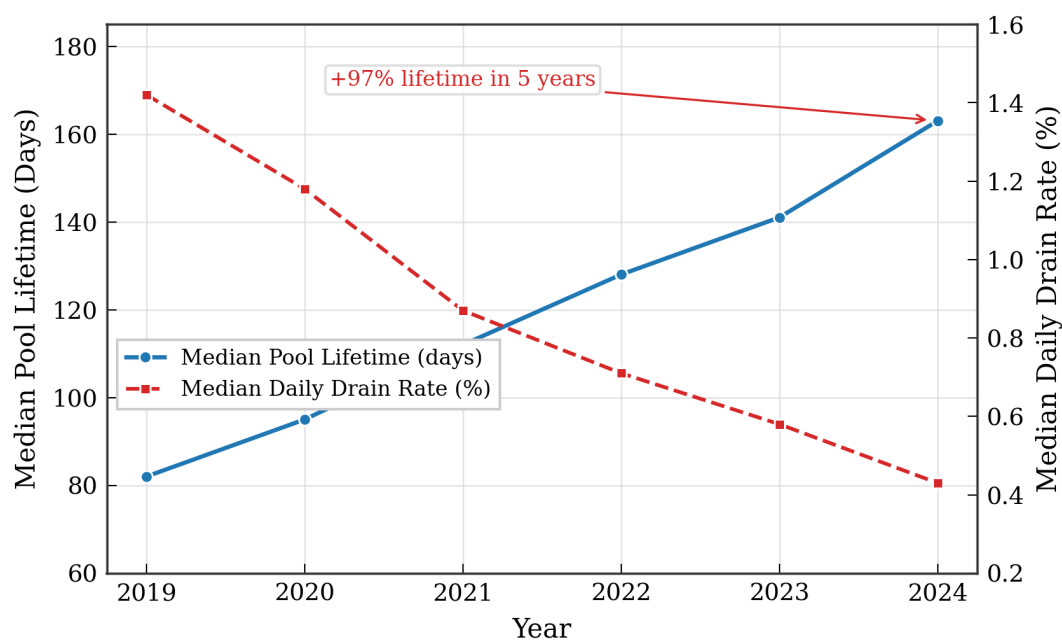


Figure 2. SLID median pool lifetime and daily drain rate change from 2019 to 2024.

The trend is clear from the data, indicating that SLID pools are operating for longer durations while extracting value more slowly. Between 2019 and 2024, median pool lifetime nearly doubled from 82 to 163 days, while the median daily drain rate fell from 1.42% to 0.43%. This slow evolution is consistent with the upward shift in required detection windows observed in the yearly evaluation (Section 4.4), as slower drain rates produce weaker early-stage signals. Features that rely on cumulative profit extraction or withdrawal frequency require more observation time to accumulate sufficient signal, which helps explain why the minimum detection time-range window has been increasing year-over-year, and further supports the case for adaptive window selection.

4.2. ML SLID Detection Models Rebuilding

We rebuild the three ML models used in previous SLID detection research with the same configuration settings and hyperparameters (including Random Forest [23], Logistic Regression [24] and XGBoost [25]). The

input features follow the four categories of the baseline study (see more feature details and calculation in [9]): *owner activity* (e.g., frequency and size of owner withdrawals/sells and the intervals between owner transactions), *user activity* (e.g., number of unique traders and buy/sell volume balance), *pool dynamics* (e.g., liquidity level, reserve trajectory, and volatility), and *profit signals* (e.g., cumulative owner value extraction relative to pool size). Keeping the original design of models allows us to observe the change in SLID detection accuracy when new data arrives and their robustness to various time-range window selections. To analyze the impact of time-range window selection, from the previous time-range window used in the baseline SLID research (between 56 and 267 days after deployment), we vary the observation window for our analysis in the surrounding zone between 35 days and 85 days after pool deployment. For each time-range window, models are trained with that day-specific extracted features from the SLID and non-SLID LPs (similar technique to baseline research). Two complementary evaluation protocols are applied to the resulting per-window datasets. Since a single hold-out split provides no variance estimate, the 95% confidence intervals in Table 2 are computed via 10-fold stratified cross-validation on the per-window data, and cross-validation is used solely for these interval estimates; all point-estimate results (Figure 3, Tables 3–5) are computed on the held-out test set of an 80/20 train–test split stratified by class label. Table 6 summarizes the split protocol behind each reported experiment. Detection performance is evaluated using accuracy as the primary metric, as it aligns with the cut-off threshold argument from the baseline research.

