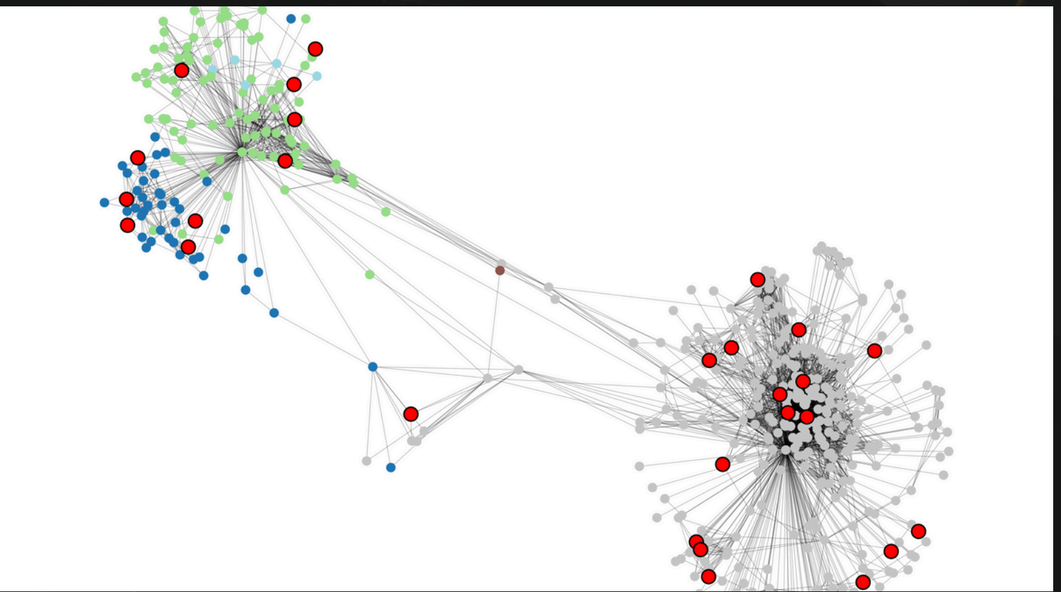
Bot Detection on Social Graphs: An Adversarial Analysis

This report details an analysis of bot detection techniques on social graphs under various adversarial attack scenarios. Using the Facebook Combined Ego Network, we evaluate the robustness of a Random Forest classifier against structural evasion and graph poisoning attacks. Our findings reveal unexpected outcomes where adversarial manipulations, rather than concealing bots, often make them more detectable due to the introduction of structural anomalies.



Dataset & Graph Overview: Foundations for Analysis

To construct a realistic social graph for evaluating bot detection, we utilized the Facebook Combined Ego Network dataset. This dataset allowed us to model real-world user connections as an undirected social graph, providing a robust environment for our experiments. We introduced a 5% bot population, randomly assigned to nodes, ensuring they initially blended seamlessly within the network.

Nodes: 4,039

Representing individual users within the social network.

Edges: 88,234

Indicating connections or relationships between users.

Bot Percentage: 5%

Bots were randomly assigned, initially indistinguishable from legitimate nodes.

This setup provided a realistic, large-scale network, crucial for evaluating bot detection techniques under challenging, adversarial conditions. The random assignment of bots made the baseline detection particularly difficult, as they lacked any inherent structural signals distinguishing them from legitimate users.

Extracted Graph-Based Features

For each node in the social graph, a comprehensive set of structural features was extracted. These features are fundamental in characterizing node behavior and connectivity patterns, forming the basis for our bot detection classifier. They were meticulously calculated and stored for analysis across baseline, evasion, and poisoning attack scenarios.

Degree

Number of connections a node has.

Clustering Coefficient

Measure of the degree to which nodes in a graph tend to cluster together.

Betweenness Centrality

Indicates a node's influence over the flow of information.

Closeness Centrality

Measures how close a node is to all other nodes in the network.

PageRank Score

Estimates a node's importance or influence within the network.

Eigenvector Centrality

Measures a node's influence based on its connections to other influential nodes.

Community Assignment

Identifies the community structure using Greedy Modularity.

