Dissertation Proposal

Learning-Based Ethereum Phishing Detection: Evaluation, Robustness, and Improvement

Ahod Alghuried

Date: November 5, 2024

Department of Computer Sciance
University of Central Florida
Orlando, FL 32816

Doctoral Committee:
Dr. David Mohaisen (Chair)
Dr. Cliff Zou
Dr. Xueqiang Wang
Dr. Sung Choi Yoo

Ahod Alghuried

Department of Electrical and Computer Engineering, University of Central Florida (UCF)
4000 Central Florida Blvd., R1-368, Orlando, FL 32816-2362 USA

Education

Ph.D. in Computer Science (2021 – Current)
University of Central Florida

M.Sc. in Computer Security and Forensics (2015 – 2017)
Technological University Dublin
CGPA: 3.62

B.Sc. in Computer Science (2009 – 2013)
University of Tabuk
CGPA: 4.74

Technical Publications And Manuscripts

1. Ahod Alghuried, Mohammed Alkinoon, Manar Mohaisen, An Wang, Cliff C. Zou, and David Mohaisen,"Blockchain Security and Privacy Examined: Threats, Challenges, Applications, and Tools", The ACM Distributed Ledger Technologies: Research and Practice, **ACM DLT**, 2024 (accepted with revisions).

2. Abdulaziz Alghamdi, Ali Alkinoon, Ahod Alghuried, David Mohaisen, "xr-droid: A Benchmark Dataset for AR/VR and Security Applications", IEEE Transactions on Dependable and Secure Computing, **IEEE TDSC**, 2024 (accepted).

3. Ahod Alghuried, and David Mohaisen,"Simple Perturbations Subvert Ethereum Phishing Transactions Detection: An Empirical Analysis", In Proceedings of the 25th International Conference, **WISA** 2024, Jeju Island, South Korea, August 22-24, 2024.

4. Hattan Althebeiti, Ran Gedawy, Ahod Alghuried, Daehun Nyang, and David Mohaisen, "Defending AirType Against Inference Attacks Using 3D In-Air Keyboard Layouts: Design and Evaluation". In Proceedings of the 24th International Conference, **WISA** 2023, Jeju Island, South Korea, August 23-25, 2023.

5. Ahod Alghuried, Ali Alkinoon, Abdulaziz Alghamdi, and David Mohaisen,"Simple Tricks, Big Threats: How Simple Perturbations Evade ML-based Ethereum Phishing Detection". In submission to Elsevier Computer Networks, **COMNET**, 2024.

6. Ahod Alghuried, Abdulaziz Alghamdi, Ali Alkinoon, Soohyeon Choi and David Mohaisen, "Fishing for Phishers: Learning-Based Phishing Detection in Ethereum Transactions". In submission to The 40th ACM/SIGAPP Symposium On Applied Computing, **ACM SAC 2025**, Sicily, Italy, March 31 - April 4, 2025.

7. Ali Alkinoon, Ahod Alghuried, Abdulaziz Alghamdi, Soohyeon Choi and David Mohaisen, "Pandora's Box Reopened: New Insights into Android Permissions". In submission to The 40th ACM/SIGAPP Symposium On Applied Computing, **ACM SAC 2025**, Sicily, Italy, March 31 - April 4, 2025.

8. Soohyeon Choi, Ahod Alghuried, Ali Alkinoon, Abdulaziz Alghamdi and David Mohaisen, "On Attributing ChatGPT-Transformed Synthetic Code". In submission to The 40th ACM/SIGAPP Symposium On Applied Computing, **ACM SAC 2025**, Sicily, Italy, March 31 - April 4, 2025.

9. Ali Alkinoon, Abdulaziz Alghamdi, Ahod Alghuried and David Mohaisen, "Troid: Temporal and Cross-Sectional Android Dataset and Its Security Applications". In submission to The 40th ACM/SIGAPP Symposium On Applied Computing, Sicily, **ACM SAC 2025**, Italy, March 31 - April 4, 2025.

10. Abdulaziz Alghamdi, Ali Alkinoon, Ahod Alghuried, Soohyeon Choi and David Mohaisen, "Unified API Call-based Detection of Android and IoT Malware". In submission to The 40th ACM/SIGAPP Symposium On Applied Computing, **ACM SAC 2025**, Sicily, Italy, March 31 - April 4, 2025.

11. Ahod Alghuried, and David Mohaisen, "Phishing in Wonderland: A Systematic Evaluation of the Learning-Based Ethereum Phishing Transactions Detection and Pitfalls". To be submitted to IEEE Security and Privacy Symposium, **IEEE S&P**, 2025.

    - Rejected from NDSS 2025 with weak split scores (2x weak accept/2x weak reject).

Contents

Abstract

Phishing attacks continue to pose a significant threat to the Ethereum ecosystem, accounting for a major share of Ethereum-related cybercrimes. To enhance the detection of such fraudulent transactions, this dissertation develops a comprehensive framework for machine learning-based phishing detection in Ethereum transactions. The framework addresses critical aspects such as feature selection, class imbalance, model robustness, and the vulnerability of detection models to adversarial attacks. By systematically evaluating these key practices, this work contributes to the development of more effective detection methods.

The first part of the dissertation assesses the current state of phishing detection methods, identifying gaps in feature selection, dataset composition, and model optimization. We propose a systematic framework that evaluates these factors, providing a foundation for improving the overall performance and reliability of detection models.

The second part explores the vulnerability of machine learning models, including Random Forest, Decision Tree, and K-Nearest Neighbors, to single-feature adversarial attacks. Through extensive experimentation, we analyze the impact of various adversarial strategies on model performance and uncover alarming weaknesses in existing models. However, the varied effects of these attacks across different algorithms present opportunities for mitigation through adversarial training and improved feature selection.

Finally, the dissertation investigates how phishing detection models generalize across datasets, focusing on the role of preprocessing techniques such as feature engineering and class balancing. Our findings show that optimizing these techniques enhances model accuracy and robustness, making detection methods more adaptable to evolving threats.

Overall, this work presents a comprehensive framework that addresses the critical elements of phishing detection in Ethereum transactions, offering valuable insights for the development of more robust and generalizable machine learning-based security models. The proposed framework has broad implications for improving blockchain security and advancing the field of phishing detection.

1 Introduction

1.1 Motivation

Ethereum, like other blockchain technologies, has transformed decentralized transactions by providing a secure, transparent, and immutable ledger [45, 61, 77, 85]. However, this same technology has also made Ethereum an attractive target for cybercriminals, with phishing scams emerging as one of the most significant threats. Since 2017, phishing scams have accounted for more than 50% of Ethereum-related cybercrimes, leading to substantial financial losses. Incidents such as the Bee token Initial Coin Offering (ICO) scam of 2018, which resulted in one million USD in losses within 25 hours, and the 2022 Uniswap Labs attack, where attackers stole over eight million USD, demonstrate the severe impact of these scams [33, 121, 9]. By 2024, more than two billion USD had been stolen from Ethereum users through phishing-related activities [24, 64]. These cases illustrate the urgent need for improved phishing detection mechanisms to protect user assets and bolster confidence in Ethereum's security [126, 7].

Phishing scams exploit the complexities of blockchain transactions, using sophisticated methods to blur the line between legitimate and malicious activities [16, 89, 128]. Addressing these evolving threats requires innovative detection methods that incorporate both traditional transaction analysis and advanced machine learning techniques [60, 108]. While classification algorithms such as Random Forest (RF), Decision Tree (DT), and K-Nearest Neighbors (KNN) have shown success in detecting phishing, concerns remain about their robustness, especially against adversarial attacks [6]. These attacks, where small perturbations are introduced into input data to deceive the models, significantly reduce the reliability of these systems [42]. In particular, methods like the Fast Gradient Sign Method (FGSM) have demonstrated how these models can be misled into incorrect classifications [101].

The vulnerability of widely used models such as KNN, RF, and DT to adversarial attacks highlights the need for stronger defense mechanisms that ensure resilience against such manipulations [75]. Developing robust detection systems capable of withstanding these adversarial inputs is crucial for maintaining the integrity of Ethereum's phishing detection efforts [5].

Current approaches to phishing detection have primarily relied on explicit transactional features, such as transaction value, gas usage, and timestamps. While these features are useful, they fail to capture the broader relational dynamics in blockchain transactions. Recent research has shown that graph-based models, which treat Ethereum transactions as a network of interacting addresses, provide a more comprehensive approach to detecting sophisticated phishing schemes [31, 116]. However, many existing studies overlook critical

Table 1: Overview of phishing detection approaches on Ethereum network. Metrics: Accuracy (A), Precision (P), Recall (R), and F1 Score (F1)

§	Task	Target	Technique	Metric
§ 3	Evaluation of phishing detection models	Robustness and effectiveness of models	DT, KNN, RF	A, P, R, F1
§ 4	Adversarial robustness	Susceptibility to feature perturbation	DT, KNN, RF	A, P, R, F1
§ 5	Phishing detection	Effectiveness of explicit and implicit features	GCN	A, P, R, F1

challenges such as class imbalance, where phishing transactions are vastly outnumbered by benign ones. This imbalance limits the generalization ability of phishing detection models, making it crucial to develop more comprehensive solutions that integrate both explicit features and graph-based models [40, 128].

1.2 Statement of Research and Contributions

In this dissertation, we propose three key tasks that tackle the challenges of phishing detection on the Ethereum network. Each task focuses on a different aspect, from systematically evaluating machine learning-based phishing detection models to improving their robustness and accuracy. Table 1 summarizes the tasks, target outputs, techniques, and evaluation metrics for each. We will elaborate on each task in the following sections.

Evaluation the Learning-Based Ethereum Phishing Transactions Detection and Pitfalls(§ 3). Ethereum's rapid growth has introduced an increasing threat of phishing scams that exploit the unique complexities of cryptocurrency transactions. Public efforts to detect phishing have employed machine learning models, yet systematic evaluations of these approaches remain scarce. In this work, we present a comprehensive evaluation framework that assesses the robustness and effectiveness of phishing detection models. Our analysis covers key elements such as feature selection, dataset composition, and model robustness under real-world conditions.

We curate two well-known datasets, Eth-PSD and CTD, for Ethereum phishing detection and use decision tree and k-nearest neighbors (KNN) models to empirically validate the results. Our evaluation reveals that many models exhibit overfitting due to artificial dataset balancing and improper feature selection. We refine these models using recursive feature elimination (RFE) and sequential forward selection (SFS), achieving significant improvements in their predictive power. Our experiments demonstrate that a leaner feature set, focusing on key attributes such as transaction values, timestamps, and block height, can enhance detection accuracy without sacrificing robustness. Evaluations using metrics such as accuracy, precision, recall, and F1-score further verify the models' improved sustainability and performance across different phishing detection scenarios.

Simple Perturbations Subvert Ethereum Phishing Transactions Detection: An Empirical Analysis(§ 4). Machine learning models, especially in the context of phishing detection, are often vulnerable to simple adversarial attacks that can significantly degrade their performance. Ethereum's rise as a platform for decentralized transactions has made it a target for such attacks. In this work, we present an empirical evaluation of Random Forest (RF), Decision Tree (DT), and K-Nearest Neighbors (KNN) models to understand their susceptibility to adversarial manipulations. Through our experimentation, we introduce perturbations to individual transaction features and measure their impact on classification performance, particularly for phishing detection.

We curate two Ethereum-based datasets, focusing on binary classification for phishing and multi-class classification for other cyber threats, including scams and fake ICOs. Our findings reveal that even minor modifications to features such as timestamps and transaction values cause significant drops in model accuracy. For instance, the RF model exhibited a decrease in accuracy from 98% to 84% under timestamp perturbation Our work also highlights the limitations of simple defenses like random perturbations, underscoring the need for more advanced adversarial training methods. We evaluate these models using metrics such as accuracy, precision, recall, and F1-score, demonstrating the inconsistency in model resilience and the critical need for robust mitigation strategies.

Learning-Based Phishing Detection in Ethereum Transactions (§ 5). This study employs a multi-stage pipeline to detect phishing scams on the Ethereum blockchain, focusing on evaluating the effectiveness of two distinct types of features: explicit transactional features and implicit graph-based features. Ethereum transaction data is collected from the Etherscan API, where phishing and benign transactions are labeled to create the dataset.

The pipeline begins by separately evaluating explicit features, such as transaction value, gas usage, and timestamps extracted from raw Ethereum transactions. Next, the study assesses implicit graph-based features, which capture the broader relational dynamics between Ethereum addresses in the transaction network. The Graph Convolutional Network (GCN) model is used to detect phishing activities by leveraging the graph structure, where nodes represent addresses and edges represent transaction relationships.

In the GCN model, the node feature matrix consists of either explicit transactional attributes or implicit relational information. The GCN layers propagate information across the graph, enabling the model to learn complex patterns in the Ethereum network relevant to phishing detection. Preprocessing steps include normalization of features using Min-Max scaling to ensure all features contribute equally during model training, and class imbalance is addressed by employing a weighted loss function to emphasize phishing transactions.

This methodology allows for a comprehensive evaluation of phishing detection using both explicit and implicit features, offering insights into how each type of feature contributes to

the effectiveness of the GCN model in identifying phishing transactions.

2 Related Work

The security and privacy of blockchains in general and cryptocurrencies in particular have been an active research area with many outstanding results highlighting the vulnerabilities and defenses [15]. Phishing detection on the Ethereum network in particular has attracted significant research interest due to the increasing vulnerability of blockchain platforms to cyber threats. Researchers have explored various approaches, including transactional features, adversarial robustness, and graph-based models, to detect and mitigate phishing scams. This section reviews key contributions from these areas.

2.1 Transactional Features

A large body of research on phishing detection has relied on explicit transactional features, such as transaction values, gas consumption, timestamps, and block numbers. These features are directly extracted from blockchain transactions and provide early indicators of phishing activities. However, while these approaches are effective for identifying straightforward scams, they often struggle to capture more sophisticated relational dynamics in phishing attacks.

Kabla *et al.* [50] employed transactional features like blockHeight and timeStamp to distinguish phishing accounts from legitimate ones, achieving high accuracy (F1 score of 0.98) through machine learning models such as KNN and decision trees. Wen *et al.* [113] focused on temporal patterns within transactional data, using neural networks to identify phishing behaviors based on subtle timing patterns. Similarly, Lin *et al.* [64]. refined phishing detection by integrating features such as gas limits and transaction counts, although their model struggled with more complex attack structures.

While these methods are highly effective for detecting simple phishing activities, their primary limitation is their inability to detect complex, multi-node attacks that involve advanced techniques to obscure malicious behaviors. Consequently, research has shifted toward more sophisticated approaches that address these shortcomings.

2.2 Adversarial Attacks and Model Robustness

In addition to transactional features, phishing detection models are vulnerable to adversarial attacks, where small modifications to transaction data can cause machine learning models to misclassify legitimate and phishing transactions. Recent research has demonstrated the susceptibility of deep learning models to such adversarial perturbations, prompting a need for more robust detection methods.

Narodytska *et al.* [71] first demonstrated the vulnerability of deep learning models to adversarial perturbations, showing how minor changes to input data could significantly impact classification outcomes [71]. Building on this, Cartella *et al.* [28] adapted adversarial attack techniques to fraud detection systems, achieving notable success in subverting models trained on imbalanced datasets. Similarly, Bhagoji *et al.* [22] introduced black-box attacks that exploited subtle modifications in transactional features like timestamps and transaction values to evade detection by machine learning models.

Generative Adversarial Networks (GANs) have emerged as an effective tool for generating adversarial examples and handling data perturbations in cryptocurrency networks. Fidalgo *et al.* [36] applied GANs to Bitcoin datasets to address class imbalance and improve phishing detection accuracy by generating synthetic adversarial data. Agarwal *et al.* [10] utilized Conditional GANs (CTGAN) to generate adversarial data for Ethereum transactions, improving model robustness in detecting phishing activities. Rabieinejad *et al.* [80] combined both CTGAN and Wasserstein GANs (WGAN) to augment Ethereum transaction datasets, enhancing the detection capabilities of phishing models in imbalanced environments.

These studies highlight the need for improved defensive strategies to mitigate adversarial attacks. The research into simple perturbations—such as manipulating timestamps and transaction values—demonstrates the susceptibility of phishing detection models to even minor alterations in input features. Addressing these vulnerabilities requires stronger feature selection techniques, adversarial training, and robust data augmentation methods to ensure that models remain resilient in adversarial environments.

2.3 Graph-Based Features

In contrast to transactional feature-based methods, graph-based approaches focus on capturing the relational dynamics inherent in blockchain networks. These methods, often powered by Graph Neural Networks (GNNs), analyze the network structure between blockchain addresses, enabling the detection of more sophisticated phishing schemes.

Zhou *et al.* [128] introduced the Edge-Featured Graph Attention Network (EGAT), which uses both node and edge features (such as transaction values, gas usage, and timestamps) to model the relationships between addresses and detect phishing activity. This approach successfully identified hidden patterns in Ethereum's transaction network that were missed by transactional models. Li *et al.* [61] proposed the Transaction Graph Contrast Network (TGC), which leverages contrastive learning to improve phishing detection by learning robust representations of Ethereum addresses within transaction subgraphs. This method effectively captures dynamic behaviors across large transaction networks.

Graph-based models offer distinct advantages over purely transactional methods by revealing the complex interactions between blockchain addresses, especially in large, dynamic

networks. However, their reliance on implicit features may sometimes overlook key transactional behaviors that are also indicative of phishing activity. This balance between transactional and graph-based features remains a critical area of exploration in phishing detection.

3 A Framework for Evaluating the ML-Based Ethereum Phishing Transactions Detection

3.1 Summary of Completed Work

This study explores the challenges of phishing detection in the Ethereum, focusing on common practices in machine learning models used to identify phishing transactions. Phishing has become a predominant security concern, significantly affecting the safety of trading within Ethereum. This research systematically examines the data processing, feature selection, and algorithmic techniques employed across multiple studies on phishing detection.

The work categorizes and evaluates the diverse methods of feature selection, balancing datasets, and robustness of features to adversarial manipulation. Key findings reveal several issues, including overemphasis on large feature sets, manipulation vulnerabilities in commonly used transaction features (e.g., time, number of neighbors, transaction direction), and the potential overfitting resulting from synthetic data generation in phishing-to-benign balancing. Moreover, the study presents a comprehensive assessment of model optimization, showing that streamlined models with fewer features could achieve comparable performance to more complex ones.

By contrasting prior works, this study establishes a clearer understanding of how the methods perform under various conditions, such as manipulated datasets or adversarial settings, and suggests improvements in feature robustness, model simplification, and sustainable detection methods for future studies.

3.2 Introduction

Machine learning has found many applications in the domain of computer and network security, particularly related to the automation processes of understanding malicious software [68, 14, 67, 94], network traces [106, 90, 30], and even behaviors [76, 107, 1]. Among the many security concerns related to Ethereum [58, 46, 119, 32, 93, 97, 59, 109], phishing scams stand out [64, 62, 61, 29, 103], constituting over 50% of Ethereum-related cybercrimes since 2017 and emerging as a significant threat to the trading security of Ethereum [47] where machine learning has a significant potential. The Bee token Initial Coin Offering (ICO) scam of 2018 resulted in a loss of one million USD within 25 hours [33]. The phishing attack on Uniswap Labs users in 2022 costed over eight million USD. Earlier in 2024, reports alleged finding 91 wallets that amassed more than 2 billion USD from illicit activities, including scams, on Ethereum [121]. All these incidents underscore the urgent need for enhanced security measures to protect both assets and confidence in the cryptocurrency [98, 8].

