PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
الجمهـــوريـــــة الجزائـريـــــة الديـمقـراطيـــــة الشـعـبيـــــة
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
وزارة التعليـم العالـــي والبحـــث العلمـي
HIGHER SCHOOL OF COMPUTER SCIENCE AND DIGITAL TECHNOLOGIES-BEJAIA
المـــدرسة العـــليا في علـــوم وتكنـولوجيـات الاعـــلام الالـــي بجـــاية

Dissertation Submitted to the Department Of Computer Science in Partial Fulfillment of the Requirements for Engineer's Degree in Computer Science

Specialty: Artificial Intelligence and Data Sciences

Submitted By:

Ahmed Yacine Bouchouareb

Theme

Evaluation of Machine Learning Models for Detecting Adversarial Attacks on Anomaly Detection Oriented Dataset

Supervised by :

Mrs. Radia Kassa ESTIN
Pr. Pierre-Martin Tardif *University of Sherbrooke*
Dr. Jordan Félicien Masakuna *University of Sherbrooke*
Mr. DJeff Kanda Nkashama *University of Sherbrooke*

Members of jury:

▪ **Dr.** HARFOUCHE Lynda	President	ESTIN
▪ **Dr.** DAOUDI Meroua	Examiner	ESTIN
▪ **Mrs.** DJENNANE Lynda	Examiner	ESTIN
▪ **Mrs.** KHELOUF Hanane	Examiner	ESTIN
▪ **Mrs.** KASSA Radia	Supervisor	ESTIN

Academic year: 2023/2024

Abstract

This report evaluates the capability of machine learning models in detecting adversarial attacks on a given dataset, with a test on the NSL-KDD dataset. The study's objectives are twofold: first, to analyze the dynamics of the autoencoder's reconstruction loss for normal, anomalous, and adversarial data points; second, to benchmark various candidate models, including Support Vector Machines (SVM), Decision Trees, and Naive Bayes, in detecting adversarial data crafted using Fast Gradient Sign Method (FGSM)[5] and Projected Gradient Descent (PGD)[10] techniques. Additionally, this research tests a feature engineering technique that considers the reconstruction loss as a vector[21], as suggested in recent literature. The results demonstrate that the reconstruction loss exhibits similar behavior between anomalous and adversarial examples, differentiating them from normal records in terms of mean and variance. Furthermore, the study reveals that the benchmarked models face significant challenges in detecting PGD attacks compared to FGSM attacks.

Keywords— Machine Learning, Adversarial Examples, Robustness, Autoencoders, FGSM, PGD, Anomaly Detection, Adversarial Attacks

Résumé

Ce mémoire analyse la performance des modèles d'apprentissage automatique pour identifier les attaques adversariales, en particulier sur le jeu de données NSL-KDD. L'étude vise, d'une part, à examiner la perte de reconstruction générée par l'autoencodeur lorsqu'il est confronté à des données normales, anormales et adversariales. D'autre part, elle compare plusieurs modèles, tels que la SVM, les arbres de décision et Naive Bayes, dans leur capacité à détecter les attaques adversariales générées par les techniques FGSM[5] et PGD[10]. En outre, une technique d'ingénierie des caractéristiques, qui traite la perte de reconstruction comme un vecteur, est testée dans cette recherche, s'appuyant sur des travaux récents. Les résultats montrent que la perte de reconstruction se comporte de manière similaire pour les exemples adversariaux et anormaux, mais diffère des données normales en termes de moyenne et de variance. L'étude souligne également que les modèles évalués rencontrent plus de difficultés face aux attaques PGD par rapport aux attaques FGSM.

