Assignment 2

Name: Rana Ashraf

1. Introduction

This assignment investigates the resilience of graph-based bot detection models under adversarial attacks. Using the Facebook Ego Network dataset from SNAP, we:

1. Built a social graph
2. Extracted essential network metrics
3. Trained a baseline machine-learning detector
4. Simulated two adversarial attacks
    - **Structural Evasion Attack**
    - **Graph Poisoning Attack**
5. Evaluated detector performance across three conditions
6. Tested the poisoned model on clean data to measure real poisoning impact
7. Visualized the graph before/after attacks

The goal is to assess how attackers can manipulate graph structure to evade or degrade detection.

2. Dataset and Graph Construction

The dataset **facebook_combined.txt** contains:

- **4039 nodes**
- **88,234 edges**

An undirected graph was constructed using NetworkX.

3. Graph Metrics Computed

For every node, the following metrics were calculated:

Metric	Purpose
Degree	Measures node popularity
Clustering Coefficient	Measures local connectivity
PageRank	Global influence
Eigenvector Centrality	Global importance
Betweenness Centrality	Control over communication paths
Community Detection (Louvain)	Structural grouping

These metrics form the **feature set** for the machine learning classifier.

4. Bot Simulation

To simulate malicious activity:

- **80 high-degree nodes** were labeled as **bots (class 1)**
- All other nodes were labeled as **normal users (class 0)**

This allows training/testing of a graph-based classifier.

5. Baseline Bot Detection Model

Model used: **Random Forest Classifier**
Training and testing was conducted with an 80/20 split.

Baseline Results (Clean Training → Clean Testing)

Metric	Value
Accuracy	0.9983

Confusion Matrix

	Pred Normal	Pred Bot
Actual Normal	1186	2
Actual Bot	0	24

Interpretation

- The baseline detector works extremely well.
- Bots are perfectly identified (recall = 100%).
- Almost no false positives.

6. Structural Evasion Attack

Attack Description

For each bot node:

1. Add edges between its neighbors
2. Create triangles to increase clustering

3. Make the bot appear more “humanlike”

This modifies local graph structure without affecting global topology heavily.

Evaluation After Structural Attack

Metric	Value
Accuracy	**0.9802**

Confusion Matrix

	Pred Normal	Pred Bot
Actual Normal	**1188**	**0**
Actual Bot	**24**	**0**

Key Observations

- All bots became **undetectable** (100% false negatives).
- Classifier predicts **everything as normal**.
- Accuracy remains high due to class imbalance (24 bots vs 1188 normal users).

Conclusion

Structural evasion is **highly effective**: Bots camouflage themselves by improving their clustering and connectivity.

7. Graph Poisoning Attack

Attack Description

Poison the training graph by injecting **200 random edges**, altering the structural distributions for both bots and normal users.

This corrupts the data used to train the model.

Evaluation After Poisoning (Training on Poisoned → Testing on Poisoned)

Metric	Value
Accuracy	**0.9802**

Exact same behavior as structural evasion:

- All bots are misclassified.
- Model collapses into predicting only normal users.

8. Testing Poisoned Model on Clean Data

To measure **true poisoning impact**:

- The model is trained on **poisoned data**
- Then tested on **clean, untouched data**

Results

Scenario	Accuracy
Baseline	0.99835
After Structural Attack	0.98020
After Poisoning	0.98020
Poisoned → Clean Test	**0.98020**

Interpretation

Even when tested on the original (clean) graph, the poisoned model:

- Completely fails to detect ANY bots
- Predicts class 0 for all cases
- Suffers from **total collapse of minority-class performance**

