Journal Pre-proof

Advanced Genetic Algorithm and Penalty Fitness Function for Enhancing DeFi Security and Detecting Ethereum Fraud Transactions

Arash Habibi Lashkari, Sepideh Hajihosseinkhani, Joshua Duarte, Isabella Lopez, Ziba Habibi Lashkari et al.

PII: S2096-7209(25)00103-4
DOI: https://doi.org/10.1016/j.bcra.2025.100376
Reference: BCRA 100376

To appear in: *Blockchain: Research and Applications*

Received date: 10 March 2025
Revised date: 17 April 2025
Accepted date: 8 June 2025

Please cite this article as: A.H. Lashkari, S. Hajihosseinkhani, J. Duarte et al., Advanced Genetic Algorithm and Penalty Fitness Function for Enhancing DeFi Security and Detecting Ethereum Fraud Transactions, *Blockchain: Research and Applications*, 100376, doi: https://doi.org/10.1016/j.bcra.2025.100376.

Advanced Genetic Algorithm and Penalty Fitness Function for Enhancing DeFi Security and Detecting Ethereum Fraud Transactions

Arash Habibi Lashkari*[a,b], Sepideh Hajihosseinkhani[b], Joshua Duarte[a], Isabella Lopez[a], Ziba Habibi Lashkari[c] and Sergio Rios-Aguilar[c,*]

[a]Behavior-Centric Cybersecurity Center (BCCC), School of Information Technology, York University, Toronto, Ontario, Canada
[b]Computer Science, Department of Computer Science and Engineering, York University, Toronto, Ontario, Canada
[c]School of Computer Engineering, Universidad Politécnica de Madrid, Madrid, Spain

ARTICLE INFO

Keywords:
Centralized Finance (CeFi), Decentralized Finance (DeFi), Financial Transactions Security, DeFi Security, Blockchain Security, Genetic Algorithm, Fraud Transactions

ABSTRACT

With the shift from Centralized Finance (CeFi) to Decentralized Finance (DeFi), financial transactions have become trustless and self-executing through blockchain platforms, creating new opportunities while exposing the ecosystem to significant fraud risks. However, due to the lack of centralized oversight and the vulnerabilities in the blockchain platforms, DeFi transactions still face several security challenges, including fraud, identity theft, insider threats, and data breaches. Various methods, including regulatory frameworks, machine learning (ML), and deep learning (DL) techniques, are employed to detect these threats, particularly fraud, in DeFi transactions. Although these approaches help identify fraudulent activities, they face challenges related to accuracy and zero-day attacks due to insufficient data and the complexity of emerging fraud patterns. This study presents a novel approach for detecting and profiling fraud attacks, including zero-day ones in DeFi transactions, thereby eliminating the reliance on wallet transaction history, a limitation that previous research has heavily depended on. The proposed approach leverages two key components: a novel analyzer named DeFiTransLyzer (V1.0) and an Advanced Genetic Algorithm (AGA) for fraud transaction profiling. DeFiTransLyzer extracts 79 features from transaction and wallet data. At the same time, the AGA incorporates advanced techniques, including Penalized Fitness Evaluation, Elite Retention Strategy, Dynamic Mutation Rate, and dynamic generation, to create precise fraud profiles. By focusing solely on transaction features, the model ensures that all fraudulent activities, including zero-day ones, initiated within the first transaction of a new account can be effectively detected, without relying on prior wallet activity. To address the scarcity of comprehensive validation datasets, we introduce BCCC-DeFiFraudTrans-2025, which comprises 1,026,867 annotated Ethereum transaction samples from the DeFi ecosystem.

Additionally, the study establishes two taxonomies for systematic classification, covering the literature on fraud detection and profiling methods. Experimental results demonstrate that the proposed method achieves superior accuracy, precision, and efficiency while offering interpretability through its profiling mechanism. These promising outcomes highlight the potential of AGA profiling to enhance the detection and identification of fraudulent activities, including zero-day ones within DeFi transactions, contributing to the security and resilience of blockchain-based financial systems.

1. Introduction

CeFi has long been the dominant structure for managing financial services, relying on intermediaries such as banks, exchanges, and payment processors to facilitate transactions and establish trust (1). CeFi platforms offer convenience but come with inherent limitations, such as high fees, limited transparency, and exposure to risks posed by centralized authorities, including fraud (2; 3). With the emergence of blockchain technology, DeFi has revolutionized the financial landscape by eliminating intermediaries through Smart Contracts (SCs) and decentralized protocols (4). This transition from CeFi to DeFi represents a significant evolution, enabling peer-to-peer transactions, improved financial inclusion, and enhanced security through transparency and immutability on the blockchain (5; 6).

fraud activities have shifted from CeFi to DeFi, introducing new security challenges (7). This study reviews existing research on fraud detection in DeFi, categorizing works based on targeted levels, including SCs (8; 9), transactions

*Corresponding author
ORCID(s):

(10; 11), and multi-levels (12; 13). Multi-level approaches involve solutions that combine SCs, transactions, and accounts. In this context, the account level specifically refers to wallet addresses. We examine the models and methods used at each level, covering both learning-based and non-learning-based techniques.

Learning-based detection approaches are gaining increased attention due to their ability to identify complex and evolving risks (14; 13; 15). Within learning-based approaches, hybrid methods combine the strengths of both ML and DL models, leveraging their adaptability to enhance fraud detection (16; 17; 18). Despite their advantages, hybrid approaches face challenges in balancing computational complexity while ensuring scalability without introducing inconsistencies. In contrast, non-learning-based approaches, such as regulatory compliance frameworks, provide simpler but less precise solutions (19; 20).

Systematic analyses provide comprehensive reviews of existing studies, offering a broad overview of the DeFi security landscape (21; 22). However, despite notable progress, significant challenges remain in effectively detecting fraud activities within DeFi systems (23; 24). A key challenge is the limitation of zero-day fraud detection in transactions. Current ML and DL models are designed to detect fraud, but they rely heavily on historical data, specifically related to wallet activity, making real-time detection complex (25; 26; 27). ML approaches utilize overly simplistic architectures that fail to capture the complexity of DeFi fraud (28). In contrast, DL models face criticism for their ”black-box” nature, which limits their transparency and adaptability in practical applications (29; 30; 31). They also struggle with insufficient and imbalanced datasets, relying heavily on the quality of training data, which restricts their generalizability to other blockchain platforms (32; 33; 34). Addressing these challenges is essential for enhancing DeFi security and expanding the scope of detection frameworks.

This study presents an advanced profiling approach to address the limitations of zero-day fraud detection in Ethereum-based transactions and effectively identify fraudulent patterns. Our proposed Advanced Genetic Algorithm (AGA) profiling method leverages principles of natural selection and genetics, enabling developers to simulate program behavior, quickly detect performance bottlenecks, and optimize execution (35; 36; 37; 38). This approach offers precise data on execution time at the function, method, and line-of-code levels while being cross-platform compatible and user-friendly. By profiling fraudulent patterns—defined as malicious or deceptive activities intended to exploit or manipulate the Ethereum network—and legitimate patterns, which refer to regular, trustworthy user behaviors that comply with the intended use of the platform, this integrated method significantly enhances the speed and accuracy of zero-day fraud detection within Ethereum transactions. As a result, it contributes to reinforcing the overall security and trustworthiness of DeFi ecosystems.

Furthermore, we constructed a new dataset, BCCC-DeFiFraudTrans-2025, to evaluate the effectiveness of our profiling approach. The dataset was curated by collecting comprehensive transaction data from Etherscan (39), a widely used blockchain explorer. It includes two primary categories of information: wallet-related data and transaction-related data, carefully selected to cover diverse samples of fraud and legitimate activities. The dataset is organized into JSON files, each corresponding to a specific wallet or transaction. It comprises 9,374 unique wallet addresses and 1,026,867 transactions, all labeled as fraud or legitimate.

Our contributions to this study are multi-faceted, offering significant advancements in DeFi fraud detection and profiling:

- To design and implement an advanced Ethereum transaction analyzer, DeFiTransLyzer-V1.0, for experiments demonstrating effective feature extraction and optimization techniques, enhancing DeFi transaction analysis performance.
- To design an advanced GA tailored for detection and profiling fraudulent activities, including the zero-day ones, incorporating four key components: penalized fitness evaluation, elite retention strategy, dynamic mutation adjustment, and dynamic generation, ensuring adaptability and robustness in detecting fraud.
- To create a large and diverse transaction dataset, BCCC-DeFiFraudTrans-2025, by collecting 1,026,867 samples from Etherscan (39), providing a comprehensive foundation for evaluating the proposed profiling model.

The remainder of this paper is organized as follows: Section 2 explores the existing body of literature in this research field, highlighting the limitations of current solutions. Section 3 provides a comprehensive discussion of our proposed model, including the development of DeFiTransLyzer-V1.0 and the AGA. Section 4 presents the detailed results of extensive experiments. Sections 5 and 6 offer an in-depth analysis of the research questions and propose common profiling patterns. Lastly, Section 7 presents the conclusions and outlines potential directions for future research.

2. Literature Review

Fraud has long plagued CeFi due to centralized control, making financial institutions susceptible to identity theft, money laundering, and unauthorized transactions (40; 41; 42). To combat this, CeFi organizations have adopted sophisticated fraud detection systems that combine rule-based techniques with AI for real-time prevention. With the emergence of DeFi, a blockchain-based financial paradigm that offers greater transparency and user control, new fraud challenges have arisen. The pseudo-anonymous nature of DeFi, along with SC vulnerabilities and the absence of regulatory oversight, has created novel opportunities for malicious behavior. As fraud shifts from CeFi to DeFi, there is a growing need for innovative detection solutions tailored to decentralized environments. This section explores this evolution and the corresponding advancements in fraud detection within the DeFi ecosystem.

2.1. Centralized Finance (CeFi)

CeFi refers to traditional financial systems governed by centralized authorities (43), yet fraud remains a significant concern despite regulated oversight (40). Early fraud detection in CeFi primarily relied on rule-based systems that flagged transactions based on fixed thresholds, though these approaches often lacked adaptability to new fraud patterns (42). With the advancement of technology, particularly AI and big data analytics, financial institutions began incorporating more sophisticated techniques. Lopez *et al.* (44) highlight the adoption of Neural Networks (NNs) and DL models, which analyze historical transaction data and behavioral trends to anticipate fraudulent activity. These modern systems enhance the detection of complex schemes such as identity theft and large-scale money laundering, pushing the boundaries of security in CeFi environments.

In parallel, the rapid digitization of financial services has led to the widespread use of ML algorithms and anomaly detection techniques for real-time monitoring of transactional data (41). Martin et al. (45) emphasize the role of both supervised and unsupervised learning models in enabling adaptive and automated fraud detection, significantly improving the accuracy of identifying novel fraud types. However, as Kim *et al.* (40) note, these advancements present new challenges, including data privacy risks, regulatory concerns, and potential algorithmic bias.

In a nutshell, fraud detection in CeFi has undergone significant changes, evolving from traditional rule-based systems to advanced AI-driven models and big data analytics. These innovations enable financial institutions to detect and respond to fraud activities with greater accuracy and scope. However, they also bring challenges such as transparency, speed, and efficiency, data privacy concerns, regulatory compliance, and biases in AI models. The following subsection examines fraud detection in DeFi, highlighting how its decentralized nature addresses the centralized challenges of CeFi.

2.2. Decentralized Finance (DeFi)

DeFi offers transparency, accessibility, and control through open, peer-to-peer financial transactions using blockchain technology (43). However, the emergence of DeFi has introduced critical challenges such as fraud and vulnerabilities (19; 33; 46). Unlike CeFi, which operates in regulated environments, DeFi is an unregulated and decentralized ecosystem, attracting malicious actors who exploit its anonymity and decentralization. This has created a demand for novel fraud detection methods tailored to the decentralized ecosystem (34). The following section examines the lifecycle of DeFi transactions and reviews recent research on detecting DeFi fraud.

2.2.1. DeFi Transaction Lifecyle

In the DeFi ecosystem, transactions follow a structured lifecycle comprising creation, verification, mining, and confirmation phases (47; 48; 49). The process begins with the **creation** phase, where users initiate transactions through blockchain wallets or dApps by specifying recipient addresses, gas fees, and amounts (47; 50). After digitally signing the transaction, it is broadcast to the blockchain network (51), ensuring cryptographic integrity and user authentication (48; 52). During **verification**, peer nodes inspect transaction validity, including checking balances, signatures, and format compliance (53; 51). Validated transactions enter the mempool, where they await miner selection—typically influenced by gas fees (47).

The **mining** phase involves incorporating verified transactions into blocks via consensus mechanisms. In PoW, miners compete to solve cryptographic puzzles (49; 54), whereas PoS selects miners based on staked cryptocurrency (48). Once a block is mined, it is distributed across the network (55), with each node independently verifying and appending it to their local blockchain copy (49; 56), completing the **confirmation** phase. This process enhances transparency, security, and efficiency compared to centralized systems.

In conclusion, the DeFi transaction lifecycle is composed of three fundamental phases: creation, verification, and confirmation. This structured process offers significant improvements over traditional centralized finance systems, particularly in terms of speed, efficiency, data privacy, and regulatory independence. In the final confirmation phase, transactions are independently verified by network nodes and, once successfully mined, permanently recorded on the blockchain—strengthening the overall integrity of the system. Nevertheless, each stage of the lifecycle remains vulnerable to fraud due to the open, programmable nature of SCs and inherent risks within transaction structures and wallets, which can be exploited by malicious actors.

2.2.2. DeFi Transaction Security

This subsection evaluates fraud detection within DeFi transaction by categorizing existing research according to the specific data sources of DeFi fraud they address —such as SCs, Transactions, and Multi-Level— and the methods they utilize. The reviewed studies present a variety of solutions, including diverse detection methods, comprehensive analyses, and the establishment of regulatory frameworks.

- ***SC Level***: Studies on the SC level are divided into two main approaches: detection approaches and systematic analysis. Detection approaches concentrate on identifying fraud by analyzing the SC source code. On the other hand, systematic analysis examines the SC's attack protocols and overall security.

  -Detection Approaches: In detection approaches utilizing learning-based techniques, researchers frequently apply ML and DL for fraud detection. Trozze *et al.* (8) investigate the rise of criminal activities, such as securities fraud, within the DeFi space. They propose an RF classifier to detect securities violations by analyzing the SC code of tokens, explicitly focusing on opcode-based features. Their method identifies DeFi projects involved in fraud by examining opcode frequency distributions in SCs. Ren *et al.* (57) introduce LookAhead, a model for detecting DeFi attacks that exploit SC vulnerabilities. They construct a large dataset of 210,643 contracts and select 375 adversarial examples to train a transformer-based model enhanced with multiple classifiers and meta-classifiers. Their approach analyzes malicious code patterns using function calls, control flows, and other structural features. Similarly, Gan *et al.* (9) utilize DL to develop DeFiAligner, an end-to-end system that detects mismatches between DeFi project documentation and deployed SCs. It leverages symbolic execution through SEVM to explore execution paths and uses LLMs to assess semantic alignment with project documentation, demonstrating high effectiveness on real-world datasets.

  In the non-learning-based category, Chen *et al.* (25) focus on identifying vulnerabilities in DeFi protocols, particularly flash loan attacks. They propose FlashSyn, an automated synthesis tool that models DeFi protocols using polynomial approximation and nearest-neighbor interpolation. FlashSyn generates and tests potential attack sequences within a forked blockchain environment, enabling the detection of flash loan vulnerabilities without relying on training data or learning-based inference.

  -Systematic Analysis: In the systematic analysis domain, attack analysis plays a key role in understanding fraud mechanisms within DeFi. Sun *et al.* (21) examine rug pull schemes, where developers abruptly withdraw liquidity or abandon projects, causing major losses. They propose the Rug Pull Analysis Framework (RPAF), which includes a literature review, a taxonomy of 34 root causes, dataset reconstruction, and evaluation of 14 detection tools. Carter *et al.* (22) investigate risk factors in DeFi, identifying five main categories: links to traditional finance, blockchain operational risks, SC vulnerabilities, governance and regulatory concerns, and scalability issues. They highlight that decentralization introduces new risks but may also strengthen financial systems, though its broader economic impact requires further study.

  Security analysis shifts focus toward underlying causes of DeFi fraud. Xue *et al.* (23) present a detailed survey on Ethereum-based DeFi fraud, identifying inherited financial system risks, architectural flaws, and development vulnerabilities. They evaluate current solutions and suggest future research directions like on-chain detection and transaction interception. Similarly, Qian *et al.* (24) explore SC vulnerability detection and automated repair. Their work categorizes DeFi attacks into six types, analyzes major incidents, and evaluates the effectiveness of tools, emphasizing the need for improved support in recovering high-level semantics and automating diverse repair tasks.

Protocol analysis further addresses system-level concerns in DeFi. Rabetti *et al.* (58) investigate the significance of auditing in promoting trust, security, and transparency. They discuss challenges such as cross-chain interactions, SC risks, and regulatory gaps. To mitigate these, the authors propose advanced auditing methods, including automation, AI integration, and continuous monitoring.

- ***Transaction Level***: At the transaction level, detection approaches are categorized into learning-based, non-learning-based, and hybrid methods. Learning-based approaches rely on labeled datasets to train algorithms, utilizing machine learning (ML), deep learning (DL), or a combination of both. Non-learning-based techniques, by contrast, depend on mathematical or rule-based logic, such as Cash Flow Tree Analysis, to detect fraudulent activities. In the ML subcategory, Aziz *et al.* (59; 10) propose a modified LGBM model using Euclidean distant structured estimation to detect Ethereum fraud. Their results demonstrate superior performance compared to RF, MLP, and KNN, particularly when using limited features. A follow-up study (10) further optimizes LGBM through hyperparameter tuning, yielding high accuracy again. Ravindranath *et al.* (11) enhance ML fraud detection by applying oversampling methods, such as SMOTENC, ADASYN, and K-Means SMOTE, thereby improving the performance of CATBoost and LGBM. Taher *et al.* (14) introduce a voting-based ensemble framework that combines high-accuracy ML detection with eXplainable AI, achieving strong results on real-world data.

  In the DL subcategory, researchers employ GNNs, Transformers, and CNNs to address fraud detection. Wang *et al.* (60) develop DeFiGuard, a GNN-based system that detects PMAs by transforming Ethereum transactions into cash flow graphs and training models to recognize manipulation patterns. Evaluated on 208 PMA and 2,080 non-PMA transactions, DeFiGuard outperforms baseline and other GNN-based models. Kanezashi *et al.* (15) compare homogeneous and heterogeneous GNN models in real Ethereum transaction networks, demonstrating the strengths of supporting multiple node and edge types in fraud detection. Similarly, Tan *et al.* (61) use web crawlers to label fraud addresses, build a transaction network, and apply a GCN informed by transaction volume and structural information, achieving high fraud detection accuracy.

  Furthering their work, Tan *et al.* (26) propose an enhanced DL framework for fraud detection by mining transaction behaviors from the public Ethereum ledger. Fraud and legitimate addresses are collected via web crawling, followed by the construction of a transaction network. A novel transaction behavior-based network embedding method is introduced to extract node features, and a Graph Convolutional Network (GCN) is utilized for classification. Their method effectively distinguishes between fraudulent and legitimate transactions, offering high performance in transaction-level fraud detection.

  The second group of learning-based methods centers on Transformer-based models, which learn from sequential inputs. An *et al.* (29) introduce Finsformer, a Transformer-based model enhanced with a cluster-attention mechanism for detecting financial attacks. By clustering similar transaction data and applying attention mechanisms within these clusters, Finsformer effectively identifies complex patterns of fraud. It outperforms RNN, LSTM, Transformer, and BERT models in precision, recall, accuracy, and F1-score. Similarly, Gunathilaka *et al.* (30) propose DeFiTrust, which combines temporal features from transaction logs with sentiment analysis of social media posts. These features are fused using a fully connected neural network to classify tokens as malicious or legitimate. Evaluated on real-world data, DeFiTrust detects scam tokens earlier and more accurately than existing models.

