

Do classifiers and compression agree? A leakage-aware comparison of Sybil detection on Ethereum

Michał Bartnicki, and Jarosław A. Chudziak

Abstract—Ethereum projects attract new users with airdrops—public giveaways of crypto tokens. This is an opportunity for Sybil bots—wallets operated by the same entity to abuse the rewards system. There are established Sybil detection methods, but every detector method family has its price: classifiers have high labeling and retraining costs; graph-based methods can be evaded by rerouting transactions through bridges or centralized exchanges; and compression-based techniques are training-free but costly to scale. Picking the correct method may be difficult because these methods have been studied in isolation. In this paper, we ask the following questions: which family of techniques is best for leakage-aware classification; which family is best for training-free Sybil candidate discovery; and do the two types of audit regimes—closed-set classification and open-world discovery—detect the same Sybils? Our approach is to combine tabular, sequential, and compression-based techniques in one evaluation scheme: our Transaction Grammar decodes the rhythm, execution structure, and intent of 14161 Ethereum wallets participating in the Hop airdrop, with all class-exclusive counterparties filtered out via the Blind-Spot protocol. We contribute a comparison of tabular (XGBoost, SVMs), sequential (Transformers, BiLSTMs), and compression-based (NCD) Sybil detectors in one leakage-controlled setting, and a two-stage pipeline that discovers more of a Sybil campaign than either technique used separately.

Keywords—Blockchain analytics; Ethereum; Sybil detection; airdrop farming; transaction grammar; normalized compression distance; data leakage

I. INTRODUCTION

SOME token airdrops and on-chain governance grant value and votes to addresses that can be created at low cost, making them natural targets for Sybil attacks, in which a single entity controls many pseudonymous wallets [1]. This is more than theoretical: airdrop campaigns allocate millions of dollars; it is known that farming groups take significant portions of these rewards [2]; and protocol designers fight back through resource-intensive audits, including community audits of the Hop Protocol airdrop campaign [3]. Meanwhile, the actions of any wallet on Ethereum leave an extensive trace because, in addition to every transaction, Ethereum archives its execution trace as well [4]. That is, how an entity acts on Ethereum—its timing, its call graph, and the functions

it invokes—can be observed on a large scale throughout the decentralized finance (DeFi) ecosystem.

At various stages, an audit team suspecting Sybil farming will require different things from its detection tool: to classify every wallet as one of the known actor classes, or to grow a few discovered Sybils into their campaigns. Not a single method family is suitable for both purposes. While supervised classifiers work well for closed-set evaluation, they require labeled data and retraining. Sequence models are able to recognize complex behavior patterns but require a high computational expense. Graph techniques require financial connections, but routing through the centralized exchanges breaks them. Compression-based similarity recognizes candidates without training but is computationally demanding when performed globally. And, as it turns out from this paper, these families have never been compared directly on the same data.

Which method, then, should the auditing team adopt? We ask ourselves these three questions, formulated as RQ1–RQ3 in Section III-A: which method is strongest for leakage-aware closed-set classification; which method is strongest for Sybil candidate discovery; and do classification and discovery find the same Sybils, or different ones? We address these questions through measurements, combining two directions from our previous research – leakage-aware classification and compression-based open-world discovery [5], [6] into a single evaluation: one population of 14161 Ethereum wallets to label, one Blind-Spot protocol eliminating class-exclusive counterparties prior to encoding, and one common Transaction Grammar encoding transaction’s rhythm, execution and purpose. Three families consume the grammar as aggregate features (XGBoost, SVMs, FT-Transformer), as ordered sequences (Transformers, BiLSTM), and as compressible strings (gzip NCD), evaluated across both audit regimes (Fig. 1).

Our paper makes the following contributions:

- First comparison between the three types of methods to detect Sybils – table-based (XGBoost, SVMs, FT-Transformer), sequence-based (Transformers, BiLSTMs), and compression-based (NCD) – on one dataset, with one type of representations and one leakage control technique;
- Two-stage audit pipeline using complementarity – classifier-based selection and then NCD-based seed expansion detects 86.0% of labeled Sybils, while classifier alone detects only 67.8% (Section IV-E).

M. Bartnicki and J. A. Chudziak are with the Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, Poland (e-mail: michal.bartnicki.stud@pw.edu.pl; jaroslaw.chudziak@pw.edu.pl).



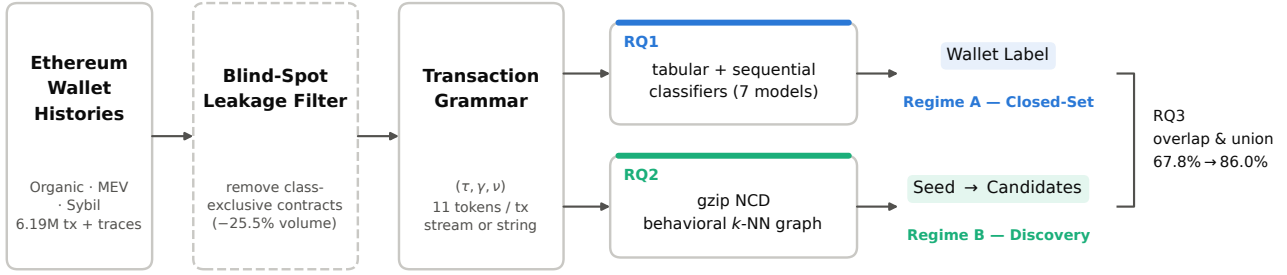


Fig. 1. Overview of the framework: one Blind-Spot-filtered population and one Transaction Grammar feed two operational regimes – closed-set classification and open-world discovery – whose recovered Sybil sets are directly compared (research questions RQ1–RQ3, Section III-A).

II. RELATED WORK

This article unifies two strands of our prior work that approached the same wallet population from opposite directions. The first strand [5] treated actor classification as a leakage-aware modeling decision: it introduced the Blind-Spot protocol together with an early form of the Transaction Grammar, and found that once high-signal contracts are removed, XGBoost stays ahead of Transformer and BiLSTM sequence models. The second strand [6] discarded labels altogether and treated Sybil detection as open-world candidate discovery, measuring behavioral similarity with gzip-based NCD over grammar strings and expanding candidates from confirmed seed wallets. Each study answered its own question on its own dataset variant, protocol, and metric suite; neither could say when a supervised classifier should be preferred over training-free retrieval, which is the question an audit team actually faces. The present article makes the two strands directly comparable – one population, one leakage protocol, one representation – and measures where their outputs overlap and where they complement each other (Section IV-E).