Phishing scams have evolved to exploit cryptocurrencies' unique complexities, posing significant security threats and challenging the foundation of trust and integrity that cryptocurrencies seek to establish [126]. Financial losses from successful attacks also undermine confidence in blockchain technology as a secure digital transaction platform. The persistent threat of such sophisticated cybercrimes casts a shadow over the trust in digital currencies [33, 50, 124]. This evolving threat has prompted a surge in research efforts to fortify cryptocurrencies against these sophisticated cybercrimes.

Numerous research initiatives combine machine learning techniques with cryptocurrency analyses to combat phishing attacks. Since transactions form a graph, many of these studies detect phishing employing graph-based analysis [31, 61, 108]. Analyzing blockchain activity alone aims to develop machine learning methods to identify malicious activities. For example, one notable approach, by Chen *et al.*[31], employed convolutional networks and auto-encoders alongside transaction graphs to identify phishing accounts. Another approach, by Lou *et al.*[66], used a Convolutional Neural Network (CNN) to improve the precision and recall of detection. Wu *et al.*[116] implemented a one-class learning for deciphering structural relationships in Ethereum's transaction network using Graph Convolutional Networks (GCN) and autoencoders for high precision and recall for transaction classification.

Nevertheless, these works fail to systematically understand the circumstances under which those approaches deliver outstanding results. In particular, proper implementation, analysis, and understanding of those systems are lacking as they often make ad hoc assumptions about how phishing works in reality in a way that appeases the implementation of the machine algorithms to deliver high effectiveness. Such assumptions may not be necessarily grounded in the reality of the cryptocurrency.

This work investigates the common practices in machine learning-based phishing detection in Ethereum, focusing on its operation, data composition, feature selection, robustness examination, and model optimization. This study is driven by the pressing need to bolster phishing detection mechanisms [34, 21]. We categorize, map, and assess features, algorithms, and techniques, evaluate their strengths and shortcomings, and highlight pitfalls in various detection methods in following standard and sustainable practices.

We show the disarray in this research space. Feature selection is often ignored, and dataset composition is often done for artificial reasons to balance label distribution and achieve superior performance. Features are prone to manipulation, and robustness analysis is often ignored, questioning the sustainability of the achieved performance. Models are bloated without justification, and we show more streamlined models with fewer features could achieve the same performance.

Contributions. In this paper, we make the following contributions. (1) We build a systematic evaluation framework that unveils the underlying practices in learning-based phishing

detection in Ethereum, including the common practices with feature selection, dataset composition, feature robustness examination, and feature selection and model optimization. We analyze and categorize a comprehensive list of recent studies in this space and highlight where they fail to implement good and sustainable practices. (2) We empirically assess several studies from the literature using our analytical framework, addressing various research questions by quantifying concerns highlighted within the framework. Additionally, we examine practices surrounding these studies, highlighting the risks of overlooking fundamental tests and assessments.

3.3 Research Questions and Methodology

Research Questions. This effort is built around five questions to understand the robustness, generalization, reproducibility, and comparison of various works on Ethereum phishing transaction detection. We highlight those questions and the need to address them.

RQ1. Are the prior works in the literature on phishing detection doing the proper feature selection? This question stems from the observation that the features designed for detection in the prior work [31, 112, 127, 33, 61, 60, 113, 38] lack thorough consideration and justification. Frequently, these features are devised hastily and lack proper statistical analysis. In other cases, those features result from a black-box learning module that is difficult to comprehend.

RQ2. What is the impact of the representation of phishing as benign on the performance of the phishing detection algorithms? This question stems from the observation that, in particular works, an arbitrary number of phishing transactions is assumed in the datasets [33, 50, 33, 74, 60, 61, 64, 113]. To achieve a balanced ratio between benign and phishing samples, these studies frequently introduce random phishing transactions into their datasets without considering the impact on other features utilized in the classification process, regardless of whether they are dependent or independent.

RQ3. How do different algorithms compare to one another? This question arises from observing that various approaches [116, 64, 74, 113, 31, 112, 126, 33, 38] employing distinct features and feature groups, and different algorithms, are typically conducted independently and not compared against each other in terms of their performance using comparable evaluation metrics and consistent evaluation settings. An additional aspect of answering this question involves determining the reproducibility of the findings from previous studies on cryptocurrency phishing detection.

RQ4. What impact does dataset preprocessing have on the overall effectiveness of the detection algorithm's generalization? This question is driven by the observation that many studies incorporate a preprocessing phase, e.g., reducing a larger network to a

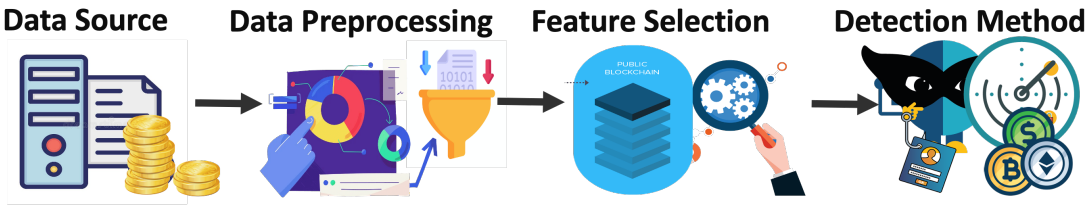

Figure 1: Detection pipeline in cryptocurrency transactions.

smaller one to make the analysis and detection algorithm implementation more feasible computationally [61, 60, 116, 64, 31]. However, such preprocessing excludes certain underlying transaction networks while possibly exaggerating others that may not be as prominent in the original network. Therefore, comprehending the influence of preprocessing steps on the outcome of the detection algorithm is essential for understanding its generalization.

RQ5. How do different features and feature groups compare for robustness to manipulation? An effective detection and sustainable algorithm should depend on inherent features and not be easily manipulated by adversaries to evade detection. However, it is evident that the feature sets in these detection schemes (e.g., [116, 64, 127, 113, 74, 61]) are non-uniform and differ in this aspect. Conducting a systematic and comprehensive analysis of these features is crucial.

3.4 Methodology

Our methodology is built around the research questions highlighted in section 3.3. The high-level depiction of the methodology is shown in the pipeline in Fig. 1. Our pipeline comprises the following steps: data collection, data preprocessing, feature selection, and detection. We will review our pipeline to address the research questions below. We limit ourselves in this discussion to the high-level and general techniques description since the individual techniques are studied in the subsequent sections when discussing specific works.

Data Source/Collection. Our pipeline begins by gathering data associated with the transactions from a data source. Such data includes full transaction information confirmed in blocks and cryptocurrency-specific application features and details. For instance, this encompasses block numbers, time, sender and receiver addresses, transaction values, and gas fees. This comprehensive dataset forms the foundation for our subsequent preprocessing and analysis steps, facilitating a thorough examination of the transactional ecosystem to identify and classify phishing attempts effectively.

Data Preprocessing. Different approaches utilize different preprocessing techniques, which we will discuss further.

Normalization. This step involves converting transactions into a usable format for classification, e.g., converting hexadecimal to integers and extracting transaction relationships. For instance, in [31], a graph is derived where nodes are the addresses and the edges are the transaction values and time.

Scaling. Scaling is a vital step in data preprocessing to adjust the range of data features to a standard scale, ensuring that each feature contributes equally to the analysis [50]. Given the significant variation in transaction data magnitude, this step is crucial to avoid potential biases in the results. Commonly employed approaches include Min-Max scaling and Z-score normalization. Min-Max scaling adjusts the data to fit within a specific range, e.g., 0 to 1 or -1 to 1, proving beneficial when dealing with features to be bounded within a specific range.

Feature Selection. Identifying the most relevant features within a dataset is critical for developing robust detection models [52]. This involves selecting features that significantly affect the accuracy of phishing detection, which may include the amount and frequency of transactions, distinct transaction patterns, and deviations in transactional behaviors [60, 113]. Considerations might include account age, network breadth, and other behavioral indicators [31, 50].

General Selection. Feature selection includes a voting strategy and various ranking techniques. Potential features are evaluated, followed by a voting system that selects features based on a predetermined threshold.

Correlation-based Selection. This approach is used to identify the most relevant and impactful features for inclusion in the model [50] based on their pair-wise correlations: features with high correlation are excluded from the candidate features.

Engineered Featues. Another technique employs the graph-based method that analyzes the transaction network through a graph perspective, denoted as $G(V, E)$. This approach zeroes in on a particular subgraph, $G_s(V_s, E_s)$, where the analysis includes node characteristics and the adjacency matrix. Attention is paid to aspects such as transaction nodes' in-degree and out-degree and the patterns and frequencies of transactions [31, 64]. This method provides detailed insight into the network structure, allowing a nuanced understanding of transaction behaviors and potential links to phishing activities.

Detection Methods. The stage involves identifying phishing activities by employing several machine-learning models to analyze and predict the likelihood of suspicious or fraudulent transactions. By training on labeled data, these models are adept at discerning patterns, enabling the distinction between normal and malicious activities.

Supervised Learning. Supervised learning approaches train algorithms on labeled data where the outcome is known, allowing models to learn from past data and apply this learning to new, unseen transactions. We deploy classifiers such as LightGBM, Decision Trees (DT),

Table 2: The literature on phishing accounts detection using various features, features groups, and algorithms across various evaluation metrics.

Work	Group	Features	Algorithms	Performance	
				F1	AUC
Chen *et al.* [33]	Transaction	T, A	LightGBM	0.80	0.81
Chen *et al.* [31]	Transaction	ID and OD, D, IS, OS, S, N	LightGBM	0.16	0.56
Palaio. *et al.* [74]	Transaction	BN, GF, NT, ST	SVM, CNN, XGBoost	0.85	–
Wen *et al.* [112]	Transaction	T, A, ID, OD, N, from, to	SVM, KNN, AdaBoost	0.94	0.92
Kabla *et al.* [50]	Transaction	T, BN, from, input	KNN, DT	0.97	0.97
Li *et al.* [60]	Transaction	ID, OD, D, A, T, NT	LightGBM	0.81	0.92
Wu *et al.* [116]	Transaction	T, A	SVM	0.90	–
Li *et al.* [61]	Transaction	ID, OD, D, A, T, NT	XGBoost	0.92	–
Lin *et al.* [64]	Transaction	BN, A, GL, GF	Neural network	0.82	–
Wen *et al.* [113]	Transaction	T, A, TD, GF, B, NT	Neural Network	0.97	–
Zhou *et al.* [127]	Transaction	T, A, GF, ID, OD	EGAT	0.97	–
Chen *et al.* [31]	Behavioral	ID, OD, D, IS, OS, S, N	LightGBM	0.16	0.56
Li *et al.* [61]	Behavioral	ID, OD, D	TGC	0.92	–
Li *et al.* [60]	Behavioral	ID, OD, D	TTAGN	0.81	0.92
Wen *et al.* [112]	Behavioral	ID, OD, N	SVM, KNN, AdaBoost	0.94	0.92
Zhou *et al.* [127]	Behavioral	ID, OD	EGAT	0.97	–
Dong *et al.* [38]	Content	ID, OD, NT	SVM, LR	–	0.97
Wen *et al.* [113]	Content	T, A, TD, GF, B, NT	CNN	0.97	–

(1) Features: Time (T), amount (A), in-degree (ID), out-degree (OD), degree (D), in-strength (IS), out-strength (OS), strength (S), neighbors (N), block number (BN), gas fee (GF), number of transactions (NT), successful transactions (ST), gas limit (GL), balance (B), transfer direction (TD). (2) Metrics: F1 score and area under the curve (AUC).

and K-nearest neighbors (KNN). More details on these algorithms are in the appendix.

Evaluation and Validation. We use various metrics for evaluation: accuracy, precision, recall, and F1 score. ① *Accuracy.* The accuracy denotes the ratio of correctly predicted observations to the total observations, offering an insight into the model's differentiation between phishing and legitimate transactions. ② *Precision.* The precision is the ratio of correctly predicted positive observations to the total predicted positive observations. ③ *Recall.* The recall, or sensitivity, quantifies the proportion of correct positives correctly identified. ④ *F1 Score.* The F1 score, also known as the harmonic mean of precision and recall, verifies the model's accuracy by factoring in the false positives and false negatives. ⑤ *The Area Under the Curve (AUC)* AUC is indispensable, especially in situations characterized by class imbalance, a frequent occurrence in phishing detection. The AUC metric assesses the model's ability to differentiate between phishing and legitimate transactions, with a perfect model achieving an AUC of 1. An AUC of 0.5 suggests a performance no better than random chance.

3.5 Analytical Evaluation

To answer the research questions in section 3.3, we pursue two directions: a theoretical analysis based on a deep understanding of the context in which the phishing detection schemes are used and an experimental work based on the actual implementation of some of those schemes to support our theoretical analysis. The theoretical analysis follows the same order in section 3.3 to answer each research question.

3.5.1 Feature Selection Techniques

The prior works have employed various techniques to select features for building the learning model. Those features are shown in Table 2 for the different techniques.

3.5.2 Feature Selection

Our primary questions are: (1) How are features selected? (2) Is there a standard for their selection? (3) Does the technique for selecting those features directly address their importance? (4) Are they all essential for the functioning of these schemes? To contextualize our discussion, we present the different feature selection techniques in the works discussed in Table 2. Then, based on the discussion, we answer the above questions.

Trans2vec. Introduced by Wu *et al.* [116], Trans2vec stands out as a specialized network embedding technique designed for transaction networks like Ethereum. It is notable for its feature extraction method, embedding transaction-specific attributes into node representations focusing on transaction amount and time. By constructing a multidimensional feature space, Trans2vec represents each node within the transaction network, where transaction amount and timestamp are not merely additional data points but are intricately incorporated into each node's representation structure.

Cascading Method. Proposed by Chen *et al.* [33], the cascading method initially analyzes individual accounts, focusing on transaction details such as frequency and amount as preliminary features obtained from a graph structure of the Ethereum transactions and associated accounts. It then gradually expands the features to consider the account's direct connections (first-order connections), extending further to second-order connections and beyond. This layered feature set reflects individual account behavior and broader interaction patterns within the network, which is vital for detecting phishing activity.

Correlation Analysis. Employed by Palaiokrassas *et al.*[74], correlation analysis is a pivotal technique for identifying linear relationships between features, thereby detecting redundancy among variables. The high correlation between features may suggest interdependence, indicating potential redundancy for machine learning models, which is crucial in optimizing

model performance. Methods based on the correlation analysis omit features with high correlation.

Feature Engineering. Wen *et al.*[112] takes a targeted approach to Ethereum transaction analysis, distinguishing between Account Features (AFs) and Network Features (NFs). To uncover unusual patterns, AFs focus on individual behaviors, such as account balance, transaction types, and volumes. NFs map an account's network interactions, examining its connectivity and transaction flows with other entities. These features are selected manually without significance testing, pointing towards a tailored but less empirical approach to feature engineering.

Sequential Forward Selection (SFS). SFS is characterized by its incremental selection of features based on their contribution to predictive power. Initialized with no features, SFS adds one feature at a time, selected according to a specific criterion to optimize model performance [104]. This addition continues until further inclusion of features yields no significant improvement.

Recursive Feature Elimination (RFE). RFE operates by iteratively removing the least impactful features based on their contribution to the performance and evaluating the remaining set [48]. This process helps understand which features are least helpful in predicting phishing and optimizes the efficiency by focusing on a smaller, more impactful feature set [70].

Voting-Based. Introduced by Kabla *et al.*[50], the voting-based feature selection method applies three distinct ranking algorithms to evaluate and select features. This method incorporates a holistic view by assessing each feature's correlation with the outcome, its importance within a specific classifier, and the collective effectiveness of feature sets through a majority vote. Such an approach aims to refine the feature set for phishing scam detection models, emphasizing consensus among different evaluative methods.

TTAGN. Li *et al.*[60] present Temporal Transaction Aggregation Graph Network (TTAGN), a method leveraging Temporal Edge Representation to capture Ethereum transaction dynamics. Using Long Short-Term Memory (LSTM), it analyzes temporal interactions between nodes, enhancing edge representations. An Edge2Node module aggregates these representations with attention mechanisms. Additionally, a Structural Enhancement Module equipped with a GCN analyzes the graph topology, thereby enriching node profiles for more comprehensive feature representation.

TGC. Due to Li *et al.*[61], the Transaction Graph Contrast Network (TGC) constructs a node's ego network as a starting point for comparative pair creation and subgraph training. Through Random Walk with Restart (RWR) sampling, diverse subgraphs are generated for contrastive learning. The feature extraction process is split into node-level, distinguishing a node against its neighbors, and context-level, differentiating the structural patterns of

Table 3: A comparison of the feature selection algorithms.

Work	*Feature Group*	*Selection*	*Algorithm*	*Selection Type*	*Interpretation*	*Computation*	*Performance*
Chen *et al.* [33]	Transaction	✓	Cascading	Hybrid	◐	◐	→
Chen *et al.* [31]	Transaction	–	–	–	–	–	–
Palaio. *et al.* [74]	Transaction	✓	Correlation	Traditional	●	○	↓
Wen *et al.* [112]	Transaction	✓	FE/RFE	Traditional	●	○	↓
Kabla *et al.* [50]	Transaction	✓	Voting	Traditional	●	○	↓
Li *et al.* [60]	Transaction	✓	TTAGN	Learning	○	●	↑
Wu *et al.* [116]	Transaction	✓	Trans2vec	Learning	○	●	↑
Li *et al.* [61]	Transaction	✓	TGC	Learning	○	●	↑
Lin *et al.* [64]	Transaction	✓	Phish2vec	Learning	○	●	↑
Wen *et al.* [113]	Transaction	✓	NN	Learning	○	●	↑
Zhou *et al.* [127]	Transaction	–	–	–	–	–	–
Chen *et al.* [31]	Behavioral	–	–	–	–	–	–
Li *et al.* [61]	Behavioral	✓	TGC	Learning	○	●	↑
Li *et al.* [60]	Behavioral	✓	TTAGN	Learning	○	●	↑
Wen *et al.* [112]	Behavioral	✓	FE	Traditional	●	○	↓
Zhou *et al.* [127]	Behavioral	–	–	–	–	–	–
Dong *et al.* [38]	Content	✓	NN	Learning	○	●	↑
Wen *et al.* [113]	Content	✓	NN	Learning	○	●	↑

(1) Abbreviations: Feature engineering (FE), neural networks (NN).
(2) Metrics: Interpretability, computation, and performance.
(3) Values: unsatisfied (○), satisfied (●), partially satisfied (◐), better performance (↑), worse performance (↓), average performance (→), and no basis (–).

phishing versus normal addresses. The contrastive approach leverages Graph Neural Network (GNN) encoders to unveil distinct transactional and structural patterns.

Phish2vec. Due to Lin *et al.* [64], Phish2vec is a feature extraction technique that accounts for temporal dynamics. The process starts with the Temporal-based Sequences Generator (TSG), which utilizes a random walk technique that incorporates both amount- and time-based factors to generate sequences that accurately portray the transactional values and their temporal sequence within the network. Moreover, the Heterogeneous-based Sequences Generator (HSG) is introduced to handle various account types, such as Contract Accounts (CAs) and Externally Owned Accounts (EOAs), by generating sequences that capture the interactions and operational patterns among these accounts. These sequences are subsequently analyzed using a word2vec model to extract low-dimensional vector representations for each account, capturing the network's contextual and structural nuances.