  The next group incorporates CNNs with RNNs for fraud detection. Smith *et al.* (62) present a hybrid deep learning approach that combines CNN and RNN models to analyze transaction data in DeFi networks. This model aims to improve fraud detection accuracy while reducing false positives. In the final subcategory of learning-based approaches—hybrid methods—Jia *et al.* (31) propose TLMG4Eth, which integrates ML and DL techniques. The model uses a Transaction Language Model (TLM) to convert numerical transaction data into sentence-like representations, capturing semantic features. It also builds a transaction attribute similarity graph and a wallet interaction graph. A deep multi-head attention network then integrates these features for comprehensive anomaly detection in Ethereum transactions.

  In the non-learning-based category, Wu *et al.* (19) introduce DEFIRANGER, which applies Cash Flow Tree Analysis to detect price manipulation attacks. The system reconstructs raw Ethereum transaction traces into Cash Flow Trees to identify high-level DeFi actions, such as trades and token transfers. When tested on over 350

million Ethereum transactions, DEFIRANGER successfully detects 432 price manipulation attacks, including four known incidents and five previously undisclosed (zero-day) vulnerabilities.

- ***Multi-Level***: Studies on the multi-level focus on multiple levels and are categorized into three branches: detection approaches, regulatory and monitoring techniques, and systematic analysis. (12; 16; 17).

  -Detection Approaches: In the learning-based category, ML techniques have been applied to detect fraudulent wallets and activities. Norouzi *et al.* (12) propose an ML classifier for identifying fraud wallets engaged in money laundering, phishing, and Ponzi schemes. Their model employs a novel feature selection strategy that combines Recursive Feature Elimination with Cross-Validation (RFECV) and RF Feature Importance (RF-FI), thereby enhancing classification accuracy for wallet-level fraud.

  Within DL methods, Wang *et al.* (13) introduce DeFiScanner, a DL-based system that detects logic vulnerabilities in SCs. DeFiScanner analyzes both global transaction features and local internal operations, integrating them using a fusion model to identify DeFi attacks. Evaluated on a dataset of 50,910 real-world Ethereum transactions, DeFiScanner outperforms traditional ML models in detecting complex vulnerabilities.

  Hybrid methods integrate both ML and DL to tackle fraud detection across different blockchain environments. Palaiokrassas *et al.* (16) propose a cross-chain fraud detection framework evaluated on transaction data from 23 DeFi protocols across 12 chains, using DNN, XGBoost, and a fine-tuned LLM to detect fraudulent wallets. Mothukuri *et al.* (17) present a multi-model AI framework that calculates a "TrustScore" for DeFi projects by integrating GPT for SC auditing, Prophet for price anomalies, FinBERT for sentiment analysis, and XGBoost for anomaly detection. Lastly, Palaiokrassas *et al.* (18) focus on DeFi credit risk, predicting wallet liquidations based on six transactional events and using SMOTE for oversampling. Among the tested models, CatBoost performs best, emphasizing the value of DeFi-specific features in risk prediction.

  Moving on to the non-learning category, Kitzler *et al.* (20) examine the complexity of DeFi compositions involving multiple interacting protocols within single transactions, which complicates risk assessment and interoperability. To address this, they develop an algorithm that decomposes protocol calls into nested building blocks, enabling detailed analysis and disentanglement of DeFi compositions. Applied to Ethereum data from 23 major protocols and over 10 million wallets, the algorithm effectively uncovers hidden risks. In the behavior-based category, Xie *et al.* (27) propose DeFort, a framework designed to detect price manipulation attacks in DeFi applications. DeFort captures real-time manipulation behavior, guides on-chain detection using a behavior model, and automatically identifies abnormal transactions, involved functions, attackers, and victims.

  -Regulatory and Monitoring Techniques: Research on regulatory and monitoring techniques highlights efforts to ensure DeFi compliance and detect fraud. Wang *et al.* (63) propose BLOCKEYE, a real-time attack detection system for Ethereum-based DeFi projects. Using symbolic reasoning, BLOCKEYE analyzes data flow in key service states, such as asset prices, to detect external manipulation and monitors off-chain transactions for policy violations. It successfully uncovers previously unknown vulnerabilities, demonstrating its effectiveness. Jin *et al.* (34) present Meta-IFD, a fraud detection framework that refines interaction behavior using generative and contrastive self-supervision. Luo *et al.* (28) introduce a comprehensive taxonomy of DeFi fraud across five project lifecycle stages and review detection techniques, showing that tree-based models are effective in early stages. At the same time, advanced graph-based methods perform better during maturity and decline.

  -Systemic Analysis: Investigations into systemic analysis emphasize understanding DeFi as an interconnected system, focusing on relationships, risks, and user behaviors. Carpentier *et al.* (64) explore profit-driven crime in DeFi by categorizing various crimes and mapping them to layers of the DeFi stack. They highlight how SC misuse and vulnerabilities can lead to significant financial losses, calling for improved cybersecurity through audits, user training, and awareness. Liu *et al.* (32) highlight the limitations of current practices, such as 2FA, and recommend enhancing user education and integrating decentralized regulatory mechanisms to enhance user security and decision-making. Weingartner *et al.* (65) compare DeFi risks to those in CeFi, distinguishing between systematic and unsystematic threats, such as SC flaws, governance issues, and scalability concerns. They propose the "risk wheel" as a practical tool for risk evaluation, management, and user education in DeFi contexts.

In summary, existing research demonstrates that fraud detection approaches are applied across multiple levels of the DeFi ecosystem. However, because not all DeFi transactions involve SCs, approaches that rely solely on SC

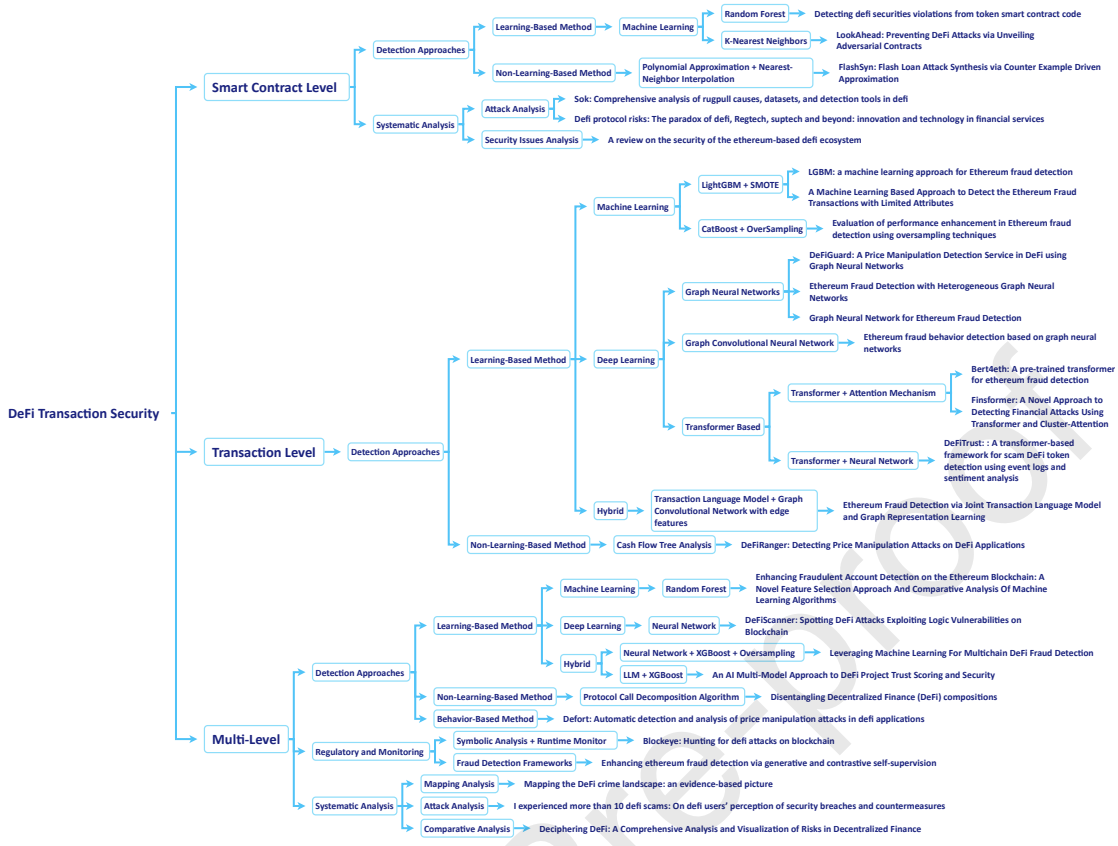

Figure 1: Taxonomy of Current Works

analysis—even when supplemented with other data—are insufficient for reliably identifying malicious activities. Most studies focus on transaction-level detection, where hybrid deep learning (DL) and machine learning (ML) models have demonstrated improved accuracy. Despite their effectiveness, these methods often rely on a wallet's transaction history, which limits their ability to detect zero-day attacks, such as fraudulent behavior in a wallet's initial transaction. Additionally, the complexity and limited interpretability of many advanced models can hinder their practical adoption. While detection strategies span various layers, their cross-level interactions remain underexplored, underscoring the need for more systematic evaluations. This research addresses that gap by focusing on transaction-level data to detect and profile fraudulent behavior—particularly zero-day attacks—with greater precision and efficiency.

2.3. Synthesis

In synthesizing the available literature on DeFi transactions security, we have developed a taxonomy illustrated in Figure 1. This taxonomy reveals that most research has concentrated on the transaction level. Among all levels, there has been a predominant focus on learning-based detection approaches, which are favored for their high accuracy in identifying complex risks within DeFi systems. In contrast, regulatory compliance frameworks offer more comprehensive and straightforward methods for risk identification but often lack the precision found in learning-based models. Systematic analyses offer a comprehensive examination of the DeFi landscape, frequently reviewing numerous articles to provide an in-depth overview of the DeFi environment. Despite these advancements, significant challenges in DeFi fraud detection remain:

- Inability to Detect Zero-Day Attacks: The detection method relies on historical data, such as wallet information, which limits its ability to identify zero-day attacks and detect fraud activity in a new wallet's initial transaction.
- Ineffective DeFi fraud Detection: Although learning-based detection methods are available, the ML techniques utilize overly simplistic architectures that fail to capture the complexities of DeFi fraud patterns. On the

other hand, more advanced DL approaches are criticized for their "black-box" nature, which makes them less transparent and more challenging to adapt for practical use.

- Real-Time Detection Challenges: The real-time detection performance of regulatory frameworks is limited due to the inflexibility of rule-based systems, highlighting the need for future improvements in dataset quality and model scope.
- Insufficient Data Size: Datasets with a low sample size provide an insufficient scope of fraud and impede model training.
- Imbalanced Data: Works that use datasets with a small fraction of fraud compared to the entire dataset are imbalanced, leading to model training and evaluation challenges.
- Platform Specificity: Current models and algorithms focus primarily on Ethereum, which limits their applicability to other blockchain platforms.
- Single-Level Concentration: Focusing on a single level of DeFi fraud, such as SC, transactions, or wallets, may not provide a multidimensional view of the fraud aspects.

Continued research and development are crucial to addressing the first five issues and enhancing the effectiveness of fraud detection and identification in DeFi. This includes focusing on multi-level approaches and enhancing dataset diversity. It is crucial to continuously evaluate and refine existing methods while exploring innovative techniques to advance DeFi fraud detection, as further discussed in the next section. This study introduces a new profiling technique that effectively identifies and detects zero-day fraud activity in the first transaction of a new wallet.

3. Proposed Model

This section presents a comprehensive methodology for profiling fraud and legitimate DeFi Ethereum-based transactions. The process begins with *Data Pre-Processing* (subsection 3.1), which establishes the framework's foundation by preparing transaction and wallet data. This stage ensures the data is cleaned, structured, and ready for subsequent analysis. Following this, the methodology progresses to *Feature Extraction* (subsection 3.2), where critical features are identified and extracted from the transactions and wallets. This stage is pivotal for uncovering the fundamental attributes necessary to distinguish between the characteristics of fraud and legitimate transactions.

The subsequent stage, *Profiling Model* (subsection 3.3), focuses on designing and implementing an AGA to profile fraud and legitimate transactions. The proposed enhancements consist of four key components: Penalized Fitness Evaluation, Elites Retention Strategy, Dynamic Mutation Rate, and Dynamic Generation.

The final stages, *Classification and Evaluation Criteria* (subsection 3.4), involve implementing a straightforward classifier to assess the performance of the AGA profiling model. These stages also define the metrics and standards used to evaluate the model's effectiveness. The overarching goal is to present an advanced analytical framework that integrates existing insights with novel methodologies, pushing the boundaries of DeFi transaction analysis. The overall architecture of the proposed model is illustrated in Fig. 2, with each stage elaborated upon in the following subsections.

3.1. Data Pre-processing

This subsection presents the initial stage of the proposed model, *Data Pre-processing*, illustrated in Fig. 2. This foundational stage focuses on collecting and processing data, a crucial step in establishing a practical framework for fraud detection and analysis. A key contribution of this stage is the introduction of a new dataset, BCCC-DeFiFraudTrans-2025, which is comprehensively detailed in subsection 4.1.2.

3.2. Feature Extraction

This section presents the second stage of the proposed model, Feature Extraction, as depicted in Fig. 2. The initial subsection provides a background on feature extraction in Ethereum-based transactions for fraud detection. The second subsection introduces a newly proposed analyzer, DeFiTransLyzer-V1.0, specifically designed to extract 79 features distributed across two main categories—wallets and transactions—and 13 subgroups.

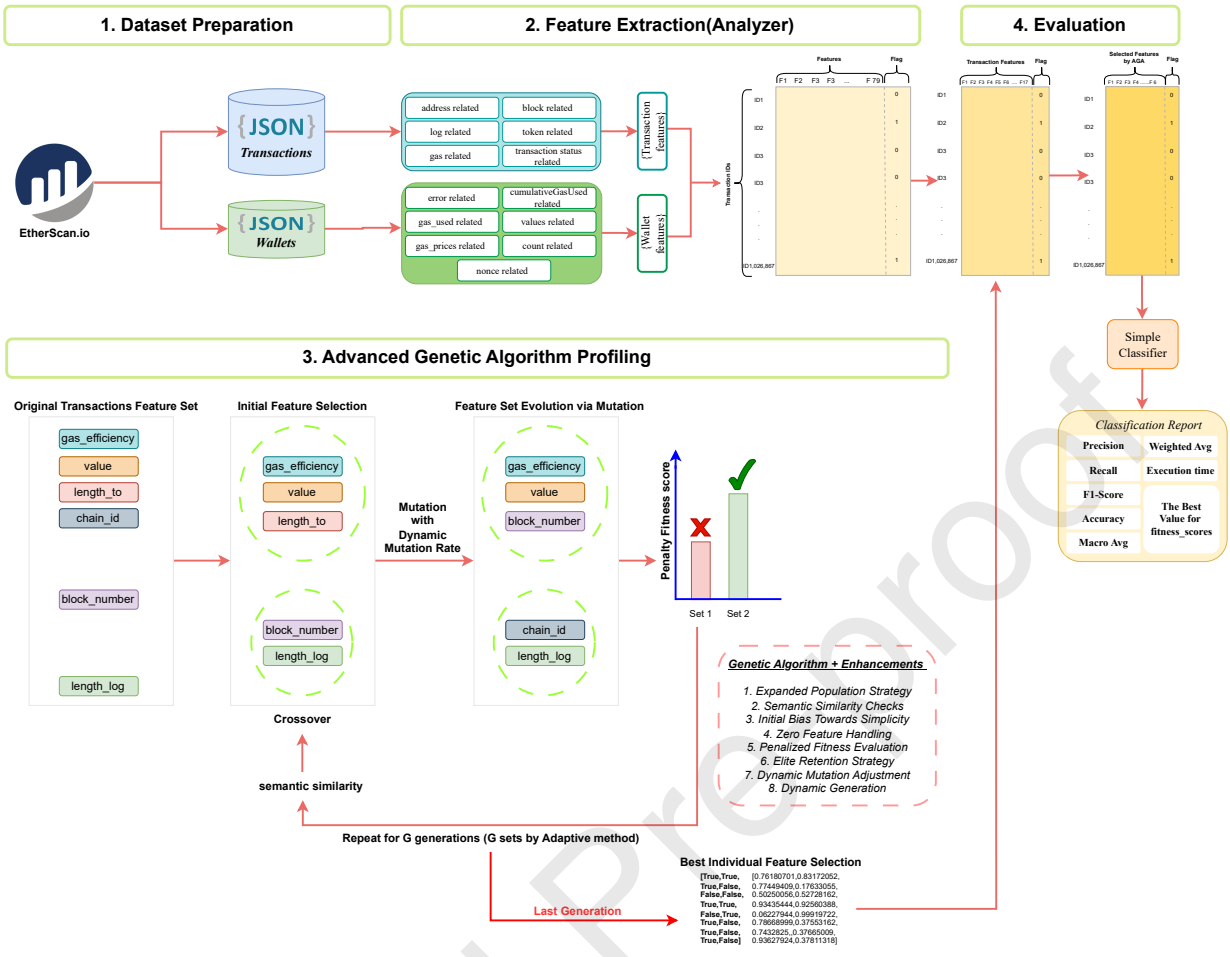

Figure 2: Architecture of the Proposed Framework

3.2.1. Feature Extraction Background

Feature extraction is a crucial component in analyzing Ethereum-based transactions, as it helps identify characteristics indicative of fraudulent activity and facilitates predictive modeling of such behaviors (37). This subsection offers a comprehensive background on feature extraction from Ethereum transactions, categorizing the features into structural and semantic types, and includes a detailed summary in Table 1.

The initial category of features comprises structural attributes. Research in this area primarily concentrates on quantifying specific parameters derived from Ethereum transactions. For instance, Hu *et al.* (33) develop a model that generates seven transaction attributes, covering both categorical and non-categorical types. This model categorizes non-categorical features, such as amount and count, through binning and includes attributes like address, wallet type, transaction direction (in/out), amount, count, timestamp, and position. Additionally, Aziz *et al.* (59**?**) investigate the incorporation of ERC20 features, transaction values, and addresses across various models, ensuring consistency in feature utilization while applying different analytical methods.

Similarly, Kanezashi *et al.* (15) track the volume and total value of transactions sent and received by an wallet node alongside metrics such as the in-degree and out-degree of transactions and PageRank scores. Moreover, Jin *et al.* (34) extract and analyzes comprehensive transaction data, including block numbers, timestamps, transaction costs, gas usage, emitted events, and associated wallet details, emphasizing the statistical analysis of specific transaction functions. Furthermore, Jia *et al.* (31) introduce a transaction language model that converts numerical transaction data into interpretable sentences, enhancing the model's ability to articulate and analyze transaction semantics explicitly.

Table 1
Background: Extracted Features

Cat.	Ref.	Description
Structural	(33)	Extracts seven transaction attributes combining categorical and non-categorical types with embedding techniques for feature encoding.
	(59**?**)	Includes ERC20 features, transaction values, and addresses.
	(15)	Tracks transaction counts sent to and received from a wallet node.
	(34)	Integrates global information such as block number, timestamp, transaction cost, and the gas used, with statistical features like the number of transfer functions.
	(31)	Implements a transaction language model to convert numerical data into interpretable transaction sentences, facilitating explicit semantic learning.
Semantical	(66)	Identifies transactions between two nodes using edge attributes like 'TimeStamp,' 'BlockHeight,' and 'Value.'
	(61)	Designs an Ethereum-based transaction network using an amount-based Node2vec approach.
	(19)	Constructs cash flow trees from raw Ethereum transactions to elevate low-level semantics to higher-level interpretations.
	(60)	Constructs cash flow graphs using Node Type, Transfer Frequency, Transfer Diversity, and Profit features.
	(13)	Integrates global and semantic features extracted from local models during events.