Sybil attacks, characterized by one operator controlling several identities, have been proposed by Douceur [1] and are often tackled in blockchain environment using graph-based analysis. Graph neural networks have been used for classification on interaction graphs [7], [8], including specialized models for Sybil attacks, assessed on airdrop data [9]. Our labels come from production airdrop audits, that also base on the funding trails and shared interactions [2]. The weakness of such methodology is that links are the easiest ones to break for the attacker: passing the funds via centralized exchanges disrupts the funding graph without affecting the behavior. Therefore, we test the approaches that do not use any financial links and avoid comparing graph-based approaches: our labels themselves are based on the links analysis, and hence the graph detector would partially reproduce the supervision, which means circularity in any potential benefit (Section III-A).

Tabular pipelines that use feature engineering and gradient-boosted trees have remained a staple practice in real-world workflows [10]. A newer line of research utilizes sequence encoders inspired by the success of language models, e.g., Transformers [11], as well as pre-trained encoders of transactions like BERT4ETH [12]. RQ1 of our paper seeks to redefine the contrast between tabular and sequential approaches, by

controlling for potential leakage and keeping a fixed representation (Section IV-C). Formal open-set recognition, which equips classifiers with a rejection option [13], is complementary to both of our regimes and deferred to future work (Section III-A).

Whether deep learning models are superior to gradient-boosted trees on tabular problems is an ongoing debate that appears to have been solved: on most tabular problems, they are not [14]–[17]. Tabular attention models purposefully designed for this specific problem, like the FT-Transformer model [18]–[20], narrow the performance difference, but fail to close it – similarly to cryptocurrency price prediction, where the use of selected features outperforms full sequence modeling [21], and long sequence architectures like Informer, used for financial option pricing [22], [23], target the sequence length aspect of the problem instead of the tabular model gap. Our architecture draws on that literature by using their control condition: the FT-Transformer baseline consumes the byte-identical feature vector as XGBoost, so the tree-versus-attention effect is separated from the aggregate-versus-sequential one (Section IV-C), and the remaining case for sequence models must rest on transaction order.

Normalized compression distance estimates the uncomputable normalized information distance by replacing Kolmogorov complexity with a real compressor [24]; the measure has no parameters, does not require training, and is independent of the token alphabet used. Compressor selection is known to be significant in cases when input data exceeds the search window of the compressor [25]; the grammar strings generated in Section IV-D fall into the kilobyte scale, well within this window, hence we use gzip throughout. The computation of exact pairwise NCD between population elements has quadratic complexity; approximate alternatives such as MinHash and locality-sensitive hashing [26] exist but are not evaluated here.

Leakage in data mining was formalized in [27]; TESSER-ACT revealed how bias from experiments, specifically spatio-temporal bias in that case, consistently overrates malware classifiers [28]; and shortcut learning refers to the tendency of models to learn decision-making rules that do not hold out-of-distribution [29]. The Blind-Spot protocol (Section III-D) is an example of how this applies to counterparty leakage in actor classification.

III. PROBLEM AND APPROACH

With the problem positioned against prior work, this section fixes the four elements every later result rests on. The research problem and its two operational regimes, along with the three research questions, are defined in Section III-A; the Transaction Grammar is formally introduced in Section III-B; and the three technique families that use it are described in Section III-C. Section III-D specifies the leakage control applied before any evaluation.

A. Problem Formulation

We analyze the wallet population operating in proximity to the blockchain incentive event of an airdrop. There are three kinds of actors in it. Organic actors are independent individuals pursuing their own goals. MEV bots are automated programs that benefit from transaction ordering – in our dataset, arbitrage bots – and are machine-like but benign for the purposes of an airdrop audit. Sybil bots are large numbers of wallets operated by one operator and programmed to mimic organic behavior in order to increase their eligibility. Thus, the goal of the task is not the detection of bots versus humans: automation in itself is not the offense, and a detector that detects only machine-like behavior will include MEV bots among its alerts, which misses the point of the audit. We need a three-way separation: separate the coordinated Sybils from the Organic actors and from benign automation. The operationally difficult boundary lies between Sybils and the Organic actors they imitate.

Detection is used in two distinct operational regimes that are confused by single-number comparisons. In the regime of closed-set classification (A), the auditor possesses labeled training examples for each known class and seeks to assign a label to all wallets in the population. In the regime of open-world candidate discovery (B), the auditor assumes neither a trained model nor a label set; starting with one or a few confirmed wallets, the auditor expands toward behaviorally similar candidates, or asks which wallets behave like near-clones of one another. The regimes differ not only in supervision and output, but also, as argued in this paper, in the most suitable method family, and this is the reason why the selection of a detection method should start from the application task rather than from a leaderboard.

Formally, the wallet w is represented via its history of transactions and encoded as the behavioral sequence $S_w = (x_1, \dots, x_{L_w})$ over the token alphabet defined in Section III-B; the class space is $\mathcal{Y} = \text{Organic, MEV Bot, Sybil Bot}$. In regime A, the task is to learn a classifier $f(S_w) \in \mathcal{Y}$ based on labeled examples, where the evaluation metrics are the Matthews correlation coefficient (MCC, primary), macro-F1, accuracy, and class-specific recall. In regime B, we need only a distance function on the sequence space, $d(S_i, S_j)$, which induces an implicit behavioral graph over the population of wallets (equivalently, a k -nearest-neighbor retrieval space). Discovery queries find the k nearest neighbors of a seed sequence and are evaluated using leave-one-out 1-NN accuracy and the Sybil purity and recall of top- k neighborhoods. Splits, seeds, and imbalance handling are specified in Section IV-A.

The evaluation is organized around three research questions:

- **RQ1 (Classification)**. With label-leaking shortcuts removed, which method family classifies the three actor classes best? (Section IV-C)
- **RQ2 (Discovery)**. Can compression-based similarity recover coordinated Sybils with no labels and no training? (Section IV-D)
- **RQ3 (Complementarity)**. Do the two regimes recover the same Sybils, or does each catch wallets the other misses? (Section IV-E)

Section IV-B comes before all three questions with an analysis without a model that shows that the shared representation carries class information.

The study was constrained by five scope decisions. First, the evaluation is leakage-aware: each method is evaluated following the removal of counterparty shortcuts by the Blind-Spot protocol in Section III-D. Second, no method relies on financial links: each wallet is represented only by its behavioral signature, without using funding connections with other wallets. This keeps the framework applicable even when funding is routed through exchanges. We also avoid circularity: our ground truth – the Hop Sybil labels – was derived through link-based investigation. Therefore, link-based detection approaches, including graph neural networks, are excluded from consideration rather than given an unfair advantage by construction (Section II). Third, regime B is open-world, not open-set: while the discovery task deals with an unlabeled population, there is no formal option to reject unknown classes during classification; we leave calibrated open-set recognition for future work. Fourth, all results are within-campaign: the data comes from one airdrop campaign, and cross-campaign transfer is neither tested nor claimed. Fifth, the current study neither tests the methods under adversarial camouflage nor assesses deployment costs; RQ1–RQ3 establish the basis of comparison and the complementarity findings using only Blind-Spot-filtered data, with robustness and deployment left for future research.