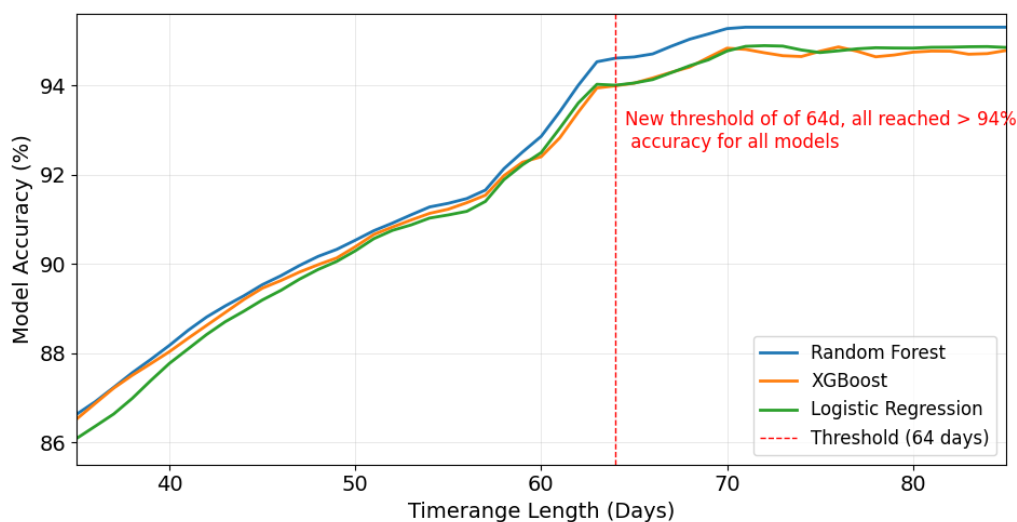


Figure 3. SLID detection accuracy by models in different time-range window data based on the 2025 updated dataset.

Table 2. Detection accuracy (%) at selected time-range windows, reported as mean $\pm$ half-width of the 95% confidence interval over 10-fold cross-validation (CV). Log. Regr.: Logistic Regression.

Days	Random Forest	XGBoost	Log. Regr.
40	86.3 $\pm$ 1.8	87.1 $\pm$ 1.6	84.9 $\pm$ 2.1
50	90.2 $\pm$ 1.4	90.8 $\pm$ 1.3	88.7 $\pm$ 1.7
57*	91.4 $\pm$ 1.2	92.1 $\pm$ 1.1	91.5 $\pm$ 1.4
64†	94.3 $\pm$ 0.9	94.7 $\pm$ 0.8	94.1 $\pm$ 1.1
70	95.1 $\pm$ 0.7	95.6 $\pm$ 0.6	94.8 $\pm$ 0.9
80	95.8 $\pm$ 0.6	96.2 $\pm$ 0.5	95.4 $\pm$ 0.8

* Previous threshold; † Newly discovered threshold.

Table 3. XGBoost results on SLID detection in various time-range windows.

Days	Precision	Recall	F1-Score
40	72.4	68.1	70.2
50	81.3	78.6	79.9
57	85.7	83.2	84.4
64	91.2	89.8	90.5
70	93.1	91.7	92.4
80	94.8	93.5	94.1

Table 4. Minimum time-range window for yearly SLID detection to reach 94% detection accuracy.

Year	2019	2020	2021	2022	2023	2024
Min. days	55	56	58	59	61	69

Table 5. Yearly XGBoost feature importance at 64-day window (↑/↓: substantial increase/decrease; ↗: slight increase; →: stable).

Category	2019	2020	2021	2022	2023	2024	Trend
Owner Activity	38.2	36.7	33.1	30.8	28.4	24.6	↓
User Activity	18.4	19.1	21.3	23.7	25.9	28.2	↑
Pool Dynamics	25.1	25.8	27.4	27.2	27.8	29.1	↗
Profit Signals	18.3	18.4	18.2	18.3	17.9	18.1	→

Table 6. Training/validation split protocol for each reported experiment. All splits are stratified by class label and include both SLID and non-SLID LPs.