Mots-clés— Apprentissage automatique, Exemples adversariaux, Robustesse, Autoencodeurs, FGSM, PGD, Détection d'anomalies, Attaques adversariales

الملخص

هذا التقرير يقيّم قدرة نماذج التعلم الآلي على اكتشاف الهجمات العدائية على مجموعة بيانات معينة، مع إجراء اختبار على مجموعة بيانات *NSL-KDD*. تهدف الدراسة إلى تحقيق هدفين رئيسيين: أولاً، تحليل ديناميكيات فقدان إعادة البناء في الترميز التلقائي للنقاط العادية والشاذة والعدائية؛ ثانياً، تقييم أداء نماذج متنوعة، بما في ذلك آلات المتجهات الداعمة *(SVM)* وأشجار القرار و*Naive Bayes*، في اكتشاف البيانات العدائية التي تم إنشاؤها باستخدام تقنيات طريقة العلامة ذات التدرج السريع *FGSM* والتدرج المتناقص المسقط *PGD*. بالإضافة إلى ذلك، تختبر هذه الدراسة تقنية هندسة الميزات التي تعتبر فقدان إعادة البناء كشعاع، كما هو مقترح في الأدبيات الحديثة. تُظهر النتائج أن فقدان إعادة البناء يظهر سلوكاً مشابهاً بين الأمثلة الشاذة والعدائية، مما يميزها عن السجلات العادية من حيث المتوسط والتباين. علاوة على ذلك، تكشف الدراسة أن النماذج التي تم تقييمها تواجه تحديات كبيرة في اكتشاف هجمات *PGD* مقارنة بهجمات *FGSM*.

الكلمات المفتاحية: التعلم الآلي، الأمثلة العدائية، القوة، الترميز التلقائي، *PGD*، *FGSM*، اكتشاف الشذوذ، الهجمات العدائية.

Contents

List of Figures

List of Acronyms

AD Anomaly Detection

Adam Adaptive Moment Estimation

AL Adversarial Learning

DNNs Deep Neural Networks

ECG Electrocardiogram

FGSM Fast Gradient Sign Method

FL Federated Learning

GANs Generative Adversarial Networks

ML Machine Learning

MLP Multilayer Perceptron

MSE Mean Squared Error

PGD Projected Gradient Descent

SGD Stochastic Gradient Descent

SimCLR Simple Framework for Contrastive Learning of Visual Representations

SVM Support Vector Machines

General Introduction

0.1 Background

Machine Learning (ML) has become integral to a wide range of applications, from image recognition and natural language processing to autonomous driving and healthcare diagnostics [8, 19]. As these models are increasingly deployed in critical and high-stakes environments, ensuring their robustness—especially against adversarial attacks—has become a significant concern [5, 20].

Adversarial attacks involve deliberately crafting inputs that are designed to deceive machine learning models, often leading to incorrect or even dangerous predictions [2]. These attacks highlight vulnerabilities in machine learning systems, especially in models that perform well under standard conditions but fail under adversarial manipulation. Given the potential consequences of such failures, enhancing the robustness of machine learning models is not just an academic challenge but a practical necessity [10].

0.2 Objective

The primary objective of this study is to benchmark the performance of machine learning models in detecting adversarial attacks on the NSL-KDD dataset. This involves two main components:

1. Analyzing the Dynamics of Reconstruction Loss: Examining how the autoencoder's reconstruction loss varies across normal, anomalous, and adversarial data points.

2. Benchmarking Candidate Models: Evaluating a range of models, including SVM, Decision Trees, and Naive Bayes, on their ability to detect adversarial examples crafted using FGSM[5] and PGD[10].