B. Transaction Grammar

Each approach in this paper uses a common representation, defined by a common theme: capture behavior, not identity. The raw data contains information about counterparty identity, but Section IV-A illustrates that identities may reveal the class label in their own right. The Transaction Grammar consequently generalizes each transaction to a set of time and execution shape information, as well as declared function, and does not name any addresses. Identity can still influence the representation indirectly – a contract specific to the class provides a unique fingerprint of selectors and construction – and that is the reason why the Blind-Spot protocol in Section III-D further discards interactions with the high-signal contracts prior to encoding. It is precisely the fact that the common alphabet is used that makes the analysis of Section III-C possible.

Each transaction x_t is represented by a three-part tuple $x_t = (\tau_t, \gamma_t, \nu_t)$, encoding its rhythm, structure, and intent. The rhythm token τ_t represents the discrete inter-arrival time

Δt between transactions in sixteen different tokens: a special START token together with fifteen logarithmically spaced bins, from same-block ($\Delta t = 0$) to seconds, minutes, hours, and days up to dormancy ($\Delta t > 30$ d). The structure token γ_t represents the execution pattern of the transaction as nine features of the Ethereum Virtual Machine (EVM) call tree, i.e., number of internal calls, max call depth, max branching factor, the number of distinct function selectors used in the call tree, the number of reverted calls, a flag indicating root call success, and the number of `delegatecall`, `staticcall`, and contract creation calls, each of which is discretized into four to six coarse buckets (transactions without trace have a special token). The intent token ν_t is the function selector of the calldata, i.e., a 4-byte prefix of the calldata itself, sampled out of the $|V| = 100$ most common selectors in the training set, stratified by class; rare selectors map to [UNK], and plain value transfers to NO_CALLDATA.

Three different encodings with progressively more information are used throughout: `rhythm_only` (τ only), `rhythm+intent` (τ, ν), and `full_tokens`, the full encoding of all eleven tokens per transaction – one rhythm, nine structure, one intent (Fig. 2). The history of a wallet, limited to the last $L_{\max} = 256$ transactions, can therefore be expressed in two alternative ways: as a sequence of tokens, kept in order by the sequence models and aggregated into counts by the tabular ones, and as a space-separated string whose binary content is seen by the NCD compressor family.

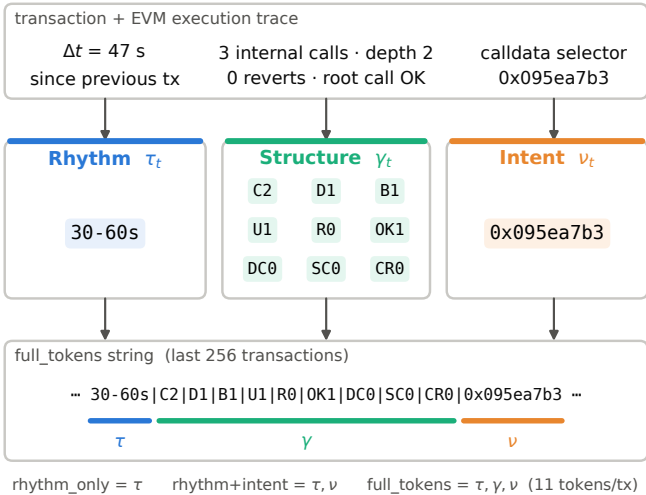


Fig. 2. Transaction Grammar encoding. One transaction and its EVM trace map to the (τ, γ, ν) triplet; a wallet's history serializes to eleven tokens per transaction.

C. Method Families: Tabular, Sequential, and Compression-Based

The grammar defined in Section III-B is intentionally fed in three different ways: first as a bag of features, second as an ordered sequence of tokens, and third as a raw string to compress. The above three perspectives constitute the method families that are being compared in this paper, and since each perspective comes from exactly the same token sequence, any performance difference can be attributed to inductive

biases – what each method family assumes about the signal – not different input data. The hyperparameters for each configuration are listed in Table III.

The tabular family discards order entirely. XGBoost, the gradient-boosted tree ensemble algorithm, uses one bag-of-grammar vector per wallet, which means the frequency of all grammar tokens (124 unigrams) and three summary statistics over tokens – token entropy, compression ratio, and token richness – resulting in 127-dimensional feature vectors. Two linear SVM algorithms use weighted views of the same input stream (TF-IDF of 1- and 2-grams and raw counts of 2-grams), two baselines that observe order in fragments of local order but lack long-range order. The FT-Transformer completes the family: it is a deep tabular model that uses self-attention over exactly the same 127-dimensional feature vectors with one feature per token.

The sequential family preserves order and lets the model decide what to make of it. The Hierarchical Transformer and Hierarchical BiLSTM are transaction-level models: the eleven tokens of each transaction are vectorized and pooled to one transaction vector, and the self-attention (respectively recurrence) operation is run on the wallet's sequence of up to 256 transaction vectors. The Flattened Transformer is, conversely, token-level, attending over the concatenated sequence of $256 \times 11 = 2816$ tokens – in theory, much more expressive since it can align structure fields from different transactions, but also more costly to train. They are all trained end-to-end as three-class classifiers; how much it helps them to know about the order is a question this article does not test directly.

The compression-based family does not require any training whatsoever. The normalized compression distance (between grammar strings for) the two wallets is given by

$$\text{NCD}(x, y) = \frac{C(xy) - \min(C(x), C(y))}{\max(C(x), C(y))}, \quad (1)$$

where $C(\cdot)$ denotes the compressed length using a generic compressor (gzip in our case, see Table III), and captures the shared algorithmic regularity of the two sequences: if two wallets follow the same script, the compression of their concatenated string will be much better than that of each string alone. When computed for all pairs of wallets, NCD defines the implicit behavioral graph from Section III-A, with wallets as nodes and similarity $1 - \text{NCD}$ as edge weights, where discovery consists in a k -nearest neighbor search of a wallet in the graph. Because the exact graph costs $O(N^2)$ compressions, population-scale deployment requires an approximate front end such as MinHash/LSH, which we leave to future work.

Each family has its own natural regime. The families under supervision provide an answer to the closed-set question (regime A), while compression-based retrieval provides an answer to the discovery question (regime B). It is feasible to use each regime from the other side: NCD can classify using k -NN majority voting – which is how retrieval performance is evaluated – and a trained classifier can suggest discoveries by scoring all wallets out-of-fold – which is how the comparison between the two regimes takes place in Section IV-E.

