Collected by: Shikah J. Alsunaidi

**Table I.** Features Description.

| # | Feature | Type | Description | Ref |
|---|---|---|---|---|
| 1 | burn_events | DA | Total number of burn events | [1] |
| 2 | cluster_coeff | DA | Clustering coefficient | [1] |
| 3 | difference_token_pool | DA | Number of blocks between token and pool creation | [1] |
| 4 | liquidity | DA | Total liquidity | [1] |
| 5 | n_syncs | DA | Total sync events | [1] |
| 6 | n_unique_addresses | DA | Total number of unique addresses | [1] |
| 7 | num_transactions | DA | Total number of transactions | [1] |
| 8 | prices | DA | Price of token | [1] |
| 9 | weth | DA | Total weth | [1] |
| 10 | Avg_GasPrice_Sent | DA | The average gas price value used in sent transactions | [2] |
| 11 | Failed_Txn_Received | DA | The total number of failed received transactions | [2] |
| 12 | Failed_Txn_Sent | DA | The total number of failed sent transactions | [2] |
| 13 | Gas_Sent | DA | Total of gas value used in sent transactions | [2] |
| 14 | GasPrice_Sent | DA | The total number of gas price values used in sent transactions | [2] |
| 15 | GasUsed_Contract_Create | DA | Gas spent during contract creation | [2] |
| 16 | Min_Eth_Sent | DA | The minimum Ether value sent | [2] |
| 17 | Total_Eth_Received | DA | The total ether value received | [2] |
| 18 | Total_Txn | DA | The total number of transactions sent and received | [2] |
| 19 | Txn_Fee_Received | DA | The total of transaction fees spent in received transactions | [2] |
| 20 | addrGetProfit | DA | The number of addresses that receive proceeds from the addresses traded with the contract | [3] |
| 21 | avgFee | DA | The average cost of all transactions in the contract | [3] |
| 22 | Gini | DA | Gini coefficie | [3] |
| 23 | maxSend | DA | The maximum amount of ETH sent in a single contract | [3] |
| 24 | totalGet | DA | The number of all ETH received by the contract | [3] |
| 25 | totalSend | DA | The number of all ETH received by the contract | [3] |
| 26 | N_maxpay | DA | The maximum total return received by a single investor in the contract | [4] |
| 27 | Paid_rate | DA | Proportion of investors who receive income at least once | [4] |
| 28 | from_in_mean_min | DA | NA | [5] |
| 29 | from_in_min_std | DA | NA | [5] |
| 30 | from_in_sum_min | DA | It reflects the in transaction of the node’s from friend | [5] |
| 31 | from_value_mean | DA | NA | [5] |
| 32 | from_value_min | DA | NA | [5] |
| 33 | in_block_ptp | DA | NA | [5] |
| 34 | In_block_std | DA | The standard deviation of blockNumber of all in transaction for a node | [5] |
| 35 | in_count_unique_ratio | DA | NA | [5] |
| 36 | in_unique | DA | NA | [5] |
| 37 | out_block_std | DA | NA | [5] |
| 38 | out_count | DA | NA | [5] |
| 39 | to_out_min_std | DA | NA | [5] |
| 40 | to_out_sum_median | DA | It reflects the overall situation (i.e., sum) of all the to friends’ out-transactions | [5] |
| 41 | to_out_sum_min | DA | NA | [5] |
| 42 | normalTransactionValueMean | DA | The mean of normal transaction value | [6] |
| 43 | Avg_min_between_received_tnx | DA | Average time between received transactions for account in minutes | [7] |
| 44 | Avg_min_between_sent_tnx | DA | Average time between sent transactions for account in minutes | [7] |
| 45 | avg_val_received | DA | Average value in Ether ever received | [7] |
| 46 | avg_val_sent | DA | Average value of Ether ever sent | [7] |
| 47 | max_value_received | DA | Maximum value in Ether ever received | [7] |
| 48 | min_val_sent | DA | Minimum value of Ether ever sent | [7] |
| 49 | min_value_recived | DA | Minimum value in Ether ever received | [7] |
| 50 | Time_Diff_between_first_and_last_(Mins) | DA | Time difference between the first and last transaction | [7] |
| 51 | total_ether_balance | DA | Total Ether Balance following enacted transactions | [7] |
| 52 | Unique_received_from_addresses | DA | Total Unique addresses from which account received transactions | [7] |
| 53 | ERC20_Uniq_Rec_Contract_Addr | DA | Number of tokens gotten from a unique contract address | [8] |
| 54 | ERC20_unique_received_token_name | DA | Number of unique tokens received | [8] |
| 55 | ERC20_unique_sent_token_name | DA | Number of unique tokens sent | [8] |
| 56 | Minimum_value_received | DA | The minimum “Ether” value received | [8] |
| 57 | Minimum_values_sent | DA | The minimum “Ether” value sent | [8] |
| 58 | Total_ether_sent | DA | The total number of “Ethers” sent from a specific address | [8] |
| 59 | fundFlowCase83 | DG | The frequency of deposits to a contract from its creator | [6] |
| 60 | numSourceCodeLines | M | The number of lines in the source code | [6] |
| 61 | ADD | O | Addition | https://ethervm.io/ |

