Malak Mohamed Osman Mohamed-2205059

1. Dataset

The Facebook Egonets dataset represents friendship networks collected from real Facebook users. Each network is an ego network, meaning it contains:

- An ego user
- Their friends (nodes)
- The friendship links between those friends (edges)

Dataset summary:

- **4,039** nodes (users)
- **88,234** edges (friendships)
- No real bots → we inject **271 synthetic bots** based on structural anomalies:
  - **Degree above average + very low clustering**
    Nodes with many connections but almost no triangles, behaving unlike real social users.
  - **Low eigenvector centrality**
    Nodes connected to weak or unimportant parts of the network.
  - **Low betweenness centrality**
    Nodes that rarely lie on important shortest paths, meaning they do not bridge communities.
  - **Low closeness centrality**
    Nodes that are far from the rest of the network, structurally isolated.

Nodes meeting these anomaly conditions were labeled as synthetic bots → 271 total bots.

2. Graph Metrics

For each node we compute:

a) Degree

- Measures the number of direct connections.

b) Clustering coefficient

- Shows how strongly a node's neighbors are interconnected.

c) Centrality measures

- **Eigenvector centrality**: influence based on neighbors' influence

- **Betweenness centrality**: how often a node lies on shortest paths
- **Closeness centrality**: how close a node is to all others

d) Community detection

- We use the Louvain algorithm to assign each node to a community.

```
PS C:\Users\MALAK\Desktop\Social networks-Assignment2> python test.py
   node  degree  clustering   eigenvector  betweenness  closeness  community
0     0     347    0.041962  3.391796e-05     0.169475   0.353343          4
1     1      17    0.419118  6.045346e-07     0.000000   0.261376          4
2     2      10    0.888889  2.233461e-07     0.000000   0.261258          4
3     3      17    0.632353  6.635648e-07     0.000006   0.261376          4
4     4      10    0.866667  2.236416e-07     0.000000   0.261258          4
```

These metrics form the core features for the classifier.

3. Interpretion of three models

Accuracy stays very high (93–96%) in all 3 scenarios because:

- The dataset is extremely imbalanced
  - 1136 humans
  - ~76 bots
- That means bots represent only ~6% of the dataset.

So even if the model completely fails to detect bots, accuracy stays high:

- If the model predicts "human" for everything →
  Accuracy ≈ 1136 / 1212 = 94%

So, Accuracy hides model failures as it is misleading in imbalanced classification and should not be used alone.

3.1 Baseline Bot Detection Model

We use:

- Node2Vec to generate 64-dimensional graph embeddings
- GraphSAGE (2-layer GNN) for classification
- Train/test split: 70% / 30%
- Labels: bot = 1, human = 0

The baseline model is trained on the clean graph (no attack).

```
=========== BASELINE GNN RESULTS ===========
Accuracy: 0.9636963696369637
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1136
           1       0.80      0.57      0.66        76

    accuracy                           0.96      1212
   macro avg       0.88      0.78      0.82      1212
weighted avg       0.96      0.96      0.96      1212
```

Interpretation

- Strong performance on humans.
- Bot recall = 0.57, meaning the model detects 57% of bots and misses 43%.

3.2. Structural Evasion Attack

Objective:

- Make bots look structurally similar to humans.

Method:

- Each bot connects to high-degree influencer nodes
- Adds 7 new edges per bot
- Bots move deeper inside real communities
- Their embeddings become more "human-like"

Effect:

- Graph structure changes, but labels remain unchanged.

```
=========== EVASION ATTACK RESULTS ===========
Accuracy: 0.943069306930693
              precision    recall  f1-score   support

           0       0.94      1.00      0.97      1136
           1       0.77      0.13      0.22        76

    accuracy                           0.94      1212
   macro avg       0.86      0.56      0.60      1212
weighted avg       0.93      0.94      0.92      1212
```

Interpretation

- Bots successfully hide inside densely connected human clusters.
- Bot recall drops from 0.57 → 0.13 (a ~77% decrease).
- The model still performs well on humans, but becomes nearly blind to bots.
- Evasion is effective, but not as damaging as poisoning.

3.3 Graph Poisoning Attack (Label Flipping)

Objective:

- Corrupt training labels so the model learns wrong rules.