D. Blind-Spot Leakage Protocol

The counterparties of a wallet are strongly indicative of its class – and this is exactly the issue. When a contract is almost solely used by a specific class, any model trained on this contract will be able to perform well simply by remembering who that wallet calls instead of learning how that wallet behaves: shortcut learning in the purest form [27], [29]. In our dataset, the strongest leaks are not those popular venues used by all classes – they would withstand any purity test – but the venues exclusive to one particular class, such as the contracts of a consumer smart wallet or an executor contract of a bot operator (Table II). For these venues, knowing the counterparty means knowing the label. Without being addressed, they will trivialize actor classification into address lookup, skewing the different method families disproportionately. Removal of direct identities performed in Section III-B helps with the problem, yet the high-signal counterparties leave their footprint via distinctive selector-and-structure pattern in the representation.

Thus, the Blind-Spot procedure precludes high-signal interactions prior to encoding. For each counterparty c , we sum all the transactions going to it in our population and derive three quantities: support n_c , the number of such transactions; purity $P_c = \max_y p_c(y)$, where $p_c(y)$ denotes the proportion of those transactions originating from class y ; and the Shannon entropy H_c of the distribution p_c . A counterparty is marked whenever $n_c \geq 50$ and $P_c \geq 0.95$ – enough activity for importance and class proportion sufficiently close to a label. All the transactions directed at the marked counterparties are purged, the Intent vocabulary of Section III-B is re-created using the purged dataset – to ensure that high-signal selectors will not make a come-back via vocabulary formation –, and each wallet is then re-encoded based on its residual transactions. The exact same purged dataset is processed by every method family: no cross-comparison in this paper involves a purged and an unpurged dataset.

At the committed thresholds, the protocol marks 4744 addresses and filters out 25.5% of the transaction volume; 443 wallets lose too many transactions to satisfy the minimum history length constraint and exit the population (Section IV-A provides details on the impact on a per-class basis, and Table II provides the most significant filtered sources). There are two aspects of the protocol that one must remember. First, it is label-informed: purity is calculated from the same labels the evaluation predicts, which makes the protocol an evaluation control rather than a preprocessing step available in label-free deployment. Second, its thresholds are configurable parameters, not constants of nature, and we do not vary them in sensitivity sweeps in this paper, treating the corresponding uncertainty as a limitation (Section V).

IV. EXPERIMENTS AND RESULTS

With the framework in place, we now put it to the test. Section IV-A specifies the dataset and the per-regime evaluation protocol, and Section IV-B verifies – before any model is trained – that the grammar carries class signal at all. The three research questions of Section III-A then follow in order: classification (RQ1), discovery (RQ2), and complementarity (RQ3).

A. Experimental Setup and Dataset

Organic users and Sybil wallets are taken from the Hop Protocol airdrop investigation (the published `eliminatedSybilAttackers` list and its verified legitimate counterparts) [3], while MEV arbitrage bots are identified through Dune Analytics’ curated arbitrage-trader labels [30]. For every address we retrieve the complete transaction history, including internal EVM execution traces, from Google BigQuery, with a hard cutoff at the Hop snapshot date of May 13th, 2022. Wallets with fewer than $L_{\min} = 10$ transactions are discarded, leaving approximately 6.19M transactions. Applying the Blind-Spot protocol of Section III-D removes 25.5% of transaction volume; as a side effect, 443 wallets drop below $L_{\min}$ and leave the dataset (Table I). The loss is concentrated in the MEV class (356 of 443), whose activity centers on the removed high-signal contracts, while only 9 Sybils are lost. The Blind-Spot-filtered population of 14161 wallets is the main evaluation setting throughout; the raw population is retained solely for the leakage-sensitivity check of Section IV-D.

TABLE I
WALLETS PER CLASS BEFORE AND AFTER BLIND-SPOT FILTERING; Δ
COUNTS WALLETS THAT DROP BELOW $L_{\min}$ ONCE HIGH-SIGNAL
CONTRACTS ARE REMOVED.

Class	Raw	Filtered	Δ
Organic	9282	9204	78
MEV Bot	3558	3202	356
Sybil Bot	1764	1755	9
Total	14604	14161	443

Table II makes the removal concrete with the three strongest flagged contracts per class, scored by the protocol’s per-counterparty statistics: support n (the number of transactions sent to the address), purity P (the dominant class’s share of those transactions), and the Shannon entropy H of the contract’s class distribution, near zero when one class accounts for almost all traffic.

TABLE II
STRONGEST REMOVED LEAKAGE SOURCES PER CLASS.

Contract	Class	TXs removed	P	H	Volume
Argent TransferManager	Organic	76818	1.000	0.00	1.5%
Argent wallet module	Organic	74220	0.983	0.12	1.4%
Argent RelayerManager	Organic	36001	0.999	0.01	0.7%
Bot executor 0x00...d54	MEV Bot	8970	1.000	0.00	1.3%
Bot executor 0x70...5a1	MEV Bot	8382	1.000	0.00	1.2%
Bot executor 0x00...f56	MEV Bot	7836	1.000	0.00	1.1%
DEX exchange contract	Sybil Bot	2377	0.976	0.16	1.0%
Gnosis Safe proxy	Sybil Bot	408	1.000	0.00	0.2%
Consolidation EOA	Sybil Bot	349	1.000	0.00	0.1%

What leaks is not where the classes trade but infrastructure exclusive to one class. The three strongest Organic sources are all module contracts of Argent, a consumer smart-contract wallet: every transaction an Argent user makes is routed through these modules, and bots and farming scripts have no reason to touch them (the linch aggregator router and the

CryptoKitties game rank close behind). The MEV sources are the operators’ own bot contracts – private, unverified contracts that execute arbitrage bundles, deployed at gas-optimized vanity addresses and called by essentially no one but their owners. The Sybil sources betray coordination rather than a product choice: a single decentralized-exchange contract, one Gnosis Safe multisig wallet shared by many farmed accounts, and even a plain externally owned account (EOA) to which hundreds of Sybil wallets send funds – a consolidation address collecting the farmed value. The busiest venues survive filtering precisely because every class uses them: the Uniswap V2 router (568828 transactions, $P = 0.946$) and the main OpenSea exchange (415666 transactions, $P = 0.909$) both fall below the purity threshold.

Each regime has its own evaluation protocol. Closed-set classification uses a single stratified 80/20 split of the filtered population (11328 train / 2833 validation wallets), generated once with a fixed seed, persisted, and reused verbatim by every method, so that all models see identical training data and are scored on identical validation wallets. Each model is trained with five random seeds; we report the mean and standard deviation of the Matthews correlation coefficient (MCC) as the primary metric – it remains informative under the 5.2:1.8:1 class imbalance – together with macro-F1, accuracy, and per-class recall. Class imbalance is compensated identically across families through inverse-frequency weighting: class-weighted cross-entropy for the neural models and balanced sample weights for XGBoost and the SVMs. Discovery is evaluated on balanced samples of 300 wallets per class ($N = 900$) drawn with multiple seeds, using 1-NN accuracy, top-10 Sybil purity, and top-10 Sybil recall; the balanced Sybil prior of 0.333 is deliberately optimistic relative to the population prior of 12.4%, which the full-population analysis of Section IV-E controls for. Table III lists the configuration of every method. Sequence-model training and all efficiency measurements of the deployment section were carried out on one machine (NVIDIA RTX 4060 Ti, 16 GB VRAM, 6 vCPU; PyTorch 2.12, CUDA 13.0).