Moving to the next category, semantic features, this segment predominantly explores graph-based techniques to delve deeper into the meanings embedded within transaction data. Ratra *et al.* (66) construct a directed graph where nodes represent transaction wallets and edges represent transactional links, enriched with attributes like 'TimeStamp,' 'BlockHeight,' and 'Value.' This graph also incorporates additional node attributes such as wallet balance, in-degree, and out-degree, enriching the structural representation of the transaction network.

Similarly, Tan *et al.* (61) refine feature extraction from Ethereum transaction networks using an amount-based Node2vec approach, effectively capturing significant transaction amount characteristics. Wang *et al.* (13) streamline the normalization of unstructured emitted events and extracts both transaction-related and semantic features using a holistic model.

Lastly, focusing on cash flow, Wu *et al.* (19) begin by constructing a cash flow tree from raw Ethereum transactions, which are elevated from low-level data to high-level semantic interpretations, covering aspects like token trade and liquidity operations. Similarly, Wang *et al.* (60) develop a methodology to evaluate cash flow graphs using four distinct features: Node Type, Transfer Frequency, Transfer Diversity, and Profit, capturing diverse trading behaviors and integrating various analytical methods for model training.

In conclusion, feature extraction from Ethereum transactions has progressed significantly, yet the categories of features currently utilized reveal a marked disparity in complexity and coverage that impairs their practical utility. Structural features, as identified by works such as (34; 59; 15), tend to oversimplify the dynamic and intricate nature of Ethereum-based transactions. Conversely, semantic features (19; 60; 13), while promising in their analytical depth, present substantial challenges in terms of complexity and resource consumption. These semantic analyses often employ complex graph-based methods, necessitating significant computational resources and advanced data management skills to derive meaningful insights. Consequently, there is a pressing need to extract more balanced features that provide thorough coverage without imposing excessive resource demands, thereby ensuring effective and efficient fraud detection in Ethereum-based transactions.

3.2.2. Developed DeFi Transaction Analyzer and Feature Extractor (DeFiTransLyzer-V1.0)

This subsection details the second component of the proposed model: a novel analyzer developed to extract features from DeFi wallets and transactions on the Ethereum platform. A review of the pertinent literature underscores a notable challenge in extracting and analyzing features from diverse data sources such as wallets and transactions (15; 59). It also reveals that semantic features pose significant challenges in term of complexity and resource consumption, while structural features lack comprehensiveness and robust design (66; 61).

To address these issues, our framework introduces a new analyzer that categorizes features into two primary categories: wallet-based and transaction-based. This classification scheme is tailored to capture various quantitative elements from both sources, enhancing the breadth of our feature extraction.

The Wallet Analyzer module is designed to extract and analyze transaction data related to Ethereum wallet activities. It processes JSON files containing transaction records, focusing on extracting meaningful features and providing a detailed summary of wallet behaviors. The Wallet Analyzer computes a comprehensive range of statistical features, including average, variance, skewness, and coefficient of variation, for all transaction activities combined, such as gas price and cumulative gas usage, within a single wallet. These features are systematically aggregated to provide a comprehensive view of wallet activity, including transaction durations, error rates, and patterns in transaction

values. Additionally, the analyzer identifies unique transaction addresses, providing further insights into wallet-specific activity. This detailed analysis is returned as a flattened, feature-rich dictionary that encapsulates wallet behavior in a structured and interpretable format.

Moreover, the Transaction Analyzer module complements the Wallet Analyzer by focusing on extracting and processing detailed transaction-level data. It parses JSON files containing individual transaction records, extracting and calculating key features that reflect the behavior and efficiency of transactions on the Ethereum blockchain. This includes metrics such as gas used, gas price, effective gas price, transaction value, and cumulative gas usage. The module also incorporates log-level data, which identifies and aggregates information about events and token transfers within each transaction. By normalizing token transfer amounts and computing efficiency ratios, such as gas price-to-efficiency ratios, the analyzer provides a nuanced understanding of each transaction's characteristics. In addition to basic transaction details, the module processes complex nested logs to extract additional insights, including log lengths, removed states, and normalized transfer values. It further identifies whether transactions involve contract creation or interactions between identical addresses. The final output is a flattened dictionary containing all relevant features, offering a structured and detailed summary of transaction behaviors.

Together, the Wallet Analyzer and Transaction Analyzer provide a comprehensive framework for analyzing Ethereum blockchain activity, enabling in-depth exploration of wallet- and transaction-level insights. The architecture of the DeFiTransLyzer is illustrated in Figure 3, and Figure 4 presents the taxonomy of features it extracts. Tables 2 and 3 detail the categories, definitions, and measurements of each extracted feature.

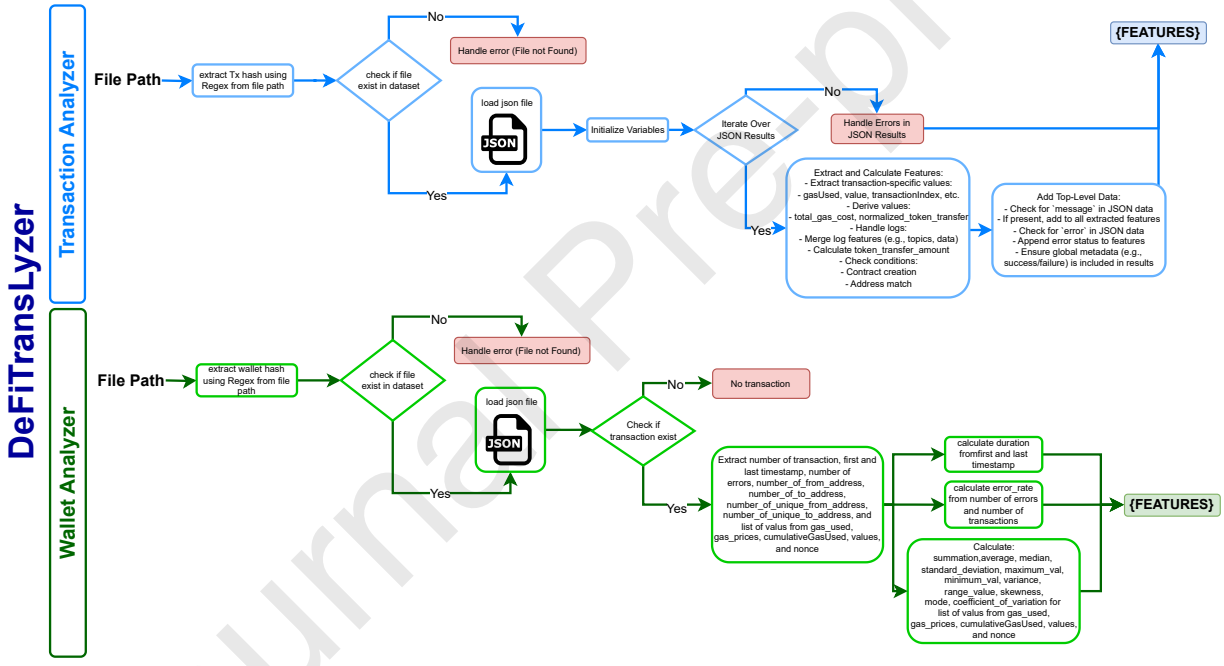

Figure 3: Architecture: DeFiTransLyzer

3.3. Profiling Technique

Profiling is a crucial technique for evaluating the performance of systems, algorithms, or applications to optimize effectiveness and identify areas for improvement (67). This process involves assessing various operational aspects, including execution time, resource usage, and the accuracy and reliability of the model. It also examines the interaction between input features and the predictions derived from them (68; 69). The primary goal of profiling is to enhance the system's effectiveness, reliability, and overall quality (69). This subsection analyzes the profiling background and then explains this study's proposed AGA profiling model.

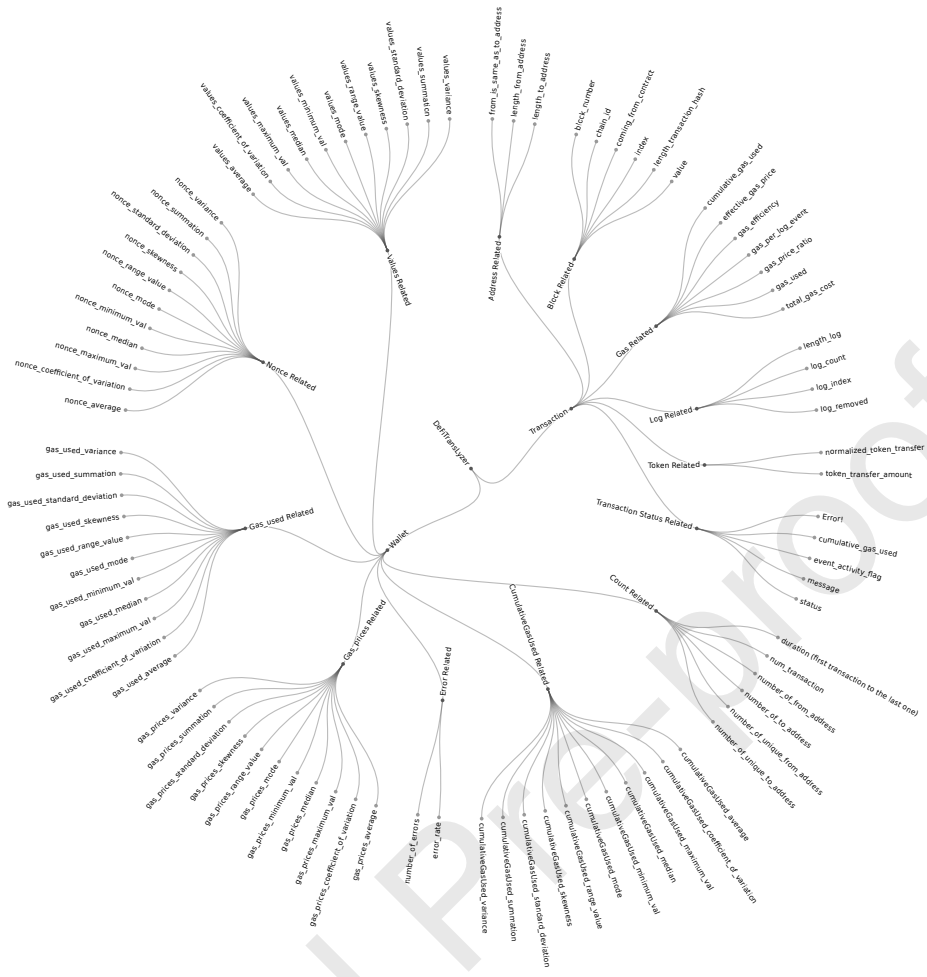

Figure 4: Extracted Features by DeFiTransLyzer

3.3.1. Profiling Algorithms Background

This section offers an in-depth examination of diverse profiling algorithms. Figure 5 presents a thorough classification of current profiling methods, offering a clear perspective on the field. Analyzing existing approaches is essential to identify their strengths, limitations, and suitability for various types of fraud detection in DeFi. By understanding the design choices and outcomes of prior methods, we can highlight gaps in performance, interpretability, or scalability that inform the development of more effective solutions. This comparative analysis also provides a foundation for justifying the need for the proposed profiling framework.

- ***Statistical:*** Statistical profiling assesses the performance of systems, algorithms, or applications by collecting and analyzing data on their operation (67). Key statistical methods used include the Anderson–Darling test, Mahalanobis distance, Hotelling's T-squared test, hypothesis testing, the Girvan–Newman algorithm, the Louvain method, the T-test, the Kolmogorov–Smirnov test, and the Kullback–Leibler divergence.

  - Mahalanobis distance: Hoffelder *et al.* (70) employed the Mahalanobis distance as a multivariate similarity measure to compare group profiles. They proposed T2EQ, a technique grounded in this distance metric, to define an internal equivalence margin for evaluating dissolution profiles in compliance with regulatory standards. Additionally, Kim *et al.* (71) introduced Mahalanobis Distance Cross-Correlation (MDCC), a robust similarity method that transforms pixel values within support windows into Mahalanobis Distance Transform (MDT) space. The similarity between MDT pairs is assessed using cross-correlation, incorporating an asymmetric weight function based on the Mahalanobis distance.

Table 2
DeFiTransLyzer Features Details: Transaction

Cat.	SubCat.	Feature name	Description
Transaction	*Address Related*	length_from_address	The length of the 'from' address in the transaction.
		length_to_address	The length of the 'to' address in the transaction.
		from_is_same_as_to_address	Boolean indicating if the 'from' and 'to' addresses are the same.
	Gas Related	*gas_used*	The amount of gas used for the transaction.
		effective_gas_price	The price per unit of gas paid in the transaction.
		total_gas_cost	The total cost of gas used (gas_used * effective_gas_price).
		gas_per_log_event	Gas used per log event in the transaction.
		cumulative_gas_used	Total gas used in the block up to this transaction.
		gas_price_ratio	Ratio of gas price to some baseline or previous price.
		gas_efficiency	A measure of transaction efficiency based on gas usage.
	Log Related	*log_count*	Number of log entries in the transaction.
		log_removed	Indicates whether any logs were removed.
		log_index	Index of the log within the transaction.
		length_log	Length of the log data.
	Block Related	*value*	The value transferred in the transaction.
		index	Transaction index in the block.
		chain_id	Identifier of the blockchain network.
		block_number	The block number in which the transaction is recorded.
		coming_from_contract	Boolean indicating if the transaction originated from a contract.
		length_transaction_hash	Length of the transaction hash.
	Transaction Status Related	*event_activity_flag*	Flag indicating specific event activity.
		message	Message associated with the transaction.
		status	Status of the transaction (e.g., success, fail).
		Error!	Any errors returned by the transaction execution.
	Token Related	*token_transfer_amount*	Amount of tokens transferred in the transaction.
		normalized_token_transfer	Normalized value of tokens transferred.

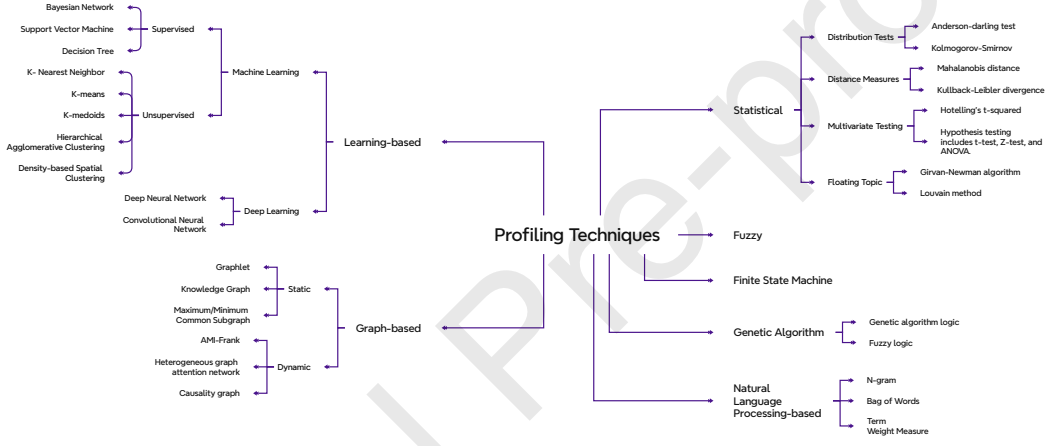

Figure 5: Profiling Taxonomy

- Girvan-Newman: Iorio *et al.* (72) applied the Girvan-Newman algorithm to build a drug similarity network based on genome-wide Gene Expression Profiles (GEPs) from over a thousand compounds, where connections represent shared molecular targets. The algorithm works by calculating shortest paths, determining edge betweenness, and iteratively removing edges with the highest betweenness to uncover a hierarchical structure of drug communities. Similarly, Gunduz *et al.* (73) used the Girvan-Newman algorithm on a real-world Twitter-based social network composed of students from four geographically close universities. In this graph, students were nodes and followership relations formed the edges. The algorithm effectively detected community clusters within the densely connected structure.

- Kolmogorov-Smirnov: Rassokhin *et al.* (74) applied the Kolmogorov-Smirnov test in a novel experimental design for medicinal chemistry. They used this technique to balance traditional design objectives with diverse criteria, optimizing various parameters simultaneously to create a diverse and efficient library of chemical compounds.

- Kullback-Leibler divergence: Zhang *et al.* (75) used the Kullback-Leibler divergence in a study to combine spatial resolution images, classifying pixels into types at MODIS 250m scale and assessing temporal NDVI changes. This approach enabled the identification of similarities between pixel types and pure winter wheat samples, demonstrating efficacy in agricultural regions.

Table 3
DeFiTransLyzer Features Details: Wallet

Cat.	SubCat.	Feture name	Description
Wallet	*Error Related*	number_of_errors	Total number of errors encountered.
		error_rate	Proportion of transactions that resulted in errors.
	GasUsed Related	gas_used_summation	Sum of gas used across transactions.
		gas_used_average	Average gas used per transaction.
		gas_used_median	Median gas used per transaction.
		gas_used_standard_deviation	Standard deviation of gas used.
		gas_used_maximum_val	Maximum gas used in a single transaction.
		gas_used_minimum_val	Minimum gas used across transactions.
		gas_used_variance	Variance of gas used across transactions.
		gas_used_range_value	Range between maximum and minimum gas used.
		gas_used_skewness	Skewness of the gas used distribution.
		gas_used_mode	Mode of the gas used across transactions.
		gas_used_coefficient_of_variation	Coefficient of variation of gas used.
	Gas Prices Related	gas_prices_summation	Sum of gas prices across transactions.
		gas_prices_average	Average gas prices per transaction.
		gas_prices_median	Median gas prices per transaction.
		gas_prices_standard_deviation	Standard deviation of gas prices.
		gas_prices_maximum_val	Maximum gas price in a single transaction.
		gas_prices_minimum_val	Minimum gas prices across transactions.
		gas_prices_variance	Variance of gas prices across transactions.
		gas_prices_range_value	Range between maximum and minimum gas prices.
		gas_prices_skewness	Skewness of the gas prices distribution.
		gas_prices_mode	Mode of the gas prices across transactions.
		gas_prices_coefficient_of_variation	Coefficient of variation of gas prices.
	Cumulative Gas Used Related	cumulativeGasUsed_summation	Sum of cumulative gas used across transactions.
		cumulativeGasUsed_average	Average cumulative gas used per transaction.
		cumulativeGasUsed_median	Median cumulative gas used per transaction.
		cumulativeGasUsed_standard_deviation	Standard deviation of cumulative gas used.
		cumulativeGasUsed_maximum_val	Maximum cumulative gas used in a single transaction.
		cumulativeGasUsed_minimum_val	Minimum cumulative gas used across transactions.
		cumulativeGasUsed_variance	Variance of cumulative gas used across transactions.
		cumulativeGasUsed_range_value	Range between maximum and minimum cumulative gas used.
		cumulativeGasUsed_skewness	Skewness of the cumulative gas used distribution.
		cumulativeGasUsed_mode	Mode of the cumulative gas used across transactions.
		cumulativeGasUsed_coefficient_of_variation	Coefficient of variation of cumulative gas used.
	Values Related	values_summation	Sum of values transferred across transactions.
		values_average	Average value transferred per transaction.
		values_median	Median value transferred per transaction.
		values_standard_deviation	Standard deviation of values transferred.
		values_maximum_val	Maximum value transferred in a single transaction.
		values_minimum_val	Minimum value transferred across transactions.
		values_variance	Variance of values transferred across transactions.
		values_range_value	Range between maximum and minimum values transferred.
		values_skewness	Skewness of the values transferred distribution.
		values_mode	Mode of the values transferred across transactions.
		values_coefficient_of_variation	Coefficient of variation of values transferred.
	Nonce Related	nonce_summation	Sum of nonce values used across transactions.
		nonce_average	Average nonce used per transaction.
		nonce_median	Median nonce used per transaction.
		nonce_standard_deviation	Standard deviation of nonce used.
		nonce_maximum_val	Maximum nonce used in a single transaction.
		nonce_minimum_val	Minimum nonce used across transactions.
		nonce_variance	Variance of nonce used across transactions.
		nonce_range_value	Range between maximum and minimum nonce used.
		nonce_skewness	Skewness of the nonce used distribution.
		nonce_mode	Mode of the nonce used across transactions.
		nonce_coefficient_of_variation	Coefficient of variation of nonce used.
	Count Related	number_of_from_address	Total number of 'from' addresses used.
		number_of_unique_from_address	Number of unique 'from' addresses.
		number_of_to_address	Total number of 'to' addresses involved.
		number_of_unique_to_address	Number of unique 'to' addresses.
		num_transaction	Total number of transactions.
		duration	Time duration from the first to the last transaction.

