Temporal Behavioral Clustering and Deanonymization in Blockchain Networks

Mark Stevenson

June 11, 2025

Summary

This project explores temporal behavioral clustering as a method for deanonymizing pseudonymous entities in Ethereum transaction networks. By modeling address activity using entropy (interaction diversity) and degree (transactional breadth), we construct a feature space for clustering behavioral roles.

We apply density-based clustering algorithms - DBSCAN and HDBSCAN - across sliding temporal windows to observe the evolution of behavioral signatures over time. Instead of identifying exact moments of identity collapse, our approach tracks the gradual emergence of stable behavioral patterns. These stable roles, once crystallized, reduce anonymity by enabling cohort-based characterization.

Notably, the Low Stability cohort exhibits a 12.98x higher odds ratio of containing OFAC-sanctioned addresses, underscoring the forensic value of entropy and volatility metrics. Our pipeline demonstrates that anonymity in blockchain systems can degrade over time as behavioral roles become increasingly consistent, measurable, and statistically distinguishable.

Introduction

The pseudonymous design of blockchain networks such as Ethereum offers a veneer of privacy, where users interact through address identifiers rather than real-world identities. While this model obscures direct attribution, transactional behavior often leaks patterns over time. These patterns - rooted in interaction diversity, temporal frequency, and structural roles - can compromise anonymity. This project investigates a temporal-behavioral framework for deanonymizing blockchain entities by clustering Ethereum addresses based on their evolving entropy and transactional degree.

This study aims to characterize Ethereum address behavior over time by observing how behavioral patterns stabilize into identifiable roles. Rather than treating deanonymization as a binary identification event, we model it as a gradual erosion of anonymity - a process in which distinct transaction behaviors emerge through temporal clustering. To construct compact behavioral profiles, we focus on two features: entropy, representing the diversity of outbound transaction targets, and degree, the number of unique recipients. These features enable unsupervised clustering of addresses into evolving cohorts, whose identities become increasingly inferable across successive time windows.

We employ two unsupervised, density-based clustering algorithms: DBSCAN and HDBSCAN. These methods are selected for their ability to capture non-convex cluster shapes, tolerate noise, and operate without requiring the number of clusters as a parameter. Compared to more rigid techniques like k-Means or Gaussian Mixture Models, DBSCAN and HDBSCAN better reflect the organic, irregular nature of transactional data in decentralized systems.

Our approach extends prior work in blockchain deanonymization by incorporating time as a first-class analytical axis. While previous efforts [1] have applied clustering to static transaction graphs, they have largely omitted the dynamic nature of behavioral shifts. Similarly, general-purpose clustering improvements like those proposed by [2] have not been adapted to blockchain-specific data. Our contribution is a framework that not only clusters behavior, but tracks its evolution - enabling insights into cohort stability, volatility, and potential role transitions.

This temporal framing is particularly important for forensic applications, where identifying the persistence or transformation of behaviors is as crucial as static attribution. We hypothesize that certain actors only become behaviorally distinct after accumulating enough transactional history - particularly those with low stability or high entropy transition. Through this lens, deanonymization becomes less about a single point of failure and more about behavioral drift toward identifiability.

Methodology

This project follows a structured pipeline for temporal behavioral clustering and cohort detection in Ethereum transaction networks. The methodology encompasses the following major steps:

Software Environment and Libraries

All code was developed in Python 3.10, executed using Google Colab (GPU-enabled instances). The following primary libraries were used:

- `google.cloud.bigquery` for querying Ethereum data
- `pandas`, `numpy` for data manipulation
- `networkx` for graph modeling
- `hdbscan`, `sklearn` for clustering and metrics
- `scipy` for Fisher's Exact Test
- `matplotlib`, `seaborn`, `plotly` for visualizations
- `powerlaw` for distributional analysis

Data Acquisition and Preprocessing

Transaction records were extracted from the Google BigQuery public Ethereum dataset. Preprocessing included:

- Removing null addresses
- Excluding self-transfers (sender = receiver)
- Removing low-value noise transactions (¡ 0.001 ETH)
- Tagging sanctioned addresses via an OFAC-derived list

Feature Engineering

Each Ethereum address was modeled as a node in a directed transaction graph. For each node, two behavioral features were computed:

- **Degree:** Total unique outbound counterparties
- **Entropy:** Shannon entropy across outbound counterparties

Entropy was calculated using:

$$H(a) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

where p_i is the fraction of outbound transactions to counterparty i.