B. Behavioral Grammar Analysis

Before any method is applied, we ask whether the three actor classes already differ in the raw statistics of their grammar sequences – the signal every downstream model must feed on. The classes differ sharply in history length (median 35 transactions for Sybil Bots, 156 for Organic, 202 for MEV Bots), so all statistics below are per-wallet and length-normalized where applicable. Table IV reports per-class medians of the Shannon entropy of the structure and intent token streams (H_G , H_V), the LZ76 complexity of the structure stream (C_{LZ}^G), and the timing entropy over rhythm bins (H_τ). All four differ across classes (Kruskal–Wallis, $p < 0.001$ throughout; pairwise Dunn tests with Holm correction significant except where noted).

The ordering is consistent: Organic users are the most structurally diverse on every metric, Sybils the least – their scripted, templated behavior yields the lowest structure entropy and LZ complexity – with MEV bots in between, machine-regular but operationally richer than farming scripts. Timing

TABLE III
HYPERPARAMETERS OF ALL EVALUATED METHODS.

Method	Configuration
XGBoost	800 trees, depth 8, $\eta = 0.05$, subsample 0.8, col-sample 0.8, $\lambda = 1$, hist grower; unigram counts + 3 global statistics (token entropy, compression ratio, richness)
TF-IDF SVM	linear SVM, balanced classes; TF-IDF over 1–2-grams, $\leq 100k$ features, min. df 2
Bigram SVM	linear SVM, balanced classes; bigram counts, $\leq 100k$ features, min. df 2
Neural (shared)	AdamW, lr $3 \cdot 10^{-4}$, weight decay 0.01; class-weighted cross-entropy; ≤ 8 epochs, early stopping on validation MCC (patience 2); input truncated to last $L_{\max} = 256$ transactions
Hier. Transformer	$d = 256$, 4 heads, 4 layers, dropout 0.1; tokens mean-pooled per transaction, [CLS] over 256 transactions; batch 256
Hier. BiLSTM	embedding 256, hidden 256×2 dir., 2 layers, dropout 0.1; per-transaction pooling as above; batch 256
Flat. Transformer	$d = 256$, 4 heads, 4 layers, dropout 0.1; token-level sequence ($256 \times 11 = 2816$ tokens); batch 32
FT-Transformer	$d = 256$, 4 heads, 4 layers; one token per feature of the XGBoost vector (124 counts + 3 statistics, $\log(1+x)$, z-scored) + [CLS]; batch 256
NCD	gzip (level 9) over <code>full_tokens</code> strings, last 256 transactions

alone, however, is a weak separator between the two bot classes: Organic differs from both (Cliff’s $\delta \approx 0.28$ –0.29), but MEV Bot versus Sybil Bot timing entropy is negligible ($\delta = 0.07$), so rhythm tokens by themselves cannot carry the classification.

TABLE IV
PER-CLASS MEDIANS OF GRAMMAR-SEQUENCE STATISTICS.

Metric	Organic	MEV Bot	Sybil Bot
Structure entropy H_G	3.43	3.05	2.64
Intent entropy H_V	3.19	2.93	2.86
LZ complexity C_{LZ}^G	3.37	3.12	2.83
Timing entropy H_τ	3.32	3.20	3.17

The second view is pairwise: if Sybil wallets are generated by shared scripts, they should compress well against each other. Table V compares within-class and between-class similarity ($1 - \text{NCD}$) on balanced samples of 300 wallets per class, and Fig. 3 shows the full within-class distributions for the main encoding. Sybils are the clear outlier: their within-class similarity exceeds their similarity to other classes by about 0.10 under every encoding (Mann–Whitney $p < 0.001$, Cliff’s $\delta = 0.50$ –0.58 versus Organic, with bootstrap confidence intervals excluding zero), and their distribution is right-shifted with a long tail of near-clone pairs that the other classes lack.

TABLE V
SYBIL BOT WITHIN- VERSUS BETWEEN-CLASS SIMILARITY ($1 - \text{NCD}$); δ IS CLIFF’S DELTA VERSUS ORGANIC.

Encoding	Within	Between	Δ	δ
rhythm_only	0.364	0.255	0.109	0.56
rhythm+intent	0.328	0.219	0.109	0.58
full_tokens	0.302	0.205	0.098	0.50

Together the two views identify the shared signal the rest of the paper exploits from two directions: Sybil behavior is low-entropy and mutually compressible. Aggregate models can read it from per-wallet statistics – the entropy and compressibility features of the XGBoost representation – which sets up the closed-set comparison, while the pairwise compressibility is exactly what NCD retrieval navigates, which sets up candidate discovery.

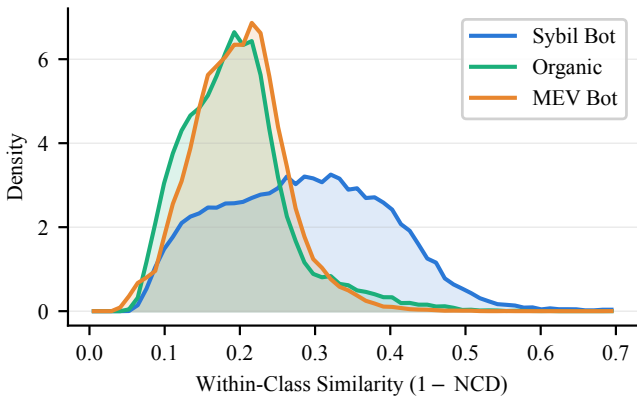


Fig. 3. Distribution of within-class pair similarity ($1 - \text{NCD}$, full_tokens) per class.

C. Closed-Set Classification

We now evaluate the closed-set regime: given the three actor classes, which method family labels Blind-Spot-filtered wallets best? Seven models consume the shared grammar through the three views of Section III-C. XGBoost operates on bag-of-grammar features (unigram counts plus global entropy, compressibility, and richness statistics); the FT-Transformer, the added deep-tabular baseline, applies attention over exactly the same 127-dimensional feature vector, one token per feature; two linear SVMs consume TF-IDF and bigram views of the token stream; and three sequence models – a token-level Flattened Transformer, a per-transaction Hierarchical Transformer, and a Hierarchical BiLSTM – consume the grammar in order. The tabular-sequential contrast thus asks directly whether preserving transaction order earns its cost once the leaked shortcuts are gone.