0.3 Structure Overview of the Report

- **General Introduction** – This chapter outlines the background of the study, clearly stating the objectives and scope. It sets the stage for the subsequent chapters by providing the necessary context and motivation behind exploring machine learning security and adversarial learning.
- **Chapter 1: Literature Review** – This chapter provides a comprehensive overview of the current state of machine learning security and adversarial learning. It examines various attacking techniques, such as the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD), and discusses defensive strategies like adversarial training and anomaly detection using autoencoders. The chapter also identifies significant gaps in the current knowledge, highlighting areas that require further research.
- **Chapter 2: Environment Setup** – Discusses the technical environment used for conducting experiments, covering essential tools and libraries such as Python, GitHub, PyTorch, pandas, and NumPy. This chapter also provides practical insights into the configuration and management of these tools to facilitate reproducibility and scalability of the experiments.
- **Chapter 3: Methodology** – This chapter details the research design and methodology employed in the study. It includes the data preprocessing steps, model training processes, and the techniques used for generating adversarial examples. The chapter also covers the training of autoencoders with adversarial examples, the visualization of error, and the methods used for dataset augmentation and evaluation.
- **Chapter 4: Results** – This chapter presents the findings of the study, focusing on the dynamics of the autoencoder's reconstruction loss across different types of data—normal, anomalous, and adversarial. It also evaluates the effectiveness of the feature engineering technique that uses reconstruction loss as a vector, comparing its performance against various benchmark models.
- **Chapter 5: Conclusion** – This final chapter summarizes the key takeaways from the research, emphasizing the contributions made to understanding and improving the robustness of machine learning models. It also discusses the practical implications of the findings and their relevance to ongoing and future research in adversarial learning.

0.4 Scope

This study tested on the NSL-KDD dataset, commonly used in network intrusion detection research. The analysis includes normal, anomalous, and adversarial data points generated using FGSM and PGD. The effectiveness of a feature engineering technique, which considers the reconstruction loss as a vector, is tested in improving model performance. Limitations include the reliance on a single dataset and

the specific types of adversarial attacks examined, which may not generalize to other datasets or attack methods.

CHAPTER 1

Literature Review

1.1 Introduction

The security of ML models has become a critical concern as these models are increasingly deployed across diverse sectors, including healthcare, finance, autonomous vehicles, and social media. Adversarial attacks, which involve subtle manipulations of input data to deceive ML models, pose a significant threat to the integrity and reliability of these systems. This chapter provides a comprehensive review of the current state of ML security, focusing on the challenges posed by adversarial attacks and the corresponding defense mechanisms. Drawing on insights from two recent surveys, as well as a selection of key papers, the review covers various attack strategies, including the FGSM and PGD, and evaluates the effectiveness of existing defensive techniques such as adversarial training, autoencoder-based Anomaly Detection (AD), and the novel concept of vector reconstruction error. The chapter concludes by identifying gaps in the current research and proposing areas for future investigation, particularly in the context of enhancing the robustness of ML models against sophisticated adversarial threats.

1.2 Comprehensive Overview of ML Security and Adversarial Learning (AL)

The security landscape in ML is rapidly evolving as the deployment of ML models expands across various domains, including medical, military, automotive, and social networking. Despite their widespread success, these models are increasingly vulnerable to a range of adversarial attacks that threaten their integrity and reliability. Two comprehensive surveys provide a detailed examination of these security concerns, offering insights into both the nature of the threats and the defenses developed to counter them.

1.2.1 Overview of Security Threats in Machine Learning

Machine learning (ML) systems are increasingly integrated into various applications, making them attractive targets for adversaries. The survey titled "Machine Learning Security Threats, Countermeasures, and Evaluations" by [23] discusses several key threats to ML systems:

- **Adversarial Attacks**: These involve manipulating input data to deceive ML models into making incorrect predictions. Techniques include adding noise to images or altering text inputs.
- **Data Poisoning**: Attackers can introduce malicious data into the training set, leading to compromised model performance. This is particularly concerning in environments where data is continuously updated.
- **Model Inversion**: This threat allows attackers to reconstruct sensitive data used in training by querying the model, potentially exposing private information.

- **Evasion Attacks**: These attacks occur during the inference phase, where adversaries modify inputs to evade detection or mislead the model without altering the training data.
- **Model Theft**: Attackers can steal the intellectual property of a machine learning model by extracting its functionality and recreating it. This can be achieved by querying the model and analyzing its outputs to approximate the original model's behavior.