Method:

- Select bots inside the training set only
- Flip 30% of bot labels from 1 → 0
- No structural changes to the graph

Effect:

- The model is trained to believe some bots are humans, destroying decision boundaries.

```
=========== POISONING ATTACK RESULTS ===========
Accuracy: 0.9364686468646864
              precision    recall  f1-score   support

           0       0.94      1.00      0.97      1133
           1       1.00      0.03      0.05        79

    accuracy                           0.94      1212
   macro avg       0.97      0.51      0.51      1212
weighted avg       0.94      0.94      0.91      1212
```

Interpretation

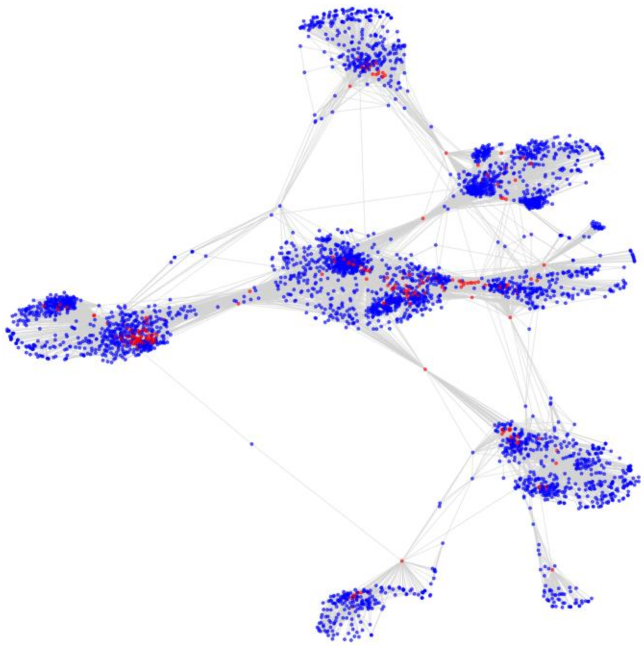
- Bot recall collapses from 0.57 → 0.03 (a 94% reduction). This means the model only detects 3 bots out of 100 on average.
- The model almost completely loses the ability to detect bots.
- Accuracy remains high due to imbalance, but this is misleading.
- Poisoning is the most destructive attack by far.

4. Visualization of Attack Scenarios:

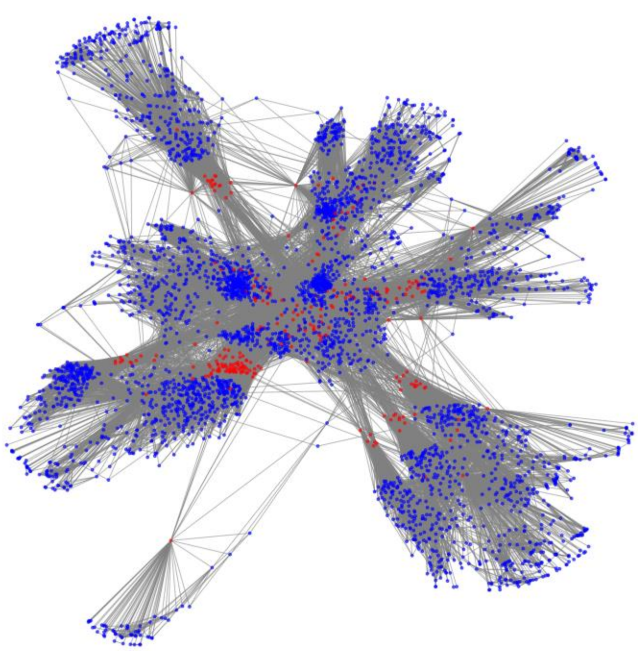
Color scheme:

- Blue = humans
- Red = bots

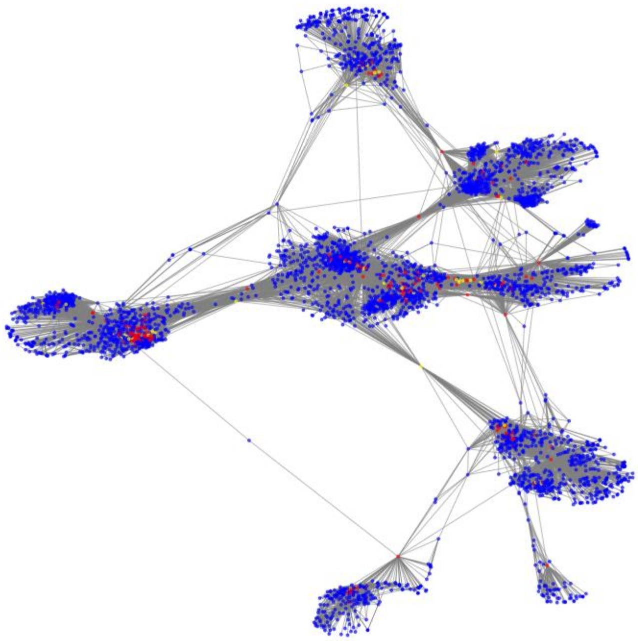
a) Baseline graph:



b) Evasion attack graph:



c) Poisoning attack graph:



- Yellow nodes = flipped nodes (1->0)