- ***Fuzzy:*** Fuzzy profiling leverages fuzzy logic and fuzzy sets—representing values with varying degrees of membership—to evaluate system performance across multiple interacting variables (76; 77). This approach enhances efficiency, reliability, and resource allocation optimization (76; 78). Alhabashneh *et al.* (79) introduced a fuzzy IR system combining relevance feedback and fuzzy rule-based summarization, creating unified indexes based on task, user, and document aspects. Their method significantly improved precision and recall in document retrieval. Xu *et al.* (80) proposed a multi-agent system for intelligent student profiling, dynamically generating personalized learning plans based on comprehensive learning activity records. Meanwhile, Dickerson *et al.* (81) developed a fuzzy network profiling technique for intrusion detection, effectively distinguishing normal from abnormal traffic by managing uncertainty in network data through fuzzy logic.

- ***Finite State Machine (FSM):*** FSM profiling, especially when integrated with fuzzy logic, has demonstrated strong potential for analyzing complex human behaviors and managing uncertainty in data-driven environments (82; 83). Smith (82) introduced a deterministic FSM to profile restriction endonuclease recognition sites, offering substantial improvements in efficiency with a time complexity of $O(N)$ and optimized memory usage from $O(M^4)$ to $O(M)$, enabling single-pass sequence analysis without the need for backup storage. Langensiepen *et al.* (35) applied fuzzy FSMs (FFSM) to monitor worker activities in intelligent office settings, aiming to enhance productivity. Fernandez *et al.* (83) developed VISUVER, a profiling and visualization framework based

on DFSMs and text-based similarity measures, which provides insights into individual behavior, motivations, and challenges.

- ***Genetic Algorithm (GA):*** GA-based profiling techniques, leveraging the principles of GA logic and fuzzy logic, serve as dynamic and robust methods for analyzing and enhancing system performance (84; 85). GA utilizes iterative evolutionary processes where a population of options evolves over generations, improving through natural selection and genetic mechanisms until an optimal solution is reached.

  In their investigation into optimizing machining parameters, Asokan *et al.* (86) aimed to achieve continuous finished profiles from cylindrical stock, focusing on selecting settings that would reduce production costs for continuous profile machining. Given the problem's complexity, the authors employed a genetic algorithm (GA) to efficiently solve this machining optimization challenge.

  Similarly, Gupta *et al.* (87) developed a GA-based profiling technique for addressing security vulnerabilities within enterprises. Their method allowed for the selection of a security profile that minimized costs while maximizing vulnerability coverage. This approach utilized GAs to balance cost-effectiveness with comprehensive vulnerability protection effectively.

  Additionally, Resende *et al.* (88) explored anomaly-based intrusion detection, where deviations from "normal" behavior are flagged as potential security threats. They introduced a GA-based adaptive method for profiling and parameterizing anomaly detection systems. Their experiments and evaluation of the CICIDS2017 dataset demonstrated robust performance, achieving a superior detection rate compared to baseline models.

  Finally, Hamamoto *et al.* (89) proposed using GAs to detect network anomalies by creating digital signatures of network segments through flow analysis. This method analyzed network flow data to anticipate traffic patterns over time. Furthermore, they integrated fuzzy logic with GA to assess whether specific scenarios constituted anomalies. The authors advocated for an expert system to monitor IP flows continuously, maintaining regular network behavior while quickly identifying potential issues.

- ***Learning-based:*** Learning-based profiling techniques, such as ML and DL methodologies, play a crucial role in improving the performance, reliability, and accuracy of predictive models. These methodologies focus on identifying areas for improvement to optimize both effectiveness and efficiency (90; 91; 92).

  -Machine learning: In the machine learning domain, KNN-based profiling has been effectively applied to various tasks. Haque *et al.* (93) tackled indoor localization by employing a KNN technique using low-power, cost-effective wireless devices, achieving superior accuracy in determining Cartesian coordinates within buildings. Tsalera *et al.* (94) used KNN to classify environmental noise, analyzing temporal and spectral sound characteristics with the UrbanSound8K dataset. Their results varied depending on the distance metrics applied, demonstrating KNN's flexibility. Nagaraj *et al.* (95) combined KNN with feature-weighted algorithms to enhance graduate admissions by recommending institutions based on successful student profiles. Similarly, Michalakopoulos *et al.* (96) introduced a novel framework for load profiling using K-means, K-medoids, Hierarchical Clustering, and DBSCAN. They also incorporated Explainable AI to improve the interpretability of household energy usage data.

  Beyond KNN, other ML techniques have also been leveraged for profiling tasks. Bayot *et al.* (97) employed SVMs combined with word embeddings for multilingual author profiling, improving adaptability and accuracy across languages. Decision tree models have been applied in both health and marketing sectors. Batterham *et al.* (98) used decision tree profiling to identify suicide risk factors such as anxiety, depression, and substance use, enabling personalized intervention strategies. In a different context, Duchessi *et al.* (99) applied decision trees to profile users of online and mobile platforms at ski resorts, optimizing targeted promotions and increasing sales among different demographic groups.

  -Deep learning: In the realm of DL techniques, several profiling innovations have emerged across diverse domains. Hawley *et al.* (100) proposed a deep auto-encoder conditioned on control settings and time-domain samples to profile audio effects, though further refinement is needed to reduce audible noise. Yu *et al.* (101) introduced SKYLINE, an interactive DNN training tool designed to enhance transparency and model development. Li *et al.* (102) developed APPDNA, a method for automated app profiling using function-call graphs. In the context of transportation, Cura *et al.* (103) conducted a comparative study evaluating LSTM and CNN architectures for driver profiling, providing insights into behavior modeling and system performance.

In addition to DL, Bayesian networks have also demonstrated strong capabilities in profiling tasks. Baumgartner *et al.* (104) applied them in criminal profiling, leveraging probabilistic reasoning to understand behavioral patterns. Xiang *et al.* (105) used Bayesian networks for anomaly detection in video surveillance systems, effectively managing noise and uncertainty. These approaches highlight the versatility and robustness of Bayesian methods in extracting inferences from complex, real-world environments.

- ***Graph-based:*** Graph-based profiling utilizes graph structures to analyze system performance and optimize efficiency by visually representing relationships among system elements (106). Karagiannis *et al.* (106) introduced a method that captures and visualizes user behavior through flow-level data, highlighting the dynamic nature of user interactions. Han *et al.* (107) proposed the AMI-Frank model for personalized search in folksonomies, integrating community clustering and an ant algorithm-based evaporation strategy to enhance personalization. Chen *et al.* (108) developed the Heterogeneous Graph Attention Network (HGAT), which refines user profiles by learning from complex, heterogeneous graph structures.

  Extensions of these approaches have demonstrated wide applicability. Xue *et al.* (109) and Labadie *et al.* (110) applied HGAT to app behavior profiling and tweet classification, respectively. Asai *et al.* (111) introduced causality graphs to accurately identify applications via packet trace analysis, particularly excelling at distinguishing P2P traffic. In the context of personalized content delivery, Daoud *et al.* (112) proposed a semantic user profiling method based on graph structures, using common subgraph techniques to rank documents according to topic relevance for individual users.

- ***Natural Language Processing (NLP)-based:*** NLP techniques have shown transformative potential in generating semantically enriched document profiles, significantly improving document identification and retrieval. Guillén *et al.* (36) and Anrig *et al.* (113) demonstrated how semantic metadata extracted via NLP can be tailored to specific user needs. Anrig *et al.* (113) further emphasized the importance of integrating NLP with algorithmic methods, highlighting the dual role of algorithms in both extracting and validating data, as well as identifying trends and patterns. This combined approach allows for the creation of document profiles that are more accurate and effective than those produced manually. Together, these studies highlight the effectiveness of NLP and algorithms in providing precise, user-centric document profiling solutions.

In summary, a comprehensive examination of profiling methodologies—including graph-based, NLP-based, finite-state machine (FSM), machine learning, and statistical approaches—has demonstrated their value across various domains, such as document profiling and behavioral analysis. Graph-based techniques facilitate semantic optimization for system enhancement, while NLP-based methods contribute to intelligent document classification through the extraction of semantic metadata. Despite these advantages, such methods encounter notable limitations when applied to fraud detection in Ethereum-based transactions. The inherently dynamic, decentralized, and adversarial nature of blockchain systems demands adaptive profiling strategies—something that static or narrowly focused techniques struggle to provide.

In contrast, genetic algorithm (GA)-based profiling offers a more robust and adaptive solution for detecting fraudulent activity in Ethereum transactions. The core strengths of GAs—namely, their iterative, population-based optimization and self-adjusting nature—enable them to effectively respond to the evolving patterns of fraudulent behavior. Unlike conventional models, GAs are not constrained by rigid learning boundaries and can dynamically optimize detection strategies over successive generations. The incorporation of fuzzy logic further strengthens GA-based systems, enhancing their ability to identify subtle and ambiguous anomalies that may elude deterministic or rule-based techniques. This makes them particularly well-suited for securing decentralized platforms where uncertainty and complexity are prevalent.

Although integrating multiple techniques—such as combining graph-based, machine learning, and NLP methods with GAs—might seem advantageous, such hybrid approaches often result in conflicting optimization objectives and excessive computational overhead. These trade-offs can compromise both the efficiency and interpretability of the profiling process. Instead of relying on a multi-method fusion, our approach focuses on refining and extending the capabilities of genetic algorithms themselves. By enhancing GA's internal mechanisms and leveraging its evolutionary nature, we provide a streamlined, resource-efficient solution that remains responsive to emerging patterns of fraud. These improvements are elaborated in detail in Section 3.3.2, where we introduce our advanced GA-based profiling framework tailored for Ethereum fraud detection.

3.3.2. Advanced Genetic Algorithm-based Profiling Technique

Using principles of natural selection and genetics, the GA-based profiling technique is a robust computational optimization method well-suited for tackling complex logical structures and relationships. This approach leverages evolutionary mechanisms to effectively navigate extensive, multidimensional search spaces where traditional methods often fall short (114). Through iterative refinement over generations, the algorithm assesses various potential solutions. It employs genetic operations such as crossover and mutation to reveal hidden patterns. This method is particularly effective in detecting fraud within complex platforms and transactions, such as those encountered on the DeFi Ethereum platform (115).

GAs thoroughly explore the search landscape, achieving a balanced integration of deep exploration and strategic exploitation. This equilibrium allows for the efficient identification of promising solution regions (exploitation), while simultaneously introducing new variations to prevent premature convergence on local maxima (exploration) (116). By utilizing a fitness function, GAs are driven towards increasingly refined solutions, enhancing both the stability and accuracy of the outcomes (117). Moreover, the genetic mechanisms of selection, crossover, and mutation help maintain population diversity, thereby preventing stagnation and facilitating adaptation to new or changing environmental patterns (118).

However, traditional Genetic Algorithms (GAs) face several limitations that hinder their effectiveness in complex fraud detection scenarios. One of the primary challenges is premature convergence, where the algorithm quickly settles on sub-optimal solutions due to early-stage loss of diversity. As genetic diversity diminishes over generations, the algorithm's ability to explore alternative solution paths is significantly reduced, increasing the likelihood of entrapment in local optima and degrading overall performance. Additionally, static mutation rates and insufficient variation among candidate solutions can slow convergence and reduce adaptability.

To overcome these challenges, we introduce an advanced profiling framework—an evolved version of the standard GA—enhanced with targeted refinements described in detail in (37). The proposed model is designed to address core GA weaknesses while eliminating the need for traditional feature engineering. Rather than relying on manually crafted features, the strength of AGA lies in its dynamic search capabilities, fine-grained control over population evolution, and explainable profiling process. This helps isolate the source of performance improvements and demonstrates that the gains achieved in fraud detection stem from the algorithm's design, rather than handcrafted data transformations. Each enhancement is modularly integrated into the AGA profiling method. The base GA structure is denoted as item "1," with improvements enumerated from "2" to "9" as follows:

1. **Fundamental Genetic Algorithm**: The GA is a search and optimization tool inspired by evolutionary biology (86). It evolves a population of candidate solutions through processes analogous to natural selection, including crossover, mutation, and survival of the fittest (88). The fitness function evaluates each individual's viability as a solution. Over time, this evolutionary process ideally steers the population toward optimal or satisfactory solutions. GAs are especially valuable for solving complex problems that defy traditional optimization methods.
2. **Expanded Population Strategy**: The AGA uses an increased population size of 10,000 individuals, enhancing the genetic diversity and broadening the exploratory scope of the algorithm (37). This size increase is crucial in preventing premature convergence and ensuring a thorough search of the solution space, thereby enhancing the likelihood of identifying globally optimal solutions.
3. **Semantic Similarity Checks**: Prior to crossover, semantic similarity checks ensure that genetic recombination occurs between sufficiently diverse yet not overly dissimilar individuals. This strategy maintains genetic diversity within the population while preventing the fusion of incompatible genetic materials (37).
4. **Initial Bias Towards Simplicity**: Starting the population with a bias towards simpler genetic configurations allows the AGA to begin the evolutionary process conservatively, introducing complexity only when it demonstrably improves fitness outcomes (37).
5. **Zero Feature Handling**: A crucial mechanism in the AGA prevents the generation of solutions without any active features, ensuring the continued relevance and utility of the evolutionary outputs (37).
6. **Penalized Fitness Evaluation**: This function penalizes solutions that exhibit undesirable traits, such as an overly simplistic feature set, guiding the evolution towards more effective and robust configurations.
7. **Elite Retention Strategy**: Preserving a subset of the best-performing individuals ensures that high-quality genetic materials are carried over across generations, fostering faster convergence and enhancing solution quality.
8. **Dynamic Mutation Adjustment**: The AGA adapts the mutation rate in response to the evolution phase, starting with higher rates to encourage exploratory diversity and decreasing the rate to refine promising solutions, maintaining an effective balance throughout the evolutionary process.

9. **Dynamic Generation**: The AGA employs a dynamic generation selection mechanism that intelligently determines the optimal number of generations needed for convergence. This feature prevents overtraining by halting the evolutionary process once no significant improvements are detected in the fitness scores over a set number of generations.

To evaluate the effectiveness of the adaptive mutation rate used in our AGA-based profiling approach, we compared it with a fixed-rate baseline (0.01) in Figure 6. The adaptive mutation rate, which increases gradually every 10 generations, enables the algorithm to strike a balance between early convergence and continued exploration. As shown in the figure, the dynamic approach achieves faster and more stable convergence to higher fitness values. In contrast, the fixed-rate approach converges more slowly and tends to plateau earlier. These results empirically validate the adaptability and efficiency of our dynamic mutation strategy.

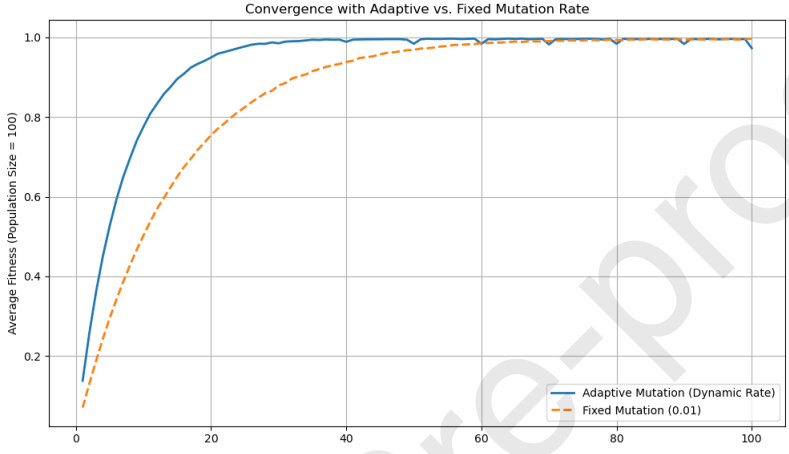

Figure 6: Convergence with Adaptive vs. Fixed Mutation Rate

The AGA profiling technique represents a significant advancement over traditional genetic algorithms by incorporating targeted enhancements that improve both adaptability and performance. Built upon core GA principles, AGA extends the algorithm's search capabilities, increasing the efficiency and quality of the generated profiling outcomes. As outlined in Algorithm 1, each enhancement is systematically integrated to support comprehensive behavioral profiling. Importantly, the nature of AGA's application—profiling rather than classification—minimizes the need for explainability. The goal is not to produce interpretable decisions at the individual transaction level, but rather to derive representative patterns and characteristics across groups of transactions. Furthermore, explainability is inherently supported through the system's other component, DeFiTransLyzer-V1.0, which is responsible for extracting meaningful and interpretable features from transaction data. Because these features already carry domain-specific semantic value, the profiling output of AGA can be understood and trusted without requiring additional explanation from the algorithm itself. Therefore, the combination of a meaningful feature extractor and a non-decision-making profiling process makes traditional explainability mechanisms unnecessary in this context.

3.4. Evaluation Criteria

The evaluation of the profiling model is described in the concluding subsection of the proposed architecture. For evaluation purposes, a straightforward classifier is employed during the initial training phase to determine the effectiveness of the profiling model in detecting fraud transactions on the Ethereum platform. This simple classifier helps to highlight the impact of the profiling process by ensuring that the outcomes predominantly reflect the quality of the pre-processed and structured data rather than the complexity of the classification algorithm.

Furthermore, this subsection delineates the metrics used for evaluation and elucidates their relevance. Table 4 provides a detailed explanation of each evaluation metric, facilitating a thorough understanding of the criteria employed to assess the model's efficacy. An extensive analysis of the metric values and the insights derived from this evaluation is elaborated in Section 4.

Algorithm 1 AGA for Profiling with Dynamic Generation

Require: population size P, number of generations G, initial mutation rate μ, elite count E, semantic similarity parameters α, β, feature space X, labels y.
Ensure: Best individuals per label after G generations.

