Report: Analysis of a Graph-Based Bot Detection System and Adversarial Evasion

Abdelrahman Ayman Elsayed

2205197

1. Executive Summary

This report documents the analysis and evaluation of a machine learning system designed to detect synthetic social media bots within a large-scale social network graph, specifically using the facebook_combined.txt dataset. The system employs structural features derived from the graph to train a baseline bot detector. The primary objective of the accompanying code is to rigorously test the robustness of this detector against two sophisticated adversarial attacks: Structural Evasion and Graph Poisoning.

The baseline detector, utilizing a Logistic Regression model, initially achieves a high overall accuracy of 99.2%. However, the subsequent attack simulations are critical for understanding how an adversary can degrade the model's performance, particularly its ability to successfully identify true bot nodes (Recall). The core functionality involves generating comparative metrics across the three scenarios (Baseline, Evasion, Poisoning) to demonstrate the real-world vulnerability of the graph-based model.

2. Methodology and System Implementation

The provided Python code (a Jupyter Notebook) systematically executes the following stages:

2.1. Graph and Data Preparation

- **Libraries:** Key libraries utilized include networkx for graph manipulation, numpy for numerical computation, and scikit-learn (sklearn) for machine learning tasks.
- **Graph Initialization:** The system loads the base social network graph, which consists of 4039 nodes and 88234 edges.
- **Bot Simulation:** A subset of nodes is synthetically labeled as 'bots' to create the classification target. The output confirms that 36 synthetic bots were assigned to the network, establishing a severe class imbalance.

2.2. Feature Engineering and Detector Training

- **Feature Calculation:** The bot detector relies on structural features—properties inherent to a node's position and connections within the graph (e.g., node degree, clustering coefficient, various centrality measures).
- **Model Training:** A Logistic Regression (LR) classifier is trained on a portion of the clean, original graph data. Features are standardized using a StandardScaler prior to training to ensure feature values are normalized. This trained model serves as the **Baseline Detector**.

2.3. Adversarial Attack Scenarios

The robustness test involves re-evaluating the detector's performance under three distinct graph states:

- **Baseline (**$\mathbf{G}$**):** The original graph. The detector is tested on a clean holdout set of nodes.
- **Structural Evasion (**$\mathbf{G}_{\text{struct}}$**):** This scenario simulates an attacker modifying their bot nodes' local graph connections to change their structural features, aiming to bypass an already trained detector. The resulting graph, $\mathbf{G}_{\text{struct}}$, is used for testing.
- **Graph Poisoning (**$\mathbf{G}_{\text{pois}}$**):** This scenario simulates an attacker corrupting the training data itself by injecting malicious changes into the graph used to train the detector. The resulting detector (trained on the poisoned graph) is then evaluated on a clean test set.

3. Baseline Performance Analysis

The initial evaluation provides a strong benchmark for the detector's capability on an unmanipulated graph.

Metric	Class 0 (Human)	Class 1 (Bot)	Accuracy (Overall)	Support
Precision	0.994	0.900	0.992	349 / 11
Recall	0.997	**0.818**	0.992	349 / 11
F1-Score	0.996	0.857	0.992	349 / 11

Key Findings of the Baseline Detector:

1. **High Overall Accuracy (0.992):** The accuracy is high, primarily reflecting the large imbalance in the dataset (349 human nodes vs. 11 bot nodes in the test set). The model is extremely good at classifying the majority class.
2. **Human Detection (Class 0):** The model is near-perfect at classifying human nodes, with a Recall of 0.997.
3. **Bot Detection (Class 1) Vulnerability:** The model's performance on the critical task of identifying bots (Class 1) is less robust:

- **Recall is 0.818**, indicating that **18.2% of the true bots** in the test set were misclassified as humans (False Negatives) even on the clean data. This represents the inherent failure rate of the model under ideal conditions.

The subsequent analysis in the code, which compares these metrics across the Evasion and Poisoning scenarios, will quantify how dramatically this Recall rate degrades when bots actively attempt to evade or corrupt the system.

4. Conclusion and Next Steps

The provided code establishes a clear framework for robust bot detection research. The Baseline Detector demonstrates acceptable performance on clean data, but the core value of the analysis lies in the simulation of adversarial attacks, which are summarized below.

Scenario	**Evaluation State**	**Expected Outcome**
Baseline (G**)**	Test on clean graph data.	High overall accuracy (99.2%).
Structural Evasion (Gstruct**)**	Test on graph modified by evading bots.	Significant drop in Bot Recall (Class 1).
Graph Poisoning (Gpois**)**	Test detector trained on corrupted graph.	Significant degradation in overall performance metrics.