Experiment	Split Protocol	Train/Eval.
Figure 3 (accuracy vs. window)	80/20 hold-out	80%/20%
Table 2 (95% CIs)	10-fold CV	90%/10% per fold
Table 3 (precision/recall/F1)	80/20 hold-out	80%/20%
Table 4 (yearly min. window)	80/20 hold-out	80%/20% per year
Table 5 (feature importance)	80/20 hold-out	importances from model trained on the 80% split

4.3. Changes in Detection Performance over Different Time-Range Windows

Figure 3 demonstrates the changes in each ML model for SLID detection with the corresponding selected time-range window for feature extraction. The shorter this time-range window is, the faster and more robust the detection model can perform in detecting the SLID from its behaviors. In previous research, the lowest reported number is 57 days to maintain 94% of accuracy, which is no longer applicable to the new data and the new pivot point to get to the previous accuracy standard of 94% is now 64 days with the arrival of new data when we rerun the experiments.

The shift in this minimum required time-range window to maintain the stable high accuracy can be observed in all models, showing that the shift comes not from model instability itself, but primarily from the newly arrived transaction features (until January 2025). This observation indicated that the SLID detection accuracy is influenced by the data patterns of SLID characteristics, which might change and evolve through time, and pointed out that the original method of using the static time-range window threshold might limit the robustness and adaptability of the detection activities in the future with new real-world updated data.

To better understand performance variability for industrial deployment, we report per-model accuracy at key time-range window thresholds with 95% confidence intervals computed over 10 stratified cross-validation folds (Table 2). As stated in Section 4.2, this cross-validation run serves only to estimate the intervals in this table; all other reported results use the stratified 80/20 hold-out protocol (see Table 6).

With the newly updated recent DEX data, all three models' accuracy falls below the previous standard of 94% SLID detection accuracy when applying the old time-range window of 57 days, with the upper confidence of 92.6%, 93.2% and 92.9%. Applying the newly found 64-day window, all models' mean accuracy has exceeded the target (of 94%), with narrower confidence intervals than at any shorter window, which supports the robustness of the new time-range window. Accuracy alone can be misleading in imbalanced detection settings. Since SLID pools constitute only ~1.27% of all pools, we additionally report precision, recall, and F1-score for XGBoost (the best-performing model) across key time-range windows (Table 3).

Using the previous window threshold of 57 days, XGBoost observed a recall of 83.2%, which rose to 89.8% after applying the new window threshold. The gaps in precision and recall of the XGBoost model in SLID detection across various window lengths (Table 3) also validated that the dynamic window is more efficient in detecting SLID compared to the proposed fixed window in previous work. These trade-offs between window length and

precision enable a more adaptive framework (more detail in Section 5) where applications/developers can choose the appropriate precision-window options based on their preference and available resources.

4.4. Yearly Evaluation

In order to evaluate the stability of different time-range windows, we conduct a yearly analysis focusing on SLID detection from pools created in the corresponding year only. From 2019 to 2024, we repeat the detection process (feature extraction with selected time-range windows, heuristic and ML detection) to evaluate the minimum window to reach the standard accuracy (of 94%, stated in previous work) for each year. The evaluation result is demonstrated in Table 4, which indicates that the minimum window for the yearly detection to reach the standard accuracy is unstable and changes yearly (with a slight but gradual increment for every year from 55 days for 2019 SLID detection to 69 days in 2024 detection).

Significantly, in the last recorded year of 2024, the minimum window increased notably from 61 days (2023) to 69 days (2024). The sudden increment has a significant impact on the global optimal detection threshold: despite the fact that in this extended work, we only add about half a year of new data, the aggregate threshold for the minimum optimal time-range window has shifted from 57 to 64 days on average. This effect is due not only to the larger gap between recent and past detection windows, but also to the significantly higher number of SLID scams that occurred during this period (1429 new SLID LP cases detected in the second half of 2024 were added to the analysis). Together, these factors amplify the impact of recent data, accelerating the breakdown of static detection thresholds inferred from previous datasets.

4.5. Feature Importance Analysis across Data Vintages

In order to find out whether similar scam traits remain distinguishable as SLID behavior changes over time, we evaluate the change in relative importance of trait groups over yearly transaction data from DEXs. Applying the Gini importance index from XGBoost, we calculate the average feature importances that are grouped by category within our new 64-day observation window (Table 5 summarizes the result and Figure 4 visualizes it).