Table VI shows a clear hierarchy. XGBoost leads every aggregate metric by a wide margin: MCC 0.759 against 0.666 for the best non-tree model, with negligible seed variance ($\sigma = 0.004$). The FT-Transformer result isolates why: it receives the byte-identical feature matrix and still trails by 0.093 MCC, so the gap is attributable to the model class – gradient-boosted trees versus attention – rather than to the aggregate-

versus-sequential view of the data. Among the sequence models, the token-level Flattened Transformer outperforms both hierarchical variants, and the Hierarchical Transformer is the weakest and least stable model overall ($\sigma = 0.032$). The per-class picture is consistent across families: MEV bots are easy to separate (recall ≥ 0.81 everywhere), and errors concentrate on the Organic-Sybil boundary. Notably, the sequence models trade Organic recall for Sybil recall (0.68–0.74 versus XGBoost’s 0.639) – a trade the aggregate metrics penalize. Whether preserving transaction order earns its computational cost at all is a question we return to in Section V.

D. Sybil Candidate Discovery

We now evaluate the open-world regime: a seed wallet queries its k nearest neighbors in the implicit behavioral graph induced by NCD, with no training and no labels in the loop. Following the protocol of Section IV-A, retrieval runs on balanced samples (300 wallets per class, three seeds) over the Blind-Spot-filtered population, and is scored by leave-one-out 1-NN accuracy, top-10 Sybil neighborhood purity, and top-10 Sybil recall. All retrieval metrics are means over the three sample seeds; we show the seed standard deviation for the primary 1-NN metric only, as the corresponding spread for purity and recall never exceeds 0.039. As a baseline we use TF-IDF cosine similarity over the same token strings – a “bag-of-symbols” retriever that sees token frequencies but no sequential structure.

TABLE VII
DISCOVERY RESULTS: GZIP-NCD VERSUS TF-IDF COSINE PER ENCODING.

Method	Encoding	1-NN Accuracy	Top-10 Purity	Top-10 Recall
NCD	rhythm_only	0.522 $\pm$ 0.019	0.552	0.836
NCD	rhythm+intent	0.543 $\pm$ 0.014	0.586	0.817
NCD	full_tokens	0.698 $\pm$ 0.016	0.727	0.919
TF-IDF	rhythm_only	0.527 $\pm$ 0.017	0.452	0.853
TF-IDF	rhythm+intent	0.523 $\pm$ 0.018	0.507	0.824
TF-IDF	full_tokens	0.637 $\pm$ 0.009	0.595	0.833

Table VII shows the same two regularities under every reading. First, information stacks: adding intent to rhythm, and execution structure on top, improves NCD monotonically, with full_tokens best on every metric (1-NN accuracy 0.698, top-10 purity 0.727 against a class prior of 0.333, recall 0.919). Second, structure matters beyond frequency: NCD beats the TF-IDF baseline by 0.061 1-NN accuracy and 0.13 purity on the full encoding, so the compressor exploits sequential regularity that a bag-of-symbols view cannot see. Notably, discovery is essentially insensitive to the Blind-Spot filtering (full_tokens 1-NN 0.689 raw versus 0.698 filtered): the similarity structure does not depend on the leaked contracts the protocol removes.

E. Classification vs. Discovery: Complementarity

Having evaluated the two regimes separately, we now ask whether they recover the same Sybils. We compare the sets

TABLE VI
CLOSED-SET CLASSIFICATION RESULTS.

Model	MCC ($\mu \pm \sigma$)	Macro-F1	Accuracy	MEV Bot	Recall Organic	Sybil Bot
XGBoost	0.759 $\pm$ 0.004	0.829	0.878	0.903	0.915	0.639
FT-Transformer	0.666 $\pm$ 0.012	0.765	0.815	0.893	0.813	0.688
Flattened Transformer	0.662 $\pm$ 0.008	0.762	0.812	0.902	0.805	0.682
Hierarchical BiLSTM	0.632 $\pm$ 0.020	0.740	0.784	0.907	0.751	0.732
TF-IDF SVM	0.613 $\pm$ 0.000	0.736	0.790	0.871	0.800	0.595
Bigram SVM	0.596 $\pm$ 0.000	0.723	0.781	0.808	0.798	0.641
Hierarchical Transformer	0.560 $\pm$ 0.032	0.692	0.733	0.827	0.699	0.744

of labeled Sybils that each regime retrieves from the identical Blind-Spot-filtered population of 14161 wallets. Set A contains the labeled Sybils that the closed-set classifier flags as Sybil. To score every wallet without train-test contamination, we use out-of-fold predictions: stratified 5-fold cross-validation with the XGBoost configuration of Table VI, so each wallet is predicted by a model that never saw it in training (out-of-fold MCC 0.763 ± 0.002 over five seeds). We take a majority-vote consensus over the five seeds (≥ 3 of 5), giving $|A| = 1190$ of the 1755 labeled Sybils. Set B contains the labeled Sybils that discovery recovers: each labeled Sybil is used as a query against the full population in the implicit behavioral graph (gzip NCD, `full_tokens`), and its ten nearest neighbors are recorded. This computation is deterministic—no sampling and no training seeds. Because “recovered by retrieval” admits two readings, we report both: B_{purity} , the Sybils whose own top-10 neighborhood is Sybil-majority (≥ 5 of 10; $|B| = 1170$), and $B_{\text{expansion}}$, the Sybils that appear in the top-10 neighborhood of at least one other labeled Sybil, i.e., the wallets a seed-expansion audit would surface ($|B| = 1444$). Sybils are only 12.4% of the population, yet the mean top-10 neighborhood of a labeled Sybil contains 6.2 Sybils—a five-fold enrichment over the base rate. The purity reading is not an artifact of the majority threshold: $|B_{\text{purity}}|$ declines smoothly from 1458 (≥ 3) through 1170 (≥ 5) to 848 (≥ 7), with no threshold cliff.

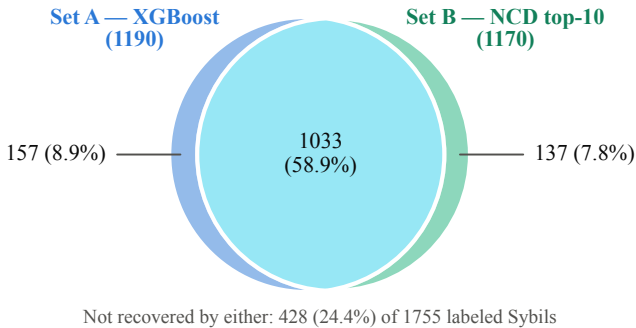


Fig. 4. Overlap between Set A and Set B_{purity} (Jaccard 0.78); percentages are relative to all 1755 labeled Sybils.