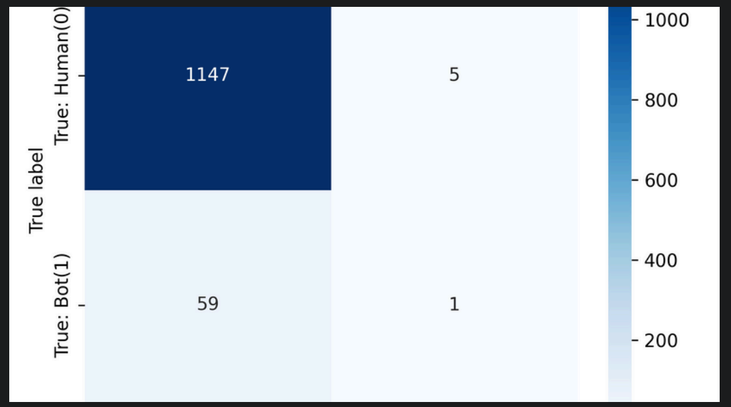
Baseline Model Analysis: The Challenge of Random Bots

The initial bot detection model, a Random Forest classifier, was trained on the extracted graph-based features without any adversarial intervention. Due to the random assignment of bots, which resulted in them being structurally indistinguishable from legitimate nodes, the model faced significant challenges. The core issue was the lack of distinctive structural signals that could differentiate bots from regular users, leading to poor detection performance.

Performance Metrics

- **Accuracy:** 0.9472
- **Bot Recall:** 0.0167 (Nearly zero bots detected)
- **Bot Precision:** 0.1667

Interpretation: The Random Forest classifier, despite high overall accuracy, virtually failed to identify bot nodes. This is because the randomly placed bots blended naturally into the network, lacking any unique structural characteristics that the model could leverage for detection. Their structural similarity to legitimate users made them effectively invisible to the algorithm.



Confusion Matrix - Baseline Model

The confusion matrix clearly illustrates the model's inability to detect bots (low recall), as it misclassifies almost all actual bots as legitimate users.

Key Takeaway: When bots are randomly distributed and exhibit no anomalous structural patterns, graph-based features alone are insufficient for effective detection. This highlights the need for more sophisticated detection mechanisms or analysis of bot behavior under adversarial attacks.

Adversarial Attacks: Evasion and Poisoning

We next explored how two distinct adversarial attacks—structural evasion and graph poisoning—impact bot detectability. Paradoxically, both attacks, intended to conceal bots, made them more identifiable by introducing detectable anomalies into their network structure.

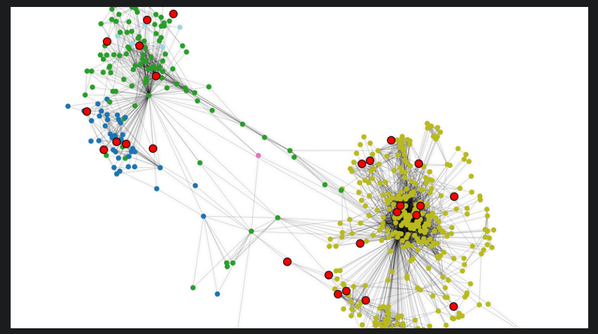
Structural Evasion Attack

Bots attempt to disguise themselves by removing a few existing edges and adding new ones to random nodes.

Graph Poisoning Attack

Bots aim to become highly connected hubs by adding multiple fake edges, inflating their centrality metrics.

Structural Evasion: Unintended Exposure



Graph After Structural Evasion

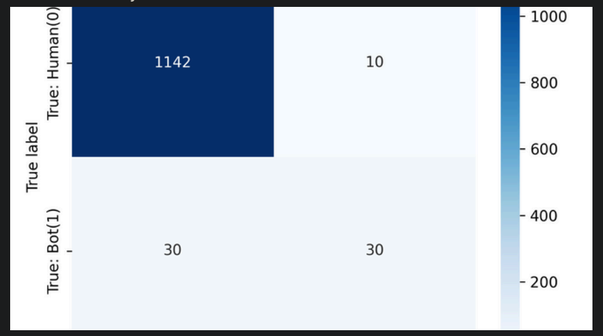
The visual representation shows a subtle but noticeable change in bot connectivity after evasion.

In an attempt to evade detection, bots subtly modified their connectivity by dropping existing edges and forming new, random connections. The unexpected outcome was that this manipulation made their structural patterns less natural. The classifier was able to pick up on these anomalies, leading to a significant increase in bot detection.

Performance Metrics

- **Accuracy:** 0.9670
- **Bot Recall:** 0.50
- **Bot Precision:** 0.75

Interpretation: The attack, rather than hiding bots, unintentionally created abnormal connectivity that the classifier effectively captured. Bots became more isolated or unusually connected, increasing their detectability.



