Adversarial Training for Network Intrusion Detection

Filippo di Gravina[1]

Abstract

This paper addresses the vulnerability of Machine-Learning based Network Intrusion Detection Systems (NIDS) to adversarial attacks by proposing a robust defense based on a regularized Multi Layer Perceptron. Our method incorporates early stopping, dropout, batch normalization, and adversarial training to enhance the robustness against gradient-based attacks like the Fast Sign Gradient Method and the Projected Gradient Descent. It achieves more than 98% accuracy under clean conditions and restores up to 97% accuracy after the adversarial fine tuning on the CIC-IDS-2017 dataset. Our approach bridges the gap between high classification performance and adversarial robustness, enabling more secure deployment of NIDS in real world environments.

Keywords

Adversarial training, CIC-IDS-2017, Network intrusion detection

1. Introduction and Motivations

With the exponential increase in cyber threats, Network Intrusion Detection Systems have become increasingly important for network security. Recent methods have addressed some challenges like data imbalance [1], robustness against adversarial attacks remain an underexplored issue.

During model selection, we applied a paired t-test to identify the best-performing architecture before the attacks, ultimately choosing a Multi Layer Perceptron. We then focused on improving the model resilience to perturbations by adopting dropout, batch normalization, and adversarial fine-tuning techniques. Lastly, we evaluated the defense effectiveness by comparing the pre-defence results to the post-defence results, demonstrating significant improvements in robustness.

2. Related Work

As a baseline for evaluating performance on clean data, we refer to the recent work by Talukder et al. [1], which performs binary classification on multiple intrusion detection datasets, including CIC-IDS-2017. Their machine learning-based approach combines oversampling techniques with stacking feature embedding and feature extraction to handle big and imbalanced data. We used their reported performance as a benchmark to contextualize our results in the non-adversarial setting.

3. Proposed Approach

3.1. Description of the solution and dataset

The CIC-IDS-2017 dataset [2] is a network intrusion detection dataset that contains benign and malicious labeled traffic.

Due to the high number of instances in the CIC-IDS2017 dataset, executing training and evaluation became computationally demanding. After merging the originally split CSV files into a single dataset, we applied some preprocessing steps to reduce dimensionality and improve the data quality. Specifically, we removed rows with missing values, dropped duplicate entries, eliminated zero-variance features, and computed the correlation matrix to identify highly correlated attributes:

Machine Learning Course 2024-2025

f.digravina2@studenti.uniba.it (F. d. Gravina)

https://github.com/fdigravina/ (F. d. Gravina)

