ARTEMIS Brief

Summary of ARTEMIS Method

ARTEMIS is a GNN system designed to detect airdrop hunters in NFT markets. Its core idea is to represent a NFT market as a graph where wallet addresses are nodes and transactions are edges. It then uses a GNN to learn the behavioral patterns that distinguish hunters from legitimate users. I will summarize the core ideas of ARTEMIS by: 1. how the graph is constructed, and 2. how the GNN learns from its graph

How the Graph is constructed: Each node (wallet) is described by a feature vector containing its behavioral and financial information (transaction frequency, account age, token balances ...). Each edge (transaction) contains both financial details and a multimodal embedding of the NFT involved. The multimodal embedding of NFT is generated by processing the NFT's image and text through a pretrained ViT and BERT, which creates a representation of the asset's content.

How GNN learns from the graph: it uses `ArtemisNet` which processes the graph layer by layer to build a contextual understanding of each node. Its first layer is a custom message-passing module that creates an initial signal by concatenating the features of a neighboring node with the features of the edge connecting to it. This combined information is then aggregated from all neighbors to form an initial "fingerprint" for each node. Subsequent layers, using a standard GraphSAGE architecture, expand this process to incorporate information from nodes up to three hops away.

Because of scale of the transaction graph, ARTEMIS uses `NeighborSamplerbyNFT` to sample and construct a small localized subgraphs for it to train on. It samples by iteratively selects nodes and a small subset of their neighbors.

After ARTEMIS has processed a subgraph, the final learned feature vector for each node is concatenated with its original input features and passed to a MLP layer, which performs the final classification, outputting a probability score of the node being an airdrop hunter.

Limitations of the ARTEMIS Model

I will not state one but 5 limitations I see in ARTEMIS:

1. Bias from Heuristic Based Ground Truth

ARTEMIS' understanding of a hunter is shaped by the heuristics used to label the training data (`y.pt`) (how the ground truth is constructed). This introduces a significant definitional bias, which might hinder the model's ability to learn to find all hunters, but rather to find users who match a specific, pre-defined stereotype. I will quickly analyze somet of the consequences of this bias:

- **Advanced attackers that violate heuritics assumptions (False Negatives):** The labeling heuristics (single funding source, rapid consolidation, ...) are effectively describing simple scripted Sybil attacks. However, a more advanced adversary could evade detection by:

    1. Using exchanges for funding and consolidation (Funding each Sybil wallet from a unique exchange address and sending the airdropped tokens back to unique exchange deposit addresses.) This would break the core heuristics assumptions.
    2. Obfuscating timing (Spreading transactions over days or weeks to mimic organic human behavior) This would also break the core heuristics assumptions.

    The model was never trained on these patterns and would likely classify these advanced attackers as legitimate users.

- **Risk of Punishing Legitimate Users (False Positives):** Legitimate activities can sometimes mimic the patterns of a Sybil attack which in terms will create a risk of false positive. Here are sone examples I came up with:

    1. Centralized Payouts: A company can pay hundreds of freelancers or a foundation distributing grants, and this would appear as a suspicious "fanout" of funds in the heuristic assumptions.
    2. Power Users: some traders might use multiple wallets for asset management which could trigger flags related to fund consolidation

The **core limitation** in this is that the constructed ground truth promotes the model to equate hunting with a specific set of programmatic behaviors, rather than a deeper, more conceptual understanding of the attacker's goal. (overfit to the heuritics). **I think this is the most significant limitation of ARTEMIS** (in experiment results and data, how much do they actually reflects the true ability of ARTEMIS).

Architectural and Methodological Limitations

The limitation above is from the data and all of the limitations below are related to model's architecture and design choises.

2. **Locality Problem of Graph CNN:** ARTEMIS uses a 3 layer GNN, and it has a receptive field of only **three hops** on the transaction graph. This causes nearsightedness in its "vision" for information: It is architecturally blind to events and relationships beyond this local neighborhood. This makes it vulnerable to attackers who use long funding chains (for example 5+ transactions) to obscure the source of their funds. And the model cannot see the full picture.

3. **Incomplete Information due to sampling:** ARTEMIS uses a neighborhood sampler that only considers a small, fixed number of a node's neighbors at each step (8 in the first layer). I think the motivation is to handle the massive scale of the graph: so to reduce noise and unimportant information and it is better for computation. But as a result, if a wallet has thousands of transactions, the model only sees a tiny fraction of its activity. The most important evidence and information may be in the transactions that were not sampled, which will lead the model to make decisions based on incomplete information.

4. **Learn and biased on Multimodal Contents:** ARTEMIS's use of multimodal NFT features (from images and text) can be a double-edged sword. It may learn (overfit) incorrect correlations between the content of popular NFTs and suspicious behavior. But both legitimate collectors and airdrop hunters could target the same high-value collections. This could lead to flagging legitimate users simply because they trade in a "risky" asset class, without sufficient evidence of malicious behavior.

5. **Ignoring Time (Static Graph):** ARTEMIS treats the transaction history as a single, static graph. While timestamps are included as features, the GNN architecture does not explicitly model sequence, causality, or flow of events over time. This is a major limitation, as the order of operations is a critical piece of evidence in detecting financial exploits. The model has a poor understanding of concepts like "A happened *before* B," which is incrediably important and fundamental informtion to this problem domain.