Confusion Matrix - Structural Evasion

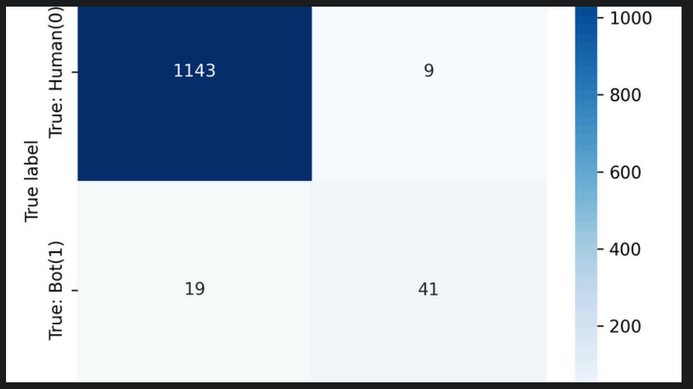
Graph Poisoning: The Hub Effect

The graph poisoning attack involved bots aggressively adding numerous fake edges, effectively transforming themselves into hyper-connected hubs. This manipulation drastically inflated their structural metrics, such as Degree, Betweenness Centrality, PageRank, and Eigenvector Centrality. This extreme deviation from normal node behavior made bots highly anomalous and easily detectable.

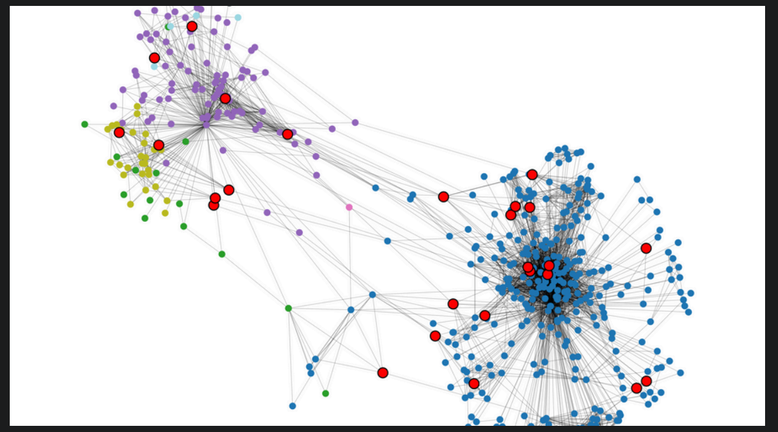
Performance Metrics

Accuracy: 0.9769
Bot Recall: 0.6833
Bot Precision: 0.82

Interpretation: Bots subjected to poisoning became extremely anomalous due to their excessive connectivity. Their artificially inflated centrality scores sharply distinguished them from legitimate users, allowing the classifier to detect them with high reliability and precision.



Confusion Matrix - Graph Poisoning



Graph After Poisoning Attack

The graph clearly shows bots as highly prominent, over-connected hubs after the poisoning attack.

Comparative Performance Analysis & Key Insights

A comparative analysis of bot detection performance across baseline, structural evasion, and graph poisoning scenarios reveals crucial insights into the impact of adversarial attacks on social graph analysis.

Scenario	Accuracy	Bot Recall	Bot Precision
Baseline	0.9472	0.0167	0.1667
Structural Evasion	0.9670	0.50	0.75
Poisoning Attack	0.9769	0.6833	0.82

Key Insights from the Analysis

1

Baseline Challenges

Bots in the baseline scenario remained largely indistinguishable due to their random, natural placement within the network, yielding very low recall.

2

Evasion's Backfire

The structural evasion attack, intended to hide bots, inadvertently introduced structural anomalies that made bots more easily detectable by the classifier.

3

Poisoning's Potency

Graph poisoning dramatically increased bot detectability by turning bots into hyper-connected hubs, leading to highly anomalous centrality scores.

Overall Conclusion

> Both adversarial attacks, paradoxically, made bots easier to detect. The graph poisoning attack produced the strongest performance boost for the classifier, making bots highly visible due to their extreme structural deviations. This suggests that sophisticated bot detection models can leverage unusual network activity, even when it's designed for malicious purposes, to identify and mitigate bot presence. The classifier significantly improved its ability to identify the anomalous connectivity introduced by these attacks.