Neural Networks. Wen *et al.* [113] leverage a composite of neural networks for data representation, anchoring on a Back Propagation (BP) neural network for processing transfer and state features. This layer encodes relational patterns into vectors, which are further scrutinized by Fully Convolutional Networks (FCN) and LSTM units to discern and prior-

itize transaction features. Dong *et al.* [38] integrates GNNs, with a spotlight on node2vec, alongside feature derivation methods for a nuanced extraction of blockchain transaction data features. This approach, focusing on structural and transactional data, e.g., time and values, aims to distill key indicators of transaction behaviors for enhanced feature selection.

3.5.3 Analysis and Results

The exploration of feature selection techniques reveals a lack of a uniform standard across different studies. Each employs its unique method, often deviating from conventional approaches, making it challenging to compare these methods directly regarding their effectiveness in feature selection. This diversity raises critical questions about whether observed discrepancies in model accuracy stem from the learning techniques themselves or from the specific features considered, which are often not transparently disclosed. Several feature representation and selection techniques are utilized. Some indirectly limit their explainability by trading the computational features for performance or vice versa, as shown in the last three columns in Table 3.

Methods. The feature representation and selection methods can be categorized into three groups, which we review below.

Learning-based Techniques. These techniques begin with a rudimentary representation based on an initial concept of nodes and edges and gradually evolve into the final representation using a learning component, e.g., a neural network. Among the discussed methods thus far, the category includes Trans2vec, TTAGN, TGC, Phish2vec, and GNN-based techniques.

Traditional Techniques. These methods start with a fixed set of features from transaction, network, or content-related data. They iteratively refine this selection by choosing features that significantly impact the detection algorithm's performance, as evaluated by their contribution to the loss function. This category includes correlation, voting, and feature engineering techniques among those reviewed.

Hybrid Techniques. These methods often demonstrate a blend of attributes from both learning-based and traditional techniques. Among the techniques we have explored thus far, the cascading method is the only one that fits this description.

Feature Selection Comparison. The *learning-based techniques* provide feature representations and automate feature selection, reducing the need for human intervention. However, they often lack interpretability since feature selection is indirect. These methods typically involve weighting features in an initial embedding and projecting them into different dimensions through multiple iterations determined by the neural network architecture. This process depends on the specific network used. It may not generalize well to different datasets,

leading to a computationally expensive feature extraction process that needs to be repeated for each dataset or fold.

The *traditional techniques* differ from learning-based ones in ignoring certain features entirely instead of optimizing them through weight adjustments based on the loss function. While this approach generally performs less than automated learning techniques, it provides more interpretable features. Knowing which features are most influential for detection helps us analyze their manipulability and design defenses, which helps us in our experimental evaluation in subsection 3.8. Traditional methods are typically computationally lightweight but require human intervention to understand features, select ones that generalize, and evaluate their suitability for different datasets.

The *hybrid techniques*, predominantly represented by the cascading method, incorporate elements reminiscent of learning-based techniques. However, they refrain from fully learning feature representations and instead cascade the representation by integrating it with adjacent entity features (e.g., nodes and edges). Thus, these techniques can be automated relatively easily while allowing for some interpretation.

Statistical Analysis. Certain studies [31, 127] involve statistical analysis on the detection features, e.g., by calculating the min, max, means, etc., from the distributions. However, such analyses often do not contribute to determining the importance of features, e.g., in cases where the min, max, and mean values fall within a narrow range from each other.

Summary. Table 3 shows a summary of the feature selection algorithms utilized in the various schemes of phishing detection, if any, categorized against their type, interpretability, computational complexity, and performance. We note that 4 out of 18 detection schemes did not use any form of feature selection. In contrast, 9 (50%) used learning-based techniques, 4 used traditional feature selection approaches, and only 1 used hybrid approaches. This makes the learning-based techniques the most widely used category despite their clear disadvantages in the security domain in terms of their computational complexity and limited interpretability.

Answering RQ1. Various techniques are utilized for feature selection in phishing transaction detection. Five of the nine techniques examined employ learning-based methods, often yielding uninterpretable features. In contrast, three techniques focus on feature selection, and one adopts a hybrid approach. While there is no standard protocol for feature selection, most works incorporate it to some extent. However, only a subset of techniques directly addresses feature importance, with interpretations remaining unclear despite emphasizing feature importance across all methods.

Table 4: Overview of various studies categorized by work, year of publication, the number of transaction, number of accounts, original data distribution, method of balancing, modified data distribution, and the corresponding ratios before and after balancing.

Work	Transactions	Accounts	Original Data		Method	Modified Data		Phishing-to-benign Ratio	
			Phishing	Benign		Phishing	Benign	Original	Modified
Chen *et al.* [33]	7,795,044	534,820	1,683	7,793,359	Filter rules	323	534,497	0.0216%	0.0604%
Chen *et al.* [31]	–	2,973,382	1,157	2,972,225	No balancing	1,157	2,972,225	0.0389%	0.0389%
Palaio. *et al.* [74]	54,000,000	550,000	10,000	540,000	SMOTE	81	10,000	1.818%	0.81%
Wen *et al.* [112]	20,667,671	52,380	3,135	49,245	Filter rules	992	4,066	6.366%	22.675%
Kabla *et al.* [50]	–	84,664	5,448	79,216	Over-sampling	38,143	79,216	6.877%	48.15%
Li *et al.* [60]	208,847,461	6,844,050	4,932	6,839,118	Filter rules	4,932	–	0.072%	–
Wu *et al.* [116]	3.8 billion	500 million	1,259	1,259	No balancing	1,259	1,259	100%	100%
Li *et al.* [61]	219,927,673	9,237,535	5,639	9,231,896	Upsampling	5,639	25,000	0.061%	22.556%
Lin *et al.* [64]	22,594,499	4,116,315	5,168	4,111,147	Sliding window	301	4,116,013	0.1257%	0.0073%
Wen *et al.* [113]	739,790	44,709	4,709	40,000	Sliding window	43,125	74,838	11.772%	57.624%
Zhou *et al.* [127]	332,670	3,359	1,659	1,700	No balancing	1,659	1,700	97.59%	97.59%
Dong *et al.* [38]	4,161,444	944,705	1,660	1,700	No balancing	1,660	1,700	97.64%	97.64%

3.6 Phishing-to-Benign Ratio

In phishing detection, distinguishing benign and malicious transactions is crucial, quantified by the phishing-to-benign ratio (Table 4). Our investigation delves into key questions: (1) Is the balance between phishing and benign transactions manipulated in machine learning-based phishing detection? (2) What techniques are employed for balancing the distribution? (3) How does the balancing affect the representation of benign relative to phishing transactions? (4) How does this balancing impact the generalization?

3.6.1 Balancing Phishing Ratios

Phishing refers to malicious activities where attackers seek to deceive users into revealing sensitive information, such as private keys or wallet passwords. In contrast, benign refers to transactions or accounts that are not malicious. The phishing-to-benign ratio measures the prevalence of phishing attempts against benign activities within datasets. Creating balanced training sets for machine learning algorithms is sometimes necessitated by the logic of the underlying machine learning algorithms to ensure high accuracy in phishing detection systems [118].

In real-world applications, biased models can reduce accuracy if the classes are imbalanced, leading to a preference for the overrepresented class [100]. Achieving a balanced ratio enhances the detection models' adaptation to diverse scenarios, reducing overfitting risks [81]. Consequently, maintaining a balanced representation in training datasets is critical for the accuracy and fairness of machine learning models in detecting phishing activities. However, this artificial balancing may not reflect the real-world scenario where the imbalance is intrinsic.

Next, we explore whether these assumptions are maintained in model training and their impact on generalization.

3.6.2 Dataset Balancing Techniques

The diverse approaches outlined in Table 4 use various techniques to balance the phishing and benign ratio to optimize the dataset's composition to meet the learning algorithm's requirements. As such, we delve into the employed balancing methods and pitfalls.

No Balancing. Several studies, such as those by Chen *et al.* [31], Zhou *et al.* [127], Dong *et al.* [38], and Wu *et al.* [116] opted for unbalanced datasets. This choice poses significant challenges for machine learning models in training to detect phishing in the minority class. With phishing examples scarce, models may skew towards recognizing the majority of benign, undermining their ability to identify phishing accurately. Such a scenario perfectly balances preserving the dataset's authenticity and enhancing the model's detection capabilities. Moreover, it represents the real-world setting where the machine learning models are eventually supposed to operate.

Filtering Rules. Data cleaning techniques are applied to eliminate outliers or specific data types, thus improving detection accuracy. Chen *et al.*[33] refined their dataset by filtering out transactions and accounts with significant transaction volumes, substantially lowering their count of phishing addresses. Similarly, Wen *et al.*[112] applied a similar strategy by removing accounts with fewer than four incoming transactions or balances below 5 *ether*. Li *et al.*[60], on the other hand, removed transactions before August 2, 2016, and those from exceptionally active addresses. While aimed at pinpointing phishing activities with greater precision, these strategies come with a notable drawback: the inadvertent removal of legitimate transactions that fall outside standard patterns. This risk could diminish the model's capacity to detect a wider array of phishing behaviors, potentially weakening their effectiveness.

SMOTE. The Synthetic Minority Over-sampling Technique (SMOTE) aims to achieve dataset balance by creating synthetic examples of underrepresented classes, thereby improving predictive model accuracy. Palaiokrassas *et al.*[74] used SMOTE to analyze over 54 million Ethereum transactions, identifying only 81 addresses associated with illicit DeFi protocol activities. While balancing phishing and benign instances for model training, this approach introduces the risk of artificial pattern creation. Such deviations could compromise the model's effectiveness on genuine data, impeding generalization.

Over-sampling. Over-sampling techniques, employed to balance datasets by increasing the representation of underrepresented classes, were utilized with variations by researchers in their studies. Kabla *et al.*[50] duplicated phishing records in datasets, aiming to achieve balance. Similarly, Li *et al.*[61] used upsampling to amplify the presence of phishing addresses in their extensive dataset for a more equitable distribution. Despite effectively augmenting phishing instances, this approach bears the risk of model bias towards these inflated exam-

ples, potentially compromising the distinction between benign and phishing activities due to the likelihood of overfitting.

Sliding Window. The sliding window technique analyzes subsets of data within a larger dataset or stream. It involves moving a fixed-size window over the data and processing or analyzing the data contained within that window at each step. Lin *et al.*[64] introduced a Statistics-Based Sampling (SBS) approach, selecting transactions within specific blocks to distribute phishing activities evenly. Alternatively, Wen *et al.*[113] utilized a sliding window of size 16 to cover overlapping transaction segments, seeking to identify and balance phishing and benign activities by acknowledging temporal patterns. However, this approach might not encompass all phishing behaviors, potentially diminishing the model's efficiency.

Summary. Altering the phishing-to-benign transaction ratio can significantly affect detection algorithms' performance. Introducing synthetic or duplicated phishing transactions for dataset balancing poses a risk of distorting the feature distribution learned by algorithms. Consequently, models might excel with altered training data but struggle to apply their insights to real, untouched data. A common risk is overfitting, where the model overly attunes to the artificial noise from these manipulations rather than discerning genuine phishing patterns. Therefore, while balancing may boost algorithm performance in training, it risks undermining effectiveness with real-world data the algorithm ultimately faces.

Answering RQ2. With the diverse methods used to alter the phishing-to-benign ratio, it is clear that manipulation can significantly compromise detection algorithms' accuracy and real-world applicability. The infusion of synthetic transactions for dataset balancing carries notable repercussions for the algorithms' ability to generalize. While these techniques aim to enhance the representation of phishing transactions, they risk introducing biases and promoting overfitting.

3.6.3 Balancing Analysis And Result

Dataset balancing, as highlighted in section 3.6.2, reveals a landscape marked by diversity. Different studies introduce techniques shaped preset, although *ad hoc*, circumstances. This variety complicates the task of directly comparing the efficacy of these approaches. It raises a critical question: are the observed variances in model performance due to the balancing methods or the complex nature of the data/algorithms? Our work sets the stage for a detailed examination of how each category of balancing technique influences the phishing detection models.

No Balancing. Four studies did not implement any balancing methods: Chen *et al.*[31], Wu *et al.*[116], Zhou *et al.*[127], and Dong *et al.*[38]. Specifically, Chen *et al.*.[31] exhibited notably low phishing ratios at 0.039%, introducing challenges to machine learning models

due to significant class imbalances. Conversely, Zhou *et al.*[127] and Dong *et al.*[38] presented higher ratios, at 97.59% and 97.64%, respectively, indicating a different set of challenges for accurate phishing detection, such as overfitting and a decreased ability to identify legitimate communications effectively. These extremes in dataset composition underscore the pivotal role of balanced data in training machine learning models for phishing detection.

Filter Rules. Applying filter rules, Chen *et al.*[33], Wen *et al.*[112], and Li *et al.*[60] achieved varied outcomes. The strategy adopted by Chen *et al.*[33] led to a reduction in phishing transactions to 323 and benign transactions to 534,497. This adjustment significantly altered the phishing-to-benign transaction ratio, raising it from 0.0216% to 0.0604%. Wen *et al.*[112] decreased the number of phishing transactions to 992 and benign transactions to 4,066, elevating the phishing-to-benign ratio from 6.366% to 22.675%. Such an increase, by approximately 256.19%, underscores deviation from reality. Li *et al. et al.*[60] did not adjust the phishing ratio, which remained consistent at around 0.072%. This steadiness suggests that the filtering had negligible impact on the dataset or that any modifications were not documented. In comparison, the methods employed by Chen *et al.*[33] and Wen *et al.*[112] highlight the impact of filtering rules in enhancing detection.

SMOTE. Utilizing SMOTE, oversampling, and upsampling enables the inflation of minority classes to adjust imbalances. Palaiokrassas *et al.*[74] applied SMOTE to refine the phishing ratio to 0.81%, albeit at the risk of introducing synthetic biases. Conversely, Li *et al.*[61] leveraged upsampling for 219.9 million transactions, elevating the phishing detection potential by increasing its ratio from 0.061% to 22% (more than 36 folds!). Similarly, Kabla *et al.*[50] utilized oversampling to enhance the phishing class's visibility, boosting the ratio from 6.877% to 48.15% (almost seven folds).

These strategies highlight the nuanced roles preprocessing plays in improving phishing detection. Each alters the dataset in a way that optimizes for challenge resolution. Nevertheless, the risk of model overfitting and the ability to faithfully represent real-world scenarios differ among the techniques. SMOTE offers a diverse yet synthetically augmented solution compared to the direct increase by oversampling and upsampling.

Sliding Window. The sliding window technique involves analyzing a subset of data over a fixed period, which shifts progressively over the entire dataset, to capture dynamic changes in data characteristics. Using the sliding window technique, Wen *et al.*[113] and Lin *et al.*[64] observed divergent outcomes in their ratios after implementing their balancing methods. Wen *et al.*[113] witnessed their dataset's phishing to benign ratio increase from 11.77% to 57.62%, equating to more than a 389% surge in phishing instances. Lin *et al.*[64] achieved reduced the phishing to benign ratio from 0.126% to 0.0073%; i.e., more than 94% reduction. Such outcomes emphasize the sliding window technique's flexibility, which can significantly diminish or considerably amplify the visibility of phishing.

Summary. Our analysis examines preprocessing techniques across multiple studies, outlined in Table 4. Notably, seven out of twelve studies showed significant alterations in phishing ratios following the application of balancing methods, with increases reaching as high as 22 folds. These changes underscore the substantial modifications made to address dataset imbalances, reflecting the adoption of various strategies. Nevertheless, among the studies we surveyed, four studies chose not to alter the datasets, staying faithful to the dataset origin in the real-world deployment scenario.

Answering RQ4. Preprocessing impacts the effectiveness of detection algorithms. Imbalances in some studies resulted in very low phishing ratios, posing challenges in model training. Employing filter rules and generating synthetic data enhances data, but overfitting and non-authentic patterns diverge from real-world scenarios. These practices underscore the importance of maintaining data authenticity and relevance, which is essential for detection algorithms capable of reliably identifying phishing in diverse situations.

Table 5: An analytical evaluation of the robustness of phishing detection models, utilizing an array of features.

Work	*T*	*A*	*ID*	*OD*	*D*	*IS*	*OS*	*S*	*N*	*BN*	*GF*	*NT*	*ST*	*From*	*To*	*Input*	*TD*	*B*	*Prone*	*Suscept.*	*Resl.*
Chen *et al.* [33]	○	◐																	1	1	0
Chen *et al.* [31]			◐	◐	◐	◐	◐	◐	○										1	6	0
Palaio. *et al.* [74]										●	◐	○	○						2	1	1
Wen *et al.* [112]	○	◐	◐	◐					○					◐	◐				2	5	0
Kabla *et al.* [50]	○	◐								●				◐	◐	◐			1	4	1
Li *et al.* [60]		◐	◐	◐	◐							○							1	4	0
Wu *et al.* [116]	○	◐																	1	1	0
Li *et al.* [61]	○	◐	◐	◐	◐							○							4	2	0
Lin *et al.* [64]		◐								●	◐								0	2	1
Wen *et al.* [113]	○	◐									◐	○					○	◐	3	3	0
Zhou *et al.* [127]	○	◐	◐	◐							◐								1	4	0
Dong *et al.* [38]		◐	◐	◐								○							1	3	0

(1) Features: The features and abbreviations are listed in Table 2.
(2) Metrics: prone (○), susceptible (◐), and resilient (●) to manipulation. The last three columns are the sum of these metrics for each studied scheme.

3.7 Features Robustness

Delving into the insights gathered from various studies showcased in Table 5, we systematically classify features from prior research according to their susceptibility to adversarial manipulation and the complexities involved in their detection [91]. Our analysis revolves around how features fare when evaluated based on their robustness against manipulation. We classify the features used in the literature into three categories: (1) prone to manipulation, (2) resilient to manipulation, or (3) susceptible to manipulation. In the following, we

identify those features, contrast the literature based on them, and answer some key research questions along the way.

3.7.1 Features Prone to Manipulation

Adversaries can avoid detection by standard security models via transaction features manipulating (e.g., adversarial learning). For instance, altering the timing and volume is simple, but detecting these manipulations is challenging because they can seamlessly blend with legitimate activity. Based on our evaluation, we identify the following features that are prone to manipulation: time, neighbors, number of transactions, and transaction direction. In the following, we make the case for why those features are prone to manipulation, thus considered not robust.

Time. Time refers to the timestamp associated with each transaction, indicating when it was confirmed on the Ethereum network. Despite the immutability of blockchain timestamps post-confirmation, attackers can subtly manipulate transaction timings to their advantage [111]. For instance, by strategically choosing when to broadcast transactions, they can dodge detection during peak network activity or take advantage of specific market conditions. Moreover, by adjusting transaction fees, attackers can influence the priority of their transactions, either delaying them to decrease visibility to miners or hastening their confirmation through higher fees [111, 44].

Neighbors. This feature indicates the number of unique addresses a wallet transacts with and is easily manipulated. For instance, an adversary can create fictitious addresses to increase the neighbor count artificially, conduct transactions with reputable addresses to merge illegitimate transactions with legitimate ones, and churn funds between controlled accounts to obscure the origins of funds [55, 88].