Temporal Slicing and Clustering

Transaction history was segmented into rolling 14-day time windows (totaling 26 windows). Within each window:

1. Extract address-level degree and entropy features
2. Apply log-normalization to degree
3. Apply DBSCAN and HDBSCAN clustering algorithms
4. Store cluster labels and membership probabilities

Hyperparameter Selection

DBSCAN (ϵ, minPts) and HDBSCAN (minClusterSize, minSamples) parameters were selected via grid search using:

- Silhouette score
- Calinski-Harabasz index
- Davies-Bouldin index

Cohort Formation and Role Annotation

Addresses were assigned to behavioral cohorts based on their cluster volatility and entropy swings across windows:

- High Stability
- Low Stability
- Frequent Appearance
- Entropy Transition

Stability score was computed as the fraction of unique cluster labels assigned across all windows.

Statistical Enrichment Testing

Fisher's Exact Test was used to compute enrichment of sanctioned entities within cohorts, reporting odds ratios and p-values.

Computational Runtime

The full pipeline required approximately 1 hour per complete run on GPU-enabled Colab instances, due to repeated clustering across temporal slices.

Results

The final dataset included 4.2 million addresses across 26 temporal windows. Of these, 475,989 addresses appeared in multiple windows, enabling temporal tracking.

Cohort Distribution:

- High Stability: 162 addresses
- Low Stability: 80,980 addresses
- Frequent Appearance: 32,862 addresses
- Entropy Transition: 2,683 addresses

Cohort Enrichment (Fisher's Exact Test):

- Low Stability: OR = 12.98, $p < 10^{-80}$
- High Stability: OR = 0.00, $p < 10^{-44}$
- Entropy Transition: OR = 0.01, $p < 10^{-40}$

These results indicate a strong enrichment of sanctioned entities in unstable behavioral cohorts, particularly Low Stability. High Stability and Entropy Transition cohorts showed significant depletion.

Visualizations included:

- Entropy and degree histograms (log scale)
- Temporal entropy drift plots by cohort
- 3D cohort trajectories over time

Conclusions

This study demonstrates that Ethereum address behavior becomes increasingly clusterable over time. Entropy and degree, when tracked temporally, reveal latent roles that are forensically relevant.

Most Effective Methods:

- HDBSCAN: Robust clustering with soft membership and hierarchical insight.
- Temporal entropy tracking: Captures dynamic obfuscation or convergence behavior.

Least Effective:

- DBSCAN: Less flexible and overly coarse, especially in sparse windows.

Trends:

- High Stability implies operational or infrastructure wallets.
- Low Stability reflects airdrop exploitation, laundering, or disposable actors.
- Entropy transitions capture strategy pivots.

Limitations:

- No off-chain attribution
- Assumes transaction independence
- Cohort definitions rely on heuristic thresholds

Future work may incorporate token semantics, contract calls, or multichain identities.

References

[1] Meng Li. Application of cluster analysis in bitcoin deanonymization. In *Advances in Cyber Security and Intelligent Analytics*, pages 337–347. Springer, 2022.

[2] Claudia Malzer and Marcus Baum. Hdbscan$^{\epsilon}$: An alternative cluster extraction method for hdbscan, 2019. Available at SciSpace.

Appendices

A. Code Repository

All source code used for this project is available at:

`https://github.com/stevensoma21/eth-deanon`

The repository includes:

- Data preprocessing and feature extraction scripts
- Temporal clustering functions
- Cohort assignment logic
- Enrichment testing and statistical analysis
- Visualization utilities and plot generation