Initialize population with size P randomly
Initialize best individuals and fitness per label in y
Initialize mutation rate μ
Initialize elite count E
function PENALTY_FITNESS_FUNCTION(individual)
    Compute base fitness based on X and y
    Apply penalty for non-compliance with constraints
    return base fitness - penalty
end function
function RETAIN_ELITES(population, fitness_scores)
    Sort population by fitness scores in descending order
    Select top E elites
    return elites
end function
function ADJUST_MUTATION_RATE(generation, initial_rate μ)
    Decay mutation rate by 5% per generation or adjust based on performance
    return new mutation rate
end function
function SSD(st1, st2)
    Compute semantic similarity distance based on features
    return distance
end function
function SEMANTIC_SIMILARITY(st1, st2)
    Check if $\alpha <$ ssd($st1$, $st2$) $< \beta$
    return True if within bounds, False otherwise
end function
function EVALUATE_CONVERGENCE(current_generation, threshold)
    Evaluate if average fitness improvement over the last $threshold$ generations is below 1%
    return True if the improvement is minimal, False otherwise
end function
$generation\_counter \leftarrow 0$
$convergence\_threshold \leftarrow 5$
while $generation\_counter < G$ and not EVALUATE_CONVERGENCE($generation\_counter$, $convergence\_threshold$) **do**
    $\mu \leftarrow$ ADJUST_MUTATION_RATE($generation\_counter$, μ)
    Calculate fitness for each individual using PENALTY_FITNESS_FUNCTION
    RETAIN_ELITES(population, fitness_scores)
    for each individual in population to maintain size P **do**
        Perform crossover and mutation with rate μ
    end for
    Update population
    Update best individuals per label
    Print current best fitness
    $generation\_counter \leftarrow generation\_counter + 1$
end while
return best individuals per label

Table 4
Details of Evaluation Metrics

Metric	Definition	Formula
Precision	Measures the accuracy of positive predictions.	$Precision = \frac{TP}{TP+FP}$
Recall	Measures the fraction of positives correctly identified.	$Recall = \frac{TP}{TP+FN}$
F1-score	Harmonic mean of precision and recall.	$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$
Accuracy	Fraction of all correct predictions.	$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
Macro Average	Average metric computed independently for each class.	$MacroAvg = \frac{1}{N}\sum_{i=1}^{N} Metric_i$
Weighted Average	Average metric weighted by class size.	$WeightedAvg = \sum_{i=1}^{N} w_i \cdot Metric_i$
Execution Time	Total duration required for a process or algorithm to complete its task.	$T_{exec} = T_{end} - T_{start}$
Best Fitness	The highest value of a fitness function achieved by an individual in a population.	$F_{best} = \max(F(x_i))$

4. Experiments and Results

This section explores the experimental setup and methodologies employed to validate the proposed profiling approach effectively. A critical component of this validation is developing a comprehensive, purpose-built dataset designed to serve as a robust testing ground for evaluating the model's performance. The validation process encompasses

multiple stages, beginning with configuration and hyperparameter tuning and culminating in experimental results that analyze fraud and legitimate transactions. The following subsections detail these steps.

4.1. Dataset

4.1.1. Available Datasets

In this subsection, we review the datasets employed in prior research and evaluate their strengths and limitations. Table 5 presents a comparative analysis of the most prominent datasets, evaluating them based on several criteria, including sample size, public availability, DeFi-level granularity (e.g., smart contracts, wallets, transactions), feature diversity, and their respective advantages and drawbacks.

Several prior works, such as (32), have introduced publicly available datasets, which are valuable for promoting transparency, reproducibility, and community-driven advancements. However, nearly half of the datasets reviewed—including those proposed in (61; 13; 19)—are private and thus inaccessible for independent validation or broader use. This lack of availability limits their utility for comparative analysis and practical application. Additionally, many existing datasets focus narrowly on a single level of DeFi data, such as only smart contracts, wallets, or transactions. Notably, only two studies, (15; 31), offer datasets that cover both wallet- and transaction-level data. However, these still suffer from limitations, either due to reliance on third-party sources or restricted accessibility, which undermines their reliability and transparency.

The richness and diversity of features across existing datasets also vary significantly, which impacts their utility in modeling complex fraud behaviors. Several datasets, including (60; 66; 33), provide only a narrow range of features, thereby limiting the depth of analysis and representation. Sample size is another critical limitation: while some datasets, such as (34; 32), contain relatively small numbers of samples—constraining their diversity—others, like (33), offer larger datasets but suffer from substantial class imbalances, with significantly fewer fraudulent samples than legitimate ones. Moreover, datasets that heavily rely on third-party platforms for labeling or data aggregation, such as (59**?** ; 15), raise concerns about data consistency and labeling integrity, even when publicly available.

Overall, the analysis reveals widespread limitations among current datasets in terms of accessibility, balance, data diversity, and feature completeness. While publicly available datasets enhance transparency and reproducibility, their limited scope and shallow feature sets often restrict their practical applicability. In contrast, private datasets, despite their potential depth, hinder independent validation and broader adoption. Furthermore, datasets with small sample sizes or severe class imbalances fall short in capturing the complexity of fraud patterns in decentralized finance. These observations underscore the pressing need for a comprehensive, multi-level, and balanced dataset—one that is publicly available, feature-rich, and capable of supporting advanced research and robust fraud detection solutions in the DeFi ecosystem.

4.1.2. New Dataset (BCCC-DeFiFruadTrans-2025)

This research introduces a new dataset designed to overcome the challenges and limitations of existing ones by offering a larger sample size and integrating a broader range of attributes and information sources, including transactions and wallets. The dataset was constructed by collecting information from Etherscan (39), a widely used blockchain explorer that offers comprehensive access to Ethereum blockchain data.

The dataset includes two primary categories of information: wallet-related data and transaction-related data. These categories were carefully curated to encompass a diverse range of both fraudulent and legitimate activities. Each transaction is labeled based on Etherscan's classification system: transactions marked as legitimate indicate no known issues, whereas those labeled as fraud are associated with contracts or activities flagged for fraudulent behavior by Etherscan. To ensure the reliability of these labels, we incorporated additional validation steps such as anomaly detection, consistency checks, and duplicate removal.

The data collection process spanned from 2017 to 2024, offering a comprehensive snapshot for analysis. Organized into JSON files, each file corresponds to a specific wallet or transaction, comprising 9,374 unique wallet addresses and 1,026,867 transactions labeled as either fraud or legitimate. This structured and validated approach enhances the credibility of the dataset and supports its application in fraud detection research within the decentralized finance sector. Additional details about the dataset's structure and characteristics are provided in Table 6. Moreover, examples of a transaction file and a wallet JSON file are provided in Figures 4.1.2 and 4.1.2.

To assess the reliability and quality of the BCCC-DeFiFraudTrans-2025 dataset, we performed two key validation steps: statistical profiling and baseline model evaluation.

Table 5
Previous Dataset Evaluation

Ref.	Name	#Samples	Public	Category	Features	Advantage	Disadvantage
(32)	I experienced more than 10 defi scams: On defi users' perception of security breaches and countermeasures	14	Yes	N/A	N/A	Qualitative interview	Analysis of user opinions only
(33)	BERT4ETH: A Pre-trained Transformer for Ethereum fraud Detection	7,057	Yes	Wallets	7	Sufficient samples	Insufficient features
(59?)	A ML based Approach to Detect the Ether o Detect the Ethereum fraud Transactions with Limited Attributes	9,841	Yes	Transactions	17	Dataset includes contracts, tokens, traces and more	Insufficient samples, Insufficient features
(15)	Ethereum fraud detection with heterogeneous graph neural network	9,629 wallets, 386,612 transactions	Yes	Wallets, Transactions	7	Sufficient samples	Imbalanced Dataset
(13)	DeFiScanner: Spotting DeFi Attacks Exploiting Protocols	50,910	No	Transactions	37	Ground-truth labels manually inspected	Private Dataset
(60)	DeFiGuard: A Price Manipulation Detection Service	2,288	No	Transactions	4	Manually picked samples	Imbalanced Dataset, Private Dataset
(19)	DeFiRanger: Detecting Price Manipulation Attacks on DeFi Applications	183,363	No	Transactions	4	Sufficent samples	Private Dataset, Insufficient features, Private Dataset
(61)	Graph Neural Network for Ethereum fraud Detection	3,041	No	Transactions	34	Balanced dataset	Private Dataset

Table 6
Overview of BCCC-DeFiFraudTrans-2025 Dataset

Data Type	# Fraud Samples	# Legitimate Samples	# Features Extracted by DeFiTransLyzer-V1.0
Ethereum-based Wallet	1,763	7,611	63
Ethereum-based Transaction	88,793	938,074	26

An exploratory analysis was conducted to evaluate the structure and consistency of the dataset. We examined the class distribution, which showed a reasonably balanced ratio of fraudulent to non-fraudulent transactions. Feature values such as gas_used, value, and token_transfer_amount were reviewed for consistency and completeness. The dataset was also checked for missing values and outliers, confirming that it is clean and free from significant anomalies.

To further validate the dataset's label quality and feature effectiveness, we trained a Random Forest classifier without hyperparameter tuning. The model achieved an accuracy of 86.2%, precision of 81.4%, recall of 83.7%, and an F1-score of 82.5% in identifying fraudulent transactions. These results suggest that the dataset contains informative patterns that align with the assigned labels. Moreover, the model's feature importance scores supported the relevance of selected attributes, providing additional confidence in the dataset's design.


```
{
    "jsonrpc": [
        "2.0",
        "2.0"
    ],
    "id": [
        1,
        1
    ],
    "result": [
        {
            "blockHash": "0xaa1ac67813803ca963172956045f6911234b99b9227a9aeffc6c9357931527d9",
            "blockNumber": "0x489c8f",
            "contractAddress": null,
            "cumulativeGasUsed": "0x24dbdb",
            "effectiveGasPrice": "0x59682f000",
            "from": "0x00009277775ac7d0d59eaad8fee3d10ac6c805e8",
            "gasUsed": "0x16bba",
            "logs": [
                {
                    "address": "0x8d12a197cb00d4747a1fe03395095ce2a5cc6819",
                    "topics": [
                        "0x6effdda786735d5033bfad5f53e5131abcced9e52be6c507b62d639685fbed6d"
                    ],
                    "data": "0x0000000000000000000000000000000000000000...",
```

```
                    "blockNumber": "0x489c8f",
                    "transactionHash": "0x012480c1f78b9c4a4adf5c7efb0ba0755eb02f78a12ed4c598c61873a2312899",
                    "transactionIndex": "0x3c",
                    "blockHash": "0xaa1ac67813803ca963172956045f6911234b99b9227a9aeffc6c9357931527d9",
                    "logIndex": "0x1d",
                    "removed": false
                }
            ],
            "logsBloom": "0x00000000000000000000000...",
            "status": "0x1",
            "to": "0x8d12a197cb00d4747a1fe03395095ce2a5cc6819",
            "transactionHash": "0x012480c1f78b9c4a4adf5c7efb0ba0755eb02f78a12ed4c598c61873a2312899",
            "transactionIndex": "0x3c",
            "type": "0x0"
        },
        {
            "blockHash": "0xaa1ac67813803ca963172956045f6911234b99b9227a9aeffc6c9357931527d9",
            "blockNumber": "0x489c8f",
            "from": "0x00009277775ac7d0d59eaad8fee3d10ac6c805e8",
            "gas": "0x3d090",
            "gasPrice": "0x59682f000",
            "hash": "0x012480c1f78b9c4a4adf5c7efb0ba0755eb02f78a12ed4c598c61873a2312899",
            "input": "0x0a19b14a000000000000000...",
            "nonce": "0x285",
            "to": "0x8d12a197cb00d4747a1fe03395095ce2a5cc6819",
            "transactionIndex": "0x3c",
            "value": "0x0",
            "type": "0x0",
            "chainId": "0x1",
            "v": "0x25",
            "r": "0x5d3616b9bed293604a61d92f6d8ce05b32af33334027ea9b1724bbb867cbd9bc",
            "s": "0x4e34d98bc14627af72e000286ad8265489972547bdd30ce5f60409c34174b277"
        },
        "Error!"
    ],
    "status": [
        "0"
    ],
    "message": [
        "NOTOK"
    ]
}
```


Listing 1: Ethereum Transaction example


```
{
    "status[0]": "1",
    "status[1]": "1",
    "status[2]": "0",
    "status[3]": "0",
    "status[4]": "0",
    "message[0]": "OK",
    "message[1]": "OK",
    "message[2]": "No transactions found",
    "message[3]": "No transactions found",
    "message[4]": "No transactions found",
    "result[0]": "0",
    "result[1]": [
        {
            "blockNumber": "6026742",
            "timeStamp": "1532511199",
            "hash": "0x94917b89296051b066db2ac572987d8ec48a88716f51291a47d50e6b1e8cc20c",
            "nonce": "2560067",
            "blockHash": "0xad77b360c7a8401ea81e875a8fbc9ca0acc0dbfed7a1954dc9cc8348b4fb063d",
            "transactionIndex": "8",
            "from": "0x3f5ce5fbfe3e9af3971dd833d26ba9b5c936f0be",
            "to": "0x0a0ba956038d4a66002d612648332b9c4ab7646c",
            "value": "5000000000000000000",
            "gas": "21000",
            "gasPrice": "60000000000",
            "isError": "0",
            "txreceipt_status": "1",
            "input": "0x",
            "contractAddress": "",
            "cumulativeGasUsed": "227318",
            "gasUsed": "21000",
            "confirmations": "14994974",
            "methodId": "0x",
            "functionName": ""
        },
        {
            "blockNumber": "6030488",
            "timeStamp": "1532565565",
            "hash": "0x5c992599647f4a95919d68c4fe137612635bb9c9530b6a4572049c6197678a81",
            "nonce": "0",
            "blockHash": "0xc8cf2bab3addc5343f38020e6bb9614f434285f858a2ccd31cfa78e18876df58",
            "transactionIndex": "86",
            "from": "0x26b315a3dd31f4002df033b5e493c05cdbd9d36c",
            "to": "0x0a0ba956038d4a66002d612648332b9c4ab7646c",
            "value": "950209050000000000",
            "gas": "21000",
            "gasPrice": "2000000000",
            "isError": "0",
            "txreceipt_status": "1",
            "input": "0x",
            "contractAddress": "",
            "cumulativeGasUsed": "7055563",
            "gasUsed": "21000",
            "confirmations": "14991228",
            "methodId": "0x",
            "functionName": ""
        },
```

```
        {
            "blockNumber": "6032557",
            "timeStamp": "1532595953",
            "hash": "0x599b8b213ca56139e7389025c9292f7933a387acf8424d25ab6d9285d7cf0706",
            "nonce": "895499",
            "blockHash": "0x237fb423d459a948f753b2a3ad7a7a9bb175d3e37341e4a9ef773cfa632f313e",
            "transactionIndex": "33",
            "from": "0x0681d8db095565fe8a346fa0277bffde9c0edbbf",
            "to": "0x0a0ba956038d4a66002d612648332b9c4ab7646c",
            "value": "1990453490000000000",
            "gas": "21000",
            "gasPrice": "60000000000",
            "isError": "0",
            "txreceipt_status": "1",
            "input": "0x",
            "contractAddress": "",
            "cumulativeGasUsed": "906401",
            "gasUsed": "21000",
            "confirmations": "14989159",
            "methodId": "0x",
            "functionName": ""
        },
        {
            "blockNumber": "6032948",
            "timeStamp": "1532601553",
            "hash": "0x9c3ac7f4d6f27567e2cee4be1e8542129f8b8f6d939a0241a1274e11d87b408f",
            "nonce": "0",
            "blockHash": "0x993ea5e475bd6c0c27979853926457bac20f7050af546746b7b877c20660f3af",
            "transactionIndex": "0",
            "from": "0x0a0ba956038d4a66002d612648332b9c4ab7646c",
            "to": "0xb553b65a8290581fb43af305a97803405375ceff",
            "value": "3438583540000000000",
            "gas": "21000",
            "gasPrice": "99000000000",
            "isError": "0",
            "txreceipt_status": "1",
            "input": "0x",
            "contractAddress": "",
            "cumulativeGasUsed": "21000",
            "gasUsed": "21000",
            "confirmations": "14988768",
            "methodId": "0x",
            "functionName": ""
        },
        {
            "blockNumber": "6035043",
            "timeStamp": "1532633215",
            "hash": "0x7a7ed80927e7fa27793619a96667e12746f4523ab3719853a184977f2f9c3e65",
            "nonce": "7",
            "blockHash": "0xec90aa6a26d62dc358c33629f6574ed6da52803866cea42447c09976417e46f4",
            "transactionIndex": "92",
            "from": "0x69ad70a4224e8ec4e4b334437e28c8418a45df9f",
            "to": "0x0a0ba956038d4a66002d612648332b9c4ab7646c",
            "value": "1000000000000000000",
            "gas": "35000",
            "gasPrice": "41000000000",
            "isError": "0",
            "txreceipt_status": "1",
            "input": "0x",
            "contractAddress": "",
            "cumulativeGasUsed": "2074797",
            "gasUsed": "21000",
            "confirmations": "14986673",
            "methodId": "0x",
            "functionName": ""
        },
        {
            "blockNumber": "6064709",
            "timeStamp": "1533063242",
            "hash": "0xa3fa5808d3fa8c73216bea3be9fe7fad9dde9bc928bfd85c593db65b550d4475",
            "nonce": "1",
            "blockHash": "0xc456aad31265cd3a710a29bfcff19e7431f394a242d347ab4cc7cc1559468f68",
            "transactionIndex": "0",
            "from": "0x0a0ba956038d4a66002d612648332b9c4ab7646c",
            "to": "0xb553b65a8290581fb43af305a97803405375ceff",
            "value": "998530000000000000",
            "gas": "21000",
            "gasPrice": "70000000000",
            "isError": "0",
            "txreceipt_status": "1",
            "input": "0x",
            "contractAddress": "",
            "cumulativeGasUsed": "21000",
            "gasUsed": "21000",
            "confirmations": "14957007",
            "methodId": "0x",
            "functionName": ""
        }
    ],
    "result[2]": [],
    "result[3]": [],
    "result[4]": []
}
```


Listing 2: Ethereum Wallet example

4.2. Experimental Configuration

This study employs a robust computational framework and specialized tools to address the complexities of fraud detection in Ethereum-based transactions. We established a dedicated environment to execute the proposed model

Table 7
Hyper-parameter Tuning Results: This table illustrates the influence of alpha and beta on the number of features selected during AGA profiling. An asterisk (*) denotes the final values implemented in the AGA.

Alpha	Beta	# Final Generation	Fitness Score
0.5	0.7	8th	7.63
0.5	1.0	9th	7.71
0.5*	1.5*	10th	7.78
0.5	2.0	10th	7.78
0.1	1.0	99th	2.14
0.1	1.5	99th	2.14
0.1	2.0	99th	2.14
0.3	1.0	99th	3.64
0.3	1.5	99th	3.64
0.3	2.0	99th	3.64

and optimize data processing and analytical efficiency. The foundation of our analytical methodology is the Python programming language, specifically version 3.11.5. The computational experiments were conducted on a system equipped with a 13th-generation Intel Core i7-13700 processor, clocked at 2.10 GHz, and supported by 32.0 GB of RAM. The experiments utilized a 64-bit Windows 10 operating system running on an x64-based processor, ensuring a powerful and effective platform for our analyses.

4.3. Hyper-parameter Tuning

This subsection examines the optimization of hyperparameters crucial to the performance of the proposed AGA profiling, with a specific focus on alpha (the lower SSD boundary) and beta (the upper SSD boundary). These parameters are crucial for striking a balance between exploring new solutions and exploiting known ones. They refine the algorithm's approach to semantic similarity, with alpha promoting the discovery of novel solutions and beta balancing new explorations with refinements of existing solutions.

Analyzing the hyperparameter tuning results from Table 7, it is clear that the choice of the alpha parameter significantly impacts the efficiency and effectiveness of the genetic algorithm. With alpha set to 0.5, the algorithm consistently achieves higher fitness scores, ranging from 7.63 to 7.78, across various beta settings (0.7 to 2.0). Additionally, it maintains a relatively low number of generations (8th to 10th), indicating that a higher alpha value fosters an optimal balance between exploration and exploitation, thus allowing the algorithm to converge rapidly to optimal or near-optimal solutions. Conversely, lower alpha values (0.1 and 0.3) result in considerably lower fitness scores (2.14 to 3.64), despite the algorithm running through a significantly higher number of generations (99th). This pattern suggests inefficient convergence, as the algorithm expends extended cycles without significantly improving fitness scores.

Furthermore, the interaction between the beta parameter and alpha reveals insightful trends for optimizing the performance of AGA profiling. Beta significantly influences the refinement process by managing the exploration-exploitation trade-off within the algorithm. The data indicate that with an alpha level of 0.5, all beta values from 0.7 to 2.0 yield high fitness scores. However, selecting a beta of 1.5 offers a distinct advantage by providing an optimal balance. This mid-range setting enables the algorithm to explore new solutions adequately without excessive directional shifts, which can impede performance when too high and prevent the stagnation of exploration when too low.