Number of Transactions. This feature refers to the total transfers or trades linked to a specific address or within the network over a period and is vital for gauging the activity level and potential behavioral patterns of users [23]. Adversaries can deploy various methods to affect detection, such as executing numerous microtransactions or spreading transactions over several accounts [51, 79, 87]. These strategies aim to either obscure the movement of illicit funds or simulate heightened activity, thus making it difficult for analytical tools to differentiate between legitimate and suspicious activities. The strategic timing of these transactions, intended to bypass surveillance, adds a further layer of complexity to their detection [79].

Transaction Direction. This feature refers to whether a transaction for a particular wallet is incoming or outgoing. Understanding the flow of funds is key to identifying potential fraudulent patterns. Attackers may manipulate this feature by, for example, address

hopping—moving funds through multiple controlled addresses to blur the distinction between incoming and outgoing transactions and transaction cycling [65], where funds are circulated among attacker-controlled addresses in a closed loop. These techniques, aimed at disguising fraudulent activity and efficiently partitioning cryptocurrency networks, challenge detection systems [44, 89].

3.7.2 Features Resilient to Manipulation

Features associated with inherent blockchain characteristics or historical data are difficult to alter without leaving a trace. Due to its decentralized and immutable nature, blockchain is supposed to ensure these features are secure and formidable to attackers [84].

Successful Transactions. This feature refers to validated and confirmed transactions. The decentralized and immutable characteristics of blockchains make manipulation a formidable challenge. Given the distributed ledger's nature, altering a transaction record would require a consensus from most network participants, a near-impossible feat [123, 86]. Moreover, any attempt to modify a transaction would necessitate recalculating the cryptographic hashes for all subsequent blocks in the chain, which is infeasible [78]. The ledger's transparency further bolsters security by enabling independent verification of transactions, thereby enhancing trust and integrity [122].

Block Number. This feature refers to the chronological order of blocks in the blockchain, starting from the genesis block. The task of altering the block number is exceptionally challenging for attackers due to the blockchain's immutability and the required consensus [123]. Moreover, the possibility of manipulating block height, such as through a 51% attack where an entity gains control over the majority of the network's hashing power [91], remains theoretically conceivable but is practically improbable [19, 83]. Thus, blockchain design serves as a defense mechanism, preserving the integrity of block numbers.

3.7.3 Features Susceptible to Manipulation

While vulnerable to manipulation by determined adversaries, those features will likely be identified as suspicious by detection systems that can flag manipulative behaviors in these features through anomaly analysis for further investigation.

Amount. This feature is the transfer volume of a cryptocurrency. Attackers opt for round numbers to mimic benign activity or stay below reporting thresholds. However, genuine transactions typically involve fractional amounts due to natural trading fluctuations and price changes [114]. Advanced techniques can identify these anomalies by contrasting them with historical transaction patterns, seeking out regular transactions at specific intervals or sudden spikes in activity that may indicate automation or scripting [12].

From and To Addresses. The *from* address is a unique identifier for the sender's wallet and is debited the sent amount, including any fees. Similarly, the *to address* identifies the recipient's wallet, crediting it with the received amount. Although these addresses become immutable once a transaction is confirmed, adversaries can manipulate them in several ways [115]; e.g., they can scatter or aggregate funds using multiple addresses [25]. Utilizing mixers to combine transactions hides the funds' source. Engaging in address hopping disrupts pattern analysis [20]. Given their similarity, the *from* and *to* features are susceptible to similar manipulations.

Account Balance. This feature is the net digital currency available in a wallet, calculated as the total of all incoming minus outgoing transactions. Adversaries may manipulate account balances by, for example, simulating legitimate activity [49]. This could involve dispersing transactions across several accounts or using wash trading to obscure funds [18]. Moreover, attackers might exploit vulnerabilities in smart contracts to divert funds or hijack accounts for changing their balances, camouflaging fraudulent activities within normal patterns [92].

Strength. This feature, gauging both in-strength and out-strength (total funds entering/leaving a wallet), offers a view of a wallet's transaction volume and direction. Attackers can manipulate these features to conceal illicit actions within ordinary transaction flows [102]. Methods include altering transaction timing to align with typical user behavior, splitting or merging transactions to mask transfer amounts, or executing circular transactions among controlled accounts [26].

Gas Limit and Fee. On Ethereum, transactions require gas, with the gas limit and fee determining the computational work a transaction can use and its processing speed, respectively [73]. Adversaries may manipulate these features to cause issues like front-running or network congestion.

Therefore, gas limits and fees share inherent characteristics; both are vulnerable to analogous manipulation tactics.

Degree. This feature represents the total transaction count associated with an address, combining both in and out-degrees, where the in-degree counts incoming transactions and the out-degree counts outgoing transactions [31]. Adversaries may alter these features, obscuring transaction patterns to bypass detection systems. They might create fictitious transactions to inflate the degree artificially, distribute funds across multiple addresses to elevate the out-degree or employ these tactics in active and complex network engagement [125, 56].

Smart Contract Input. refers to the data fed into a contract when activated, including commands and operational parameters. While typically secure, these inputs can be manipulated due to vulnerabilities in smart contract programming. Adversaries might exploit weak validation to alter transaction data, such as through integer overflows [54], causing unintended

behaviors. Contracts with poor input validation are particularly at risk, enabling attackers to, for example, redirect funds or perform unauthorized actions. Effective detection relies on robust contract design and stringent validation [39, 110].

Summary. This analysis, summarized in Table Table 5, unveils a dynamic interplay between the vulnerability and security of blockchain transaction features. The table, which aligns features and abbreviations, illustrates that while some features are highly susceptible to manipulation, requiring sophisticated detection methods, others are fortified by the inherent security properties of blockchain. This dichotomy highlights the critical need for judicious feature selection and the continuous evolution of detection strategies to counter sophisticated threats. The findings from our examination advocate for a nuanced and flexible approach to feature selection, which is essential for the progressive enhancement of phishing detection capabilities and the overall strengthening of blockchain security.

Answering RQ5. Features range from highly susceptible to resistant to manipulation. Adversaries can mask fraudulent activities using transaction timing, volume, direction, and number of neighbors, complicating detection. Conversely, successful transactions, block numbers, and block IDs benefit from blockchain's inherent security.

3.8 Experimental Evaluation and Discussion

We now experimentally substantiate our findings and quantify the impacts of the raised issues in subsection 3.5. Our experimental work extends to the logic behind feature selection and modeling optimization (§3.9), the balance between phishing and benign transactions (§3.10), the effect of preprocessing (§3.10.3), and feature robustness (§3.11).

Experimental Setup. We base our experimental evaluation on two datasets from Ethereum's transaction network curated for phishing studies, namely Eth-PSD [50] and CTD [31]. Moreover, we use two widely used algorithms in interpretable literature, DT and KNN, as described in section 3.4. We chose those techniques over alternatives for three reasons: first, they are the most performant according to Table 2; second, they are easy to interpret; and third, they are widely used. For feature selection, we use SFS and RFE, two interpretable techniques. We use the parameters from the original works where the datasets were introduced for the initial feature sets and their size. We used the stratified 5-fold cross-validation in all experiments to evaluate the model's performance and report the average. In the following, we review the two datasets and their features.

Eth-PSD Dataset Overview. The Eth-PSD, due to Kabla *et al.*[50], is a comprehensive data collection that provides insights into the mechanics of Ethereum transactions, facilitating the analysis and detection of phishing activities. Eth-PSD incorporates TxHash, BlockHeight, TimeStamp, From, To, Value, Input, and a category label of the transaction as

benign or phishing. Features like TxHash and TimeStamp allow for examining transactions and timing, whereas From, To, and Value highlight the flow and magnitude of transferred funds.

CTD Overview. CTD, due to Chen *et al.*[31], primarily focuses on transaction analysis for detecting phishing. CTD's features include transaction flow metrics and account balance information, and it captures the nuanced dynamics of transactions susceptible to phishing attacks. Transactional features, such as in-degree and out-degree, capture the transaction counts, while aggregate transaction volumes are captured through degree and strength metrics: in-strength, out-strength, and total strength. The number of neighbors and the inverse number of transactions further discern phishing activity.

3.9 Feature Selection and Model Optimization

3.9.1 Feature Selection

We experimentally assess the efficacy of feature selection over Eth-PSD [50] and CTD *et al.* [31] to pinpoint key predictive features in both datasets. **Eth-PSD Feature Selection.** The SFS method, described in §3.5.2, was implemented to determine the optimal number of features for a phishing detection model, and the results are shown in Figure 2. The performance, evaluated by the negative mean squared error, plateaus after adding the third feature, indicating that additional features do not significantly enhance the predictive accuracy. This trend is consistently observed with stabilized performance from three to seven features, highlighting the diminishing returns of additional features. Consequently, we conclude that a leaner model with just three features is optimal, striking a balance between simplicity and effectiveness without compromising the model's performance.

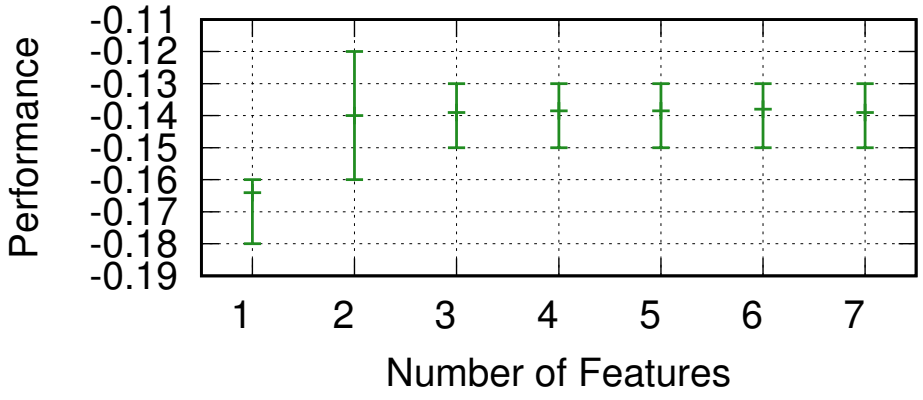

Figure 2: The performance of phishing detection, measured by the negative mean squared error, as a function of the number of features assessed using SFS over the Eth-PSD dataset.

To signify important features for classification outcome, the Pearson correlation coeffi-

Table 6: Visualization of the Pearson correlation.

Corr	TxHash	BlockHeight	TimeStamp	From	To	Value	Input	Class
TxHash	1	0.0056	0.0061	0.0018	0.0023	-0.0043	0.0012	0.0022
BlockHeight	-0.0056	1	1	0.35	-0.28	0.0039	0.26	0.43
TimeStamp	-0.0061	1	1	0.34	-0.27	0.0042	0.26	0.43
From	-0.0018	0.35	0.34	1	-0.28	-0.008	0.26	0.36
To	-0.0023	-0.28	-0.27	-0.28	1	0.0025	-0.28	-0.27
Value	-0.0043	0.0039	0.0042	-0.008	0.0025	1	-0.016	-0.013
Input	0.0012	0.26	0.26	0.26	-0.28	-0.016	1	0.62
Class	0.0022	0.43	0.43	0.36	-0.27	-0.013	0.62	1

cients are calculated and visually shown in Table 6, highlighting that the input (0.62), block height (0.43), and time (0.43) features are the top features due to their high correlations with the target label.

CTD's Feature Selection Process. All of CTD's features were normalized to the same scale. For feature selection, consistent with the original work, we use RFE, which provided insights into the predictive dynamics underpinning Ethereum phishing detection. In all cases, and as in the original work, the initial set of features was the following: in-degree, out-degree, degree, in-strength, out-strength, strength, neighbors, and inverse number of transactions.

3.9.2 Feature Sets Evaluation

We analyze the sufficiency of feature sets and their combinations in previous work. This analysis is grounded in empirical data from performance metrics across diverse feature combinations and uses the work of Kabla *et al.*[50] as an example. The evaluation focuses on the implications of these features on model efficacy, as measured by accuracy, precision, recall, and F1 scores. Table 7 compares multiple configurations, incorporating features such as those top-ranked (from, block height, time, and input). These model performance results reveal several insights.

❶ **High Efficacy Combinations.** Both DT and KNN models, employing features such as from, block height, time, and input (F, B, T, I), demonstrate superior performance with AUC scores reaching a peak of 0.98 and F1 scores ascending to 0.96. This blend epitomizes a holistic strategy with multiple dimensions. However, the negligible performance disparity with simpler models supports the need for feature selection.

❷ **Impact of Simplified Sets.** Simplifying the feature set by excluding input (I) still results in high AUC (0.98) and F1 (0.96), challenging the indispensability of the input feature for achieving elevated precision. This reduction implies that models can achieve similar efficacy with fewer features, casting doubt on including features without justification.

❸ **Minimal Pairs Analysis.** Examining minimal feature pairs like block height and time (B, T) results in an AUC of 0.97, which is only marginally less than those achieved by more complex combinations. This finding shows the considerable scope for optimizing the feature

Table 7: Performance of decision trees (DT) and k-nearest neighbor (KNN) models for various feature combinations.

Combination	KNN					DT				
	AUC	A	P	R	F1	AUC	A	P	R	F1
F, B, T, I	0.97	0.95	0.92	0.95	0.93	0.98	0.97	0.93	1.00	0.96
F, B, T	0.97	0.95	0.92	0.95	0.93	0.98	0.97	0.93	1.00	0.96
F, B	0.96	0.96	0.91	0.97	0.94	0.98	0.97	0.93	1.00	0.96
F, T	0.96	0.96	0.91	0.97	0.94	0.98	0.97	0.92	0.99	0.95
F, I	0.93	0.90	0.87	0.82	0.84	0.97	0.92	0.88	0.87	0.88
F, B, I	0.97	0.94	0.91	0.93	0.92	0.98	0.97	0.92	0.99	0.95
B, T	0.97	0.94	0.91	0.92	0.92	0.97	0.96	0.91	1.00	0.95
B, I	0.97	0.95	0.91	0.93	0.92	0.98	0.97	0.92	1.00	0.96
T, I	0.97	0.95	0.91	0.93	0.92	0.98	0.96	0.90	0.98	0.94
F, B, T, V	0.98	0.97	0.93	0.99	0.96	0.98	0.97	0.93	0.99	0.96
B, T, V	0.97	0.94	0.91	0.92	0.91	0.98	0.97	0.91	0.99	0.95
F, T, V	0.98	0.95	0.91	0.94	0.93	0.98	0.96	0.91	0.98	0.95
T, V	0.97	0.94	0.91	0.92	0.91	0.98	0.95	0.89	0.98	0.93

(1) Features: from (F), block height (B), timestamp (T), input (I), value (V). (2) Metrics: area under the curve (AUC), accuracy (A), precision (P), recall (R), and F1 score (F1).

selection processes.

3.9.3 Model Optimization

With a larger set of features, we examine the feature selection as a model optimization problem on CTD *et al.* [31] under a fixed dataset size (40k transactions). Motivated by the findings in the previous section, we look into how far one can push the model simplification while maintaining the performance. For this experiment, we use LightGBM, which performs well on complex networks, as shown in Table 2. We start with the comprehensive model and then reduce the complexity by eliminating some features using the RFE and Pearson correlation calculations. We make the following observations based on the results shown in Table 8.

❶ **Comprehensive Model.** The comprehensive model (M1) encompassing all the features has resulted in the highest AUC, at 0.88, with modest precision, recall, and F1 scores.

❷ **Large is not Always Better.** M2, a simpler model realized by dropping two features from M1, is only 0.02 worse than the optimal in terms of AUC. This trend, however, does not hold universally since M3 and M4, two models that are different in size, produced the same performance in terms of AUC. Similarly, M5, M6, and M7, three models of the same size, produced different performances in terms of AUC. Moreover, M6, a model that is only half the size of M3 and smaller than M4, produced twice as much precision as the former and outperformed the latter in terms of precision. These findings highlight the importance of model optimization in striking a balance between model complexity and performance.

Table 8: Model performance under a variable feature set with a fixed sample size of 40K from CTD.

Model	Feature Group	AUC	Precision	Recall	F1 Score
M1	ID, OD, D, IS, OS, S, N, INT	0.88	0.27	0.09	0.14
M2	OD, IS, OS, S, INT	0.86	0.11	0.03	0.04
M3	IS, OS, S, INT	0.85	0.09	0.03	0.05
M4	OD, D, IS	0.85	0.06	0.02	0.03
M5	D, IS	0.85	0.01	0.006	0.009
M6	S, INT	0.82	0.09	0.02	0.03
M7	IS, OS	0.81	0.01	0.006	0.008

(1) Features: in-degree (ID), out-degree (OD), degree (D), in-strength (IS), out-strength (OS), strength (S), neighbors (N), inverse number of transactions (INT).

Answering RQ1. Beyond the analytical results in subsection 3.5, we show that simplifying the feature set does not affect the model accuracy detrimentally. The nuanced decrease in performance metrics underscores the potential overemphasis on specific features without substantial evidence of impact.

3.9.4 Comparison of Models

The algorithms, parameters, and the features they operate on define the models. To provide some insight into the performance of the different models, and driven by the insights realized on the importance of feature selection earlier, we evaluate DT and KNN across various feature combinations, as shown in Table 7. In the following, we provide insights into the performance.

Overall Efficacy. The DT and KNN models show robust performance across feature combinations. However, DT consistently outperforms KNN in most cases, although marginally.

Feature Combination Impact. The performance of both algorithms fluctuates with the variation in feature sets. Moreover, the DT model consistently excels with specific sets (e.g., <from, block height>, <from, time stamp>, and <block height, time stamp>), maintaining an AUC of 0.98 and high precision and recall. Although the KNN model shows high performance, its precision and F1 score decreased with fewer features, hinting at its sensitivity to the feature set composition.

Precision and Recall Balance. The DT model exhibits an exceptional balance between precision and recall in configurations that include block height and time stamp, achieving a perfect recall in several cases. This underscores the DT model's adeptness at minimizing false negatives.

Table 9: Model performance for different sample sizes.

Size	Benign	Phishing	AUC	Precision	Recall	F1 Score
30K	29,896	103	0.88	0.20	0.09	0.12
40K	39,864	135	0.89	0.25	0.10	0.14
50K	49,838	161	0.90	0.20	0.09	0.13
100K	99,723	276	0.90	0.22	0.16	0.18
300K	299,440	559	0.82	0.11	0.15	0.12
500K	499,271	729	0.75	0.12	0.16	0.13

Answering RQ3. Our analysis highlights the differences in performance influenced by model choice. The DT models generally offer a marginal advantage over the KNN and stand out in precision, whereas KNN models have an edge concerning recall. The evaluation highlights the importance of choosing the right features over the algorithm choice.

3.10 Phishing-to-Benign Ratios Under Sampling

We explore the influence of the phishing-to-benign ratios on model performance to unravel how they affect performance and understand the impact of the balancing methods.

Sampling. In sampling transaction networks, we follow the random walk-based method to extract nodes from the larger CTD dataset as done in [31]. This approach starts from an initial seed node and conducts a walk over one of its neighbors with a probability proportional to the degree. If the next node is not in the list of the visited nodes, it is added, and the process is repeated until the total number of nodes (sample size) is exhausted. Once the sample size is exhausted, the edges between those visited nodes are derived from the original graph. The sample networks are shown in Table 9. For phishing detection, we use the DT algorithm.