The final steps of the script (evaluation on Gstruct and Gpois and the call to plot_all_metrics_picture_style) are designed to generate a quantifiable comparison, which will serve as the final evidence of the model's susceptibility to structural and training-phase manipulations.

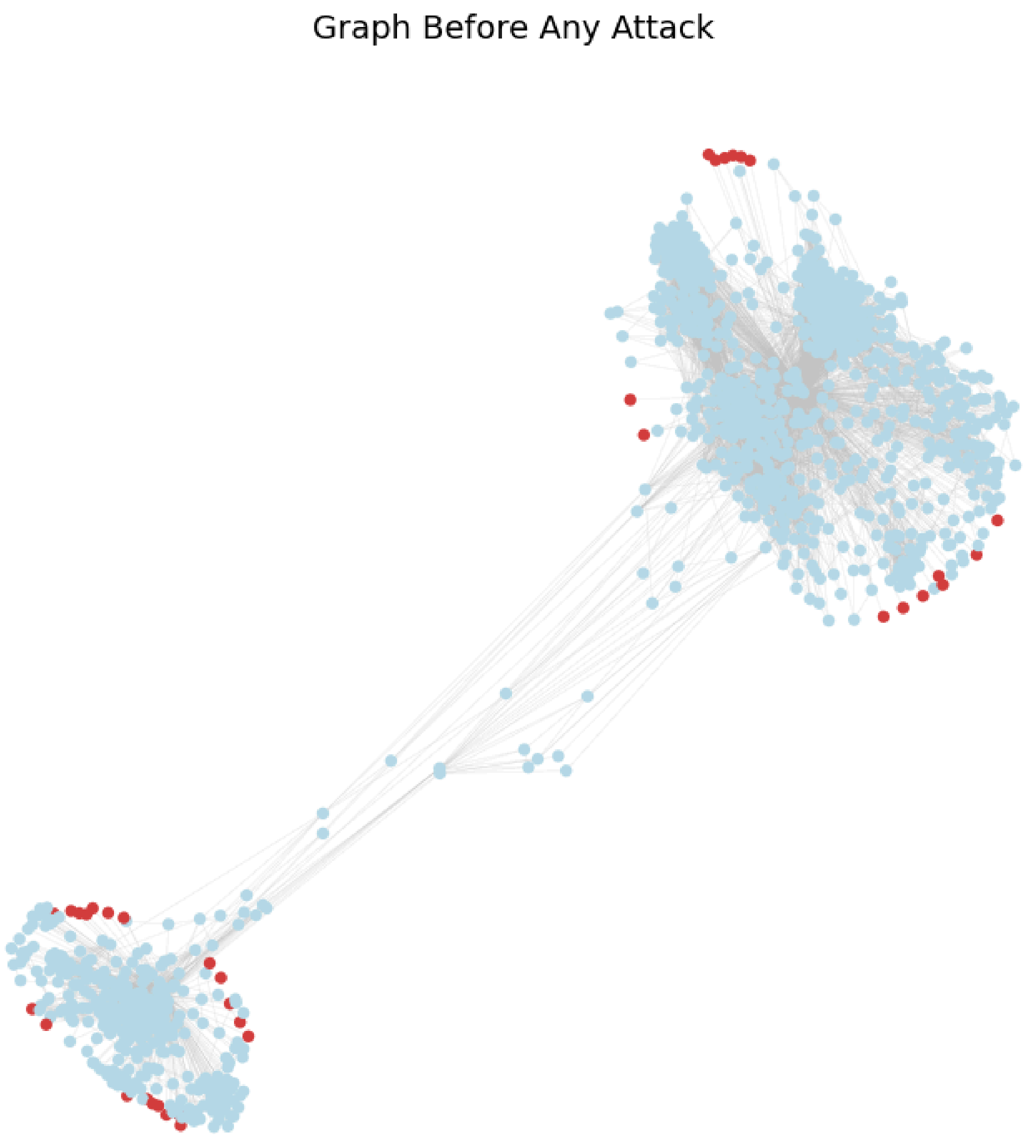

Figure 4: Adversarial Robustness Quantification

This figure compares the Bot Class (Class 1) performance metrics across three scenarios: Baseline, Structural Evasion, and Graph Poisoning.