In conclusion, an alpha of 0.5 paired with a beta of 1.5 establishes a robust framework for the algorithm to function efficiently across diverse scenarios. This configuration ensures that the algorithm avoids premature convergence on local optima—a common pitfall in genetic algorithms—while maintaining the agility to refine solutions effectively. The consistent achievement of high fitness scores across different beta settings, with alpha set at 0.5, underscores the critical role of alpha. In contrast, beta at 1.5 optimizes the interplay between the depth and breadth of the search.

4.4. Experiments Results

In this study, we perform a classification to assess the efficacy of the proposed profiling model in identifying Ethereum-based transactions. Due to the historical nature of wallet information, it is less effective in classifying zero-day fraud transactions, which are the first transactions of a new wallet. As a result, while extracting wallet information and transaction analyzer features, we focus solely on transaction analyzer features for identification and profiling in this study.

Table 8
Data Distribution in the Classification Model Training Process

Set	# Fraud	# legitimate	# Total
Training	66,629	66,561	133,190
Testing	22,164	22,233	44,397

Table 9
Classification Report: Reported metrics are averaged over 10 independent runs with 95% confidence intervals.

Classification Report				
	precision	recall	f1-score	support
Legitimate	0.96 ± 0.01	0.95 ± 0.02	0.95 ± 0.01	22233
Fraud	0.95 ± 0.01	0.96 ± 0.01	0.96 ± 0.01	22164
macro avg	0.96 ± 0.01	0.96 ± 0.01	0.96 ± 0.01	44397
weighted avg	0.96 ± 0.01	0.96 ± 0.01	0.96 ± 0.01	44397
accuracy		**0.96 ± 0.01**		
Execution time		**37.49 Seconds**		

Table 8 presents a detailed breakdown of the data proportions used in the classification process. Additionally, Table 9 illustrates the model's ability to distinguish between fraudulent and legitimate transactions, utilizing the evaluation metrics outlined in Table 4. Further insights into the model's complexity, performance analysis, and comparisons with existing models are discussed in Section 5.

5. Analysis

This section comprehensively analyzes the proposed model by addressing four critical questions. First, it evaluates the model's performance by examining the results presented in Section 4.4 and Table 9. Next, it examines the model's time and space complexity, and analyzes its stability and generalizability through a K-fold cross-validation process. Finally, it compares the proposed model with available methods regarding their effectiveness in detecting fraud Ethereum transactions.

(1) RQ1: How efficient is the proposed model in terms of the defined evaluation parameters outlined in Table 4?

The classification report provides a detailed evaluation of the proposed model's performance in distinguishing between fraudulent and legitimate Ethereum transactions. To accurately interpret this performance, three key metrics are used: precision, recall, and F1-score. Precision measures the proportion of true positive predictions among all transactions labeled as positive by the model, indicating how many of the identified transactions, whether fraudulent or legitimate, were correct. Recall, on the other hand, indicates the model's ability to identify all actual instances of a given class—it measures how many fraudulent or legitimate transactions were successfully detected out of all that truly exist. The F1-score is the harmonic mean of precision and recall, offering a balanced measure that accounts for both false positives and false negatives.

As shown in the classification report, the model achieves a precision of 0.90 for legitimate transactions and 0.87 for fraudulent ones, indicating a high level of accuracy in its positive predictions for both categories. These precision scores suggest that when the model identifies a transaction as legitimate or fraudulent, it is very likely to be correct. The recall values further demonstrate the model's ability to capture actual cases: 0.87 for legitimate and 0.90 for fraud transactions. These results imply that the model is highly effective at identifying the majority of true cases, with slightly better sensitivity in detecting fraud—a crucial capability for a fraud detection system aimed at securing blockchain-based transactions.

The F1-scores for legitimate and fraudulent transactions are closely matched at 0.88 and 0.89, respectively. This balance underscores the model's consistent and reliable performance in handling the inherent complexity of distinguishing between transaction types. The overall classification accuracy stands at 0.88, based on a dataset comprising 44,397 Ethereum transactions, which reinforces the model's robustness in maintaining a high rate of correct predictions across various scenarios.

To gain further insight into the model's generalization across classes, both macro and weighted averages are reported. The macro average, which treats all classes equally regardless of their frequency, and the weighted average, which adjusts for class imbalance, both report values of 0.88 for precision, recall, and F1-score. These consistent figures highlight the model's fairness and stability in evaluating both fraud and legitimate transactions, ensuring that neither class is overlooked or disproportionately favored. Together, these metrics affirm the model's suitability for real-world deployment in Ethereum fraud detection tasks, offering both accuracy and interpretability across a diverse range of transaction behaviors.

RQ2: How efficient is each component of the proposed profiling model regarding time and space complexity?

The proposed AGA profiling model showcases robust efficiency and reliability, as illustrated by the execution time metrics detailed in Table 9. The model demonstrates a streamlined and efficient execution, completing tasks in approximately 37.49 seconds. This performance underscores the model's rapid processing capabilities with minimal delays.

Further analysis delves into the specific space and time complexities associated with each component of the AGA, along with the phases of feature selection and model training. The advanced elements of the algorithm, designed to enhance performance and efficiency, are outlined below:

- ***Penalty Fitness Function***: The space complexity remains constant at $O(1)$, as it primarily depends on the size of a fixed-length individual array and does not scale with input size. The time complexity is linear, $O(n)$, where n is the length of the individual, due to power calculations and summation operations performed on each element.
- ***Retention Strategy (`retain_elites`)***: This component maintains a fixed-size list of elites, leading to a constant space complexity of $O(1)$. The time complexity is dominated by sorting the fitness scores, which takes $O(m \log m)$, where m is the population size. Selecting the top-performing individuals for retention adds a linear component, resulting in an overall time complexity of $O(m \log m)$ per generation.
- ***Adaptive Learning Rate***: Both time and space complexities are constant at $O(1)$, as the mechanism performs simple mathematical adjustments that neither depend on population size nor require additional memory.
- ***SSD Checks***: This component performs direct comparisons across arrays like `st1` and `st2`, resulting in a linear time complexity of $O(k)$, where k is the length of the input arrays. Since no additional data structures are created, space complexity remains constant at $O(1)$.
- ***Adaptive Generation Control***: This mechanism leverages existing metrics computed during the genetic algorithm (GA) process, requiring no extra memory and thus maintaining a space complexity of $O(1)$. The evaluation of convergence criteria across generations is carried out through lightweight comparison operations, contributing a constant time complexity of $O(1)$ per check. As these checks occur periodically, their overall impact on the total time complexity is negligible.
- ***Main AGA Loop***: The space complexity of the main AGA loop scales with both the population size and the length of each individual, resulting in $O(mn)$, where m is the population size and n is the individual length. The time complexity consists of several components:
  - *Fitness Evaluation:* Each individual's fitness is computed in $O(n)$, leading to $O(mn)$ for the full population.
  - *Elite Retention:* Sorting for elite selection contributes $O(m \log m)$.
  - *Crossover and Mutation:* These genetic operations run in $O(mn)$.

  Together, the per-generation complexity is $O(mn + m \log m)$. Over g generations, the overall time complexity becomes $O(g \cdot (mn + m \log m))$.

The model demonstrates its capability for high-performance operations within constrained time frames, as evidenced by the impressive execution time detailed in Table 9. The thorough examination of the space and time complexities presented in Table 10 further underscores the strategic design of the model's components, enabling efficient optimization of computational resources. This optimization is exemplified by innovative features, including the Penalty Fitness Function, Adaptive Learning Rate, and Adaptive Generation Control. These elements significantly contribute to the model's efficacy and reliability, making it valuable in scenarios that demand precision and speed.

(3) RQ3: How does the proposed model perform in terms of stability and generalizability?

The robustness and generalizability of the proposed AGA profiling model are rigorously verified through extensive cross-validation, which involves dividing the dataset into 'k' folds and iteratively training and testing the model 'k'

Table 10
Space and Time Complexity of Each Component

Component	Space Complexity	Time Complexity
Penalty Fitness Function	O(1)	O(n)
Retention Strategy (retain_elites)	O(1)	O(m log m)
Adaptive Learning Rate	O(1)	O(1)
SSD Checks	O(1)	O(k)
Adaptive Generation Control	O(1)	O(1)
Main GA Loop	O(mn)	O(g * (mn + m log m))

Table 11
K-Fold Comparison: AGA Stability and Generalizability (Results include 95% confidence intervals over K-fold runs.)

K-fold	Precision	Recall	F1 Score
5	0.93 ± 0.01	0.94 ± 0.02	0.93 ± 0.01
10	0.93 ± 0.01	0.95 ± 0.01	0.94 ± 0.01
15	0.94 ± 0.01	0.95 ± 0.01	0.94 ± 0.01
20	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01

times. In each iteration, a distinct fold serves as the test set, while the remainder serves as the training set. In Table 11, we present the model's performance metrics obtained through k-fold cross-validation for k = 5, 10, 15, and 20.

The results highlight the model's exceptional generalizability, as demonstrated by its consistent performance across different metrics. The precision, recall, and F1-scores remain stable for k=15 and 20, suggesting that the model effectively generalizes across fraud and legitimate labels. Furthermore, the stability of the model is accentuated by the incremental changes in these metrics as the number of folds increases from 5 to 20. Despite the varying data splits in the validation process, the metrics show slight yet consistent improvements or maintain stability. This demonstrates the model's capability to handle diverse validation scenarios reliably. The robustness and sensitivity to data splits make this model a dependable tool in predictive scenarios that demand high accuracy and consistency.

(4) RQ4: How does the effectiveness of the proposed model in detecting fraudulent Ethereum transactions compare to existing traditional, machine learning, and neural network-based methods?

To rigorously assess the effectiveness of the proposed AGA model in detecting fraudulent activities within Ethereum-based transactions, we conducted an extensive evaluation using the BCCC-FraudDeFiTrans-2025 dataset. The AGA model was benchmarked against a diverse set of baselines, including traditional rule-based approaches such as cash flow tree analysis, as well as machine learning models like Random Forest, AdaBoost, and Decision Trees. While these conventional methods demonstrated reasonable performance across various scenarios, they consistently lagged behind AGA. This performance gap underscores AGA's strength in capturing complex fraud patterns—enabled by its advanced profiling mechanisms and a rich, domain-informed feature set that collectively improve its accuracy in identifying fraudulent behaviors.

We further compared AGA to state-of-the-art neural network-based models, including Graph Neural Networks (GNNs) and Transformer architectures. Although these models are designed to learn intricate relationships and temporal patterns, AGA either matched or outperformed them, particularly in terms of precision. A key shortcoming of deep learning models is their "black-box" nature, which often conceals the rationale behind their predictions. This opacity poses a significant limitation in the context of financial fraud detection, where the interpretability and traceability of decisions are crucial. In contrast, AGA offers a transparent decision-making process by relying on well-defined attributes that directly correspond to specific transactional behaviors. This transparency enables analysts and auditors to understand better, validate, and trust the model's assessments.

Overall, the AGA model excels in distinguishing between legitimate and fraudulent Ethereum transactions, achieving a strong balance between F1-score and accuracy. Its interpretable, feature-driven design not only ensures high performance but also makes it a practical and reliable solution for real-world fraud detection in blockchain-based financial systems. By combining precision, transparency, and robust detection capabilities, AGA significantly outperforms both learning and non-learning alternatives.

Table 12
Performance Comparison: AGA versus Previous Models

Model	Recall	Precision	F1-score
Traditional Methods			
Cash Flow Tree Analysis	0.84	0.70	0.76
Machine Learning Methods			
Random Forest + AdaBoost + Decision Tree	0.90	0.73	0.80
Neural Network-Based Methods			
Graph Neural Networks	0.83	0.93	0.87
Transformer Based	0.94	0.89	0.91
Proposed Model			
Proposed AGA	0.95	0.96	0.96

6. Discussion

The proposed model relies solely on transaction features to detect fraud transactions, including zero-day transactions, which are the first transactions of a new wallet, without depending on wallet transaction history, which previous researchers have traditionally relied on. This section presents a comprehensive analysis of profiling features using the AGA profiling method, detailing the methodology and key findings. It also illustrates how profiles are visualized, distinguishing between fraud and legitimate Ethereum-based transactions based on specific genetic markers.

6.1. Gen-selected Features for Profiling

This section provides an in-depth discussion of the profiling results derived from the proposed AGA method, visualized through violin plots. To illustrate the characteristics of fraud and legitimate samples, 16 violin plots are generated for all features, as shown in Figure 7. Each plot distinctly represents fraud transactions in red and legitimate transactions in green. These visualizations effectively highlight the disparities in feature distributions between fraud and legitimate Ethereum-based transactions. In the following, we analyze each feature based on its definition and characteristics.

The first feature, *block_number*, is crucial in establishing the sequence in which blocks are added to the blockchain. Each block number corresponds to a specific block containing a set of transactions. Once integrated into the blockchain, the block number becomes immutable, serving as a fixed historical marker. This fixed sequencing aids in the analysis of transaction patterns over time. This feature's distribution, shown in Fig 7 (part "a"), presents that distributions with higher block numbers often show elevated values, suggesting that instances of fraud transactions tend to increase in these higher ranges. This pattern suggests that the complexity and potential for fraudulent activities will increase as the blockchain evolves.

The second feature is *chain_id*, a crucial component of the transaction signature process in Ethereum. Integrating the *chain_id* into the signature ensures that each transaction is uniquely tied to a specific network, making it invalid if replayed on another network with a different *chain_id*. When analyzing the violin plot of *chain_id* distributions shown in Fig 7 (part "b"), we observe similar fluctuations between fraud and legitimate transactions. Moreover, higher values (close to 1) are predominantly associated with fraudulent transactions, and lower values (close to 0) with legitimate ones. This is due to factors including less stringent security measures and smaller user bases that are less attentive.

The next feature is *length_address_"to"*, which refers to the length of the receiver's Ethereum address in a transaction. As illustrated in Fig 7 (part "c"), fraud transactions exhibit minimal diversity. This pattern suggests that fraud activities utilize a small, consistent set of receiver addresses, which are controlled by the fraudsters themselves, to facilitate and conceal the movement of funds.

The next feature is *length_log*, representing the number of log entries a transaction generates. As shown in Fig 7 (part "d"), transactions with values near 0.0 display similar behavior across fraud and legitimate cases. However, for greater values, they display entirely different fluctuations at various values. This pattern means that while small or no log entries might be typical in regular operations, significant discrepancies in the number of higher ranges are indicative of complex transactions potentially associated with fraud activities, where multiple log entries might be used to execute or obscure malicious intents.

Focusing on gas-related features such as *cumulative_gas_used*, *gas_used*, and *effective_gas_price* is instrumental in distinguishing between fraud and legitimate activities. The first three important features are *cumulative_gas_used*,

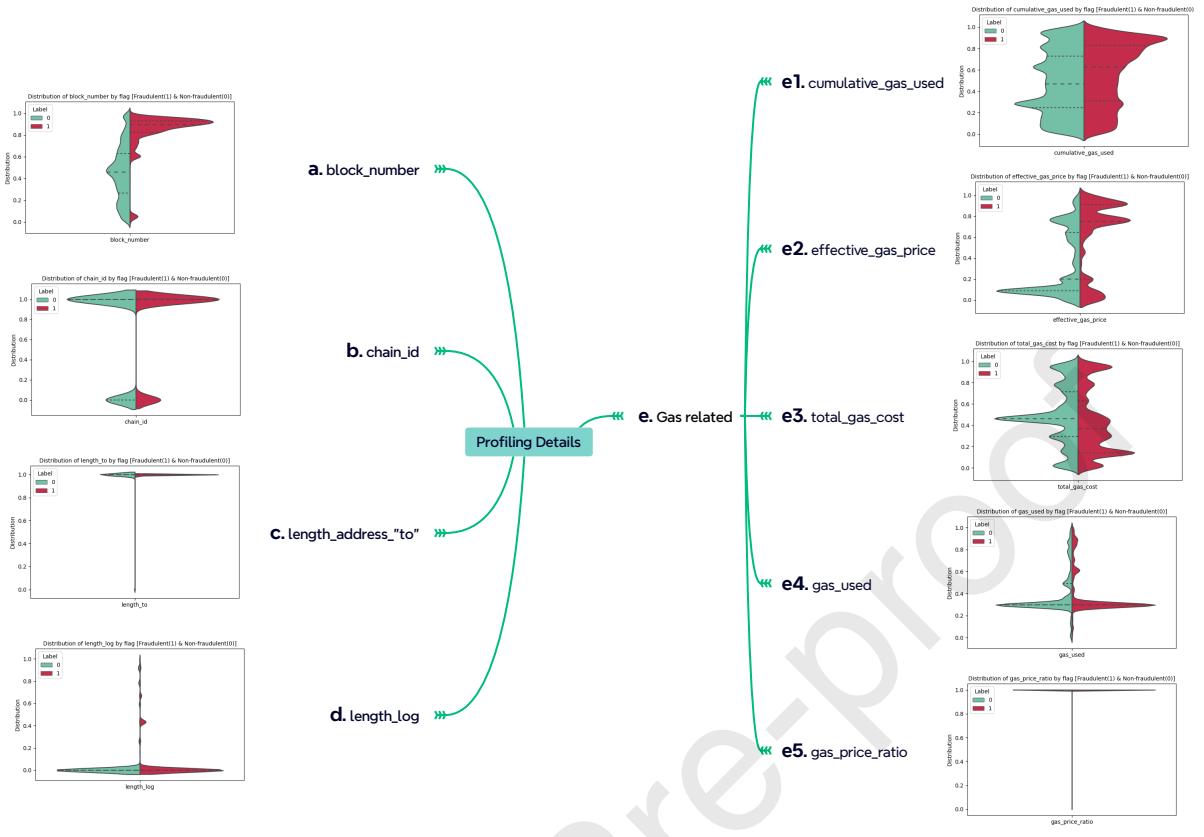

Figure 7: Profiling Details

effective_gas_price, and *total_gas_cost*. The metric *cumulative_gas_used*, as visualized in Fig 7 (part “e1"), reflects the total gas expended up to a specific transaction within a block. Anomalies in this metric, such as unusually high values, indicate complex operations often associated with schemes designed to obfuscate fraud activities.

Moreover, *effective_gas_price*, as shown in Fig 7 (part “e2"), which includes the base fee and any priority fee, reflects the cost per gas unit. This metric helps identify the urgency of a transaction; unusually high fees suggest a fraudster’s attempt to expedite processing to avoid detection, whereas significantly low fees during peak times could indicate cost manipulation in non-urgent fraud activities. Similarly, *total_gas_cost*, as shown in Fig. 7 (part “e3"), represents the aggregate gas expense for a transaction and can indicate fraud when costs are exceptionally high, potentially masking illicit activities within high-value transactions.

Furthermore, *gas_used*, as depicted in Fig. 7 (part ”e4"), represents the total gas consumed during the execution of a transaction. Peaks in gas usage—particularly in contexts where similar transactions typically require less—can suggest that a transaction carries out unusual or additional operations. Additionally, *gas_price_ratio*, as illustrated in Fig. 7 (part ”e5"), compares the *effective_gas_price* with the initially specified *gas_price* by the user. This ratio provides insights into how much more (or less) the user pays than expected. Most fraud transactions have values close to or equal to 1, indicating that the initiator paid more than the standard rate.

The proposed AGA effectively identifies fraud and legitimate patterns in Ethereum-based transactions. However, some features exhibit similar value ranges, peaks, and fluctuations that differentiate the patterns. This highlights the complex nature of transaction fraud, demonstrating that while fraud and legitimate transactions may share characteristics within specific metrics, they also possess distinct traits that distinguish them.