3.10.1 Node Size and Ratio Impact

While the absolute number of phishing nodes increases with the dataset size from 103 in the 30K dataset to 729 in the 500K dataset, the proportion of phishing to benign instances significantly drops. Specifically, in the 30K dataset, phishing nodes constituted approximately 0.34% of the data. In the 500K dataset, this proportion dropped to about 0.15%. This proportional shift highlights the challenge of pinpointing an optimal dataset composition for phishing detection, stressing the need to carefully weigh the dataset size against the representation of phishing instances. The decreasing proportion of phishing instances as datasets grow complicates detecting phishing attempts due to a diluted signal-to-noise ratio. This situation underscores the intricate balance required between dataset size and the representativeness of phishing instances for effective phishing detection.

3.10.2 Performance Evaluation

The phishing-to-benign ratio directly impacts the performance in all metrics. Regarding AUC, the performance drops from 0.9 in datasets of 50K and 100K nodes to 0.75 in the 500K node dataset, indicating a diminishing ability of the model to differentiate between phishing and non-phishing transactions. In terms of precision, a drop from 0.25 in the 40K dataset to 0.12 in the 500K dataset indicates declining accuracy in identifying phishing as dataset size increases. Concurrently, recall experiences a slight rise from 0.09 in the 30K dataset to 0.16 in the 500K dataset, suggesting an improved model in identifying relevant instances in larger datasets. However, its accuracy in correctly labeling these instances as phishing decreases, highlighting a balancing act between recall and precision as dataset size expands.

Taking the precision and recall together, the F1 scores across various datasets reveal subtle shifts in performance, with 0.12 in the 30K dataset and 0.18 in the 100K dataset, hinting at a temporary balance improvement. Yet, this improvement does not persist in larger datasets, with F1 scores reverting to 0.12 and 0.13 for the 300K and 500K datasets, respectively.

Takeaway. Modifying phishing-to-benign ratios significantly impacts algorithmic efficiency up to an equilibrium, beyond which they cease to correlate with enhanced detection capabilities. These insights accentuate the criticality of dataset structuring that reflects the real-world scenario and the imperative for a judicious balance to optimize phishing detection mechanisms' precision and recall metrics within *practical contexts*.

Answering RQ2. Manipulating the phishing-to-benign ratio, as shown in Table 4 and section 3.6, is a common practice. We show empirically that shifts in the ratio significantly influence the detection performance. Initial minor increases may improve metrics, yet substantial expansions result in performance declines, showing the critical role of maintaining a balanced ratio to preserve accuracy and authenticity.

3.10.3 The Effect of Preprocessing

Preprocessing datasets enhances computational efficiency and initial model accuracy. This process raises a crucial question: How does preprocessing influence the detection algorithm's generalization across diverse datasets? We investigate the impact of standard preprocessing techniques like dataset reduction and feature selection on the generalization.

Eth-PSD Dataset Preprocessing. In Kabla *et al.*[50], the preprocessing incorporated *oversampling* to amend class imbalance, resulting in considerable data duplication. Notably, the *Input* feature displayed a significant overlap of 98,762 instances from a subset of 113,716, contributing to overall data redundancy affecting 33,279 rows. Such a considerable extent of

Table 10: The data overlap for the Input and From features, indicating the percentage of matching entries within each category across the training, testing, and overall datasets.

Feature	Training	Testing	Overall
From	86.47	88.37	86.85
Input	69.44	74.73	70.90

duplication significantly impairs the model's discriminative aptitude, escalating the susceptibility to overfitting and casting doubt on its capacity for generalization to novel samples.

Examining the *From* and *BlockHeight* attributes underscores the significant overlap and redundancy, with the From attribute having an overlap of 3,464 unique values in 83,209 instances and the BlockHeight attribute having a unique overlap of 5,519 values. This pronounced redundancy not only elucidates the impact of the oversampling technique in fostering a more unbiased class distribution—approximately 67.5% to 32.5%, but also highlights the profound trade-offs associated with such preprocessing interventions.

CTD Preprocessing. CTD's preprocessing, as highlighted earlier, relies on random walks for efficient subgraph sampling and meta-feature engineering to extract node features [31]. This approach, adaptive to dataset size, critically impacts algorithm performance. As highlighted earlier, while this approach does not introduce redundancies, it simply alters the phishing-to-benign ratio, affecting the ability of the models to distinguish between phishing and benign transactions as the sampled dataset complexity increases.

Answering RQ4. Preprocessing datasets impacts the generalization capabilities of detection algorithms. Simplifying datasets for computational efficiency risks losing crucial nuances and patterns vital for identifying a broad spectrum of phishing activities. Moreover, while preprocessing might enhance specific performance metrics, it does not consistently bolster generalization across phishing scenarios.

3.11 Evaluating Features Robustness

To ensure a comprehensive discussion, we present some preliminary findings on how feature manipulation affects the performance of phishing detection in transactions. Although a thorough evaluation of machine learning algorithms' robustness warrants a separate paper, we examine the manipulation of key features such as time, address, input, amount, block number, and successful transactions.

3.11.1 Time Manipulation

Timestamps are crucial for identifying patterns of potential illicit activities, although they are prone to manipulations. In the following, we investigate the hypothesis that modifications to transaction timestamps can obscure the patterns these models seek to identify, thereby

Table 11: Performance comparison under manipulation.

Fea-	KNN					DT				
ture	AUC	A	P	R	F1	AUC	A	P	R	F1
T	0.78	0.78	0.76	0.50	0.60	0.95	0.95	0.91	0.95	0.93
F	0.97	0.94	0.91	0.92	0.91	0.93	0.92	0.88	0.91	0.89
I	0.83	0.94	0.62	0.32	0.42	0.84	0.92	0.94	0.60	0.58
V	0.83	0.94	0.62	0.32	0.42	0.84	0.92	0.94	0.60	0.58
M1	0.97	0.95	0.92	0.95	0.93	0.98	0.97	0.93	1.00	0.96
M2	0.98	0.97	0.93	0.99	0.96	0.98	0.97	0.93	0.99	0.96

(1) Features: Time (T), from (F), input (I), and value (V).
(2) Metrics: AUC, accuracy (A), precision (P), recall (R), and F1 score.
(3) Base models: M1 and M2 are the base models with clean features. For evaluating the manipulation of T, F, and I, we use M1 (F, B, T, I; first row in Table 7; best performance and includes all features). Similarly, we use M2 as the model of F, B, T, V for V manipulation (tenth row) in Table 7.

impeding the detection of suspicious behavior.

Manipulation Technique. We employed two manipulation techniques. ① *Randomization.* The chronological order was disrupted by randomly shuffling timestamps across the dataset to obfuscate the patterns associated with the timing of transactions. ② *Uniform Distribution.* A uniform time distribution was applied over the specified period (2017–2018) to distribute transactions and mitigate significant concentrations. Both strategies correspond to delaying confirmation, e.g., manipulating fees. We limit our evaluation to modifying a single feature at a time to simplify our discussion.

Findings. Table 11 (first row vs. M1) shows that the KNN model's accuracy dropped from 0.95 (M1) to 0.78, the precision dropped from 0.92 to 0.76, the recall from 0.95 to 0.50, resulting in a drop in F1 score from 0.93 to 0.60, compromising its ability to identify phishing activities accurately.

3.11.2 Address Manipulation

For this feature, we simulated address hopping within an Ethereum transaction dataset to emulate adversaries' strategies to elude detection.

Manipulation Techniques. A custom hashing function was utilized, incorporating the original From address, a unique salt string, and the transaction's index to generate new, distinct addresses that simulate address-hopping behavior used by adversaries to anonymize their transactions and avoid detection.

Findings. Table 11 (second row vs. M1) shows the results. Noticeably, the manipulation of this feature results in the mildest change in performance. Specifically, the performance of DT models decreased from 0.97 to 0.92, the precision dropped from 0.93 to 0.88, and recall,

which was 1.00, dropped to 0.91; similarly, F1 score decreased from 0.96 to 0.89.

3.11.3 Input Manipulation

Input is pivotal for the functionality of smart contracts [129]. Manipulation, however, is possible due to exploitable defects in the contract's validation protocols [53, 43, 82].

Manipulation Techniques. To manipulate the input, we generated synthetic adversarial examples by extending the input data length, pattern injection, and input value randomization. More details are in appendix **??**

Findings. The results are in Table 11 (third row vs. M1). In KNN, the AUC dropped from 0.97 to 0.83, the accuracy from 0.95 to 0.94, the precision from 0.92 to 0.62, and the recall dropped sharply from 0.95 to 0.32. Thus, the F1 score fell from 0.93 to 0.42. In contrast, the DT deterioration was milder.

3.11.4 Amount Manipulation

Adversarial manipulation of *value* poses a significant threat, as attackers aim to obfuscate fraudulent transactions by blending them with legitimate ones.

Manipulation Techniques. We follow two techniques. ① *Refined amount smoothing.* We adjust the transaction amounts to incorporate randomly generated fractional components. Such a technique is devised to seamlessly integrate suspicious transactions within the flow of legitimate ones, eliminating any noticeable discrepancies. ② *Distributive pattern emulation.* By analyzing the distribution patterns of legitimate transactions and replicating these patterns in manipulated transactions. This strategy aims to create a mask of normalcy.

Findings. The results are in Table 11 (fourth row vs. M2). For KNN, the AUC decreased from 0.98 to 0.83, accuracy from 0.97 to 0.94, precision from 0.93 to 0.62, and recall dropped sharply from 0.99 to 0.32; thus, the F1 score fell from 0.96 to 0.42. In contrast, the DT model's deterioration was milder.

3.11.5 Block Numbers and Successful Transactions Manipulation

The design and security protocols inherent to blockchain technology render manipulating the number of successful transactions or altering block numbers technically infeasible within the blockchain ecosystem [123]. While it is possible to violate some of those properties using partitioning attacks, those attacks tend to be very costly and are limited in practice.

Answering RQ5. There is a varying susceptibility of features to adversarial manipulation. Contrary to assumptions, features derived from primary transaction data are vulnerable to manipulation, significantly impacting the detection models' performance. These findings underscore the need to reassess feature selection and model development, focusing on inherently stable features to counter manipulation attacks.

3.12 Summary

This study has critically examined the state of machine learning-based phishing detection in the Ethereum. By systematically evaluating prior work, it has highlighted both common practices and key challenges in the current detection methods. Our analysis shows that the frequent use of large feature sets, dataset balancing techniques, and feature vulnerability to manipulation often undermines the performance of phishing detection models. A major insight is that simpler models with optimized features can achieve comparable results to more complex approaches, mitigating the risk of overfitting and improving computational efficiency.

We also observed that the lack of standardized methods for feature selection and dataset processing across the literature complicates comparisons and reproducibility. Moreover, adversarial manipulation of key features like transaction time, neighbor counts, and transaction direction remains a significant concern for maintaining the integrity of detection systems.

Our empirical results further reinforce the importance of careful feature selection, as some features add minimal value or introduce vulnerabilities. In addition, adjusting phishing-to-benign ratios through artificial balancing methods, such as oversampling or synthetic data generation, can boost short-term model performance but risks hindering generalization to real-world scenarios.

4 ML-Based Ethereum Phishing Transactions Detection Robustness Under Simple Perturbations

4.1 Summary of Completed Work

This study evaluates the robustness of machine learning models in detecting fraudulent Ethereum transactions, specifically focusing on phishing detection. Our approach applies the Fast Gradient Sign Method (FGSM) to generate adversarial examples (AEs), targeting Random Forest (RF), Decision Tree (DT), and K-Nearest Neighbors (KNN) classifiers. These adversarial inputs manipulate key features of the transactions, such as timestamp, value, and address fields, to test the resilience of the models under realistic attack scenarios.

We employed two datasets: a binary classification dataset distinguishing phishing from benign transactions, and a multi-class dataset including phishing, scamming, fake ICOs, and benign categories. The manipulated features were subjected to both targeted and untargeted adversarial attacks to assess the performance degradation of the models. Our results demonstrate that RF outperforms both DT and KNN in maintaining classification accuracy under adversarial conditions, with a particular strength in resisting temporal and value manipulations.

Additionally, we explored potential mitigation strategies, including feature selection optimization, where temporal and address-based features showed greater resistance to adversarial attacks. This work provides empirical evidence that models' susceptibility varies based on feature type and attack method, and highlights the importance of developing robust defense mechanisms tailored to specific vulnerabilities.

Our findings contribute to the understanding of adversarial robustness in machine learning models for Ethereum transaction classification, offering insights into improving model reliability in adversarial environments.

4.2 Introduction

The proliferation of machine learning models in various domains has brought significant advancements in decision-making processes. However, concerns regarding the robustness and security of these models have also emerged alongside these advancements. Adversarial attacks, wherein small, carefully crafted perturbations are introduced into the input data to cause misclassification, seriously threaten the reliability of machine learning systems [17]. Understanding the susceptibility of these models to adversarial attacks is crucial for developing robust and trustworthy AI systems.

The increasing prevalence of machine learning in cybersecurity applications has signifi-

cantly improved the detection and prevention of various cyber threats. Among these threats, fraudulent activities such as phishing, scamming, and fake initial coin offerings (ICOs) pose substantial financial and personal data security risks. Machine learning models, particularly classification algorithms, have been deployed to identify and mitigate these threats with notable success. However, studies have raised concerns about the robustness and reliability of these models [42, 6, 13, 4, 5, 3, 2]. The attacks presented in the literature (see section **??**) are effective yet sophisticated.

This study investigates the impact of extremely simple adversarial attacks on different machine learning models used in fraudulent transaction detection, specifically focusing on Random Forest (RF), Decision Tree (DT), and K-Nearest Neighbors (KNN) classifiers. By employing the Fast Gradient Sign Method, a widely recognized adversarial attack technique, we assess the performance degradation of these models when subjected to simple, realistic adversarially crafted inputs [101]. The primary objective is to understand how these models are susceptible to adversarial manipulation and explore potential mitigation strategies to enhance their robustness.

Previous research has highlighted the vulnerability of ML models to adversarial attacks [75]. This study contributes to the existing body of knowledge by providing a detailed empirical analysis of the effects of simple adversarial attacks on fraud detection models and proposing practical approaches to mitigate these effects. Our findings reveal inconsistency across algorithms for their tolerance of simple manipulation, underscoring the importance of selecting appropriate models and implementing robust defense mechanisms tailored to specific applications.

Research Gap. Despite advancements in machine learning and blockchain technology, critical gaps persist in effectively understanding mitigating adversarial attacks and emerging security threats within cryptocurrency networks. AEs grounded in the context of application are underexplored. AEs in the feature space that leverage targeted and untargeted attacks, transaction fraud, smart contract exploits, etc., are lacking. This underscores the imperative for further analysis. This study aims to bridge this gap by investigating the robustness of machine learning-based phishing detection algorithms against *simple manipulations*, comparing the effectiveness of various algorithms in resisting such attacks, and exploring mitigation strategies to enhance their resilience. Our approach involves evaluating the algorithms' susceptibility to subtle feature manipulations and conducting a comparative analysis to identify the most robust models.

4.3 Research Questions

This research explores the robustness, comparative performance, and mitigation strategies of machine learning-based phishing detection algorithms for Ethereum transactions in adversarial manipulations. The following questions highlight the core issues and the necessity to address them in light of existing literature.

RQ1.Are machine learning-based phishing detection algorithms for Ethereum robust against simple manipulations of individual features? This question is motivated by the vulnerability of machine learning (ML) models to adversarial attacks, as demonstrated in several studies [71, 28, 22, 95, 105]. In these studies, slight modifications in input data significantly altered the classification outcome, highlighting how even minor changes in key features like transaction amounts or timestamps can lead to incorrect classifications. Such manipulations can make it easier for adversaries to deceive models, underscoring the need to evaluate the robustness of ML models used in Ethereum phishing detection to ensure they remain effective and reliable despite such attacks.

RQ2.How do different machine learning algorithms compare robustness against adversarial manipulations in Ethereum phishing detection? This question stems from the observation that various machine learning algorithms, while effective in detecting phishing activities, exhibit differing levels of resilience against adversarial attacks [96, 72, 36, 10, 80, 35]. The robustness of these algorithms can vary significantly under adversarial conditions, which can be seen in studies where some algorithms perform better than others when subjected to adversarial manipulations designed to evade detection [57, 37, 99]. A comparative analysis of these algorithms is essential to identify those that provide the best defense against such threats, ensuring the highest possible security in Ethereum transaction classification.

RQ3. How can the impact of manipulations be mitigated in machine learning-based Ethereum phishing detection? This question arises from the need to develop robust defensive strategies against adversarial attacks, as highlighted in recent literature [120, 69, 27, 41, 117]. Effective mitigation strategies could include enhancing feature selection processes, employing advanced data augmentation techniques, or implementing adversarial training methods [63, 80, 130]. Understanding and developing these techniques are crucial for improving the security and reliability of ML-based phishing detection systems in Ethereum transactions, thereby reducing the risk posed by adversaries manipulating transaction data and enhancing overall network security.

4.4 Methodology

Our pipeline is shown in Figure 3 and some of its key aspects are reviewed below.

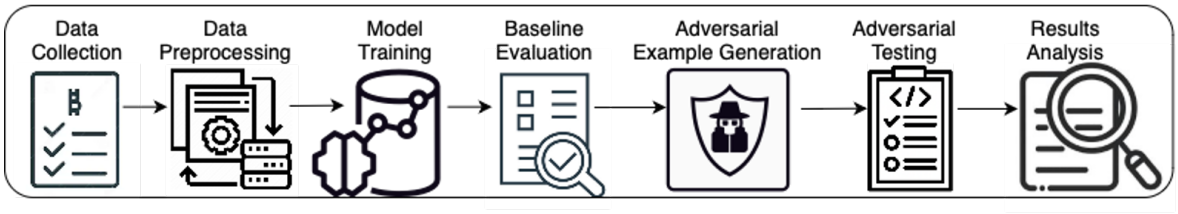

Figure 3: Pipeline in Ethereum transactions and adversarial testing.

4.4.1 Data Preparation

4.4.2 Dataset Description

For our analysis, we employed two datasets. The first dataset, as detailed by Kabla *et al.* [50], focuses on binary classification, distinguishing between phishing and benign transactions. This dataset encompasses features such as `TxHash`, a unique identifier for each transaction; `BlockHeight`, which specifies the height of the block in which the transaction was recorded; and `TimeStamp`, indicating the exact time the transaction was confirmed and added to the blockchain. It also includes the `From` and `To` addresses, representing the Ethereum addresses of the sender and receiver, respectively. The `Value` feature denotes the amount of Ether transferred in the transaction, while `ContractAddress` indicates the Ethereum address of the smart contract involved, if applicable. Additionally, the `Input` field contains any extra data provided with the transaction. The dataset labels transactions as either phishing (1) or benign (0) under the `Class` feature. Dataset 1 includes 23,472 transactions, of which 15,989 are benign and 7,483 are phishing.