| # | Feature | Type | Description | Ref |
|---|---|---|---|---|
| 62 | AND | O | And | https://ethervm.io/ |
| 63 | BALANCE | O | Address balance in wei | https://ethervm.io/ |
| 64 | CALL | O | Calls a method in another contract | https://ethervm.io/ |
| 65 | CALLDATALOAD | O | Reads a (u)int256 from message data | https://ethervm.io/ |
| 66 | CALLER | O | Message caller address | https://ethervm.io/ |
| 67 | CALLVALUE | O | Message funds in wei | https://ethervm.io/ |
| 68 | CODECOPY | O | Copy executing contract’s bytecode | https://ethervm.io/ |
| 69 | DIV | O | Division | https://ethervm.io/ |
| 70 | DUP1 | O | Clones the last value on the stack | https://ethervm.io/ |
| 71 | DUP6 | O | Clones the 6th last value on the stack | https://ethervm.io/ |
| 72 | DUP9 | O | Clones the 9th last value on the stack | https://ethervm.io/ |
| 73 | EQ | O | Equality | https://ethervm.io/ |
| 74 | EXP | O | Exponentiation | https://ethervm.io/ |
| 75 | GAS | O | Remaining gas | https://ethervm.io/ |
| 76 | GASLIMIT | O | Current block's gas limit | https://ethervm.io/ |
| 77 | GT | O | Greater than (comparison) | https://ethervm.io/ |
| 78 | ISZERO | O | Is zero | https://ethervm.io/ |
| 79 | JUMPDEST | O | Metadata to annotate possible jump destinations | https://ethervm.io/ |
| 80 | JUMPI | O | Conditional jump if condition is truthy | https://ethervm.io/ |
| 81 | LOG | O | Fires an event | https://ethervm.io/ |
| 82 | LT | O | Less than (comparison) | https://ethervm.io/ |
| 83 | MOD | O | Modulus | https://ethervm.io/ |
| 84 | MSTORE | O | Writes a (u)int256 to memory | https://ethervm.io/ |
| 85 | MUL | O | Multiplication | https://ethervm.io/ |
| 86 | POP | O | Pops a (u)int256 off the stack and discards it | https://ethervm.io/ |
| 87 | PUSH1 | O | Pushes a 1-byte value onto the stack | https://ethervm.io/ |
| 88 | PUSH2 | O | Pushes a 2-byte value onto the stack | https://ethervm.io/ |
| 89 | PUSH21 | O | Pushes a 21-byte value onto the stack | https://ethervm.io/ |
| 90 | PUSH4 | O | Pushes a 4-byte value onto the stack | https://ethervm.io/ |
| 91 | PUSH9 | O | Pushes a 9-byte value onto the stack | https://ethervm.io/ |
| 92 | RETURN | O | Return memory[offset:offset+length] | https://ethervm.io/ |
| 93 | REVERT | O | Byzantium hardfork, EIP-140: reverts with return data | https://ethervm.io/ |
| 94 | SHA | O | Hash | https://ethervm.io/ |
| 95 | SLOAD | O | Reads a (u)int256 from storage | https://ethervm.io/ |
| 96 | SSTORE | O | Writes a (u)int256 to storage | https://ethervm.io/ |
| 97 | STOP | O | Halts execution of the contract | https://ethervm.io/ |
| 98 | SUB | O | Subtraction | https://ethervm.io/ |
| 99 | SWAP1 | O | Swaps the last two values on the stack | https://ethervm.io/ |

**Note:** Opcode (**O**), Source code metrics (**M**), Dynamic features/Account-based (**DA**) Dynamic features/Network Graph-based (**DG**), Not Available (**NA**).

## References

[1] B. Mazorra, V. Adan, and V. Daza, “Do Not Rug on Me: Leveraging Machine Learning Techniques for Automated Scam Detection,” *Mathematics*, vol. 10, no. 6, p. 949, Mar. 2022.

[2] A. Aljofey, A. Rasool, Q. Jiang, and Q. Qu, “A Feature-Based Robust Method for Abnormal Contracts Detection in Ethereum Blockchain,” *Electronics*, vol. 11, no. 18, p. 2937, Sep. 2022.

[3] X. He, T. Yang, and L. Chen, “CTRF: Ethereum-Based Ponzi Contract Identification,” *Security and Communication Networks*, vol. 2022, pp. 1–10, Mar. 2022.

[4] Y. Zhang, W. Yu, Z. Li, S. Raza, and H. Cao, “Detecting Ethereum Ponzi Schemes Based on Improved LightGBM Algorithm,” *IEEE Transactions on Computational Social Systems*, vol. 9, no. 2, pp. 624–637, Apr. 2022.

[5] 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*, 2020, pp. 4506–4512.

[6] R. Camino, C. F. Torres, M. Baden, and R. State, “A Data Science Approach for Detecting Honeypots in Ethereum,” in *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2020, pp. 1–9.

[7] S. Farrugia, J. Ellul, and G. Azzopardi, “Detection of illicit accounts over the Ethereum blockchain,” *Expert Systems with Applications*, vol. 150, p. 113318, Jul. 2020.

[8] R. F. Ibrahim, A. Mohammad Elian, and M. Ababneh, “Illicit Account Detection in the Ethereum Blockchain Using Machine Learning,” in *2021 International Conference on Information Technology (ICIT)*, 2021, pp. 488–493.

[9] M. Wang and J. Huang, “Detecting Ethereum Ponzi Schemes Through Opcode Context Analysis and Oversampling-Based AdaBoost Algorithm,” *Computer Systems Science and Engineering*, vol. 47, no. 1, pp. 1023–1042, 2023.