Table VIII and Fig. 4 summarize the result. Under the conservative purity reading the two regimes agree strongly—Jaccard 0.778 (stable at 0.772–0.787 across classifier seeds), with 1033 Sybils recovered by both—yet each contributes a distinct margin: 157 Sybils only the classifier catches, 137 only retrieval catches, for a union of 1327 (75.6% of labeled Sybils). The seed-expansion reading shifts the picture from agreement to asymmetry: retrieval then recovers all but 65 of the classifier’s catches while adding 319 Sybils the classifier misses, lifting union coverage to 1509 (86.0%, versus 67.8% for the classifier alone). Neither regime dominates: the union beats either method under both readings.

TABLE VIII
OVERLAP BETWEEN CLASSIFIER-RECOVERED (SET A) AND NCD-RECOVERED (SET B) LABELED SYBILS, FOR BOTH SET- B READINGS.

Statistic	B_{purity}	$B_{\text{expansion}}$
$ B $	1170	1444
$ A \cap B $	1033	1125
$ A \setminus B $	157	65
$ B \setminus A $	137	319
$ A \cup B $	1327	1509
Coverage of labeled Sybils	75.6%	86.0%
Jaccard (seed range)	0.778 (0.772–0.787)	0.746 (0.739–0.751)

Table IX profiles the four regions of the overlap by history length (transactions per wallet), mean per-wallet Shannon entropy (bits) of rhythm and intent tokens, and the most frequent consecutive inter-transaction-gap motif with the share of wallets containing it (document frequency). The consensus region $A \cap B$ (row “Both”) is the archetypal airdrop farmer: short histories (median 28 transactions), low intent entropy, and a slow farming cadence—the dominant motif is consecutive inter-transaction gaps of 7–30 days, consistent with scripted wallets that periodically touch a protocol to accumulate eligibility. The retrieval-only region $B \setminus A$ (row “NCD”) has markedly longer histories (median 40, mean 66 transactions) and higher intent entropy (1.08 vs 0.92): wallets whose aggregate feature profile drifts toward the organic class—which is exactly where the classifier loses them—but whose sequential behavior still compresses well against other Sybils. The classifier-only region $A \setminus B$ (row “XGBoost”) is statistically closest to the consensus region; these are wallets whose neighborhoods fall just short of the purity threshold rather than a behaviorally distinct population. Finally, the 428 Sybils (24.4%) that neither method recovers look strikingly

organic: long histories (median 110 transactions), the highest rhythm and intent entropy of any region, and a daily, human-like cadence (6–24 h and 1–3 d gap motifs). These wallets—eliminated by Hop’s graph-based audit—are behaviorally indistinguishable from organic users at the granularity of our grammar, and mark the boundary of what purely behavioral, link-free detection can reach.

TABLE IX
CHARACTERIZATION OF THE OVERLAP REGIONS. n : NUMBER OF
WALLETS IN THE REGION; H : MEAN PER-WALLET SHANNON ENTROPY
(BITS) OF RHYTHM AND INTENT TOKENS.

Region	n	Length median	H_{rhythm}	H_{intent}	Top gap motif (% of wallets)
Both	1033	28	3.15	0.92	7–30 d $\rightarrow$ 7–30 d (40%)
XGBoost	157	24	3.11	0.92	7–30 d $\rightarrow$ 7–30 d (33%)
NCD	137	40	3.20	1.08	7–30 d $\rightarrow$ 7–30 d (42%)
Neither	428	110	3.28	1.41	6–24 h $\rightarrow$ 1–3 d (48%)

The two regimes are therefore overlapping but not redundant, and their disagreement is systematic rather than noise: the classifier anchors on the aggregate profile of the typical farmer, while compression-based retrieval tracks sequential self-similarity and therefore holds on to longer, more entropic Sybils that blend into the organic class in feature space. For practice this argues for a two-stage audit pipeline—the cheap classifier as triage over the full population, followed by NCD seed-expansion around confirmed Sybils to recover the coordinated groups the classifier misses—which we return to in Section V. Two caveats bound the claim. First, both sets are measured on labeled Sybils, so the comparison is recall-oriented; it does not measure false positives on unlabeled wallets (the neighborhood-purity statistics of Table VII are the closest proxy). Second, $B_{\text{expansion}}$ presupposes confirmed seed Sybils to expand from, so its 86% union coverage describes the audit workflow, not a label-free detector.

V. DISCUSSION AND FUTURE WORK

Which method an audit team should use cannot be answered without first fixing the operational regime. On the closed-set leaderboard (RQ1), gradient-boosted trees win decisively – MCC 0.759 against 0.666 for the best alternative – while on the discovery task (RQ2) the leaderboard is beside the point: only a training-free, label-free retriever operates at all once labels are unavailable. This article does not test whether the sequential inductive bias of the runner-up models earns its computational cost through a mechanistic ablation; the raw closed-set gap of 0.093 MCC is nonetheless a high bar for any such justification to clear. What the evidence here does support directly is RQ3’s two-stage pipeline: cheap classification first, behavioral expansion second, rather than a single leaderboard entry.

RQ3’s result is, to our knowledge, the first direct measurement of how much a supervised classifier and a training-free retriever agree about the same Sybil population, and the interesting part is where they disagree. The two methods converge on the archetypal short-history farmer (Jaccard 0.778), but retrieval alone reaches longer-history, higher-entropy Sybils

whose aggregate profile has drifted toward the organic class, and seed expansion around confirmed Sybils lifts coverage from 67.8% (classifier alone) to 86.0%. For an audit team this is a workflow, not just a statistic: run the classifier as near-instant triage over the full population, then expand each confirmed hit through the behavioral graph to recover coordinated groups the aggregate view alone would miss. A boundary remains – under the stricter B_{purity} reading, 24.4% of labeled Sybils elude both methods, and these are precisely the long-history, high-entropy wallets with human-like cadence that Hop’s own link-based investigation caught instead. That is direct evidence that behavioral and financial-link evidence are complementary, not that either alone suffices.

Several limitations bound these claims. All results come from a single campaign on one chain – the Hop airdrop population – so cross-campaign generalization is neither tested nor claimed here. Ground truth descends from Hop’s own link-based investigation, which makes the complementarity comparison recall-oriented over labeled Sybils rather than a measurement of false positives on the wider unlabeled population, and is also why graph-based detectors are excluded from the comparison: a link-based detector would partly re-derive the process that produced the labels, making any measured advantage circular. Discovery is evaluated on balanced samples (Sybil prior 0.333), which is optimistic relative to the population prior of 12.4%; the full-population complementarity analysis of Section IV-E is the check against that optimism. The Blind-Spot protocol is label-informed and therefore an evaluation control, not a label-free deployment step, and its thresholds are not swept for sensitivity here. Finally, this article establishes the comparison and the complementarity result on clean data only: it does not measure how any method degrades under adversarial camouflage, nor what any method costs to train, query, or scale.