B. Additional Figures

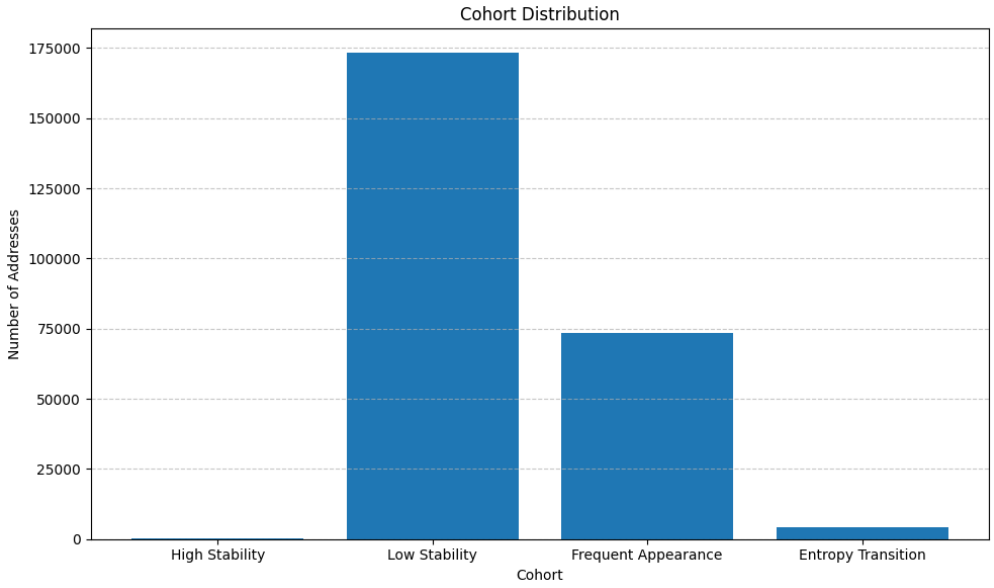

Figure 1: Cohort Distribution

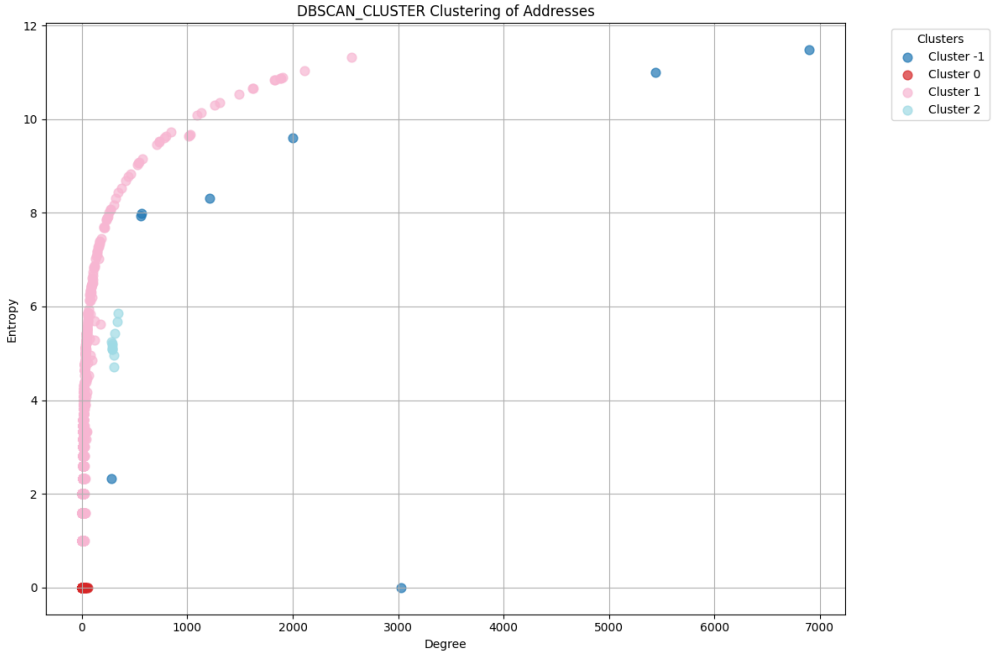

Figure 2: DBSCAN Clusters

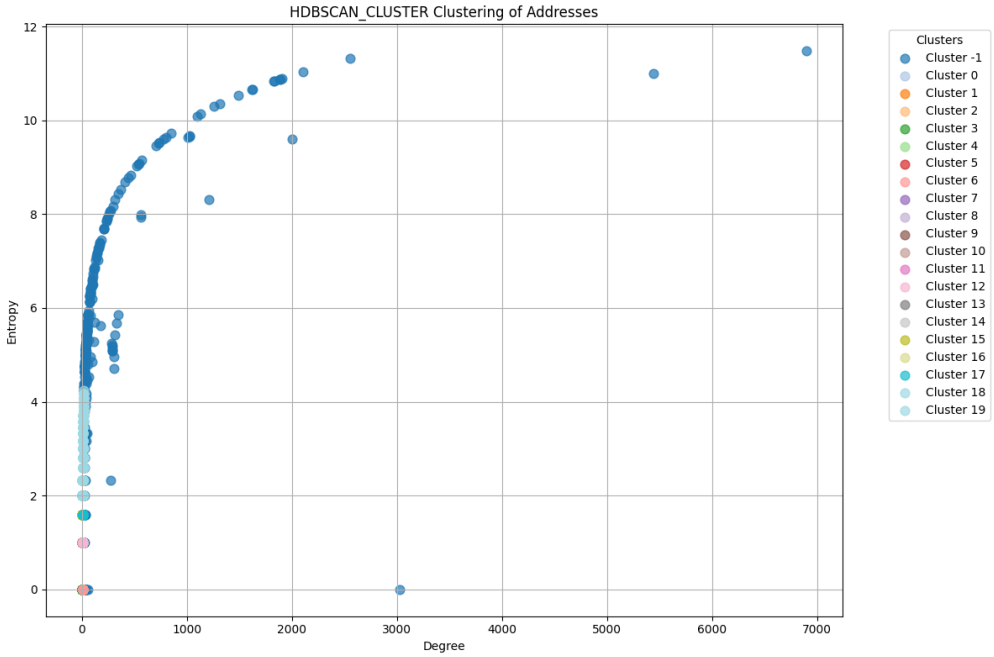