1.2.2 Countermeasures Against ML Threats

The paper outlines several countermeasures to mitigate the identified threats:

- **Adversarial Training**: This involves training models on both clean and adversarial examples to enhance robustness. This method can help models learn to recognize and correctly classify manipulated inputs.
- **Data Validation Techniques**: Implementing rigorous data validation can help detect and filter out poisoned data before it is used for training.
- **Differential Privacy**: This approach adds noise to the training data or model outputs to protect individual data points, making it harder for attackers to infer sensitive information.[1]
- **Robustness Evaluation**: Regularly evaluating model performance against known attack vectors can help identify vulnerabilities and improve defenses.

1.2.3 Evaluation Frameworks

The paper emphasizes the need for comprehensive evaluation frameworks to assess the effectiveness of various countermeasures. These frameworks should consider:

- **Attack Scenarios**: Testing models against a range of adversarial techniques to understand their weaknesses.
- **Performance Metrics**: Evaluating models not just on accuracy but also on robustness against attacks.
- **Real-World Simulations**: Conducting tests in environments that closely mimic real-world applications to better understand how models will perform under attack.

Figure 1.1 illustrates these categories of attacks, providing a visual representation of the diverse threats faced by ML models.

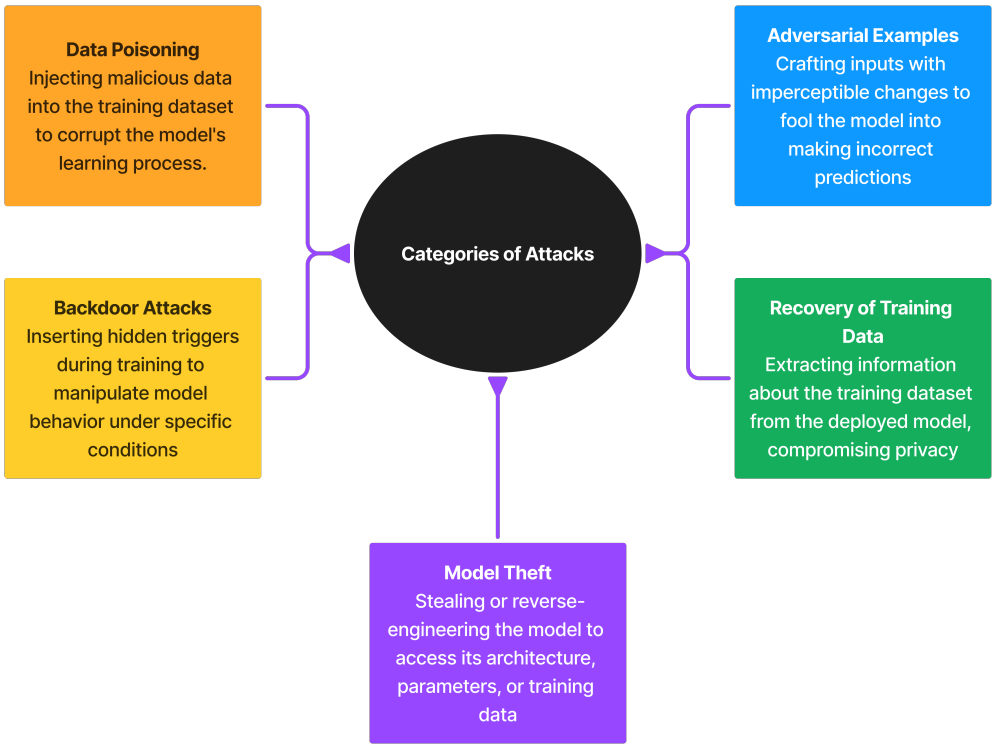


Figure 1.1: Categories of attacks in ML.