The second dataset, as described by Al-Emari *et al.* [11], is intended for multi-class classification, categorizing transactions into Phishing, Scam, Fake ICO, or Benign. This dataset features the `hash`, a unique identifier for each transaction, and `nonce`, a counter ensuring each transaction is processed only once. The `transaction_index` indicates the transaction's position within the block, while `from_address` and `to_address` denote the blockchain addresses of the sender and receiver, respectively. The `value` field specifies the amount of cryptocurrency transferred. The dataset also includes `gas`, representing the gas limit provided for the transaction, and `gas_price`, indicating the price per gas unit. The `input` field contains additional data attached to the transaction. Moreover, `receipt_cumulative_gas_used` provides the total gas used by all transactions up to and including the current one within the block, and `receipt_gas_used` specifies the gas consumed by this particular transaction. The `block_timestamp` and `block_number` detail the time and number of the block that includes the transaction, while `block_hash` serves as the block's unique identifier. Lastly, the `from_scam` and `to_scam` fields indicate whether the sender's and receiver's addresses are associated with scams (0 for no, 1 for yes). The dataset also includes categorical data, `from_category` and `to_category`, which classify the nature

of the participants, such as Phishing, Scamming, or Fake ICO. In this dataset, `Benign` had 57,000 transactions (79.28%), `Scamming` had 11,143 transactions (15.51%), `Phishing` had3,106 transactions (4.32%), and `Fake ICO` had only 1 transaction. Dataset 2 includes 71,250 transactions, with 80-20 training-testing splits.

4.4.3 Experimental Procedures

We utilized the two datasets outlined earlier to examine how simple AEs impact the classification accuracy of three classifiers: RF, DT, and KNN.

Minimal Manipulations. AEs were crafted by manipulating specific features. For the first dataset, we manipulated for feature. ① *Timestamp Manipulation (TimeStamp)*: We use predefined intervals to simulate future occurrences, testing the models' robustness to shifts against hypothetical scenarios of temporal manipulation. ② *Value Manipulation (Value)*: Transaction values were altered using two strategies: uniformly by adding a fixed percentage to each transaction's value or proportionally by introducing a random percentage change relative to each transaction's original value. ③ *Receiver Address Manipulation (To)*: The receiver's address was randomly changed to different addresses within the dataset to simulate phishing transactions directed to alternative destinations. ④ *Sender Address Manipulation (From)*: The sender's address was altered to different sender addresses to simulate phishing transactions originating from various sources.

We implemented a two-pronged approach, including both targeted and untargeted adversarial attacks on a second dataset [11]. AEs were generated by focusing on a broader array of critical transaction features, allowing us to simulate realistic attack scenarios and identify potential vulnerabilities more precisely.

Untargeted Attacks. The study of untargeted attacks involved generating AEs by applying broad, random perturbations across the entire feature space. Initially, AEs were generated using all features to evaluate the model's capacity to withstand attacks.

Individual features were then targeted to challenge the model more effectively. Modifying the *from_address* and *to_address* fields introduced new, unseen addresses to assess the model's ability to handle transaction origin and destination changes. Altering *value*, *gas* and *gas_price* simulated economic fluctuations, providing insights into the model's sensitivity to variations in transaction costs. Manipulating *block_timestamp* and *block_number* mimicked transaction timing and sequence changes to understand the model's response to variations in transaction order. Altering *input*, *receipt_cumulative_gas_used*, and *receipt_gas_used* help explore the impact of changes in content.

Targeted Attacks. We conducted targeted adversarial attacks focusing on three scenarios: benign, phishing, and scamming. We employed two methods for generating these targeted

attacks: rule-based and gradient-based (using the Fast Gradient Sign Method).
Rule-based Modifications This approach applies straightforward, rule-based modifications to critical features like transaction value and timestamps, simulating realistic variations that can alter the classification outcomes.

- **Benign Targeted Attack:** This scenario aimed to assess the model's ability to maintain a benign classification despite manipulations, simulating tactics used to camouflage malicious activities. We created artificial benign transactions by minorly adjusting features such as transaction value and block_timestamp.
- **Phishing Targeted Attack:** The focus was on modifying phishing-labeled transactions to evade detection by misclassifying them as benign. This involved altering attributes such as from_address, to_address, and value, simulating an adversary's attempt to bypass security measures.
- **Scamming Targeted Attack:** In this scenario, scam-labeled transactions were manipulated to explore whether they could be misclassified as benign or other types. Adjustments to features like gas and gas_price were made to examine the model against efforts to obscure scam activities through changes in transaction costs.

4.4.3.1 FGSM

FGSM applied small, strategically calculated changes to subtly influence the model's predictions while maintaining realistic feature values. Unlike the rule-based method, FGSM retained the existing distribution from the dataset, focusing on fine-tuning modifications based on the gradients to maximize the attack's effectiveness.

- **FGSM Details:** FGSM calculates perturbations that align with the gradient direction of the loss function. The perturbations were applied using the formula:

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y)),$$

  where x is the original feature, x' is the perturbed feature (AE), ϵ is a small scalar controlling the perturbation's magnitude, ∇_x represents the gradient of the loss function J for x, and $J(\theta, x, y)$ is the loss function dependent on the model parameters θ, input x, and true label y. The term $\text{sign}(\nabla_x J(\theta, x, y))$ provides the direction in which to perturb the feature vector to maximize the loss function.
- **Features:** FGSM perturbations were applied to transaction value, gas, gas_price, and block_timestamp. These features are critical in determining the nature of the transaction, and even small changes may affect the classification results.

4.5 Results and Analysis

4.5.1 Preliminary Results

We evaluate how different perturbations on the first dataset affect the accuracy and resilience of RF, DT, and KNN models. We examine how uniform and proportional value manipulations and address change impact model performance.

Timestamp Manipulations. We evaluate the effects of various timestamp modifications on classifier accuracy, including shifts of +24 hours, +1 hour, +30 minutes, +15 minutes, and +5 minutes. The RF classifier showed the highest resilience with only minor reductions in accuracy. For example, a one-day timestamp shift resulted in a decrease in accuracy from 98.82% to 95%, while a one-hour change led to an accuracy of 97.31%. In contrast, the DT model experienced a more pronounced decline, with accuracy dropping from 98.35% to 94.46% with a one-day shift. The KNN classifier was most affected by these temporal manipulations, with accuracy falling from 94.45% to 83.42% for a one-day change. These findings highlight the superior robustness of the RF model in handling timestamp alterations, positioning it as a preferable option for phishing detection in environments with variable timestamps (Table 12).

Value Manipulations. The uniform value manipulations caused significant declines. RF's accuracy dropped to 69%, and DT's fell to 68%, with substantial reductions in precision and recall for phishing transactions. Notably, the recall for phishing in DT decreased to 0.01% under uniform manipulation. Proportional manipulations had minimal impact, with accuracy and other metrics remaining close to their original values. KNN maintained relatively stable performance across both manipulation types, indicating robustness against such value changes (Table 13).

Address Manipulations. The robustness of the classifiers was tested against AEs of the `From` and `To` address features, involving changes in 5,000, 10,000, and 23,472 instances. For the RF model, accuracy decreased to 87% when the `From` address was manipulated and to 84% for the `To` address. Precision and recall for phishing transactions also declined significantly, with F1-scores dropping notably. The DT model showed a moderate reduction in accuracy, dropping to 92% for `From` and 93% for `To` manipulations, and a noticeable decrease in recall for phishing. KNN was the most sensitive to these manipulations, with accuracy falling to 85% for `From` and 93% for `To`, and significant drops in precision and recall for phishing transactions (Table 14 and 15).

Next, our analysis will focus on the second dataset from Al-Emari *et al.* [11], which provides a comprehensive and relevant context for multi-class classification tasks.

Table 12: RF, DT, and KNN performance under timestamp manipulations. Accuracy, precision, recall, F1 score, and counts. B stands for benign and Ph. for phishing.

Increment	Model	Dataset	Acc	Precision		Recall		F1		Count	
				B	Ph	B	Ph	B	Ph	#B	#Ph
Original	RF	Baseline	0.9882	1	0.96	0.98	1	0.99	0.98	15989	7483
	DT	Baseline	0.9835	1	0.95	0.98	1	0.99	0.97	15989	7483
	KNN	Baseline	0.9445	1	0.85	0.92	1	0.96	0.92	15989	7483
+24 hours	RF	Adversarial	0.95	0.94	0.98	0.99	0.86	0.96	0.92	15498	7974
	DT	Adversarial	0.9446	0.95	0.94	0.97	0.88	0.96	0.91	15601	7871
	KNN	Adversarial	0.8342	0.84	0.82	0.94	0.62	0.88	0.7	14376	9096
+1 hour	RF	Adversarial	0.9731	0.97	0.97	0.99	0.94	0.98	0.96	15498	7974
	DT	Adversarial	0.9626	0.97	0.95	0.98	0.94	0.97	0.94	15601	7871
	KNN	Adversarial	0.8997	0.93	0.84	0.93	0.84	0.93	0.84	14376	9096
+30 min	RF	Adversarial	0.9776	0.98	0.97	0.99	0.96	0.98	0.96	15498	7974
	DT	Adversarial	0.9649	0.97	0.95	0.98	0.94	0.97	0.94	15601	7871
	KNN	Adversarial	0.916	0.95	0.85	0.92	0.9	0.94	0.87	14376	9096
+15 min	RF	Adversarial	0.9817	0.99	0.97	0.99	0.97	0.99	0.97	15498	7974
	DT	Adversarial	0.9672	0.98	0.95	0.98	0.95	0.98	0.95	15601	7871
	KNN	Adversarial	0.9179	0.95	0.85	0.93	0.9	0.94	0.87	14376	9096
+5 min	RF	Adversarial	0.9793	0.99	0.97	0.98	0.97	0.98	0.97	15498	7974
	DT	Adversarial	0.9734	0.98	0.95	0.98	0.97	0.98	0.96	15601	7871
	KNN	Adversarial	0.9344	0.98	0.85	0.92	0.96	0.95	0.9	14376	9096

Table 13: Performance evaluation of RF, DT, and KNN models subjected to 1% uniform and proportional value manipulation strategies. Metrics are as in Table 12.

Model	Strategy	Acc	Precision		Recall		F1		Count	
			B	Ph	B	Ph	B	Ph	#B	#Ph
RF	Original	0.99	0.98	1	1	0.99	0.99	0.99	15803	7669
	Uniform	0.69	0.96	0.68	0.02	1	0.03	0.81	23353	119
	Proportional	0.99	0.98	1	1	0.99	0.99	0.99	15813	7659
DT	Original	0.98	0.95	1	1	0.98	0.97	0.99	15601	7871
	Uniform	0.69	0.75	0.68	0.02	1	0.03	0.81	23294	178
	Proportional	0.98	0.95	1	1	0.98	0.97	0.99	15619	7853
KNN	Original	0.96	0.89	0.99	0.98	0.94	0.93	0.97	15192	8280
	Uniform	0.96	0.89	0.99	0.98	0.94	0.93	0.97	15193	8279
	Proportional	0.96	0.89	0.99	0.98	0.94	0.93	0.97	15192	8280

4.5.2 Results of Targeted Attacks

Rule-based Modifications. We initially focus on rule-based AEs for specific classes.

① *Benign Class.* The RF and DT models initially demonstrated near-perfect accuracy in classifying benign transactions on the original test set. Under adversarial conditions, the accuracies for benign classifications declined markedly, with RF and DT models dropping to 84.39% and 84.65%. This represents a reduction exceeding 15%. In contrast, the KNN model sustained a higher adversarial accuracy of 90.25%.

② *Phishing Class.* The initial phishing accuracies for RF and DT were 96.34% and 96.98%,

Table 14: Performance evaluation of RF, DT, and KNN models under manipulations of the `From` feature in Ethereum transaction datasets. Metrics are as in Table 12.

Model	Strategy	Acc	Precision		Recall		F1		Count	
			B	Ph	B	Ph	B	Ph	#B	#Ph
RF	Original Strategy	0.99	1	0.96	0.98	1	0.99	0.98	15708	7764
	5000 Changes	0.96	0.96	0.97	0.98	0.91	0.97	0.94	16386	7086
	10000 Changes	0.94	0.93	0.97	0.99	0.83	0.96	0.89	17027	6445
	23472 Changes	0.87	0.84	0.97	0.99	0.6	0.91	0.74	18877	4595
DT	Original Strategy	0.98	1	0.95	0.98	1	0.99	0.97	15601	7871
	5000 Changes	0.97	0.98	0.95	0.98	0.96	0.98	0.96	15935	7537
	10000 Changes	0.96	0.96	0.95	0.98	0.91	0.97	0.93	16272	7200
	23472 Changes	0.92	0.91	0.94	0.98	0.79	0.94	0.86	17207	6265
KNN	Original Strategy	0.94	1	0.85	0.92	1	0.96	0.92	14706	8766
	5000 Changes	0.93	0.97	0.85	0.92	0.93	0.94	0.89	15209	8263
	10000 Changes	0.91	0.94	0.84	0.92	0.87	0.93	0.85	15767	7705
	23472 Changes	0.85	0.86	0.81	0.93	0.67	0.89	0.74	17288	6184

Table 15: Performance evaluation of RF, DT, and KNN models under manipulations of the `To` feature in Ethereum transaction datasets. Metrics are as in Table 12.

Model	Strategy	Acc	Precision		Recall		F1		Count	
			B	Ph	B	Ph	B	Ph	#B	#Ph
RF	Original Strategy	0.99	1	0.96	0.98	1	0.99	0.98	15708	7764
	5000	0.96	0.95	0.97	0.98	0.89	0.97	0.93	16558	6914
	10000	0.92	0.91	0.97	0.99	0.79	0.95	0.87	17377	6095
	23472	0.84	0.81	0.97	0.99	0.51	0.89	0.67	19537	3935
DT	Original Strategy	0.98	1	0.95	0.98	1	0.99	0.97	15601	7871
	5000	0.97	0.98	0.95	0.98	0.96	0.98	0.96	15884	7588
	10000	0.96	0.97	0.95	0.98	0.93	0.97	0.94	16142	7330
	23472	0.93	0.92	0.94	0.98	0.83	0.95	0.88	16872	6600
KNN	Original Strategy	0.94	1	0.85	0.92	1	0.96	0.92	14706	8766
	5000	0.94	0.99	0.85	0.92	0.98	0.95	0.91	14849	8623
	10000	0.94	0.98	0.85	0.92	0.97	0.95	0.91	14988	8484
	23472	0.93	0.96	0.85	0.93	0.93	0.94	0.89	15351	8121

respectively. However, these values **plummeted to 1.31% and 1.18% under adversarial conditions**, reflecting a reduction of over 95%. The KNN model, which began with a lower baseline accuracy of 41.49%, saw a decrease to 2.15%.

③ *Scamming Class* Initially, the RF and DT models exhibited high accuracies of 99.5% and 98.68%. Adversarial attacks reduced these accuracies to 14.27% and 14.16%, representing a reduction of over 85%. The KNN model, with an initial accuracy of 67.06%, experienced a drop to 7.6%.

4.5.3 Gradient-based Approach Using FGSM

① *Benign Class* The overall accuracy of the RF model decreased from 99.75% to 94.62%, with a significant deterioration in phishing detection metrics. Despite hat, the model maintained a high benign accuracy of 99.95%. Conversely, the DT model's overall accuracy plummeted from 99.64% to 9.54%, with benign recall dropping to 0.02, indicating extreme vulnerability.

The KNN model preserved a perfect benign accuracy of 100% even under attack, but its overall accuracy fell from 90.15% to 80.22%, reflecting a failure to detect phishing and scamming categories effectively.

② *Phishing Class* The phishing detection accuracy of the RF model decreased from 96.34% to 47.69%, with a significant drop in the F1-score. The DT model's overall accuracy declined to 10.22%, with phishing recall reducing to 0.75%, underscoring a pronounced susceptibility to adversarial attacks. The KNN model's phishing detection performance collapsed entirely, with metrics falling to zero, indicating a complete failure to detect phishing transactions under adversarial conditions.

③ *Scamming Class* The overall accuracy of the RF model dropped from 99.75% to 81.75%, with scamming accuracy decreasing from 99.50% to 76.70%. The DT's overall accuracy fell to 9.71%, with scamming recall drastically reducing to 0.30. The KNN model's scamming detection metrics also dropped to zero.

4.5.4 Results of Untargeted Attacks

① *All Features* The RF model's accuracy decreased to 95.81%, DT's to 91.16%, with phishing detection severely impaired, and KNN maintained its baseline accuracy.

② *Address Features* AEs focusing on the *from_address* and *to_address* features resulted in a decline in overall accuracy to 80.22% for all models. None of the models detected phishing or scamming, indicating a high sensitivity to address manipulations.

③ *Financial Features* AEs targeting financial features (*value*, *gas*, *gas_price*) led to reduced RF's accuracy to 79.96%. The DT's accuracy dropped to 79.42%. The KNN model's accuracy slightly decreased to 90.02%.

④ *Using Temporal Features* Adversarial manipulations of temporal features (*block_timestamp*, *block_number*) showed the RF model's accuracy fell from 99.02% to 80.25%, with phishing detection metrics nearly nullified. The DT's accuracy similarly declined to 80.25%. The KNN model's accuracy also dropped to 80.26%.

Takeaway. These results underscore the need for robust defensive mechanisms against AEs. The significant declines in performance metrics under simple conditions highlight the necessity for a more reliable classification of transactions.

4.6 Discussion

Feature Selection for Optimal Classification. The analysis of transaction classification in this study highlights the significant role of feature selection in the robustness and accuracy of ML models. Among the features tested, **timestamp** and **value** emerged as critical classifier performance determinants. Timestamp manipulations demonstrated substantial

impacts across models, with RF showing notable resilience compared to DT and K-KNN. The accuracy metrics indicate that temporal features, such as transaction time and date, are crucial for distinguishing legitimate from fraudulent transactions due to their inherent variability and relevance to transactional behaviors.

In contrast, value manipulations, including uniform and proportional changes, significantly affected model performance, particularly under uniform conditions. RF and DT models experienced considerable accuracy drops with uniform value changes, while KNN maintained relative stability. These findings suggest that while **transaction value** is a key feature for classification, it is also highly susceptible to perturbations.

Most Resistant Features to Adversarial Attacks. The study's results indicate that **address features**, specifically the `From` and `To` addresses, exhibit higher resistance to adversarial attacks compared to other feature types. Manipulations of these features resulted in moderate accuracy reductions for DT and RF models but a more pronounced impact on KNN. These features likely encode the relationship patterns between transaction entities, making them inherently resistant to straightforward changes.

The analysis showed that despite their critical contribution to accuracy, temporal features were also relatively resistant to manipulations. Timestamp shifts caused accuracy declines, but the extent was less severe than value manipulations. This indicates that while temporal features are crucial for classification, they are robust against adversarial attacks due to timestamp data's complexity and non-repetitive nature.

Best Combinations. For resilient classification, a combination of **temporal and address features** has proven effective. The synergy between these features offers a dual layer of robustness; temporal features provide a dynamic aspect that captures the temporal distribution and patterns of transactions, while address features contribute a stable relational component less prone to adversarial interference.

Combining temporal features with **financial features**, such as transaction value and gas price, also enhances robustness. The results show that despite the susceptibility of financial features to uniform manipulations, their combined use with temporal data provides a broader context that improves model resilience. The temporal features help to contextualize the financial data, mitigating the impact of adversarial value manipulations by providing a temporal frame of reference.

The following recommendations can be drawn for the effective and robust classification of digital transactions, especially in adversarial environments. ① **Focus on Temporal and Address Features**: Incorporate timestamp and address data as primary features due to their robustness against adversarial attacks and critical role in classification accuracy. ② **Integrate Financial Features with Temporal Data**: Use financial transaction data with temporal features to improve robustness and provide a comprehensive transactional context

that helps counteract adversarial manipulations. ③ **Adopt a Multi-Feature Approach**: Utilize a combination of diverse feature types to leverage their respective strengths and ensure a balanced, resilient classification model capable of withstanding various adversarial strategies.