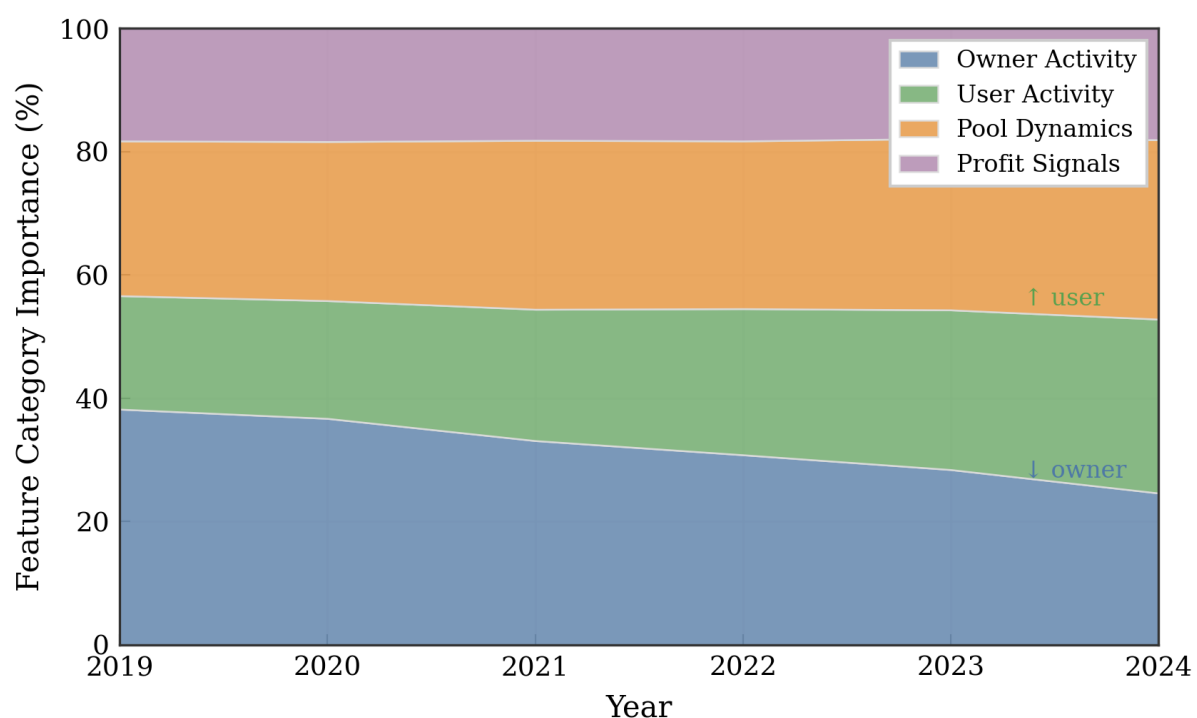


Figure 4. Feature-importance shifts over time between different category groups.

The changes in feature importance yearly have been gradual but toward a significant trend. The group of owner activity features, which in the past acted as the strongest SLID behavior (confirmed by previous work), in the latest year of 2024 has decreased in importance index (from 38.2% in 2019 to 24.6% in 2024). Meanwhile, user activity features in combination have increased their total importance (from 18.4% to 28.2%, over the same selected period) and are close to surpassing the owner activity features group in 2024. The most reasonable explanation is that SLID scam owners have adapted to the current market (where common users started to become aware of SLID operations) and have

gone further to minimize their visibility on-chain through reducing activities/footprints (with even fewer transactions, longer intervals between withdrawals/sells). Some sophisticated SLID pools in 2024 showed evidence of fabricated transactions (wash trading using trading bots) to inflate trading volume metrics and conceal illegal exploits.

From a practical perspective to create an industry-integrable detection framework, we can conclude that monitoring detection accuracy is not enough and operators should also monitor stability at the feature level. Even if detection accuracy remains 94%, a significant change of importance in any feature groups can also indicate that the current feature set is getting obsolete for a particular newer dataset. To address the issue, we later propose an adaptive framework (Section 5) to enable the inclusion of monitoring feature importance as a signal for model reconsideration.

5. Adaptive SLID Detection Framework for Industry Integration

From Section 4, the empirical evaluation has demonstrated that 1) the proposed SLID detection has an accuracy that significantly relies on the time-range window values for the pool's transaction feature extraction and 2) the importance of features can vary over time, which is a stronger indicator showing that a more adaptive solution is required. The 2 findings emphasized the real problems of stability and generality in deploying a real-time SLID detection in any running applications on and outside of the blockchain. In this section, to solve the discovered problem and enable a more adaptive SLID detection solution that is integrable to a real-time system, we propose a more adaptive SLID detection framework.