It is important to note that the labels "fraudulent" and "legitimate" were not generated using the on-chain features analyzed in this section. Instead, we used a set of pre-labeled transactions obtained from Etherscan, where fraudulent addresses and contracts are publicly tagged. Our profiling and genetic algorithm analysis was then applied to these labeled transactions to identify the most influential features associated with fraudulent behavior.

Table 13
Sensitivity of AGA Performance to Different Values of λ

λ Value	F1-Score	Accuracy
0.0	0.93	0.91
0.1	0.95	0.94
0.3	**0.96**	**0.95**
0.5	0.92	0.90

6.2. Weighted Profiling Using Gen-selected features

This subsection outlines the profiling process for fraud and legitimate transactions. It begins by explaining the AGA procedure for selecting the most relevant features and then presents the resulting profiles for both legitimate and fraud transactions. Additionally, it highlights the effectiveness of AGA through a confusion matrix, emphasizing the algorithm's role in creating accurate and reliable profiles for distinguishing between different types of transactions.

The initial step in the profiling process involves assigning weights to transaction features to measure their contribution to identifying fraud behavior. These weights are represented as

$$w = [w_1, w_2, \ldots, w_d] \quad \text{with} \quad w_j \in [0, 1],$$

and are determined using AGA optimization. The AGA iteratively learns the optimal weights w^* by maximizing a fitness function $F(w)$, which evaluates the performance of the selected features. The fitness function is formulated as:

$$F(w) = Accuracy(w) - \lambda \cdot Penalty(w),$$

where λ is a regularization coefficient that controls the trade-off between accuracy and overfitting. The penalty term mitigates noise and redundancy, ensuring the selected features provide reliable and interpretable results.

For each value of λ, we trained the AGA-based profiling model and evaluated its performance using the F1-score and accuracy. The results are summarized in Table 13, and show that:

- When $\lambda = 0$, the model tends to overfit, resulting in higher false positives.
- A moderate value of $\lambda = 0.3$ yields the best trade-off, maintaining both accuracy and robustness.
- Increasing λ to 0.5 reduces overfitting but slightly decreases overall detection accuracy.

This analysis confirms that incorporating a properly tuned penalty term in the fitness function improves the model's generalizability and reduces overfitting. The default setting of $\lambda = 0.3$ used throughout the main experiments was selected based on this evaluation.

AGA operates through an evolutionary process involving selection, crossover, and mutation. During selection, weight configurations with higher fitness values are more likely to be retained for the next generation, promoting the survival of high-performing solutions. Crossover combines two parent weight vectors $w^{(p1)}$ and $w^{(p2)}$ to generate offspring, enabling exploration of new feature weight combinations:

$$w_j^{(c)} = \begin{cases} w_j^{(p1)} & \text{if } j \le k, \\ w_j^{(p2)} & \text{if } j > k, \end{cases}$$

where k is the crossover point. Mutation further introduces diversity by perturbing weights with small random noise, allowing the algorithm to escape local optima:

$$w_j' = w_j + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

The AGA retains the best-performing configurations (elite solutions) across generations to preserve progress. The optimization process terminates once $F(w)$ converges, yielding the optimized weights w^* that balance accuracy and robustness.

These optimized weights are then used to calculate the detection score for fraud transactions, providing a quantitative measure of risk. Figure 8 illustrates the relative importance of each feature, with the AGA-calculated

weights displayed at the bottom of the plot. These weights visually highlight the contributions of individual features in distinguishing between fraudulent and legitimate activities. Additionally, Figure 9 presents the confusion matrix generated using only the AGA-optimized features, showcasing the improved detection capability achieved through the profiling process. By leveraging AGA, the profiling model effectively identifies fraud activities while minimizing false positives and irrelevant noise.

Thus, the final features selected through optimization via AGA are used to calculate the fraud detection score. Figure 8 visualizes the importance of each feature, with the corresponding weights displayed at the bottom of each feature plot. Additionally, Figure 9 presents the confusion matrix using only the features selected by AGA profiling, which highlights the effectiveness of AGA in identifying fraudulent activities.

By leveraging AGA, the profiling model effectively identifies fraud activities while minimizing false positives and irrelevant noise. This becomes particularly important in detecting zero-day attacks, where fraudulent behaviors may not follow known patterns. Unlike previous approaches that rely heavily on static rules or complex black-box representations, our method builds on interpretable, behavior-driven features such as abnormal gas usage, token transfer volume, and transaction frequency—characteristics that often shift subtly in emerging attack patterns.

The optimized weights derived by AGA reveal which features most strongly contribute to identifying anomalous behavior. For instance, in our experimental cases, AGA successfully flagged transactions exhibiting rare gas usage and sudden spikes in transfer value, which traditional models failed to detect due to reliance on pre-trained or signature-based logic. These patterns were previously unrecognized in known attack profiles, indicating the model's adaptability to new fraud behaviors. Figure 8 illustrates the influence of these features, and Figure 8 confirms the performance improvement, particularly in edge cases where fraudulent transactions did not match any known pattern. This demonstrates the model's strength in capturing zero-day attacks through meaningful feature weighting and evolutionary optimization.

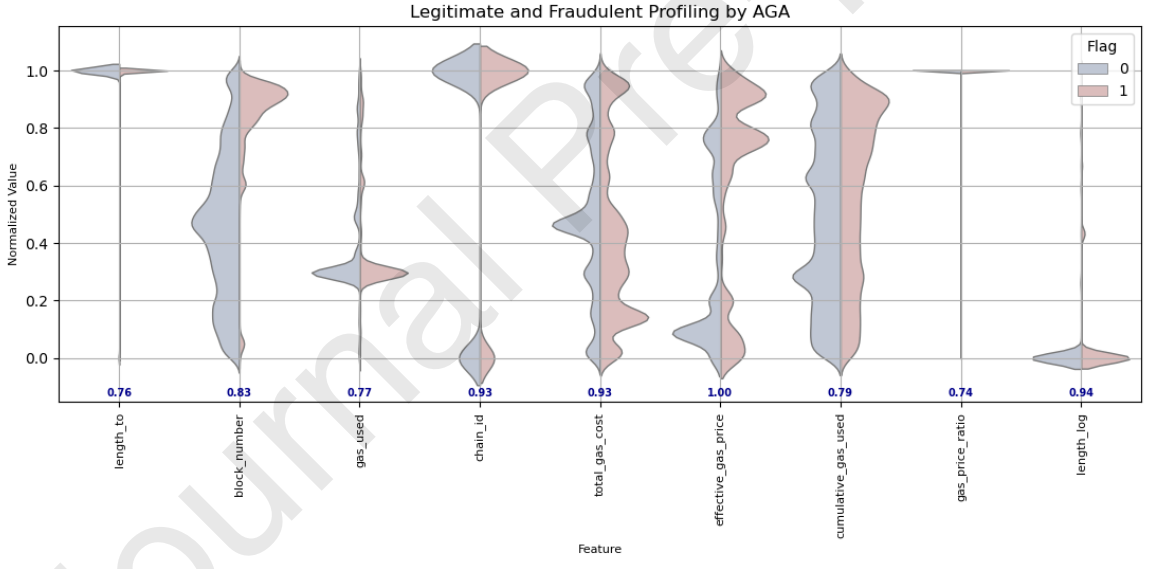

Figure 8: Legitimate and Fraud Profiling by AGA: The blue numbers at the bottom of each plot represent the weights assigned by AGA to the corresponding features. Flag 0 indicates legitimate transactions, while Flag 1 denotes fraudulent ones.

7. Conclusion and Future Works

The transition from CeFi to DeFi has revolutionized financial transactions by enabling trustless, automated operations through blockchain technology. While the shift to DeFi enhances transparency and efficiency, it also brings significant security risks, with fraud emerging as one of the most prevalent and pressing issues.

Various detection mechanisms have been implemented, which can be categorized into four primary groups, each targeting different levels of the DeFi ecosystem to detect fraud. One significant limitation of these methods is their

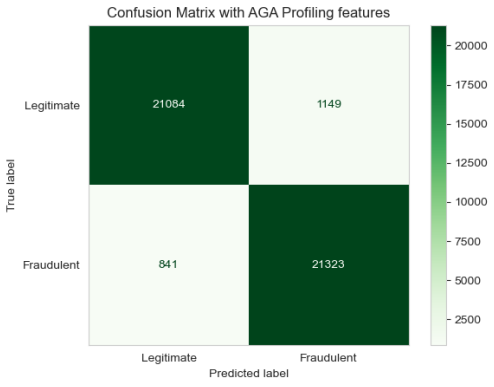

Figure 9: Confusion Matrix with AGA Profiling Features: The matrix illustrates the performance of the AGA profiling method in distinguishing between legitimate and fraudulent transactions. True labels are displayed along the vertical axis, while predicted labels are on the horizontal axis.

reliance on historical data, which restricts their ability to detect zero-day fraud attacks, such as the initial transaction of a newly created wallet. Moreover, the effectiveness of these detection tools is limited by their reliance on the quality and diversity of training data, with the lack of large, comprehensive datasets posing a significant challenge. This research introduces an advanced Genetic Algorithm (AGA) for profiling fraud and legitimate transactions. This enhanced GA version features a specially designed fitness function and adaptive mechanisms that dynamically adjust mutation rates and generations. It balances exploration and exploitation effectively, yielding robust and refined outcomes.

Additionally, Semantic Similarity Distance (SSD) checks are applied before crossover events to ensure that genetic recombination occurs between individuals with an optimal level of similarity, thus preserving genetic diversity while preventing the generation of unrealistic solutions. The mutation rate is adaptively managed, starting at a higher rate to promote broad exploration of the solution space and gradually decreasing as promising solutions are identified. We also developed an analyzer, DeFiTransLyzer-V1.0, which explicitly leverages AGA to profile transactions. Additionally, we created a comprehensive benchmark dataset, BCCC-DeFiFraudTrans-2025, consisting of 1,026,867 Ethereum-based transaction samples that cover a wide range of examples to ensure thorough validation.

Our model demonstrated superior performance in distinguishing between fraudulent and legitimate transactions through comprehensive testing and evaluation. Compared to both learning-based and non-learning-based baselines, it achieved a precision of 0.95, a recall of 0.96, and an F1-score of 0.96—reflecting high accuracy, reliability, and generalization. In addition to performance, the profiling mechanism used in our approach provides strong interpretability by identifying the behavioral contributions of individual transaction features. This transparency is essential for building trust in fraud detection systems within DeFi platforms.

From a practical standpoint, our model is designed to operate efficiently and support real-time fraud detection. The average execution time of 37.49 seconds refers to batch processing of a large number of transactions, not individual transaction latency. Once profiling is complete and optimal feature weights are established, the system can evaluate and classify new transactions in near real-time based on the learned fraud profiles. This makes it highly suitable for real-time monitoring scenarios, such as flagging suspicious transactions as they enter the network.

Moreover, the model's ability to detect zero-day attacks—even during the first transaction of a new wallet—adds a critical layer of proactive defense. Unlike conventional systems that rely on historical patterns or known addresses, our method can generalize from behavioral patterns to identify previously unseen threats. This adaptability is crucial in the rapidly evolving DeFi landscape, where attackers continually refine their tactics. Finally, the model is lightweight and modular, enabling seamless integration into the backends of Ethereum-based platforms. This integration can enable automated risk tagging, transaction blocking, or alert generation, providing a powerful tool for real-time threat prevention without the need for high-latency processing pipelines.

In the future, expanding the discussion to include the adaptability of the proposed approach to DeFi ecosystems on other blockchain platforms will be valuable. While this work primarily focuses on fraud detection within the Ethereum-based DeFi landscape, the methodology outlined in Section 3 is designed to be flexible and extendable. Our goal is to

broaden the model's applicability to detect fraudulent activities across diverse blockchain platforms and DeFi protocols, thereby enhancing its robustness and generalizability. This work lays the groundwork for several promising research avenues. While our current focus has been on detecting fraud patterns, future efforts could broaden the profiler's scope to include other types of attacks, such as phishing and money laundering activities. A more comprehensive understanding of these threats would enhance the model's flexibility, making it applicable to a broader range of vulnerabilities. Additionally, scalability remains a crucial area for enhancement, particularly in optimizing the profiler to efficiently manage larger volumes of transactions as DeFi systems expand in size and complexity. Furthermore, the taxonomies established in this study should be continuously revised and updated to reflect new developments. As profiling techniques and security protocols evolve, keeping classifications current will ensure that detection methods remain practical and relevant, ultimately strengthening the security of the DeFi ecosystem.

Acknowledgements

The authors declare that they have no competing interests related to this work. The authors acknowledge the grant from Canada Research Chair - Tier II (#CRC-2021-00340) and the Natural Sciences and Engineering Research Council of Canada — NSERC (#RGPIN-2020-04701) — to Arash Habibi Lashkari.

References

[1] Y. Karaulova, Decentralized finance to improve the performance of centralized finance, Journal of Advanced Research in Law and Economics (JARLE) 8 (26) (2017) 1167–1174.
[2] D. A. Zetzsche, D. W. Arner, R. P. Buckley, Decentralized finance, Journal of Financial Regulation 6 (2) (2020) 172–203.
[3] X. Sun, C. Stasinakis, G. Sermpinis, Decentralization illusion in decentralized finance: Evidence from tokenized voting in makerdao polls, Journal of Financial Stability (2024) 101286.
[4] H. E. Jackson, Centralization, competition, and privatization in financial regulation, Theoretical Inquiries in Law 2 (2) (2001).
[5] J. Son, D. Ryu, Competitive dynamics between decentralized and centralized finance lending markets, International Review of Financial Analysis 96 (2024) 103699.
[6] C. Dong, C. Chen, X. Shi, C. T. Ng, Operations strategy for supply chain finance with asset-backed securitization: Centralization and blockchain adoption, International Journal of Production Economics 241 (2021) 108261.
[7] C. Carpentier-Desjardins, M. Paquet-Clouston, S. Kitzler, B. Haslhofer, Mapping the defi crime landscape: an evidence-based picture, Journal of Cybersecurity 11 (1) (2025) tyae029.
[8] A. Trozze, B. Kleinberg, T. Davies, Detecting defi securities violations from token smart contract code (2023). arXiv:2112.02731.
URL https://arxiv.org/abs/2112.02731
[9] R. Gan, L. Zhou, L. Wang, K. Qin, X. Lin, Defialigner: Leveraging symbolic analysis and large language models for inconsistency detection in decentralized finance, in: 6th Conference on Advances in Financial Technologies (AFT 2024), Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024.
[10] R. M. Aziz, M. F. Baluch, S. Patel, A. H. Ganie, Lgbm: a machine learning approach for ethereum fraud detection, International Journal of Information Technology 14 (7) (2022) 3321–3331.
[11] V. Ravindranath, M. Nallakaruppan, M. L. Shri, B. Balusamy, S. Bhattacharyya, Evaluation of performance enhancement in ethereum fraud detection using oversampling techniques, Applied Soft Computing 161 (2024) 111698.
[12] B. Norouzi, Enhancing fraudulent account detection on the ethereum blockchain: A novel feature selection approach and comparative analysis of machine learning algorithms, Master's thesis, Bishop's University, https://www.ubishops.ca/wp-content/uploads/norouzi20230801.pdf (August 2023).
[13] B. Wang, X. Yuan, L. Duan, H. Ma, C. Su, W. Wang, Defiscanner: Spotting defi attacks exploiting logic vulnerabilities on blockchain, IEEE Transactions on Computational Social Systems 11 (2) (2022) 1577–1588.
[14] S. S. Taher, S. Y. Ameen, J. A. Ahmed, Advanced fraud detection in blockchain transactions: An ensemble learning and explainable ai approach, Engineering, Technology & Applied Science Research 14 (1) (2024) 12822–12830.
[15] H. Kanezashi, T. Suzumura, X. Liu, T. Hirofuchi, Ethereum fraud detection with heterogeneous graph neural networks, arXiv preprint arXiv:2203.12363 (2022).
URL https://arxiv.org/pdf/2203.12363
[16] G. Palaiokrassas, S. Scherrers, I. Ofeidis, L. Tassiulas, Leveraging machine learning for multichain defi fraud detection, in: 2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), IEEE, 2024, pp. 678–680.
[17] V. Mothukuri, R. M. Parizi, J. L. Massa, A. Yazdinejad, An ai multi-model approach to defi project trust scoring and security, in: 2024 IEEE International Conference on Blockchain (Blockchain), IEEE, 2024, pp. 19–28.
[18] G. Palaiokrassas, S. Scherrers, E. Makri, L. Tassiulas, Machine learning in defi: Credit risk assessment and liquidation prediction, in: 2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), IEEE, 2024, pp. 650–654.
[19] S. Wu, D. Wang, J. He, Y. Zhou, L. Wu, X. Yuan, Q. He, K. Ren, Defiranger: Detecting price manipulation attacks on defi applications (2021). arXiv:2104.15068.
URL https://arxiv.org/abs/2104.15068

[20] S. Kitzler, F. Victor, P. Saggese, B. Haslhofer, Disentangling decentralized finance (defi) compositions (2022). arXiv:2111.11933.
URL https://arxiv.org/abs/2111.11933

[21] D. Sun, W. Ma, L. Nie, Y. Liu, Sok: Comprehensive analysis of rug pull causes, datasets, and detection tools in defi (2024). arXiv:2403.16082.
URL https://arxiv.org/abs/2403.16082

[22] N. Carter, L. Jeng, Defi protocol risks: The paradox of defi, Regtech, suptech and beyond: innovation and technology in financial services" riskbooks–forthcoming Q 3 (2021).

[23] Y. Xue, D. Fan, S. Su, J. Fu, N. Hu, W. Liu, Z. Tian, A review on the security of the ethereum-based defi ecosystem., CMES-Computer Modeling in Engineering & Sciences 139 (1) (2024).

[24] P. Qian, R. Cao, Z. Liu, W. Li, M. Li, L. Zhang, Y. Xu, J. Chen, Q. He, Empirical review of smart contract and defi security: vulnerability detection and automated repair, arXiv preprint arXiv:2309.02391 (2023).

[25] Z. Chen, S. M. Beillahi, F. Long, Flashsyn: Flash loan attack synthesis via counter example driven approximation, in: Proceedings of the IEEE/ACM 46th International Conference on Software Engineering, 2024, pp. 1–13.

[26] R. Tan, Q. Tan, Q. Zhang, P. Zhang, Y. Xie, Z. Li, Ethereum fraud behavior detection based on graph neural networks, Computing 105 (10) (2023) 2143–2170.

[27] M. Xie, M. Hu, Z. Kong, C. Zhang, Y. Feng, H. Wang, Y. Xue, H. Zhang, Y. Liu, Y. Liu, Defort: Automatic detection and analysis of price manipulation attacks in defi applications, in: Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis, 2024, pp. 402–414.

[28] B. Luo, Z. Zhang, Q. Wang, A. Ke, S. Lu, B. He, Ai-powered fraud detection in decentralized finance: A project life cycle perspective, ACM Computing Surveys (2023).

[29] H. An, R. Ma, Y. Yan, T. Chen, Y. Zhao, P. Li, J. Li, X. Wang, D. Fan, C. Lv, Finsformer: A novel approach to detecting financial attacks using transformer and cluster-attention, Applied Sciences 14 (1) (2024) 460.

[30] M. Gunathilaka, S. Wickramanayake, H. D. Bandara, Defitrust: A transformer-based framework for scam defi token detection using event logs and sentiment analysis, Expert Systems with Applications 251 (2024) 123913.

[31] Y. Jia, Y. Wang, J. Sun, Y. Liu, Z. Sheng, Y. Tian, Ethereum fraud detection via joint transaction language model and graph representation learning, arXiv preprint arXiv:2409.07494 (2024).

[32] M. Liu, J. H. Huh, H. Han, J. Lee, J. Ahn, F. Li, H. Kim, T. Kim, I experienced more than 10 defi scams: On defi users' perception of security breaches and countermeasures, in: 33rd USENIX Security Symposium (USENIX Security 24), USENIX Association, 2024.