1.2.4 Summary of Adversarial Learning and Defenses

The survey "Adversarial Learning Targeting Deep Neural Network Classification: A Comprehensive Review of Defenses Against Attacks" by [12] provides an extensive overview of adversarial learning, focusing on the vulnerabilities of Deep Neural Networks (DNNs) to adversarial attacks. Key points include:

- **Nature of Adversarial Examples**: These are inputs specifically crafted to mislead DNNs, often imperceptibly altered from legitimate inputs. The paper discusses various methods used to generate these examples, such as the FGSM and PGD.
- **Impact on DNNs**: Adversarial examples can significantly degrade the performance of DNNs, leading to misclassifications that can have serious implications in critical applications like autonomous driving and healthcare.

1.2.5 Defense Mechanisms

The review categorizes defense mechanisms into several strategies:

- **Input Transformation**: Techniques such as image preprocessing (e.g., smoothing, cropping) can help reduce the impact of adversarial perturbations.

- **Model Regularization**: Methods like adversarial training, where models are trained with adversarial examples, can enhance robustness. This approach has shown promise but can be resource-intensive.
- **Ensemble Methods**: Using multiple models and aggregating their predictions can improve resilience against attacks, as adversarial examples may not fool all models in the ensemble.[15]
- **Detection Techniques**: Developing systems to detect adversarial inputs before they reach the model can serve as an additional layer of security.

1.2.6 Evaluation of Defenses

The paper stresses the importance of evaluating the effectiveness of defense mechanisms through:

- **Benchmarking Against Attacks**: Systems should be tested against a variety of adversarial attacks to gauge their robustness.
- **Generalization**: Defenses must not only work against specific attacks but should also generalize well to unknown attacks.
- **Trade-offs**: Evaluating the trade-offs between model accuracy and robustness is crucial, as some defenses may degrade performance on legitimate inputs.[22]

Figure 1.2 provides an illustration of the phases of attacks in ML, showing how different attack strategies evolve and impact ML systems.

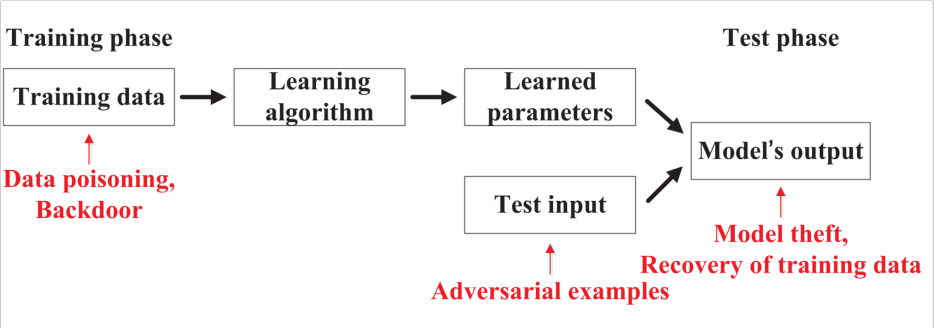


Figure 1.2: Phases of attacks in ML [23].

Together, these surveys provide a comprehensive overview of the current state of ML security, highlighting both the breadth of potential threats and the depth of defensive strategies. They underscore the importance of rigorous security evaluations and the need for ongoing research to address unresolved challenges in the field. By integrating insights from these works, this report builds a foundation for understanding the complexities of defending ML models in adversarial environments.

1.3 Attacking Techniques

1.3.1 Fast Gradient Sign Method (FGSM)

FGSM is one of the earliest and most widely used methods for generating adversarial examples. It works by adding perturbations to the input data in the direction of the gradient of the loss function with respect to the input, scaled by a small factor. This method is efficient and easy to implement but is relatively straightforward to defend against using techniques like adversarial training [5].

$$\text{Adversarial Example: } x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y)) \tag{1.1}$$

1.3.2 Projected Gradient Descent (PGD)

PGD is an iterative extension of FGSM and is considered a more powerful attack. In each iteration, the adversarial perturbation is updated by taking a small step in the direction of the gradient, followed by a projection step to ensure the perturbation remains within a specified norm ball. PGD is recognized as one of the most effective methods for crafting adversarial examples and is often used as a benchmark in evaluating model robustness [10]. The Projected Gradient Descent attack can be formulated as:

$$x_{t+1} = \text{Proj}_{\mathcal{B}_\epsilon(x)} \left(x_t + \alpha \cdot \text{sign} \left(\nabla_x \mathcal{L}(f_\theta(x_t), y) \right) \right)$$