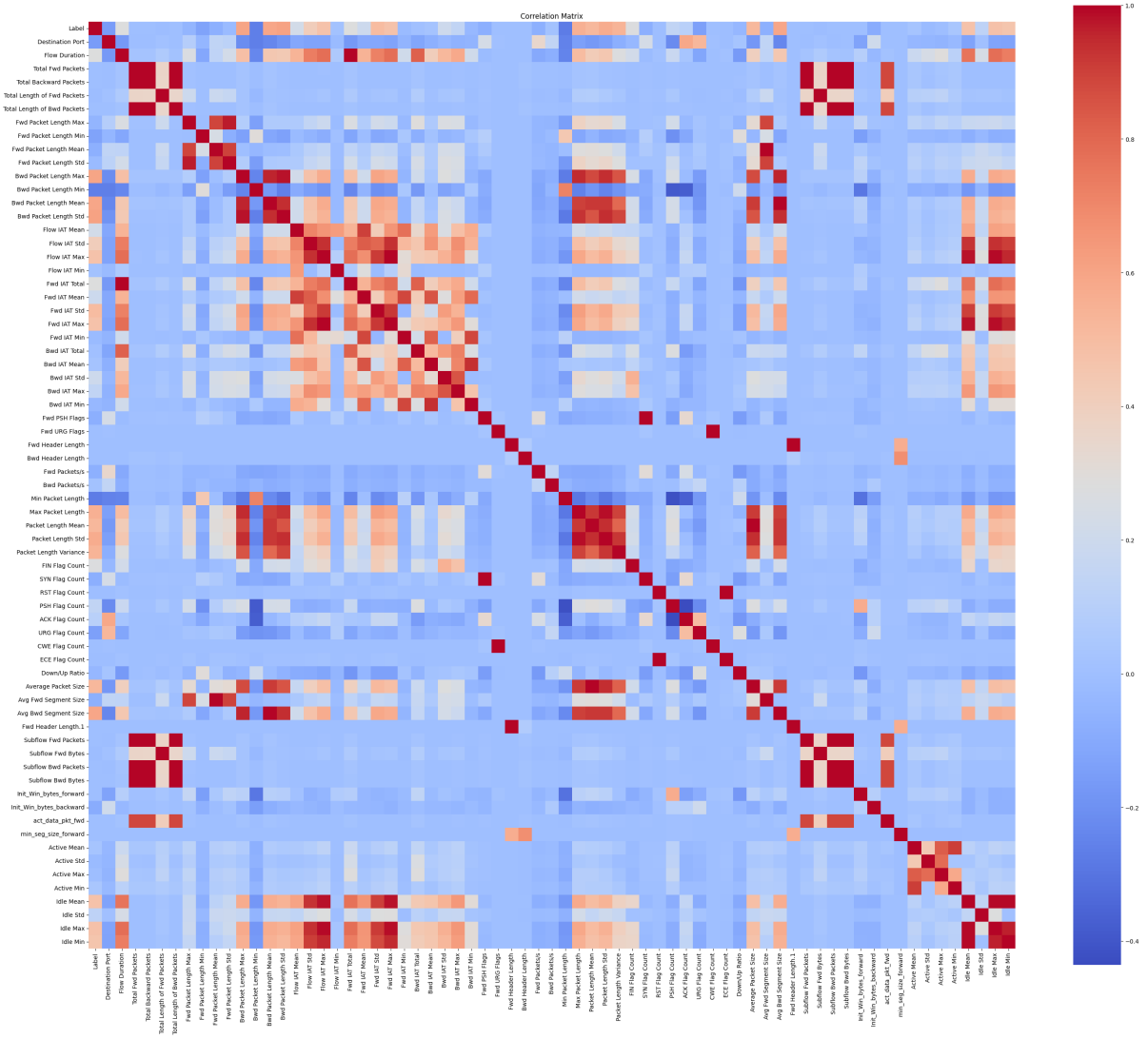

Figure 1: Correlation matrix

Adversarially perturbed examples are inputs modified by small perturbations designed to mislead a machine learning model, while remaining nearly indistinguishable from the original data.

Adversarial training is a regularization technique in which a model is trained not only on clean inputs but also on adversarially perturbed examples, with the goal of improving robustness to these attacks.

We used two different methods to generate adversarial examples: the Fast Gradient Sign Method [3] and the Projected Gradient Descent method [4].

FGSM and PGD were selected as adversarial attack methods due to their gradient-based nature, which enables the generation of adversarial examples by computing the gradient of the loss with respect to the input. Since our MLP model exposes its gradients during inference, these methods are well-suited for evaluating its robustness.

FGSM generates adversarial examples by applying a single-step perturbation in the direction of the gradient of the loss function:

$$x_{\text{adv}} = x + \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))$$

PGD is a stronger attack that performs iterative FGSM steps:

$$x^{t+1} = \Pi_{x+S}\left(x^t + \alpha \cdot \text{sign}(\nabla_x L(\theta, x, y))\right)$$

3.2. Main technical details

The model selection process involved comparing the Logistic Regression, SVM, and MLP classifiers. For Logistic Regression and SVM, hyperparameter tuning was performed using a grid search with a 5-fold cross-validation. The MLP was trained and validated separately, and to statistically compare its performance against Logistic Regression, who performed better than the SVM, we conducted a paired t-test between the cross-validation accuracies of Logistic Regression and the validation accuracies of the MLP. The results showed that the MLP significantly outperformed Logistic Regression:

$$\text{Paired t-test (MLP vs LR): } t = 6.763, p = 0.001$$

The experimental pipeline was then carried out on the MLP model and consisted of three main phases: training on clean data, testing under adversarial attack, and adversarial training. In the clean phase, the MLP was trained on the unperturbed dataset and evaluated on clean test samples. Next, we tested the model's robustness by exposing it to adversarially perturbed examples generated using FGSM and PGD. In the third phase, we applied fine-tuning on adversarial examples generated on-the-fly using the same attack used for evaluation.

We experimented with multi-attack adversarial training, but this method degraded performance compared to single-attack adversarial training. Additionally, we tested the feature squeezing defense as a pre-processing step, but it failed to improve robustness. For this reason, these methods have been discarded.

4. Evaluation

To evaluate performance on clean data, we used the results of [1] as a baseline for the non-adversarial setting. Their method achieved over 99% across all the following metrics: accuracy, precision, recall, and F1-score.

In comparison, our model achieves an accuracy of 98.9%, precision of 98.6%, recall of 94.7%, and F1-score of 96.6% on the clean dataset. While slightly lower, these results are competitive and were obtained using a simpler architecture.

These results support the choice of the use of a Multi Layer Perceptron for the adversarial training, also given that its exposure of gradients makes it particularly well-suited for evaluating model robustness against gradient-based adversarial attacks such as FGSM and PGD.

We evaluated the model across three phases: before any attack, under adversarial perturbations, and after applying adversarial training as a defense. On clean data, the model demonstrated strong performance across all metrics. However, when subjected to adversarial attacks, the model's performance dropped significantly, especially in terms of recall and F1-score, highlighting its vulnerability. Adversarial training proved to be an effective countermeasure: it restored the original performance levels and also ensured consistent robustness against both types of attacks.