4.7 Summary and Work to be Completed

This study investigates the vulnerability of machine learning models to adversarial attacks in detecting fraudulent Ethereum transactions, particularly phishing attacks. The models analyzed include RF, DT, and KNN. We employed the FGSM to generate adversarial examples, which involve carefully crafted manipulations of key transaction features such as timestamps, value, and addresses. Our findings highlight significant differences in the resilience of these models when exposed to adversarial manipulations.

The results show that RF demonstrated the highest robustness, especially when timestamps and transaction values were altered. In contrast, DT and KNN were more vulnerable, with KNN experiencing the greatest performance degradation. We also explored mitigation strategies like enhanced feature selection and adversarial training. Temporal and address-based features provided better resistance to adversarial attacks, offering valuable insights into improving phishing detection defenses.

To extend this work, we plan to broaden the adversarial training approach by generating a wider variety of adversarial examples and implementing advanced attack techniques. We will retrain the models using these new examples and compare their performance across all classes. This will help evaluate how retraining enhances model robustness, particularly addressing vulnerabilities in KNN. The results will offer deeper insights into the effectiveness of adversarial training in strengthening phishing detection models.

5 Improving ML-Based Phishing Detection in Ethereum Transactions with Implicit Features

5.1 Introduction and Motivation

Phishing attacks on blockchain networks, particularly Ethereum, have become a growing concern as the usage of decentralized platforms expands. Ethereum's transparency and secure transaction records have made it a target for cybercriminals, with phishing emerging as one of the most prevalent attack vectors. These scams exploit the trust within blockchain transactions to deceive users, often leading to significant financial losses. For instance, the 2022 phishing attack on Uniswap Labs resulted in the theft of over eight million USD, highlighting the critical need for more robust and accurate phishing detection systems.

Detecting phishing on Ethereum presents several challenges due to the complex nature of these attacks. Traditional approaches primarily rely on explicit transactional features, such as gas usage, transaction values, and timestamps. While these features provide valuable insights into transaction behavior, they often fail to capture the broader relational dynamics that are crucial for identifying more sophisticated phishing activities. Addressing this limitation requires exploring additional features, such as implicit patterns within the transaction network, which can reveal interactions and transaction flows indicative of phishing.

The motivation for this work is to systematically evaluate the effectiveness of two distinct feature sets in phishing detection on Ethereum: explicit transactional features and implicit graph-based features. By first testing the model's performance using explicit transactional features, and then separately testing it using implicit features, we aim to understand the strengths and limitations of each approach. This approach allows for a clear comparison of how different feature types impact the accuracy and robustness of phishing detection, ultimately contributing to the development of more resilient models.

5.1.1 Problem Statement

Phishing attacks on Ethereum pose a significant threat to users, leading to the loss of millions of dollars each year. As phishing schemes grow in sophistication, traditional detection methods relying on explicit transactional features, such as gas usage, timestamps, and transaction values, have proven to be insufficient in capturing the full complexity of phishing behavior. These methods often fail to account for the relational and temporal dynamics inherent in Ethereum's transaction network, leaving room for malicious activities to go undetected.

The core challenge in phishing detection lies in identifying patterns that distinguish phishing transactions from legitimate ones. Phishing transactions often blend seamlessly into

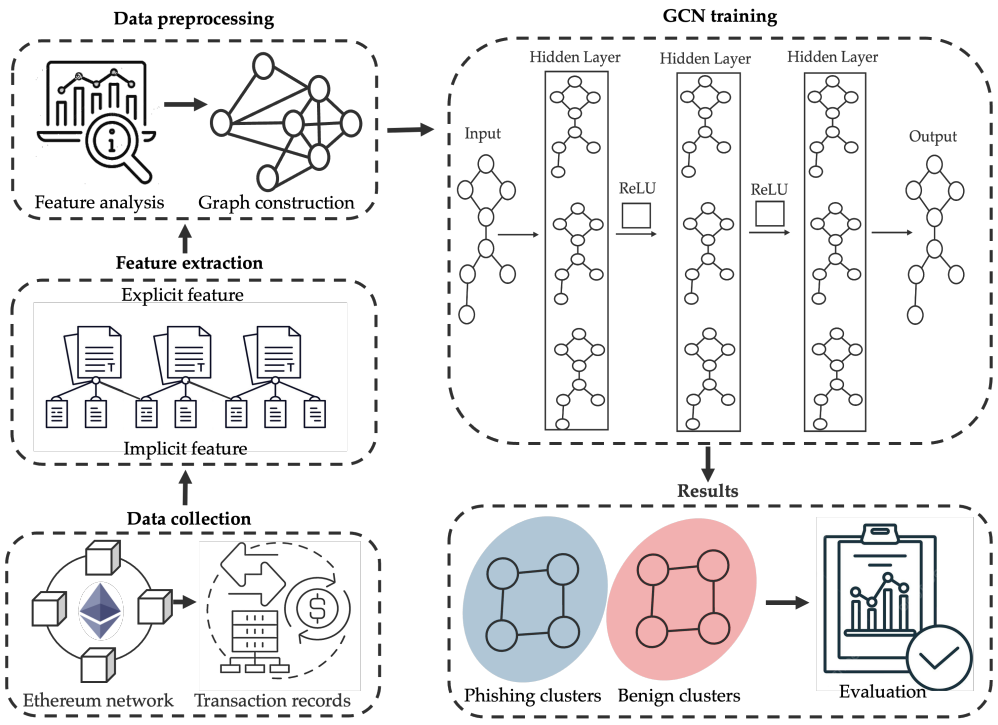

Figure 4: An illustration of the proposed pipeline, integrating explicit and implicit features from the Ethereum network.

the Ethereum network, making detection difficult when only explicit features are considered. Additionally, the inherent imbalance between phishing and benign transactions in available datasets further complicates efforts to build effective detection models, as phishing activities are heavily underrepresented.

To address these challenges, it is necessary to systematically explore the effectiveness of different feature sets in phishing detection. This work focuses on evaluating the performance of models using two distinct sets of features: explicit transactional features and implicit graph-based features. By testing the model performance separately for each feature set, this study aims to provide insight into how these feature types impact the accuracy and reliability of phishing detection in Ethereum transactions. The goal is to understand the limitations of each feature set and identify potential improvements for more robust phishing detection methods.

5.2 Model Workflow

Our model leverages transaction data from the Ethereum blockchain to detect phishing activities with high accuracy. The workflow begins with the data collection process, where we gather extensive transactional data, both phishing and benign, from trusted sources such as Etherscan. Following this, we engage in feature extraction, focusing on both explicit and

implicit attributes that define transactional behavior.As shown in Figure 4, this data is then passed through a Graph Convolutional Network (GCN) architecture, where each node represents a blockchain address and each edge a transaction. The model's layers capture both individual node attributes and network-wide interactions, enabling it to identify suspicious activities more effectively. Finally, the model is trained using labeled phishing data, and its performance is evaluated using standard classification metrics such as precision and recall.

5.3 Work to be Done

The primary focus of our upcoming work is to develop and evaluate a phishing detection model for Ethereum transactions using both explicit and implicit features. We will begin by collecting and curating transactional data from reliable sources such as Etherscan, ensuring a balanced dataset of phishing and benign transactions. The next step involves feature extraction, where we will focus on both explicit transactional details like gas usage, value, and timestamps, and implicit graph-based features that capture interactions within the transaction network.

5.3.1 Implementation

We plan to implement and experiment with GCN, to explore how each feature set affects model performance. Additionally, we will address challenges like class imbalance by employing techniques such as weighted loss functions or data resampling.

The evaluation of our models will be based on standard classification metrics, including accuracy, precision, recall, and F1-score, which will help us assess the effectiveness of each approach. These metrics will provide insights into how well the model identifies phishing activities, ensuring both robust performance and generalization in adversarial environments. By conducting these experiments, we aim to compare the efficacy of different feature sets and models, and refine our approach based on the outcomes.

References

[1] M. Abuhamad, A. Abusnaina, D. Nyang, and D. Mohaisen. Sensor-Based Continuous Authentication of Smartphones' Users Using Behavioral Biometrics: A Contemporary Survey. *IEEE Internet Things J.*, 8(1):65–84, 2021.

[2] A. Abusnaina, M. Abuhamad, H. Alasmary, A. Anwar, R. Jang, S. Salem, D. Nyang, and D. Mohaisen. DL-FHMC: Deep Learning-Based Fine-Grained Hierarchical Learning Approach for Robust Malware Classification. *IEEE Trans. Dependable Secur. Comput.*, 19(5):3432–3447, 2022.

[3] A. Abusnaina, A. Anwar, S. Alshamrani, A. Alabduljabbar, R. Jang, D. Nyang, and D. Mohaisen. Systematically Evaluating the Robustness of ML-based IoT Malware Detection Systems. In *25th International Symposium on Research in Attacks, Intrusions and Defenses, RAID*, pages 308–320. ACM, 2022.

[4] A. Abusnaina, R. Jang, A. Khormali, D. Nyang, and D. Mohaisen. DFD: Adversarial Learning-based Approach to Defend Against Website Fingerprinting. In *39th IEEE Conference on Computer Communications, INFOCOM*, pages 2459–2468. IEEE, 2020.

[5] A. Abusnaina, A. Khormali, H. Alasmary, J. Park, A. Anwar, and A. Mohaisen. Adversarial Learning Attacks on Graph-based IoT Malware Detection Systems. In *39th IEEE International Conference on Distributed Computing Systems, ICDCS*, pages 1296–1305. IEEE, 2019.

[6] A. Abusnaina, Y. Wu, S. S. Arora, Y. Wang, F. Wang, H. Yang, and D. Mohaisen. Adversarial Example Detection Using Latent Neighborhood Graph. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV*, pages 7667–7676. IEEE, 2021.

[7] A. Adeniran, M. Alkinoon, and D. Mohaisen. Understanding the Utilization of Cryptocurrency in the Metaverse and Security Implications. In *Computational Data and Social Networks - 12th International Conference, CSoNet*, volume 14479 of *Lecture Notes in Computer Science*, pages 268–281. Springer, 2023.

[8] A. Adeniran, K. Human, and D. Mohaisen. Dissecting the Infrastructure Used in Web-based Cryptojacking: A Measurement Perspective. In *International Conference Information Security Applications, WISA*, 2024.

[9] A. Adeniran and D. Mohaisen. Measuring Cryptocurrency Mining in Public Cloud Services: A Security Perspective. In *Computational Data and Social Networks - 11th*

International Conference, CSoNe, volume 13831 of *Lecture Notes in Computer Science*, pages 128–140. Springer, 2022.

[10] R. Agarwal, T. Thapliyal, and S. K. Shukla. Analyzing Malicious Activities and Detecting Adversarial Behavior in Cryptocurrency based Permissionless Blockchains: An Ethereum Usecase. *Distributed Ledger Technol. Res. Pract.*, 1(2):1–21, 2022.

[11] S. Al-Emari, M. Anbar, Y. K. Sanjalawe, and S. Manickam. A labeled transactions-based dataset on the ethereum network. In *Advances in Cyber Security - Second International Conference, ACeS*, volume 1347 of *Communications in Computer and Information Science*, pages 61–79. Springer, 2020.

[12] I. Alarab, S. Prakoonwit, and M. I. Nacer. Comparative Analysis Using Supervised Learning Methods for Anti-Money Laundering in Bitcoin. In *International Conference on Machine Learning Technologies, ICMLT*, pages 11–17. ACM, 2020.

[13] H. Alasmary, A. Abusnaina, R. Jang, M. Abuhamad, A. Anwar, D. Nyang, and D. Mohaisen. Soteria: Detecting adversarial examples in control flow graph-based malware classifiers. In *40th IEEE International Conference on Distributed Computing Systems, ICDCS*, pages 888–898. IEEE, 2020.

[14] H. Alasmary, A. Khormali, A. Anwar, J. Park, J. Choi, A. Abusnaina, A. Awad, D. Nyang, and A. Mohaisen. Analyzing and Detecting Emerging Internet of Things Malware: A Graph-Based Approach. *IEEE Internet Things J.*, 6(5):8977–8988, 2019.

[15] A. Alghuried, M. Alkinoon, M. Mohaisen, A. Wang, C. C. Zou, and D. Mohaisen. Blockchain security and privacy examined: Threats, challenges, applications, and tools. *The ACM Distributed Ledger Technologies: Research and Practice, ACM DLT*, 2024.

[16] A. Alghuried and D. Mohaisen. Simple Perturbations Subvert Ethereum Phishing Transactions Detection: An Empirical Analysis. In *The 25th International Conference Information Security Applications, WISA*, pages 1–12, 2024.

[17] A. Alghuried and D. Mohaisen. Phishing in wonderland: A systematic evaluation of the learning-based ethereum phishing transactions detection and pitfalls. In *IEEE Security and Privacy Symposium, IEEE S&P*, 2025.

[18] A. Aloosh and J. Li. Direct evidence of bitcoin wash trading. *Management Science*, 2024.

[19] F. A. Aponte-Novoa, A. L. S. Orozco, R. Villanueva-Polanco, and P. M. Wightman. The 51% Attack on Blockchains: A Mining Behavior Study. *IEEE Access*, 9:140549–140564, 2021.

[20] E. G. Barrantes, D. H. Ackley, T. S. Palmer, D. Stefanovic, and D. D. Zovi. Randomized instruction set emulation to disrupt binary code injection attacks. In *ACM Conference on Computer and Communications Security, CCS*, pages 281–289, 2003.

[21] M. Bartoletti, S. Lande, A. Loddo, L. Pompianu, and S. Serusi. Cryptocurrency Scams: Analysis and Perspectives. *IEEE Access*, 9:148353–148373, 2021.

[22] A. N. Bhagoji, W. He, B. Li, and D. Song. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *Computer Vision - ECCV*, volume 11216 of *Lecture Notes in Computer Science*, pages 158–174. Springer, 2018.

[23] M. Bhowmik, T. S. S. Chandana, and B. Rudra. Comparative study of machine learning algorithms for fraud detection in blockchain. In *International Conference on Computing Methodologies and Communication, ICCMC*, pages 539–541, 2021.

[24] S. Bibi. Cryptocurrency world identification and public concerns detection via social media: student research abstract. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC*, pages 550–552. ACM, 2019.

[25] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. In *Symposium on Security and Privacy, SP*, pages 104–121. IEEE, 2015.

[26] R. Böhme, N. Christin, B. Edelman, and T. Moore. Bitcoin: Economics, Technology, and Governance. *Journal of Economic Perspectives*, 29(2), 2015.

[27] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. Liang. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*, pages 11190–11201, 2019.

[28] F. Cartella, O. Anunciação, Y. Funabiki, D. Yamaguchi, T. Akishita, and O. Elshocht. Adversarial Attacks for Tabular Data: Application to Fraud Detection and Imbalanced Data. In *Proceedings of the Workshop on Artificial Intelligence Safety (SafeAI 2021)*, volume 2808. CEUR-WS.org, 2021.

[29] F. Cernera, M. L. Morgia, A. Mei, and F. Sassi. Token Spammers, Rug Pulls, and Sniper Bots: An Analysis of the Ecosystem of Tokens in Ethereum and in the Binance Smart Chain (BNB). In *32nd USENIX Security Symposium*, pages 3349–3366. USENIX Association, 2023.

[30] W. Chang, A. Mohaisen, A. Wang, and S. Chen. Measuring Botnets in the Wild: Some New Trends. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, ASIA CCS,*, pages 645–650. ACM, 2015.

[31] L. Chen, J. Peng, Y. Liu, J. Li, F. Xie, and Z. Zheng. Phishing Scams Detection in Ethereum Transaction Network. *ACM Trans. Internet Techn.*, 21(1):10:1–10:16, 2021.

[32] T. Chen, Y. Zhang, Z. Li, X. Luo, T. Wang, R. Cao, X. Xiao, and X. Zhang. TokenScope: Automatically Detecting Inconsistent Behaviors of Cryptocurrency Tokens in Ethereum. In *ACM Conference on Computer and Communications Security, CCS*, pages 1503–1520, 2019.

[33] W. Chen, X. Guo, Z. Chen, Z. Zheng, and Y. Lu. Phishing Scam Detection on Ethereum: Towards Financial Security for Blockchain Ecosystem. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI*, pages 4506–4512. ijcai.org, 2020.

[34] G. Cola, M. Mazza, and M. Tesconi. From Tweet to Theft: Tracing the Flow of Stolen Cryptocurrency. In *CEUR Workshop*, volume 3488. CEUR-WS.org, 2023.

[35] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. Robustbench: a standardized adversarial robustness benchmark. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks*, 2021.

[36] P. de Juan Fidalgo, C. Camara, and P. Peris-Lopez. Generation and Classification of Illicit Bitcoin Transactions. In *Proceedings of the International Conference on Ubiquitous Computing & Ambient Intelligence, UCAmI*, volume 594 of *Lecture Notes in Networks and Systems*, pages 1086–1097. Springer, 2022.

[37] Y. Ding, L. Wang, H. Zhang, J. Yi, D. Fan, and B. Gong. Defending against adversarial attacks using random forest. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR*, pages 105–114. Computer Vision Foundation / IEEE, 2019.

[38] J. Dong, Z. Qin, Z. Fang, X. Chen, Z. Huang, H. Guo, R. Liu, C. Turkay, and S. Chen. Visual Analytics for Phishing Scam Identification in Blockchain Transactions with Multiple Model Comparison. In *International Symposium on Visual Information Communication and Interaction, VINCI*, pages 17:1–17:9. ACM, 2023.

[39] M. Eshghie, C. Artho, and D. Gurov. Dynamic Vulnerability Detection on Smart Contracts Using Machine Learning. In *Evaluation and Assessment in Software Engineering, EASE*, pages 305–312. ACM, 2021.

[40] B. Fu, X. Yu, and T. Feng. CT-GCN: a phishing identification model for blockchain cryptocurrency transactions. *Int. J. Inf. Sec.*, 21(6):1223–1232, 2022.

[41] D. Gibert, L. Demetrio, G. Zizzo, Q. Le, J. Planes, and B. Biggio. Certified adversarial robustness of machine learning-based malware detectors via (de)randomized smoothing. volume abs/2405.00392, 2024.

[42] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR*, 2015.

[43] F. Gritti, N. Ruaro, R. McLaughlin, P. Bose, D. Das, I. Grishchenko, C. Kruegel, and G. Vigna. Confusum Contractum: Confused Deputy Vulnerabilities in Ethereum Smart Contracts. In *USENIX Security Symposium*, pages 1793–1810, 2023.

[44] M. U. Hassan, M. H. Rehmani, and J. Chen. Anomaly Detection in Blockchain Networks: A Comprehensive Survey. *IEEE Commun. Surv. Tutorials*, 25(1):289–318, 2023.

[45] B. He, Y. Chen, Z. Chen, X. Hu, Y. Hu, L. Wu, R. Chang, H. Wang, and Y. Zhou. TxPhishScope: Towards Detecting and Understanding Transaction-based Phishing on Ethereum. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS*, pages 120–134. ACM, 2023.

[46] H. Heo, S. Woo, T. Yoon, M. S. Kang, and S. Shin. Partitioning Ethereum without Eclipsing It. In *Network and Distributed System Security Symposium, NDSS*. The Internet Society, 2023.

[47] L. Hornuf, P. P. Momtaz, R. J. Nam, and Y. Yuan. Cybercrime on the ethereum blockchain. 2023.