Where:

- x_t is the adversarial example at step t.
- α is the step size.
- $\nabla_x \mathcal{L}(f_\theta(x_t), y)$ is the gradient of the loss function $\mathcal{L}$ with respect to the input x_t.
- $\text{sign}(\cdot)$ is the sign function.
- $\text{Proj}_{\mathcal{B}_\epsilon(x)}(\cdot)$ projects the perturbed example back onto the ϵ-ball around the original input x.

1.3.3 A Robust Analysis of Adversarial Attacks on Federated Learning (FL) Environments

FL has emerged as a prominent branch of Artificial Intelligence, particularly with the proliferation of mobile computing and IoT technologies. Unlike traditional centralized learning paradigms, FL relies on a distributed computing approach, where multiple decentralized devices contribute to the learning process. While this methodology enhances privacy by keeping data localized, it also introduces significant security challenges, as many participating devices operate outside the protective scope of a centralized system.

This paper provides a robust analysis of the security vulnerabilities inherent in federated learning environments, focusing on various adversarial attacks

that can compromise the integrity and effectiveness of FL models. Key threats include data leakage, communication issues, poisoning attacks, and system manipulation through backdoor mechanisms. These attacks are categorized based on their modus operandi, offering a detailed examination of poisoning and inferencing attacks, among others.

The study not only reviews the different types of attacks but also evaluates the effectiveness of existing defense strategies designed to mitigate these risks in federated environments. By systematically analyzing the challenges posed by adversarial examples in FL, this paper contributes to a deeper understanding of the security issues faced in Federated ML and proposes potential solutions to enhance the robustness of these systems [14].

1.3.4 A Comprehensive Overview of Generative Generative Adversarial Networks (GANs) and Their Applications

GANs have become a pivotal innovation in artificial intelligence, particularly in generating high-quality synthetic data. By leveraging two neural networks in a zero-sum game framework—where one network generates data and the other attempts to discern real data from synthetic—GANs excel in producing sharp and discrete outputs.

This survey offers an extensive review of GAN architectures, their prevalent variants, and a broad range of applications across numerous sectors. GANs have been widely used in computer vision tasks such as image processing, video generation, and prediction. Additionally, they have found applications in scientific fields like protein engineering, astronomical data analysis, remote sensing image dehazing, and crystal structure synthesis. Beyond these, GANs have also made significant inroads into finance, marketing, fashion design, sports, and music.

My interest in this survey was driven by the role GANs play in the creation of adversarial examples. GAN-based techniques have been increasingly used to generate adversarial examples that can deceive ML models, a crucial aspect of studying model robustness. Understanding the underlying mechanics and applications of GANs provided essential insights into how these networks can be harnessed for adversarial purposes, as well as how they might be defended against. The survey thoroughly covers the theoretical foundations of GANs, the various variants developed, and the metrics used for evaluation, making it a valuable resource for comprehending the potential and risks associated with GAN-generated adversarial examples [4].

The figure 1.3 illustrates the basic architecture of a GAN.