To address these requirements, we propose an adaptive SLID detection framework that separates the proposed ML models from the parameter of time-range window selection (in the previous step of pre-ML feature selection). Different from the original SLID detection in [9] that was built around a static window of time-range window, the framework allows the time-range window to be configured as a controllable system parameter that enables users to fine-tune detection behavior based on accuracy targets, latency constraints, and operational priorities. We describe two example implementation modes that reflect the two most common industry use-case groups.

5.1. Slow-Adaptive Detection Mode

The slow-adaptive detection mode is designed for industrial environments/applications that either prioritize stability, auditability, and long-term monitoring or lack more intensive resources for constant model re-training (including compliance teams, blockchain analytics providers, small startups, and applications that provide other blockchain functionalities as their main service and use SLID detection as part of their routine risk assessment). This framework includes four components with duties of:

- Data collection component: Constantly collecting DEX on-chain data for targeted pools that need to be flagged, and computing required features with given parameters (e.g., time-range windows).
- ML model component: Apply the pre-trained ML model for SLID detection to targeted pools with collected features.
- Window re-evaluation component: Actively comparing the accuracy of the model to the baseline heuristic and re-evaluating a more optimal time-range window if the current value causes inefficiency in detection.
- Controller component: Updating the new window (if the previous component computed it) and updating the data collection component with the new time-range window value.

Operational Workflow: In this framework, the detection process will maintain a fixed observation window over a long period of time (e.g., weeks or months). Within a certain interval of time (e.g., after 1 week/month), the window re-evaluation component will conduct quality assessment by comparing ML model detection results to the baseline heuristic and determine whether the ML performance still reaches the given threshold (e.g., $\geq 94\%$). If a change (whether an increase or decrease in accuracy) is detected, the controller component will update the time-range window parameter in the data collection component in corresponding ways (increase the windows if recent accuracy falls below the standard and vice versa). Otherwise, if the accuracy remains stable in the latest re-evaluation round, then the parameter remains unchanged. This process is illustrated in Figure 5.

Trade-Offs: This framework aims to optimally and naturally integrate with batch analysis systems, dashboards, and alert review processes (small-task activities). The strength of this framework is low cost: the model's behavior changes infrequently (depending on the user's setting of updating and re-evaluation period). However, as discussed, the slow adaptation method faces the trade-off of accuracy, where it is expected to be unable to catch SLID variation with sudden unexpected change in behavior/feature pattern (compared to the other common SLID behavior at the same time).

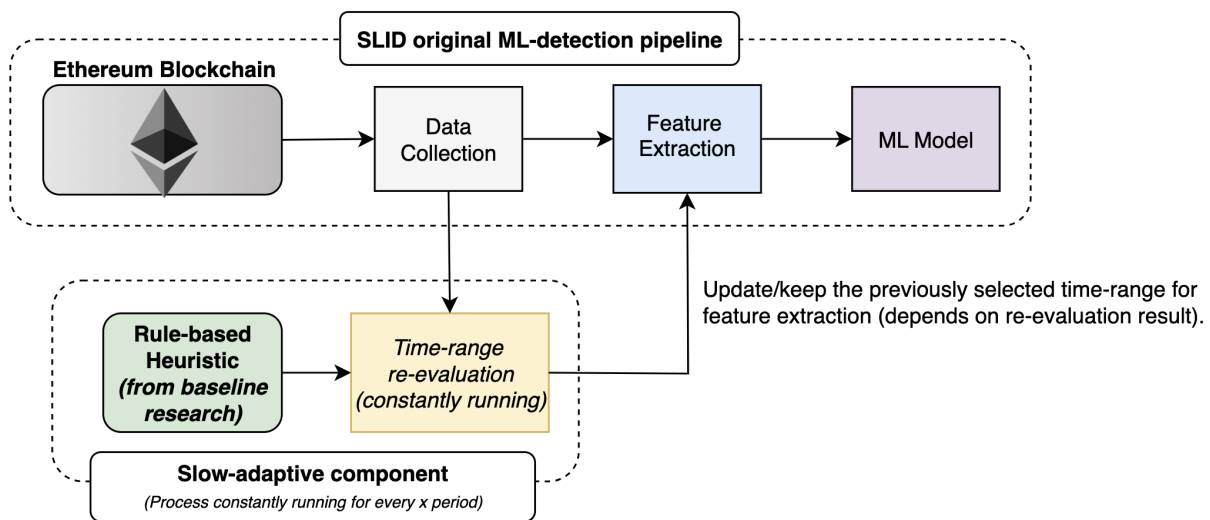


Figure 5. Components and workflow of proposed slow-adaptive SLID detection framework.