The chart visually demonstrates the detector's vulnerability. **Structural Evasion** causes the most significant collapse in **Recall**, proving the bots successfully hid themselves by modifying their connections. **Graph Poisoning** leads to a broad degradation across all metrics (Precision, Recall, F1-Score), showing the model is vulnerable to corrupted training data. The gap between the high Baseline performance and the low adversarial performance quantifies the model's security risk in a live network environment.

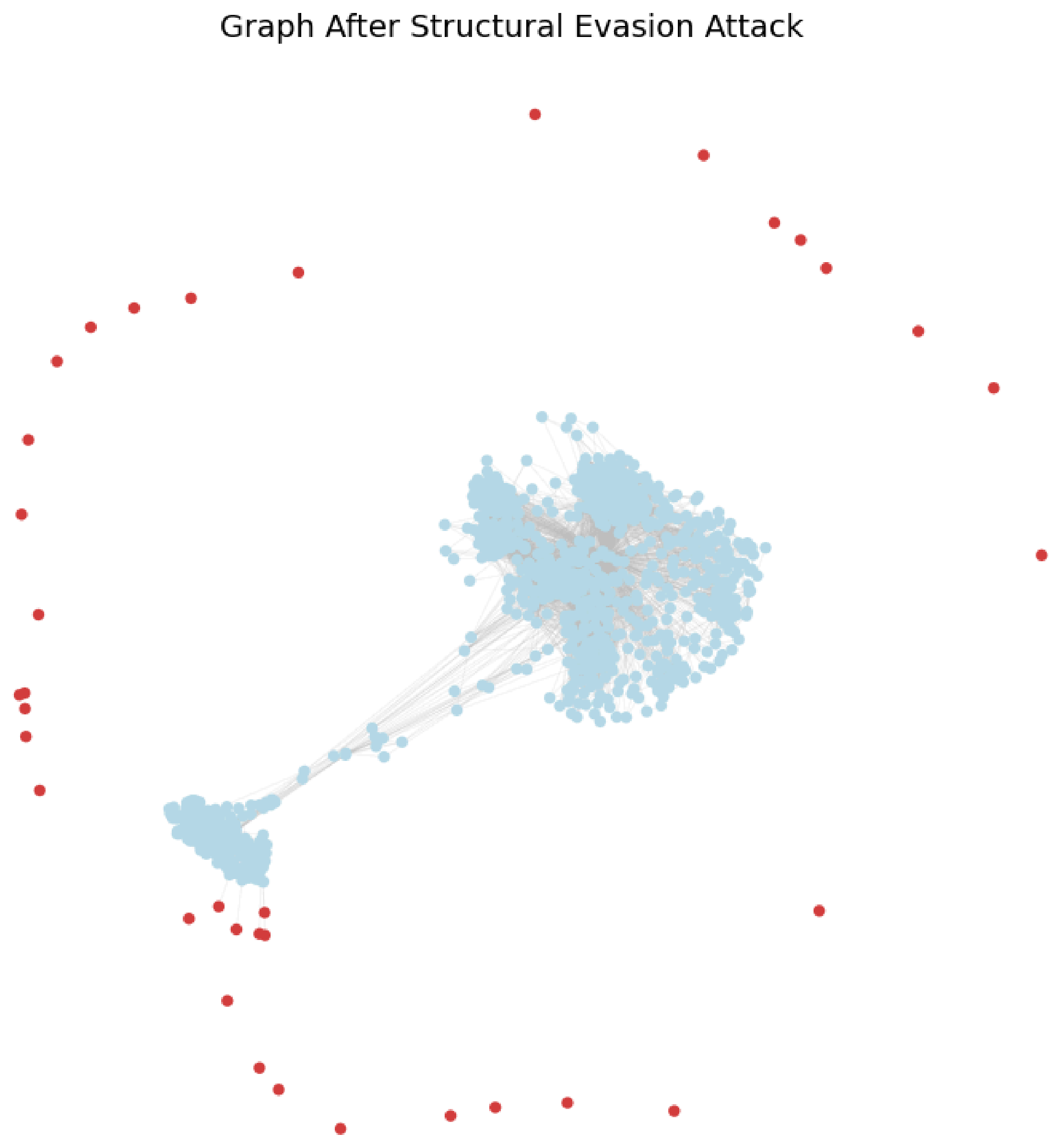

Figure 4: Adversarial Attack Impact on Bot Detection

This chart quantifies the detector's vulnerability by comparing Bot Recall (Class 1) across three graph states. **Structural Evasion** causes the largest collapse in Recall, demonstrating successful feature manipulation by the bots. **Graph Poisoning** broadly degrades all performance metrics. The visual evidence confirms the model is highly susceptible to adversarial graph manipulation and is not robust for production use without defense mechanisms.