[48] M. G. Ismail, M. A. E. Ghany, and M. A. Salem. Enhanced Recursive Feature Elimination for IoT Intrusion Detection Systems. In *International Conference on Microelectronics, ICM*, pages 193–196. IEEE, 2022.

[49] P. E. Johnson, S. Grazioli, K. Jamal, and R. G. Berryman. Detecting deception: adversarial problem solving in a low base-rate world. *Cogn. Sci.*, 25(3):355–392, 2001.

[50] A. H. H. Kabla, M. Anbar, S. Manickam, and S. Karuppayah. Eth-PSD: A Machine Learning-Based Phishing Scam Detection Approach in Ethereum. *IEEE Access*, 10:118043–118057, 2022.

[51] O. Konashevych and O. Khovayko. Randpay: The technology for blockchain micropayments and transactions which require recipient's consent. *Comput. Secur.*, 96:101892, 2020.

[52] L. P. Krishnan, I. Vakilinia, S. Reddivari, and S. Ahuja. Scams and Solutions in Cryptocurrencies - A Survey Analyzing Existing Machine Learning Models. *Inf.*, 14(3):171, 2023.

[53] J. Krupp and C. Rossow. teEther: Gnawing at Ethereum to Automatically Exploit Smart Contracts. In *USENIX Security Symposium*, pages 1317–1333, 2018.

[54] S. S. Kushwaha, S. Joshi, D. Singh, M. Kaur, and H. Lee. Systematic Review of Security Vulnerabilities in Ethereum Blockchain Smart Contract. *IEEE Access*, 10:6605–6621, 2022.

[55] B. Lal, R. Agarwal, and S. K. Shukla. Understanding Money Trails of Suspicious Activities in a cryptocurrency-based Blockchain. *CoRR*, abs/2108.11818, 2021.

[56] J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1):2, 2007.

[57] D. Li, D. Chen, J. Goh, and S. Ng. Anomaly detection with generative adversarial networks for multivariate time series. *CoRR*, abs/1809.04758, 2018.

[58] K. Li, Y. Wang, and Y. Tang. DETER: Denial of Ethereum Txpool sERvices. In *ACM Conference on Computer and Communications Security, CCS*, pages 1645–1667, 2021.

[59] P. Li, Y. Xie, X. Xu, J. Zhou, and Q. Xuan. Phishing Fraud Detection on Ethereum Using Graph Neural Network. In *International Conference on Blockchain and Trustworthy Systems, BlockSys*, pages 362–375. Springer, 2022.

[60] S. Li, G. Gou, C. Liu, C. Hou, Z. Li, and G. Xiong. TTAGN: Temporal Transaction Aggregation Graph Network for Ethereum Phishing Scams Detection. In *WWW '22: The ACM Web Conference 2022*, pages 661–669. ACM, 2022.

[61] S. Li, G. Gou, C. Liu, G. Xiong, Z. Li, J. Xiao, and X. Xing. TGC: Transaction Graph Contrast Network for Ethereum Phishing Scam Detection. In *Annual Computer Security Applications Conference, ACSAC*, pages 352–365. ACM, 2023.

[62] S. Li, R. Wang, H. Wu, S. Zhong, and F. Xu. SIEGE: Self-Supervised Incremental Deep Graph Learning for Ethereum Phishing Scam Detection. In *Proceedings of the 31st ACM International Conference on Multimedia, MM*, pages 8881–8890. ACM, 2023.

[63] X. Li, Y. Chen, Y. He, and H. Xue. Advknn: Adversarial attacks on k-nearest neighbor classifiers with approximate gradients. *CoRR*, abs/1911.06591, 2019.

[64] Z. Lin, X. Xiao, G. Hu, B. Zhang, Q. Liu, and X. Luo. Phish2vec: A Temporal and Heterogeneous Network Embedding Approach for Detecting Phishing Scams on Ethereum. In *20th Annual IEEE International Conference on Sensing, Communication, and Networking, SECON*. IEEE, 2023.

[65] L. Liu, W. Tsai, M. Z. A. Bhuiyan, H. Peng, and M. Liu. Blockchain-enabled fraud discovery through abnormal smart contract detection on Ethereum. *Future Gener. Comput. Syst.*, 128:158–166, 2022.

[66] Y. Lou, Y. Zhang, and S. Chen. Ponzi Contracts Detection Based on Improved Convolutional Neural Network. In *International Conference on Services Computing, SCC*, pages 353–360. IEEE, 2020.

[67] A. Mohaisen and O. Alrawi. Unveiling Zeus: automated classification of malware samples. In *22nd International World Wide Web Conference, WWW*, pages 829–832. International World Wide Web Conferences Steering Committee / ACM, 2013.

[68] A. Mohaisen, O. Alrawi, and M. Mohaisen. AMAL: High-fidelity, behavior-based automated malware analysis and classification. *Comput. Secur.*, 52:251–266, 2015.

[69] A. Mozo, Á. González-Prieto, A. P. Perales, S. G. Canaval, and E. Talavera. Synthetic flow-based cryptomining attack generation through generative adversarial networks. *CoRR*, abs/2107.14776, 2021.

[70] N. S. M. Nafis and S. Awang. An Enhanced Hybrid Feature Selection Technique Using Term Frequency-Inverse Document Frequency and Support Vector Machine-Recursive Feature Elimination for Sentiment Classification. *IEEE Access*, 9:52177–52192, 2021.

[71] N. Narodytska and S. P. Kasiviswanathan. Simple Black-Box Adversarial Attacks on Deep Neural Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR*, pages 1310–1318. IEEE Computer Society, 2017.

[72] V. C. Oliveira, J. A. Valadares, J. E. de Azevedo Sousa, A. B. Vieira, H. S. Bernardino, S. M. Villela, and G. D. Gonçalves. Analyzing transaction confirmation in ethereum

using machine learning techniques. *SIGMETRICS Perform. Evaluation Rev.*, 48(4):12–15, 2021.

[73] M. Pacheco, G. A. Oliva, G. K. Rajbahadur, and A. E. Hassan. Is My Transaction Done Yet? An Empirical Study of Transaction Processing Times in the Ethereum Blockchain Platform. *ACM Trans. Softw. Eng. Methodol.*, 32(3):59:1–59:46, 2023.

[74] G. Palaiokrassas, S. Scherrers, I. Ofeidis, and L. Tassiulas. Leveraging Machine Learning for Multichain DeFi Fraud Detection. *CoRR*, abs/2306.07972, 2023.

[75] N. Papernot, P. D. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. In *IEEE Symposium on Security and Privacy, SP*, pages 582–597. IEEE Computer Society, 2016.

[76] J. Park, A. Khormali, M. Mohaisen, and A. Mohaisen. Where Are You Taking Me? Behavioral Analysis of Open DNS Resolvers. In *49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN*, pages 493–504. IEEE, 2019.

[77] P. Pinio, R. Batko, and D. Lewicka. Between Theory and Value Transactions: A Multifaceted Exploration of Relevance and Resilience of Decentralised Autonomous Organisations. In *Proceedings of the 2024 7th International Conference on Software Engineering and Information Management, ICSIM*, pages 42–48. ACM, 2024.

[78] M. Platt and P. McBurney. Sybil in the Haystack: A Comprehensive Review of Blockchain Consensus Mechanisms in Search of Strong Sybil Attack Resistance. *Algorithms*, 16(1):34, 2023.

[79] N. R. Pradhan, A. P. Singh, S. Verma, Kavita, M. Wozniak, J. Shafi, and M. F. Ijaz. A blockchain based lightweight peer-to-peer energy trading framework for secured high throughput micro-transactions. *Scientific Reports*, 12(1):14523, 2022.

[80] E. Rabieinejad, A. Yazdinejad, R. M. Parizi, and A. Dehghantanha. Generative Adversarial Networks for Cyber Threat Hunting in Ethereum Blockchain. *Distributed Ledger Technol. Res. Pract.*, 2(2):1–19, 2023.

[81] R. S. Rao and A. R. Pais. Detection of phishing websites using an efficient feature-based machine learning framework. *Neural Comput. Appl.*, 31(8):3851–3873, 2019.

[82] M. Rodler, W. Li, G. O. Karame, and L. Davi. EVMPatch: Timely and Automated Patching of Ethereum Smart Contracts. In *USENIX Security Symposium*, pages 1289–1306, 2021.

[83] M. Saad, A. Anwar, S. Ravi, and D. Mohaisen. Revisiting Nakamoto Consensus in Asynchronous Networks: A Comprehensive Analysis of Bitcoin Safety and Chain-Quality. In *ACM Conference on Computer and Communications Security, CCS*, pages 988–1005, 2021.

[84] M. Saad, A. Anwar, S. Ravi, and D. Mohaisen. Revisiting Nakamoto Consensus in Asynchronous Networks: A Comprehensive Analysis of Bitcoin Safety and Chain Quality. *IEEE/ACM Trans. Netw.*, 32(1):844–858, 2024.

[85] M. Saad, S. Chen, and D. Mohaisen. Root Cause Analyses for the Deteriorating Bitcoin Network Synchronization. In *41st IEEE International Conference on Distributed Computing Systems, ICDCS*, pages 239–249. IEEE, 2021.

[86] M. Saad, S. Chen, and D. Mohaisen. SyncAttack: Double-spending in Bitcoin Without Mining Power. In *ACM Conference on Computer and Communications Security, CCS*, pages 1668–1685, 2021.

[87] M. Saad, V. Cook, L. N. Nguyen, M. T. Thai, and A. Mohaisen. Partitioning Attacks on Bitcoin: Colliding Space, Time, and Logic. In *International Conference on Distributed Computing Systems, ICDCS*, pages 1175–1187. IEEE, 2019.

[88] M. Saad, V. Cook, L. N. Nguyen, M. T. Thai, and D. Mohaisen. Exploring Partitioning Attacks on the Bitcoin Network. *IEEE/ACM Trans. Netw.*, 30(1):202–214, 2022.

[89] M. Saad and D. Mohaisen. Three Birds with One Stone: Efficient Partitioning Attacks on Interdependent Cryptocurrency Networks. In *44th IEEE Symposium on Security and Privacy, SP*, pages 111–125. IEEE, 2023.

[90] M. Saad, L. Njilla, C. A. Kamhoua, J. Kim, D. Nyang, and A. Mohaisen. Mempool optimization for Defending Against DDoS Attacks in PoW-based Blockchain Systems. In *IEEE International Conference on Blockchain and Cryptocurrency, ICBC*, pages 285–292. IEEE, 2019.

[91] M. Saad, J. Spaulding, L. Njilla, C. A. Kamhoua, S. Shetty, D. Nyang, and D. Mohaisen. Exploring the Attack Surface of Blockchain: A Comprehensive Survey. *IEEE Commun. Surv. Tutorials*, 22(3):1977–2008, 2020.

[92] S. Sayeed, H. Marco-Gisbert, and T. Caira. Smart Contract: Attacks and Protections. *IEEE Access*, 8:24416–24427, 2020.

[93] C. Schneidewind, I. Grishchenko, M. Scherer, and M. Maffei. eThor: Practical and Provably Sound Static Analysis of Ethereum Smart Contracts. In *ACM Conference on Computer and Communications Security, CCS*, pages 621–640, 2020.

[94] F. Shen, J. D. Vecchio, A. Mohaisen, S. Y. Ko, and L. Ziarek. Android Malware Detection Using Complex-Flows. *IEEE Trans. Mob. Comput.*, 18(6):1231–1245, 2019.

[95] S. H. Silva and P. Najafirad. Opportunities and challenges in deep learning adversarial robustness: A survey. *CoRR*, abs/2007.00753, 2020.

[96] H. J. Singh and A. S. Hafid. Prediction of transaction confirmation time in ethereum blockchain using machine learning. In *Blockchain and Applications - International Congress, BLOCKCHAIN*, volume 1010 of *Advances in Intelligent Systems and Computing*, pages 126–133. Springer, 2019.

[97] S. So, M. Lee, J. Park, H. Lee, and H. Oh. VERISMART: A Highly Precise Safety Verifier for Ethereum Smart Contracts. In *Symposium on Security and Privacy, SP*, pages 1678–1694. IEEE, 2020.

[98] S. Somraaj. Hackers Nab $8M$ in Ethereum via Uniswap Phishing Attack, 2022.

[99] D. Stutz, M. Hein, and B. Schiele. Disentangling adversarial robustness and generalization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 2019.

[100] P. Subba Narasimha, B. Arinze, and M. Anandarajan. The predictive accuracy of artificial neural networks and multiple regression in the case of skewed data: Exploration of some issues. *Expert systems with Applications*, 19(2):117–123, 2000.

[101] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR*, 2014.

[102] K. Thomas, D. Y. Huang, D. Y. Wang, E. Bursztein, C. Grier, T. Holt, C. Kruegel, D. McCoy, S. Savage, and G. Vigna. Framing Dependencies Introduced by Underground Commoditization. In *Annual Workshop on the Economics of Information Security, WEIS*, 2015.

[103] M. Vasek and T. Moore. There's No Free Lunch, Even Using Bitcoin: Tracking the Popularity and Profits of Virtual Currency Scams. In *Financial Cryptography and Data Security - 19th International Conference, FC*, volume 8975 of *Lecture Notes in Computer Science*, pages 44–61. Springer, 2015.

[104] D. Ververidis and C. Kotropoulos. sequential forward feature selection with low computational cost. In *European Signal Processing Conference, EUSIPCO*, pages 1–4. IEEE, 2005.

[105] D. Vos and S. Verwer. Efficient training of robust decision trees against adversarial examples. In *Proceedings of the 38th International Conference on Machine Learning, ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 10586–10595. PMLR, 2021.

[106] A. Wang, W. Chang, S. Chen, and A. Mohaisen. Delving Into Internet DDoS Attacks by Botnets: Characterization and Analysis. *IEEE/ACM Trans. Netw.*, 26(6):2843–2855, 2018.

[107] A. Wang, A. Mohaisen, and S. Chen. An Adversary-Centric Behavior Modeling of DDoS Attacks. In *37th IEEE International Conference on Distributed Computing Systems, ICDCS*, pages 1126–1136. IEEE Computer Society, 2017.

[108] J. Wang, P. Chen, X. Xu, J. Wu, M. Shen, Q. Xuan, and X. Yang. TSGN: Transaction Subgraph Networks Assisting Phishing Detection in Ethereum. *CoRR*, abs/2208.12938, 2022.

[109] W. Wang, W. Huang, Z. Meng, Y. Xiong, F. Miao, X. Fang, C. Tu, and R. Ji. Automated Inference on Financial Security of Ethereum Smart Contracts. In *USENIX Security Symposium*, pages 3367–3383, 2023.

[110] W. Wang, J. Song, G. Xu, Y. Li, H. Wang, and C. Su. ContractWard: Automated Vulnerability Detection Models for Ethereum Smart Contracts. *IEEE Trans. Netw. Sci. Eng.*, 8(2):1133–1144, 2021.

[111] P. Wei, Q. Yuan, and Y. Zheng. Security of the Blockchain Against Long Delay Attack. In *International Conference on Advances in Cryptology, ASIACRYPT*, pages 250–275. Springer, 2018.

[112] H. Wen, J. Fang, J. Wu, and Z. Zheng. Transaction-Based Hidden Strategies against General Phishing Detection Framework on Ethereum. In *International Symposium on Circuits and Systems, ISCAS*, pages 1–5. IEEE, 2021.

[113] T. Wen, Y. Xiao, A. Wang, and H. Wang. A novel hybrid feature fusion model for detecting phishing scam on Ethereum using deep neural network. *Expert Syst. Appl.*, 211:118463, 2023.

[114] C. Wronka. Money laundering through cryptocurrencies-analysis of the phenomenon and appropriate prevention measures. *Journal of Money Laundering Control*, 25(1):79–94, 2022.

[115] J. Wu, J. Liu, Y. Zhao, and Z. Zheng. Analysis of cryptocurrency transactions from a network perspective: An overview. *J. Netw. Comput. Appl.*, 190:103139, 2021.

[116] J. Wu, Q. Yuan, D. Lin, W. You, W. Chen, C. Chen, and Z. Zheng. Who Are the Phishers? Phishing Scam Detection on Ethereum via Network Embedding. *IEEE Trans. Syst. Man Cybern. Syst.*, 52(2):1156–1166, 2022.

[117] C. Xie, Y. Wu, L. van der Maaten, A. L. Yuille, and K. He. Feature denoising for improving adversarial robustness. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 501–509. Computer Vision Foundation / IEEE, 2019.

[118] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang. Machine Learning and Deep Learning Methods for Cybersecurity. *IEEE Access*, 6:35365–35381, 2018.

[119] A. Yaish, G. Stern, and A. Zohar. Uncle Maker: (Time)Stamping Out The Competition in Ethereum. In *ACM Conference on Computer and Communications Security, CCS*, pages 135–149, 2023.

[120] J. Yang, T. Li, G. Liang, Y. Wang, T. Gao, and F. Zhu. Spam transaction attack detection model based on GRU and wgan-div. *Comput. Commun.*, 161:172–182, 2020.

[121] M. Yao, R. Zhang, H. Xu, S. Chou, V. C. Paturi, A. K. Sikder, and B. Saltaformaggio. Pulling Off The Mask: Forensic Analysis of the Deceptive Creator Wallets Behind Smart Contract Fraud. In *IEEE Symposium on Security and Privacy, SP*, pages 2236–2254. IEEE, 2024.

[122] C. Zhang, C. Wu, and X. Wang. Overview of Blockchain Consensus Mechanism. In *International Conference on Big Data Engineering, BDE*, pages 7–12. ACM, 2020.

[123] P. Zhang, D. C. Schmidt, J. White, and A. Dubey. Chapter Seven - Consensus mechanisms and information security technologies. *Adv. Comput.*, 115:181–209, 2019.

[124] Z. Zhang, T. He, K. Chen, B. Zhang, Q. Wang, and L. Yuan. Phishing Node Detection in Ethereum Transaction Network Using Graph Convolutional Networks. *Applied Sciences*, 13(11):6430, 2023.

[125] J. Zhao, R. Y. K. Lau, W. Zhang, K. Zhang, X. Chen, and D. Tang. Extracting and reasoning about implicit behavioral evidences for detecting fraudulent online transactions in e-Commerce. *Decis. Support Syst.*, 86:109–121, 2016.

[126] X. Zhou, W. Yang, and X. Tian. Detecting Phishing Accounts on Ethereum Based on Transaction Records and EGAT. *Electronics*, 2023.

[127] X. Zhou, W. Yang, and X. Tian. Detecting Phishing Accounts on Ethereum Based on Transaction Records and EGAT. *Electronics*, 12(4):993, 2023.

[128] X. Zhou, W. Yang, and X. Tian. Detecting Phishing Accounts on Ethereum Based on Transaction Records and EGAT. *Electronics*, 12(4), 2023.

[129] Y. Zhou, D. Kumar, S. Bakshi, J. Mason, A. Miller, and M. D. Bailey. Erays: Reverse Engineering Ethereum's Opaque Smart Contracts. In *USENIX Security Symposium*, pages 1371–1385, 2018.

[130] F. Zola, J. L. Bruse, X. E. Barrio, M. Galar, and R. O. Urrutia. Generative adversarial networks for bitcoin data augmentation. In *2nd Conference on Blockchain Research & Applications for Innovative Networks and Services*, pages 136–143. IEEE, 2020.