5.2. Fast-Adaptive Detection Mode

The fast-adaptive detection framework, in contrast, targets deployments with the capacity of heavy computation, where greater priority is placed on detection accuracy and robustness (e.g., fraud detection system, blockchain defense application, DEX protector that runs in real-time, etc). This framework mainly shares the same structure and process with the slow-adaptive framework, and has two additional components with extra responsibilities of:

- **Real-time window evaluation component:** Constantly re-evaluate different time-range windows based on recent ML model accuracy metrics to find the most optimal window parameter in real-time (unlike periodically evaluating the performance of this activity on the slow-adaptive framework).
- **Decision-making coordination layer:** manages detection results at different confidence levels.

Operational Workflow: Incoming data is collected and evaluated rapidly and constantly in parallel or sequentially across multiple time-range windows. Shorter timeframes are used to generate early, less reliable alerts, while longer timeframes are applied to confirm suspicious behavior before escalation. The window selector (interval) adjusts its operation based on current operational objectives, such as prioritizing rapid detection during high-risk periods or accuracy during execution phases. Besides, all the other components work similarly to those of the slow-adaptive framework. The process of this framework is demonstrated in Figure 6.

Window Update Rule: Formally, let W_t denote the observation window in force at re-evaluation round t , τ the accuracy target (e.g., $\tau = 94\%$), and $\mathcal{W} = \{W^{\min}, \dots, W^{\max}\}$ a candidate grid of windows. At every interval Δ (e.g., one week or one month), the window re-evaluation component computes $A_t(W)$, the detection accuracy over the most recently labeled pools for each candidate $W \in \mathcal{W}$, and the controller selects

$$W_{t+1} = \min\{W \in \mathcal{W} : A_t(W) \geq \tau\}, \quad (1)$$

i.e., the shortest window that still meets the target. If no candidate meets τ , the largest window is retained and the operator is alerted, since this indicates degradation that window selection alone cannot absorb (e.g., feature obsolescence, Section 4.5). To avoid oscillation, an update is applied only when $|W_{t+1} - W_t|$ exceeds a stability margin ϵ . The four operator-facing parameters of interval Δ , target τ , grid $\mathcal{W}$, and margin ϵ encode the trade-off between responsiveness and stability: the slow-adaptive mode uses a long Δ and conservative ϵ , while the fast-adaptive mode evaluates $A_t(W)$ continuously over multiple windows in parallel.

Trade-Offs: The primary trade-off of this framework is to get the robustness and accuracy in SLID real-time detection, coming with the unlimited scalability of the settings, which leads to a potentially large resource usage for running it properly. The framework allows users to adjust the number of windows involved in real-time window evaluation (where each will create a parallel session of evaluation of the ML performance from data collected in that window specifically and create an infinite scaling scenario) within a real-time, fast-paced period (e.g., every hour). The framework is suitable for dedicated security-focused applications and detection/defense services with available resources to run the large-scale framework for rapid SLID detection.