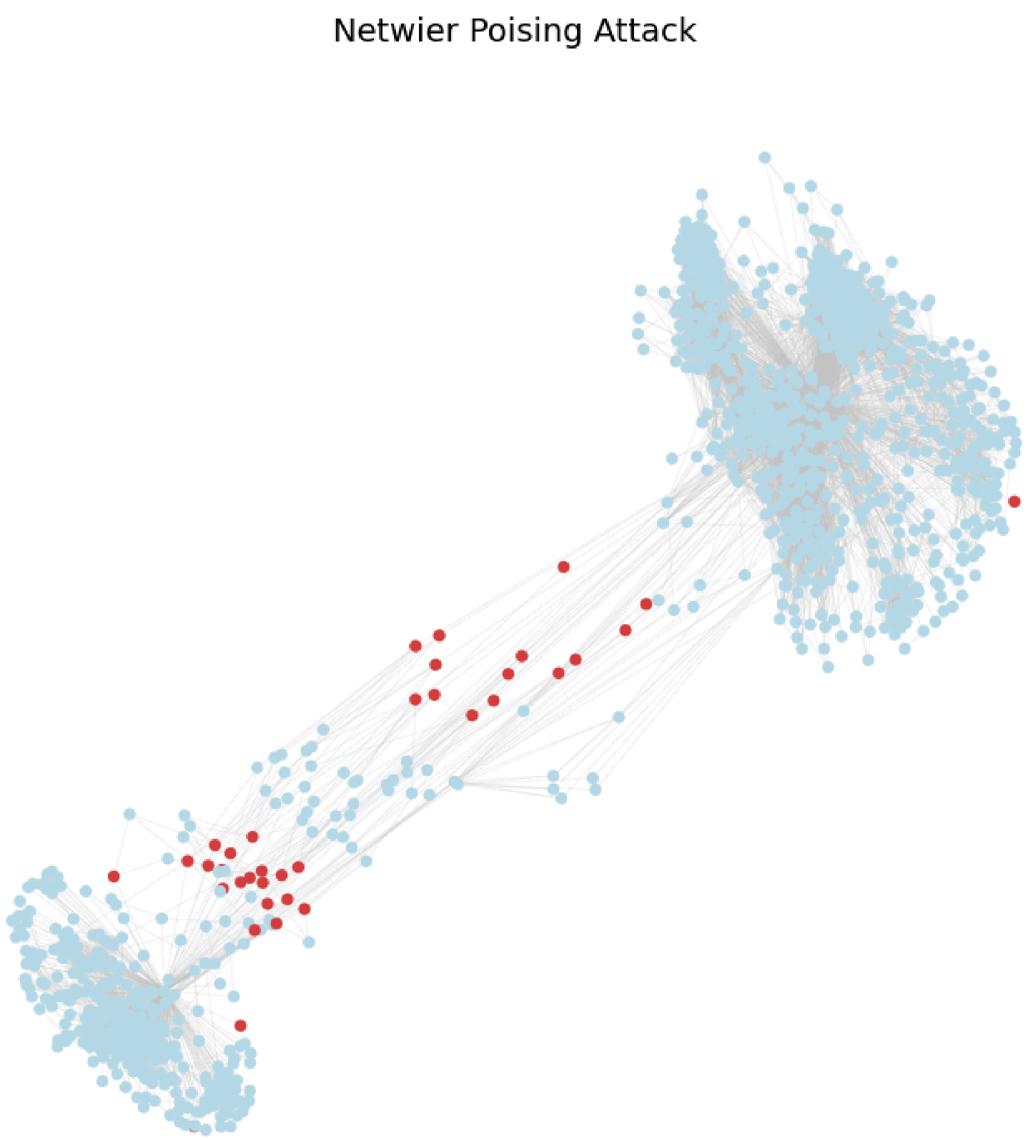

Figure 4: Adversarial Robustness Failure

The chart clearly quantifies the model's vulnerability, showing that **Structural Evasion** is the most effective attack, causing a catastrophic drop in Bot Recall (Class 1) and proving the detector is highly susceptible to adversarial graph manipulation. This failure mode occurs because the model relies heavily on local structural features, which attackers successfully manipulate by altering node connectivity. Furthermore, the degradation caused by **Graph Poisoning** highlights the risk of training on corrupted data, compromising the integrity of the learned features.