Social lab

Report

By

Omar Hossam Ahmed

Social Network Bot Detection Assignment Report

1. Introduction

The objective of this assignment was to build a machine learning pipeline to detect bots in a social network graph. Using a real-world Facebook graph dataset, we extracted graph-based features, trained a classifier, and evaluated its performance. Additionally, we implemented structural evasion and graph poisoning attacks to study model robustness.

2. Dataset

- **Graph:** Facebook combined network from SNAP.
- **Nodes:** Users of the network.
- **Edges:** Friendships (undirected).
- **Labels:** Users vs. bots (simulated 3–5% of nodes as bots).

Statistics:

Class	Count
User (0)	3917
Bot (1)	122

Note: The dataset is highly imbalanced, which affects model evaluation.

3. Pipeline Overview

The pipeline consists of several stages:

1. **Graph Loading**

```
%run src/load_graph.py --input data/facebook_combined.txt.gz --output
results/graph_baseline.pkl
```

   a. Reads the SNAP edge list.
   b. Converts to a NetworkX graph.
   c. Saves as a pickle (`.pkl`) file for fast reloading.

2. **Feature Extraction**

```
%run src/features.py --graph results/graph_baseline.pkl --out
results/features_baseline.csv
```

Extracted features per node:

   a. Degree
   b. Clustering coefficient
   c. PageRank
   d. Number of triangles
   e. Average neighbor degree
   f. Standard deviation of neighbor degrees

3. **Label Generation**

```
%run src/generate_labels.py --graph results/graph_baseline.pkl --out
results/labels.csv --bot_rate 0.03
```

   a. Bots simulated based on **anomaly scores** (high degree, high betweenness, low clustering).

4. **Model Training**

```
%run src/train.py --features results/features_baseline.csv --model_out
results/model_baseline.pkl
```

a. Random Forest classifier
b. Stratified train/test split
c. Evaluated using precision, recall, F1-score, and AUC.

5. **Attacks**

```
%run src/attacks.py --attack evasion --graph
results/graph_baseline.pkl --out results/graph_evasion.pkl --bots_file
results/bots.txt --budget 3
```

a. **Structural Evasion:** Connect bots to high-degree nodes to hide them.
b. **Graph Poisoning (optional):** Inject new attacker-controlled nodes.

6. **Evaluation After Attack**
   a. Features extracted from attacked graph.
   b. Model retrained to observe changes in performance.

4. Metrics

Baseline Model Performance:

Class	Precision	Recall	F1-score	Support
User (0)	0.95	1.00	0.97	1152
Bot (1)	0.00	0.00	0.00	60

- **Accuracy:** 0.95
- **Macro F1:** 0.49 → shows poor bot detection
- **Weighted F1:** 0.93 → dominated by majority class
- **AUC:** 0.505 → close to random

Observation: Model predicts users correctly but **fails to detect bots**, due to class imbalance and similarity between benign and bot features.

After Structural Evasion Attack:

- Accuracy remained 0.95 (still dominated by users).
- Macro F1 and AUC slightly changed (AUC ~0.506).
- Structural evasion did **not drastically change model performance**, indicating model relies on simple node-level features.

5. Visualizations

1. **Label Distribution**
    a. Bar chart of users vs bots.
2. **Feature Correlation Heatmap**
    a. Correlation between features helps detect redundancy.
3. **ROC Curve**
    a. Shows trade-off between true positive rate and false positive rate.
4. **Graph Visualization**
    a. Sampled subgraph (200 nodes) colored by label:
        i. Blue → user
        ii. Red → bot

Visuals help understand graph structure and model input features.

6. Discussion

- The model struggles with **imbalanced data**. Precision/recall for bots is zero.
- Structural evasion attacks didn't reduce accuracy but slightly affected AUC.
- Class imbalance is a key challenge for bot detection in social networks.
- Future improvements:
    - Use **oversampling** or **SMOTE** to balance labels.
    - Use **graph neural networks** (GNNs) for relational learning.
    - Combine node features with edge/graph-level features.

7. Conclusion

- Successfully implemented a **graph-based bot detection pipeline**.
- Demonstrated **feature extraction, labeling, model training, and attacks**.
- Highlighted limitations in detecting **rare classes (bots)** and importance of **robust evaluation metrics** beyond accuracy.

8. References

- SNAP Facebook dataset: https://snap.stanford.edu/data/
- NetworkX documentation: https://networkx.org/
- Scikit-learn documentation: https://scikit-learn.org/