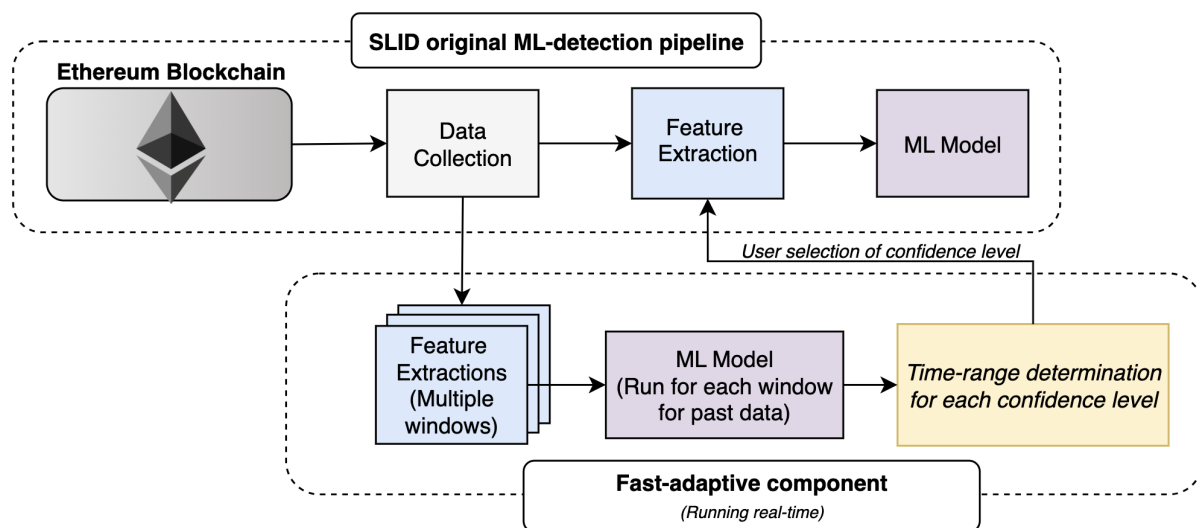


Figure 6. Components and workflow of proposed fast-adaptive SLID detection framework.

6. Discussion

The results presented in this article highlight the practical challenges of deploying SLID detection systems in a real-world DeFi environment. Our large-scale re-evaluation shows that detection performance is sensitive to both data recency and the selection of the observation time-range window. Analyzing the importance of additional features adds another dimension; we found that features that were previously crucial for detection are losing relevance as owner activity decreases and user activity increases. This pattern is consistent with the possibility that SLID operators may deliberately alter their on-chain behavior to avoid detection. This confirms that SLID detection is not a static problem, but rather evolves with market conditions, attacker strategies, and ecosystem development.

Rather than viewing this instability as a limitation of the ML models themselves, our findings suggest that much of the vulnerability arises at the system level, specifically in the selection and application of detection windows. By keeping the model architecture and features fixed, we isolate how changes over time affect both the overall detection accuracy and the relative importance of each feature, demonstrating that adaptability can be achieved without retraining or redesigning the model. This distinction is crucial for industrial deployments, where retraining can be costly, time-consuming, or potentially operationally risky.

The adaptive framework proposed in Section 5 translates these ideas into practical implementation strategies. By providing both slow and fast adaptation modes, the framework addresses a variety of operational objectives, from stability monitoring for regulatory compliance to rapid-response, security-based alerts. While these modes present different advantages and disadvantages in terms of complexity and responsiveness, they all share a common goal: to make SLID detection more robust against changes in real-world data, while maintaining scalability and comprehensibility.

7. Conclusions

In this paper, we reviewed SLID detection and the proposed methods and reran them using updated, large-scale DeFi data and discovered that using fixed parameters (of time-range windows) can cause detection instability and obsolescence in the long run if new behavioral evolution or temporal drift occurs. Our empirical evaluation and analysis have demonstrated that there is no such static optimal time-range window that can stably help the ML model catch SLID, and the values will change over time, which requires a more adaptive/dynamic approach. We also found that over time, the importance of different feature groups in detection is changing, suggesting that SLID scammers are actively adjusting their behavior to avoid being detected. In order to solve the discovered problems, we propose a SLID detection framework with adaptability that enables real-time integration. The framework separates ML models from time-range window data selection and supports multiple deployment modes with customized retraining frequency. The work bridges the academic findings of the Slow Liquidity Drain scam—an advanced variation of DEX scams—with its characteristics/detection methods to the industrial system/real-world applications that contain more advanced problems and requirements. We hope these findings will contribute to the development of SLID detection research and implementations in the future.

Author Contributions

M.T.T.: conceptualization, methodology, software, data curation, investigation, draft preparation; N.S.: supervision, methodology, validation, writing-review and editing; B.K.: software, validation, investigation, writing-review; Z.T.: supervision, writing-review and editing; T.M.: resources, project administration, funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding

The project was funded by a CRC-P grant (grant number PRJ00001687-Fast and Secure Crypto Payments for E-Commerce Merchants) and supported by Novatti Group Ltd.

Institutional Review Board Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

Not applicable.

Conflicts of Interest

The authors declare no conflict of interest.

Use of AI and AI-Assisted Technologies

During the preparation of this work, the authors have used LLM (ChatGPT-4o) for writing improvement and to enhance the content's readability. All of the paper's scientific ideas, experimental design, and result discussion originate from the authors. After using this tool, the authors have reviewed the manuscript and take full responsibility for the content of the published article.