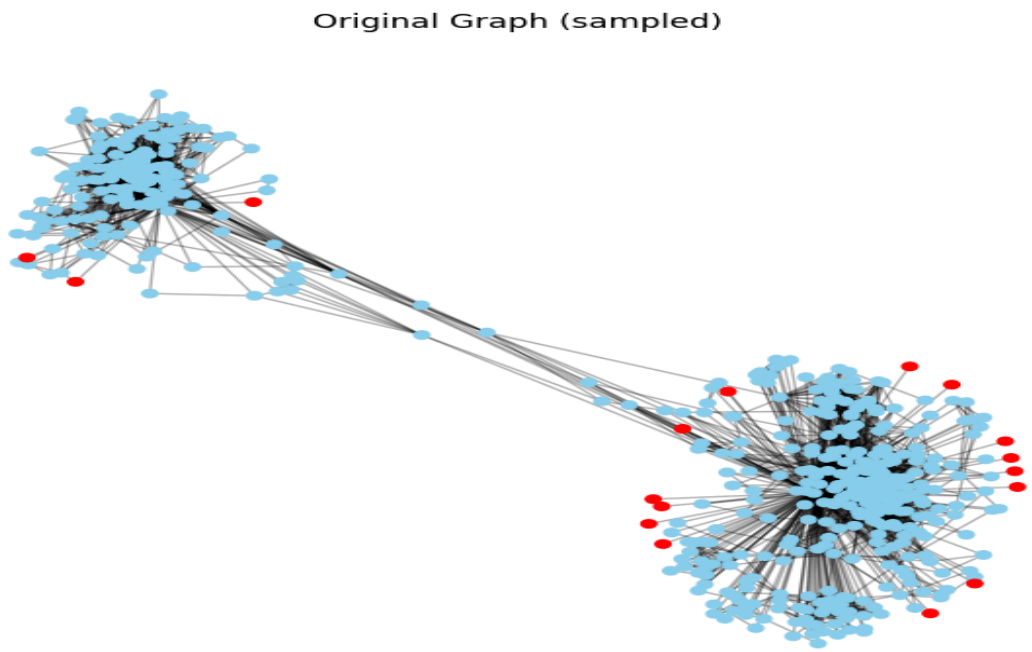
This confirms a successful poisoning attack.

9. Visualizations

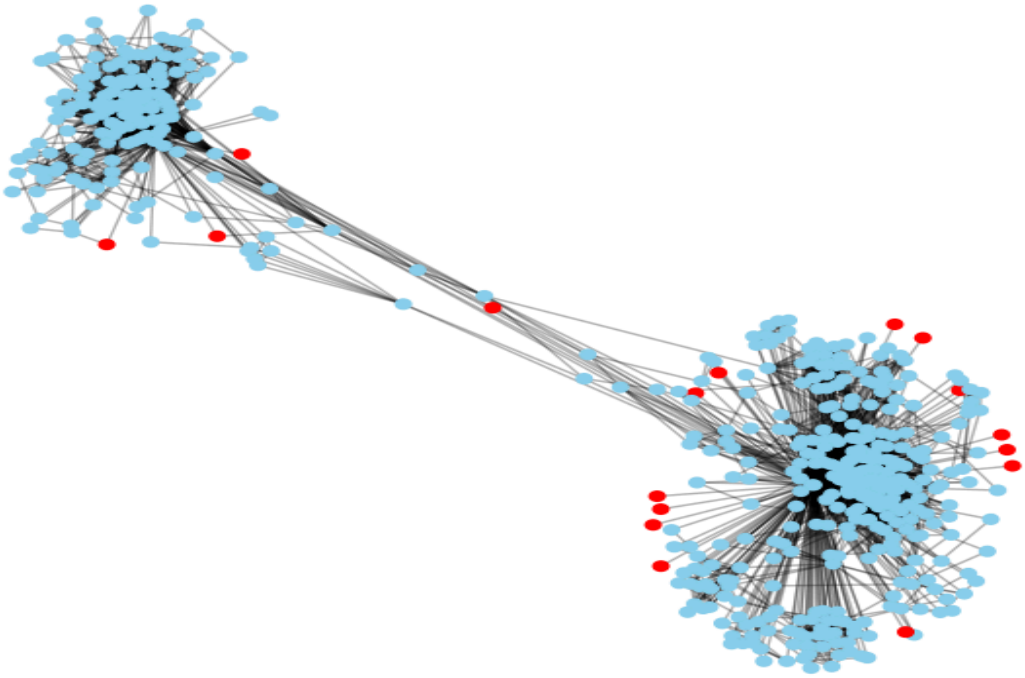
Three visualizations were generated:

1. **Graph Before Attacks**
2. **After Structural Evasion Attack**
3. **After Graph Poisoning Attack**

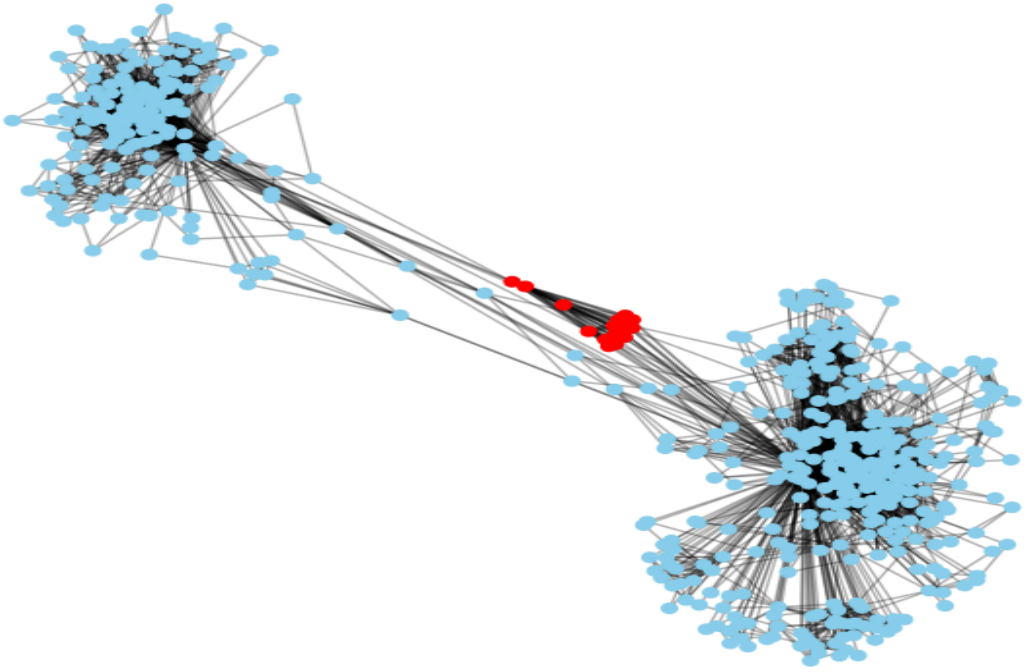
Bots were highlighted in **red** to show their transformation after attacks.



After Structural Evasion (sampled)



After Graph Poisoning (sampled)



10. Summary of Results

Scenario	Accuracy	Bot Detection	Notes
Baseline	0.9983	Excellent	Correctly detects all bots
Structural Evasion	0.9802	0%	Bots modify local patterns to appear normal
Graph Poisoning	0.9802	0%	Model collapses due to corrupted training data
Poisoned → Clean Test	0.9802	0%	Long-term degradation confirmed

11. Final Conclusions

1. **Baseline model is strong**, achieving near-perfect detection.
2. **Structural Evasion Attack is very successful** — bots adjust their graph position so they no longer appear anomalous.
3. **Graph Poisoning Attack corrupts the training process**, causing the classifier to always predict the majority class.
4. **Testing on clean data shows the damage persists**, proving poisoning permanently degrades the detector.
5. **A high accuracy after attack is misleading** due to class imbalance — the model is “broken” even though accuracy is high.

Final Takeaway

Graph-based detectors are **highly vulnerable** to simple structural and poisoning attacks. Defenses must incorporate:

- balanced training
- adversarial training
- robust community-based features
- anomaly detectors resilient to structural manipulation