Figure 3: HDBSCAN Clusters

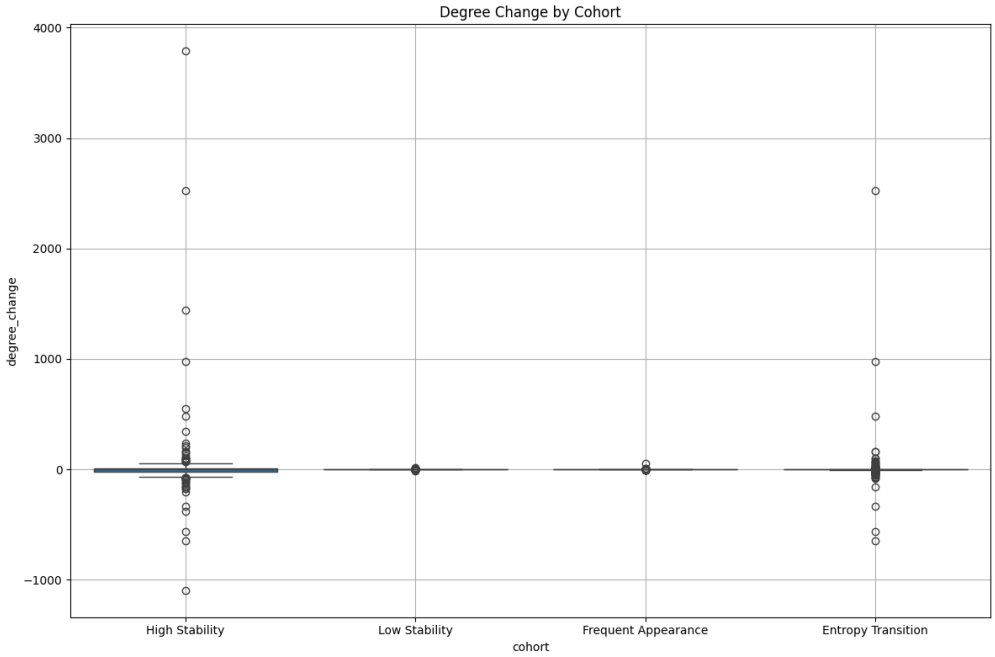

Figure 4: Cohort Degree Change Boxplot

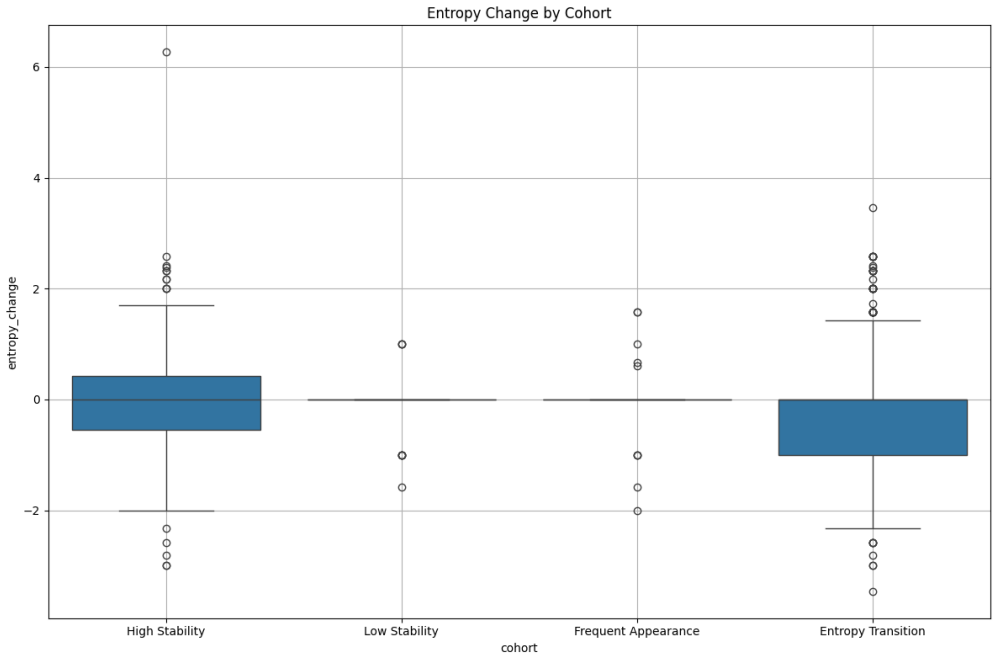

Figure 5: Entropy Change BOxplot