References

1. Werner, S.; Perez, D.; Gudgeon, L.; et al. SoK: Decentralized Finance (DeFi). In Proceedings of the 4th ACM Conference on Advances in Financial Technologies, Cambridge, MA, USA, 19–21 September 2022; pp. 30–46.
2. Jiang, E.; Qin, B.; Wang, Q.; et al. Decentralized Finance (DeFi): A Survey. *arXiv* **2023**, arXiv:2308.05282.
3. Chen, Y.; Bellavitis, C. Blockchain Disruption and Decentralized Finance: The Rise of Decentralized Business Models. *J. Bus. Ventur. Insights* **2020**, *13*, e00151.
4. Momtaz, P.P. How Efficient Is Decentralized Finance (DeFi)? Available online: https://www.researchgate.net/publication/359551946_How_Efficient_is_Decentralized_Finance_DeFi (accessed on 7 August 2026).
5. ERC-20 Token Standard. Available online: <https://ethereum.org/en/developers/docs/standards/tokens/erc-20/> (accessed on 7 August 2026).
6. How Uniswap Works. Available online: <https://docs.uniswap.org/contracts/v2/concepts/protocol-overview/how-uniswap-works> (accessed on 7 August 2026).
7. Mazorra, B.; Adan, V.; Daza, V. Do Not Rug on Me: Leveraging Machine Learning Techniques for Automated Scam Detection. *Mathematics* **2022**, *10*, 949.
8. Torres, C.F.; Steichen, M.; State, R. The Art of the Scam: Demystifying Honeypots in Ethereum Smart Contracts. In Proceedings of the 28th USENIX Conference on Security Symposium, Santa Clara, CA, USA, 14–16 August 2019; pp. 1591–1607.
9. Tran, M.T.; Sohrabi, N.; Tari, Z.; et al. Slow Is Fast! Dissecting Ethereum's Slow Liquidity Drain Scams. *arXiv* **2025**, arXiv:2503.04850.
10. Tran, M.T.; Killeen, B.; McGrath, T. Adaptive Detection of DeFi SLID Scams: A Data-Driven and Industry-Oriented Framework for Large-Scale DeFi Security. In Proceedings of the 17th Australasian Information Security Conference (AISC 2026), Melbourne, VIC, Australia, 11–12 February 2026; pp. 98–103.
11. Xu, J.; Paruch, K.; Cousaert, S.; et al. SoK: Decentralized Exchanges (DEX) with Automated Market Maker (AMM) Protocols. *ACM Comput. Surv.* **2023**, *55*, 1–50.
12. Uniswap Main Page. Available online: <https://app.uniswap.org/> (accessed on 7 August 2026).
13. How Are Prices Determined? Uniswap. Available online: <https://docs.uniswap.org/contracts/v2/concepts/advanced-topics/pricing> (accessed on 7 August 2026).

14. Cernera, F.; La Morgia, M.; Mei, A.; et al. Token Spammers, Rug Pulls, and Sniper Bots: An Analysis of the Ecosystem of Tokens in Ethereum and in the Binance Smart Chain (BNB). In Proceedings of the 32nd USENIX Conference on Security Symposium, Anaheim, CA, USA, 9–11 August 2023; pp. 3349–3366.
15. Lin, Z.; Chen, J.; Wu, J.; et al. CRPWarner: Warning the Risk of Contract-Related Rug Pull in DeFi Smart Contracts. *IEEE Trans. Softw. Eng.* **2024**, *50*, 1534–1547.
16. Gan, R.; Wang, L.; Lin, X. Why Trick Me: The Honeypot Traps on Decentralized Exchanges. In Proceedings of the 2023 Workshop on Decentralized Finance and Security, Copenhagen, Denmark, 30 November 2023; pp. 17–23.
17. Shi, L.; Li, Y.; Liu, T.; et al. Dynamic Distributed Honeypot Based on Blockchain. *IEEE Access* **2019**, *7*, 72234–72246.
18. Curve Finance Main Page. Available online: <https://curve.fi/#/ethereum/swap> (accessed on 7 August 2026).
19. Balancer Main Page. Available online: <https://balancer.fi/> (accessed on 7 August 2026).
20. SushiSwap Main Page. Available online: <https://www.sushi.com/> (accessed on 7 August 2026).
21. PancakeSwap Main Page. Available online: <https://pancakeswap.finance/> (accessed on 7 August 2026).
22. SLID Anonymized Source Code for Reproduction. Available online: <https://anonymous.4open.science/r/SLID-sourcecode-B638/> (accessed on 7 August 2026).
23. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32.
24. Kleinbaum, D.G.; Klein, M. *Logistic Regression: A Self-Learning Text*; Springer: New York, NY, USA, 2002.
25. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.