These results reinforce the suitability of the MLP model, as its gradient exposure makes it easily trainable against such adversarial threats. The results are visualized using bar charts:

Table 1
Adversarial performance

Scenario	Accuracy	Precision	Recall	F1-score
Pre-Attack	0.988	0.971	0.955	0.963
Post FGSM	0.832	0.059	0.000	0.000
Post-Defence (FGSM)	0.972	0.972	0.858	0.911
Post PGD	0.832	0.520	0.006	0.012
Post-Defence (PGD)	0.972	0.958	0.869	0.912

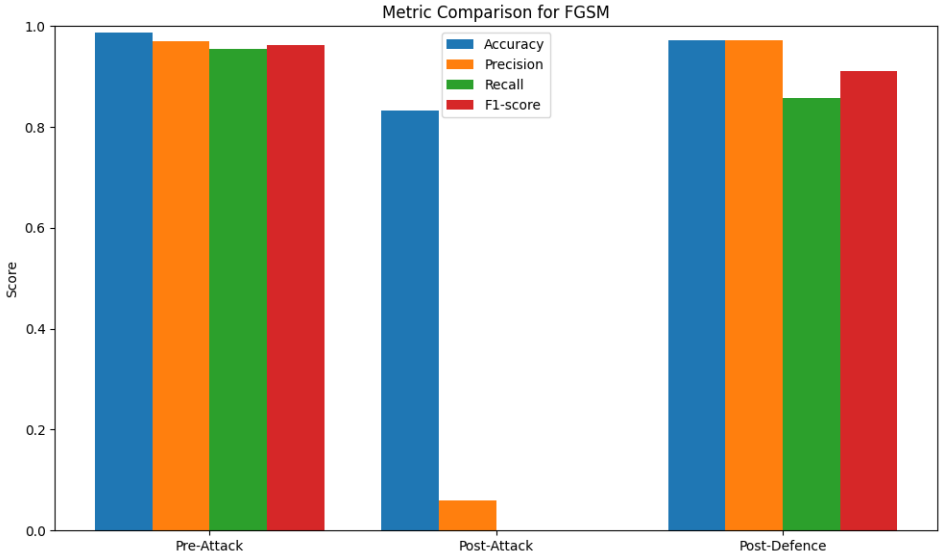

Figure 2: FGSM metric comparison

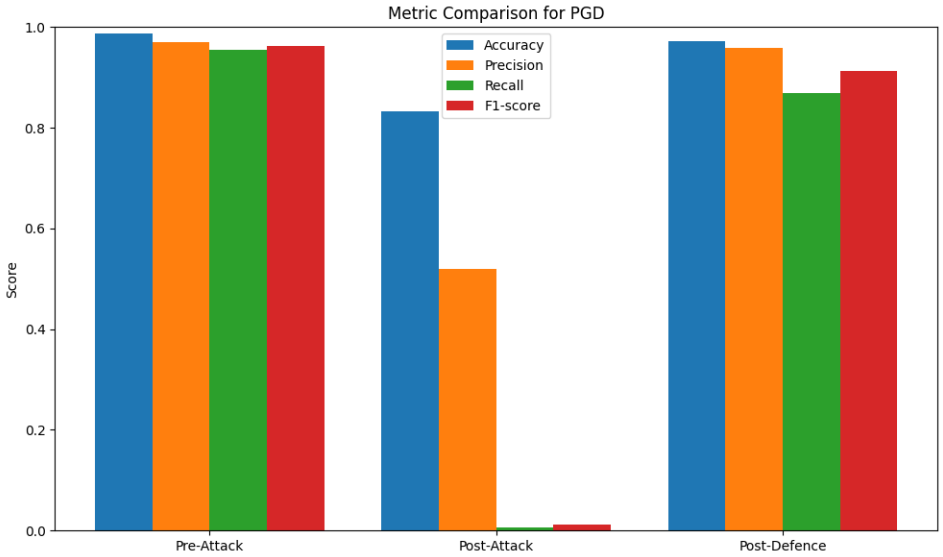

Figure 3: PGD metric comparison

5. Conclusions and Limitations

Our study demonstrates that while our MLP model performs very well on clean data, it remains vulnerable to adversarial attacks such as FGSM and PGD. However, adversarial training effectively restores robustness, confirming the value of this defense strategy. The simplicity and the gradient exposure of the MLP architecture make it a suitable baseline for further robustness evaluations. Future work could explore the application of additional and more complex adversarial attacks.

5.1. Code and data availability

The code is available on the repository: https://github.com/fdigravina/ML-2024-25. The CIC-IDS-2017 dataset is publicly available.

References

[1] M. A. Talukder, M. M. Islam, M. A. Uddin, K. F. Hasan, S. Sharmin, S. A. Alyami, M. A. Moni, Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction, 2024. URL: https://arxiv.org/abs/2401.12262. arXiv:2401.12262.

[2] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, Toward generating a new intrusion detection dataset and intrusion traffic characterization, in: International Conference on Information Systems Security and Privacy, 2018. URL: https://api.semanticscholar.org/CorpusID:4707749.

[3] I. J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, 2015. URL: https://arxiv.org/abs/1412.6572. arXiv:1412.6572.

[4] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu, Towards deep learning models resistant to adversarial attacks, 2019. URL: https://arxiv.org/abs/1706.06083. arXiv:1706.06083.