[33] S. Hu, Z. Zhang, B. Luo, S. Lu, B. He, L. Liu, Bert4eth: A pre-trained transformer for ethereum fraud detection, in: Proceedings of the ACM Web Conference 2023, 2023, pp. 2189–2197.

[34] C. Jin, J. Zhou, C. Xie, S. Yu, Q. Xuan, X. Yang, Enhancing ethereum fraud detection via generative and contrastive self-supervision, arXiv preprint arXiv:2408.00641 (2024).

[35] C. Langensiepen, A. Lotfi, S. Puteh, Activities recognition and worker profiling in the intelligent office environment using a fuzzy finite state machine, in: 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, 2014, pp. 873–880.

[36] A. Guillén, Y. Gutiérrez, R. Muñoz, Natural language processing technologies for document profiling (2017) 284–290.

[37] S. HajiHosseinKhani, A. H. Lashkari, A. M. Oskui, Unveiling vulnerable smart contracts: Toward profiling vulnerable smart contracts using genetic algorithm and generating benchmark dataset, Blockchain: Research and Applications (2023) 100171.

[38] S. HajiHosseinKhani, A. H. Lashkari, A. M. Oskui, Unveiling smart contracts vulnerabilities: Toward profiling smart contracts vulnerabilities using ega and generating benchmark dataset, Blockchain: Research and Applications (Submitted July 2024).

[39] Etherscan: Ethereum blockchain explorer.
URL https://etherscan.io

[40] J. Kim, S. Yoon, Ai bias in financial fraud detection, Financial Technology and Ethics 11 (3) (2022) 215–228.

[41] R. Johnson, H. Lee, Machine learning in financial fraud detection, Finance Technology Journal 15 (2) (2023) 67–79.

[42] J. Smith, R. Patel, The evolution of rule-based fraud detection in cefi, Journal of Financial Security 10 (3) (2022) 123–134.

[43] K. Qin, L. Zhou, Y. Afonin, L. Lazzaretti, A. Gervais, Cefi vs. defi–comparing centralized to decentralized finance, arXiv preprint arXiv:2106.08157 (2021).

[44] A. Lopez, et al., The role of neural networks in fraud detection, Journal of Artificial Intelligence in Finance 8 (1) (2023) 44–58.

[45] P. Martin, Q. Liu, Advancements in unsupervised learning for cefi fraud detection, Computational Economics 30 (4) (2022) 401–414.

[46] D. Chen, L. Feng, Y. Fan, S. Shang, Z. Wei, Smart contract vulnerability detection based on semantic graph and residual graph convolutional networks with edge attention, Journal of Systems and Software 202 (2023) 111705.

[47] S. Al-E'mari, M. Anbar, Y. Sanjalawe, S. Manickam, A labeled transactions-based dataset on the ethereum network, in: International Conference on Advances in Cyber Security, Springer, 2020, pp. 61–79.

[48] K. John, B. Monnot, P. Mueller, F. Saleh, C. Schwarz-Schilling, Economics of ethereum, Available at SSRN 4783695 (2024).

[49] M. Imran, B. Yao, W. Ali, A. Akhunzada, M. K. Azhar, M. Junaid, U. Iqbal, Research perspectives and challenges of blockchain for data-intensive and resource-constrained devices, IEEE Access 10 (2022) 38104–38122.

[50] B. Luo, Z. Zhang, Q. Wang, A. Ke, S. Lu, B. He, Ai-powered fraud detection in decentralized finance: A project life cycle perspective, ACM Computing Surveys 57 (4) (2024) 1–38.

[51] E. Hoch, The defi ecosystem game: Proof-via-simulations, in: The International Conference on Mathematical Research for Blockchain Economy, Springer, 2024, pp. 285–314.

[52] O. Tjäder, L. Ulrich, The great defi dilemma: How stakeholders can navigate the uncertain waters of decentralised finance adoption: An explorative study (2023).

[53] M. Salah, et al., Decentralized finance (defi) on blockchain: Current landscape and future trends, Journal of Innovative Technologies 6 (1) (2023) 1–13.

[54] F. Jia, J. Zheng, F. Li, Decentralized intelligence in gamefi: Embodied ai agents and the convergence of defi and virtual ecosystems, arXiv preprint arXiv:2412.18601 (2024).
[55] B. Kaplan, V. F. Benlı, E. A. Alp, Blockchain based decentralized lending protocols: A return analysis between s&p 500 and defi assets, JOEEP: Journal of Emerging Economies and Policy 8 (1) (2023) 360–378.
[56] A. Turillazzi, A. Tsamados, E. Genç, M. Taddeo, L. Floridi, Decentralised finance (defi): a critical review of related risks and regulation, Available at SSRN (2023).
[57] S. Ren, T. Tu, J. Liu, D. Wu, K. Ren, Lookahead: Preventing defi attacks via unveiling adversarial contracts (2024). arXiv:2401.07261.
URL https://arxiv.org/abs/2401.07261
[58] D. Rabetti, Auditing decentralized finance (defi) protocols, in: Proceedings of the Conference on Decentralized Finance (DEFI) Protocols, Vol. 15, 2023, pp. 1–58.
[59] R. M. Aziz, M. F. Baluch, S. Patel, P. Kumar, A machine learning based approach to detect the ethereum fraud transactions with limited attributes, Karbala International Journal of Modern Science (2022).
URL https://api.semanticscholar.org/CorpusID:248629006
[60] D. Wang, B. Wu, X. Yuan, L. Wu, Y. Zhou, H. Cui, Defiguard: A price manipulation detection service in defi using graph neural networks (2024). arXiv:2406.11157.
URL https://arxiv.org/abs/2406.11157
[61] R. Tan, Q. Tan, P. Zhang, Z. Li, Graph neural network for ethereum fraud detection, in: 2021 IEEE international conference on big knowledge (ICBK), IEEE, 2021, pp. 78–85.
[62] J. Smith, Ai-driven blockchain analytics: Leveraging deep learning for fraud detection in decentralized networks, Journal of Artificial Intelligence Research and Applications 4 (2) (2024) 50–55.
[63] B. Wang, H. Liu, C. Liu, Z. Yang, Q. Ren, H. Zheng, H. Lei, Blockeye: Hunting for defi attacks on blockchain, in: 2021 IEEE/ACM 43rd international conference on software engineering: companion proceedings (ICSE-companion), IEEE, 2021, pp. 17–20.
[64] C. Carpentier-Desjardins, M. Paquet-Clouston, S. Kitzler, B. Haslhofer, Mapping the defi crime landscape: An evidence-based picture, arXiv preprint arXiv:2310.04356 (2023).
[65] T. Weingärtner, F. Fasser, P. Reis Sá da Costa, W. Farkas, Deciphering defi: A comprehensive analysis and visualization of risks in decentralized finance, Journal of risk and financial management 16 (10) (2023) 454.
[66] S. Ratra, M. Ghosh, N. Baliyan, J. Rashmitha Mohan, S. Singh, Graph neural network based phishing account detection in ethereum, The Computer Journal (2024) bxae079.
[67] R. Gupta, N. K. Gupta, M. L. Soffa, A case for profiling-oriented software engineering, IEEE Software 13 (1) (1996) 22–31.
URL https://doi.org/10.1109/52.485327
[68] B. J. Cox, The execution time measurement and profiling of a program, Communications of the ACM 15 (10) (1972) 801–805.
[69] B. Zadrozny, C. Elkan, Profiling user sessions for fun and profit: Data, methods and models, in: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2002, pp. 627–632.
URL https://doi.org/10.1145/775047.775141
[70] T. Hoffelder, Equivalence analyses of dissolution profiles with the mahalanobis distance, Biometrical Journal 61 (5) (2019) 1120–1137.
[71] S. Kim, B. Ham, B. Kim, K. Sohn, Mahalanobis distance cross-correlation for illumination-invariant stereo matching, IEEE Transactions on Circuits and Systems for Video Technology 24 (11) (2014) 1844–1859.
[72] F. Iorio, R. Tagliaferri, D. d. Bernardo, Identifying network of drug mode of action by gene expression profiling, Journal of Computational Biology 16 (2) (2009) 241–251.
[73] A. F. Gündüz, A. Karadoğan, Community detection in social media network with maximum modularity using girvan-newman algorithm, in: SETSCI-Conference Proceedings, Vol. 1, SETSCI-Conference Proceedings, 2017, pp. 222–225.
[74] D. N. Rassokhin, D. K. Agrafiotis, Kolmogorov-smirnov statistic and its application in library design, Journal of Molecular Graphics and Modelling 18 (4-5) (2000) 368–382.
[75] X. Zhang, F. Qiu, F. Qin, Identification and mapping of winter wheat by integrating temporal change information and kullback–leibler divergence, International Journal of Applied Earth Observation and Geoinformation 76 (2019) 26–39.
[76] S. Yu, H. Zhang, T. Li, Fuzzy profiling: A framework for discovering unknown malware, in: Proceedings of the 9th International Conference on Intelligent Systems Design and Applications (ISDA), 2009, pp. 838–843.
URL https://doi.org/10.1109/ISDA.2009.214
[77] R. Zuech, A. Goodrum, Fuzzy profiling for the detection of anomalous program behavior, Computers & Security 24 (2) (2005) 123–139.
URL https://doi.org/10.1016/j.cose.2004.11.001
[78] C. Liu, G. Wang, K. Peng, Fuzzy profiling-based approach to anomaly detection in network traffic, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 37 (2) (2007) 202–211.
URL https://doi.org/10.1109/TSMCC.2006.886813
[79] O. Alhabashneh, R. Iqbal, F. Doctor, S. Amin, Adaptive information retrieval system based on fuzzy profiling, in: 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, 2015, pp. 1–8.
[80] D. Xu, H. Wang, K. Su, Intelligent student profiling with fuzzy models, in: Proceedings of the 35th Annual Hawaii international conference on system sciences, IEEE, 2002, pp. 8–pp.
[81] J. E. Dickerson, J. A. Dickerson, Fuzzy network profiling for intrusion detection, in: PeachFuzz 2000. 19th International Conference of the North American Fuzzy Information Processing Society-NAFIPS (Cat. No. 00TH8500), IEEE, 2000, pp. 301–306.
[82] R. Smith, A finite state machine algorithm for finding restriction sites and other pattern matching applications, Bioinformatics 4 (4) (1988) 459–465.
[83] A. Fernández-Isabel, P. Peixoto, I. M. de Diego, C. Conde, E. Cabello, Combining dynamic finite state machines and text-based similarities to represent human behavior, Engineering Applications of Artificial Intelligence 85 (2019) 504–516.

[84] E. Mezura-Montes, J. Velázquez-Reyes, C. A. Coello Coello, Profiling-based adaptive genetic algorithm, IEEE Transactions on Evolutionary Computation 13 (5) (2009) 1053–1070.
URL https://doi.org/10.1109/TEVC.2008.2011727
[85] G. Kendall, E. K. Burke, M. Gendreau, B. Ombuki-Berman, B. McCollum, E. Özcan, R. Qu, On the use of profiling techniques in genetic algorithm-based hyper-heuristics, Journal of the Operational Research Society 58 (6) (2007) 708–718.
URL https://doi.org/10.1057/palgrave.jors.2602257
[86] P. Asokan, R. Saravanan, K. Vijayakumar, Machining parameters optimisation for turning cylindrical stock into a continuous finished profile using genetic algorithm (ga) and simulated annealing (sa), The International Journal of Advanced Manufacturing Technology 21 (1) (2003) 1–9.
[87] M. Gupta, J. Rees, A. Chaturvedi, J. Chi, Matching information security vulnerabilities to organizational security profiles: a genetic algorithm approach, Decision Support Systems 41 (3) (2006) 592–603.
[88] P. A. A. Resende, A. C. Drummond, Adaptive anomaly-based intrusion detection system using genetic algorithm and profiling, Security and Privacy 1 (4) (2018) e36.
[89] A. H. Hamamoto, L. F. Carvalho, L. D. H. Sampaio, T. Abrão, M. L. Proença Jr, Network anomaly detection system using genetic algorithm and fuzzy logic, Expert Systems with Applications 92 (2018) 390–402.
[90] S. Lal, C. Cascaval, J. Mars, P. Dey, H. Tang, Profiling machine learning workloads, in: Proceedings of the 2019 International Symposium on Code Generation and Optimization (CGO), 2019, pp. 267–279.
URL https://doi.org/10.1109/CGO.2019.8661173
[91] J. Wang, W. Huang, H. Zhang, X. Wu, Understanding the performance of tensorflow workloads on gpus, in: Proceedings of the 2020 IEEE International Symposium on Workload Characterization (IISWC), 2020, pp. 83–92.
URL https://doi.org/10.1109/IISWC49841.2020.00015
[92] A. Samajdar, V. Sridharan, M. Zinsmaier, Z. Pan, S. Shin, R. Gao, Q. Zhou, R. K. Gupta, Auto-profiling: A framework for profiling and optimization of ml workloads, in: Proceedings of the 2020 USENIX Annual Technical Conference (ATC), 2020, pp. 631–644.
URL https://www.usenix.org/conference/atc20/presentation/samajdar
[93] I. T. Haque, C. Assi, Profiling-based indoor localization schemes, IEEE Systems Journal 9 (1) (2013) 76–85.
[94] E. Tsalera, A. Papadakis, M. Samarakou, Monitoring, profiling and classification of urban environmental noise using sound characteristics and the knn algorithm, Energy Reports 6 (2020) 223–230.
[95] P. Nagaraj, K. Saiteja, K. K. Ram, K. M. Kanta, S. K. Aditya, V. Muneeswaran, University recommender system based on student profile using feature weighted algorithm and knn, in: 2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS), IEEE, 2022, pp. 479–484.
[96] V. Michalakopoulos, E. Sarmas, I. Papias, P. Skaloumpakas, V. Marinakis, H. Doukas, A machine learning-based framework for clustering residential electricity load profiles to enhance demand response programs, Applied Energy 361 (2024) 122943.
[97] R. Bayot, T. Gonçalves, Multilingual author profiling using word embedding averages and svms, in: 2016 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA), IEEE, 2016, pp. 382–386.
[98] P. J. Batterham, H. Christensen, Longitudinal risk profiling for suicidal thoughts and behaviours in a community cohort using decision trees, Journal of affective disorders 142 (1-3) (2012) 306–314.
[99] P. Duchessi, E. J. Lauría, Decision tree models for profiling ski resorts' promotional and advertising strategies and the impact on sales, Expert Systems with Applications 40 (15) (2013) 5822–5829.
[100] S. H. Hawley, B. Colburn, S. I. Mimilakis, Signaltrain: Profiling audio compressors with deep neural networks, arXiv preprint arXiv:1905.11928 (2019).
[101] G. X. Yu, T. Grossman, G. Pekhimenko, Skyline: Interactive in-editor computational performance profiling for deep neural network training, in: Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology, 2020, pp. 126–139.
[102] A. Li, S. Xue, X.-Y. Li, L. Zhang, J. Qian, Appdna: Profiling app behavior via deep-learning function call graphs, IEEE Transactions on Emerging Topics in Computing 10 (1) (2020) 414–427.
[103] A. Cura, H. Küçük, E. Ergen, İ. B. Öksüzoğlu, Driver profiling using long short term memory (lstm) and convolutional neural network (cnn) methods, IEEE Transactions on Intelligent Transportation Systems 22 (10) (2020) 6572–6582.
[104] K. C. Baumgartner, S. Ferrari, C. G. Salfati, Bayesian network modeling of offender behavior for criminal profiling, in: Proceedings of the 44th IEEE Conference on Decision and Control, IEEE, 2005, pp. 2702–2709.
[105] T. Xiang, S. Gong, Video behavior profiling for anomaly detection, IEEE transactions on pattern analysis and machine intelligence 30 (5) (2008) 893–908.
[106] T. Karagiannis, K. Papagiannaki, N. Taft, M. Faloutsos, Profiling the end host, in: Passive and Active Network Measurement: 8th Internatinoal Conference, PAM 2007, Louvain-la-neuve, Belgium, April 5-6, 2007. Proceedings 8, Springer, 2007, pp. 186–196.
[107] K. Han, J. Park, M. Y. Yi, Adaptive and multiple interest-aware user profiles for personalized search in folksonomy: A simple but effective graph-based profiling model, in: 2015 International Conference on Big Data and Smart Computing (BIGCOMP), IEEE, 2015, pp. 225–231.
[108] W. Chen, Y. Gu, Z. Ren, X. He, H. Xie, T. Guo, D. Yin, Y. Zhang, Semi-supervised user profiling with heterogeneous graph attention networks., in: IJCAI, Vol. 19, 2019, pp. 2116–2122.
[109] S. Xue, L. Zhang, A. Li, X.-Y. Li, C. Ruan, W. Huang, Appdna: App behavior profiling via graph-based deep learning, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, IEEE, 2018, pp. 1475–1483.
[110] R. Labadie-Tamayo, D. Castro-Castro, Graph-based profile condensation for users profiling (2022).
[111] H. Asai, K. Fukuda, P. Abry, P. Borgnat, H. Esaki, Network application profiling with traffic causality graphs, International Journal of Network Management 24 (4) (2014) 289–303.
[112] M. Daoud, L. Tamine, M. Boughanem, A personalized graph-based document ranking model using a semantic user profile, in: User Modeling, Adaptation, and Personalization: 18th International Conference, UMAP 2010, Big Island, HI, USA, June 20-24, 2010. Proceedings 18,

Springer, 2010, pp. 171–182.
[113] B. W. G. M. Anrig, Bernhard, The role of algorithms in profiling (2008).
[114] M. Sedighizadeh, A. Rezazadeh, Using genetic algorithm for distributed generation allocation to reduce losses and improve voltage profile, World Academy of Science, Engineering and Technology 37 (1) (2008) 251–256.
[115] W. Zou, V. V. Tolstikov, Probing genetic algorithms for feature selection in comprehensive metabolic profiling approach, Rapid Communications in Mass Spectrometry: An International Journal Devoted to the Rapid Dissemination of Up-to-the-Minute Research in Mass Spectrometry 22 (8) (2008) 1312–1324.
[116] L. Haldurai, T. Madhubala, R. Rajalakshmi, A study on genetic algorithm and its applications, Int. J. Comput. Sci. Eng 4 (10) (2016) 139–143.
[117] N. C. Evans, D. L. Shealy, Design and optimization of an irradiance profile-shaping system with a genetic algorithm method, Applied Optics 37 (22) (1998) 5216–5221.
[118] R. Paulavičius, L. Stripinis, S. Sutavičiūtė, D. Kočegarov, E. Filatovas, A novel greedy genetic algorithm-based personalized travel recommendation system, Expert Systems with Applications 230 (2023) 120580.

Declaration of interests

☐ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☒ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Arash Habibi Lashkari reports financial support was provided by York University. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Author Contribution Statement

Arash Habibi Lashkari, as the founder and supervisor of the research team, provided strategic guidance, ensured methodological rigor, and oversaw the overall direction of the study.
Sepideh Hajihosseinkhani was responsible for data collection, model development, and experimentation. She designed and implemented the advanced genetic algorithm and penalty fitness function, conducted extensive testing on Ethereum transactions, and analyzed the results. She also wrote the experiment and conclusion sections, synthesizing key findings to strengthen the study's contribution to DeFi security and fraud detection.
Joshua Duarte and Isabella Lopez conducted a comprehensive literature review, identifying relevant works on genetic algorithms, penalty functions, and fraud detection in Ethereum transactions. They also contributed to data visualization, ensuring clear and effective presentation of the model's performance and results. Their work enhanced the interpretability of the findings through figures and graphical representations.
Ziba Habibi Lashkari and Sergio Rios-Aguilar contributed as external advisors and international collaborators, providing insights on the theoretical and methodological aspects of the study. Their expertise helped refine the research framework and ensure the study's relevance in the broader context of DeFi security and fraud detection.