Future work follows directly from these boundaries. The most consequential test is cross-campaign transfer: repeating this comparison on a second incentive event with independently produced labels, since nothing here speaks to whether the grammar’s regularities hold beyond the Hop population. Two questions this article leaves entirely open concern cost and pressure: how much of each method’s headline performance survives adversarial camouflage, and what each costs to train, query, and scale – the natural basis for a decision table that would translate these results into deployment guidance. Further out, regime A could be extended to formal open-set recognition, the “neither” region of the overlap analysis argues for fusing behavioral and financial-link evidence where non-circular labels exist, and porting the grammar to other chains would test how much of the representation is Ethereum-specific.

VI. CONCLUSION

Closed-set classification and open-world discovery of Sybil wallets have typically been studied separately, leaving an audit team no direct basis for choosing between them. This article closes that gap for three questions: seven supervised classifiers and a training-free compression-based retriever are

compared under one leakage-control protocol, one shared Transaction Grammar, and one Blind-Spot-filtered population of 14161 Ethereum wallets. With counterparty shortcuts removed, gradient-boosted trees over bag-of-grammar features remain the strongest closed-set classifier (MCC 0.759), well ahead of every sequence model (RQ1). Without any labels or training, gzip-based normalized compression distance recovers Sybil neighborhoods well above chance and above a frequency-only baseline (RQ2). The two regimes recover overlapping but distinct Sybil sets: seed expansion around confirmed Sybils lifts coverage of labeled Sybils from 67.8% to 86.0%, so classifier triage followed by behavioral expansion beats either method alone (RQ3).

We contribute the first comparison of tabular, sequential, and compression-based Sybil detection under one leakage-controlled protocol, and a two-stage classifier-triage-plus-behavioral-expansion pipeline that recovers more of a labeled Sybil campaign than either method alone. For practitioners, the significance is a change in the shape of the question: not which detector tops a single leaderboard, but which operational regime – classification or discovery – the audit actually calls for, since the two are complementary rather than competing tools. Because every result here is within-campaign, cross-campaign transfer on an independently labeled second incentive event is the decisive next step; robustness under adversarial pressure and the cost of deploying each method remain open questions this article does not address.

REFERENCES

- [1] J. Douceur, “The sybil attack,” 10 2002.
- [2] J. Messias, A. Yaish, and B. Livshits, “Airdrops: Giving money away is harder than it seems,” 2023.
- [3] Hop Protocol, “hop-airdrop: Hop Airdrop distribution,” <https://github.com/hop-protocol/hop-airdrop>, 2022, github repository (accessed 11.07.2026).
- [4] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger,” Ethereum Yellow Paper, 2025, shanghai version.
- [5] M. Bartnicki and J. A. Chudziak, “Modeling decisions in blockchain analytics: A leakage-aware evaluation of tree-based vs. sequential models,” 2026. [Online]. Available: <https://arxiv.org/abs/2607.27350>
- [6] —, “Compression-based behavioral similarity for open-world sybil discovery on ethereum,” 2026. [Online]. Available: <https://arxiv.org/abs/2607.27370>
- [7] T. Huang, D. Lin, and J. Wu, “Ethereum account classification based on graph convolutional network,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 5, 2022.
- [8] J. Liu *et al.*, “Fa-gnn: Filter and augment graph neural networks for account classification in ethereum,” *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 4, 2022.
- [9] Q. Liu *et al.*, “Detecting sybil addresses in blockchain airdrops: A subgraph-based feature propagation and fusion approach,” in *2025 IEEE Int. Conf. on Blockchain and Cryptocurrency (ICBC)*, 2025.
- [10] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proc. KDD*, 2016.
- [11] A. Vaswani *et al.*, “Attention is all you need,” in *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.
- [12] S. Hu *et al.*, “Bert4eth: A pre-trained transformer for ethereum fraud detection,” in *Proceedings of the ACM Web Conference 2023*, 2023.
- [13] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, “Toward open set recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, 2013.
- [14] L. Grinsztajn, E. Oyallon, and G. Varoquaux, “Why do tree-based models still outperform deep learning on typical tabular data?” in *Proc. 36th NeurIPS*, 2022.
- [15] A. Shmuel, O. Glickman, and T. Lazebnik, “A comprehensive benchmark of machine and deep learning models on structured data for regression and classification,” *Neurocomputing*, vol. 655, p. 131337, 2025. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925231225020090>
- [16] R. Schwartz-Ziv and A. Armon, “Tabular data: Deep learning is not all you need,” *Information Fusion*, vol. 81, 2022.
- [17] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, and G. Kasneci, “Deep neural networks and tabular data: A survey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, 2024. [Online]. Available: <http://dx.doi.org/10.1109/TNNLS.2022.3229161>
- [18] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, “Revisiting deep learning models for tabular data,” 2021.
- [19] N. Hollmann *et al.*, “TabPFN: A transformer that solves small tabular classification problems in a second,” in *NeurIPS 2022 1st Table Rep. Workshop*, 2022. [Online]. Available: <https://openreview.net/forum?id=eu9fVjVasr4>
- [20] G. Somepalli *et al.*, “Saint: Improved neural networks for tabular data via row attention and contrastive pre-training,” 2021.
- [21] A. Tokajuk and J. Chudziak, “Partial multivariate transformer as a tool for cryptocurrencies time series prediction,” in *2025 IEEE 37th International Conference on Tools with Artificial Intelligence (ICTAI)*, 2025.
- [22] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *Proc. AAAI*, vol. 35, no. 12, 2021, pp. 11 106–11 115.
- [23] F. Bańka and J. Chudziak, “Applying informer for option pricing: A transformer-based approach,” in *Proc. ICAART*. SCITEPRESS - Science and Technology Publications, 2025.
- [24] R. Cilibrasi and P. Vitányi, “Clustering by compression,” *Information Theory, IEEE Transactions on*, vol. 51, 2005.
- [25] R. S. Borbely, “On normalized compression distance and large malware,” *Journal of Computer Virology and Hacking Techniques*, vol. 12, no. 4, 2016.
- [26] S. Har-Peled, P. Indyk, and R. Motwani, “Approximate nearest neighbor: Towards removing the curse of dimensionality,” *Theory of Computing*, vol. 8, 01 2012.
- [27] S. Kaufman, S. Rosset, and C. Perlich, “Leakage in data mining: Formulation, detection, and avoidance,” vol. 6, 01 2011.
- [28] F. Pendlebury *et al.*, “Tesseract: eliminating experimental bias in malware classification across space and time,” in *Proceedings of the 28th USENIX Conference on Security Symposium*, ser. SEC’19. USA: USENIX Association, 2019.
- [29] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann, “Shortcut learning in deep neural networks,” *Nature Machine Intelligence*, vol. 2, no. 11, 2020.
- [30] Dune Analytics, “Curated labels: arbitrage traders (labels.arbitrage_traders),” <https://dune.com/queries/6372046>, 2026, dune query 6372046 (accessed 11.07.2026).