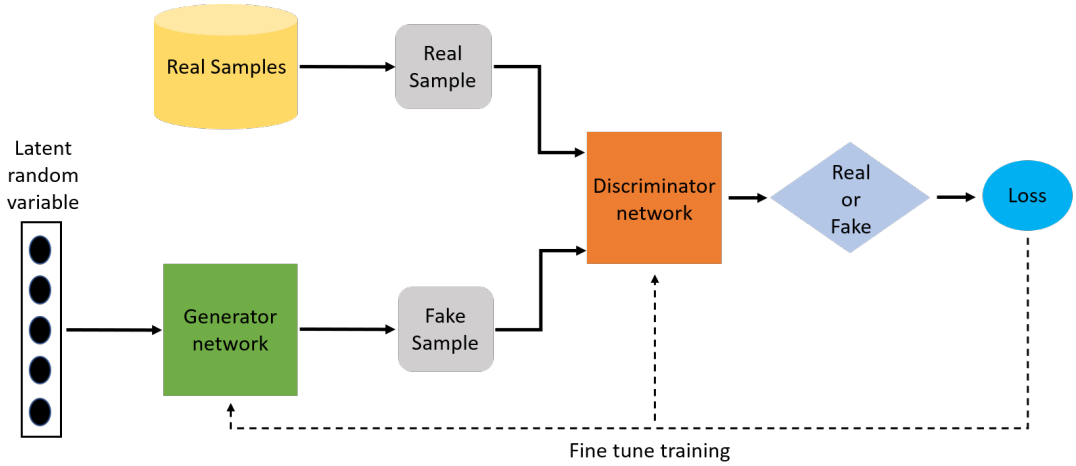


Figure 1.3: Basic GAN architecture [4]

1.4 Defensive Techniques

1.4.1 Adversarial Training

AL is one of the most widely used techniques to enhance model robustness. It involves augmenting the training data with adversarial examples, allowing the model to learn how to correctly classify perturbed inputs. This method has been shown to improve resistance against specific attacks, such as FGSM and PGD, but may require substantial computational resources and can still be vulnerable to more sophisticated attacks.

1.4.2 Training Distillation

Training distillation, or adversarial distillation, is another defense mechanism that enhances robustness. In this technique, a "teacher" model is trained with adversarial examples, and the knowledge from this model is then distilled into a "student" model, which is smaller and potentially more robust. This approach has been applied successfully in various domains, including Electrocardiogram (ECG) classification, as discussed in the paper *Cardio Defense: Defending Against Adversarial Attack in ECG Classification with Adversarial Distillation Training*. Zhang et al. [18] proposes an innovative approach for defending ECG classification models. The technique involves training a model using a mixture of normal and adversarial examples. This process distills the knowledge of adversarial robustness into the model, improving its ability to correctly classify both normal and adversarially perturbed ECG signals. The study demonstrates that this technique enhances the model's resistance to adversarial attacks without significantly compromising its accuracy on clean data.

1.4.3 Anomaly Detection (AD)

AD, also known as outlier detection, is the process of identifying rare items, events, or observations which raise suspicions by differing significantly from the majority of the data. It has critical applications in various fields such as fraud detection,

network security, fault detection in industrial systems, and health monitoring [3]. Traditional anomaly detection techniques include statistical methods, clustering-based approaches, and distance-based methods, each having its own strengths and limitations.

1.4.4 Anomaly Detection Using Autoencoders

Autoencoders (AEs) have emerged as powerful tools for anomaly detection due to their ability to learn compact, high-dimensional representations of data [7]. In a typical autoencoder-based anomaly detection scenario, the model is trained on normal data, learning to reconstruct it accurately. During inference, the model attempts to reconstruct both normal and anomalous data. Since the model has only learned the patterns of normal data, it struggles to reconstruct anomalies, resulting in higher reconstruction errors for such data points.

To distinguish between normal and anomalous data, a threshold value is set on the reconstruction error. Data points with a reconstruction error exceeding this threshold are classified as anomalies. Determining the threshold can be done through various methods, such as selecting a fixed percentile of the reconstruction errors from the training data, using validation data to optimize the threshold, or employing statistical techniques [9].

1.4.5 Reconstruction Loss as a Vector for Enhanced Anomaly Detection

A novel approach to anomaly detection using autoencoders involves considering the reconstruction loss as a vector rather than a scalar. Torabi et al. [21] introduced a method that enhances traditional autoencoder-based anomaly detection by analyzing the vector reconstruction error, which captures more nuanced differences between normal and anomalous data points. Instead of collapsing the reconstruction loss into a single scalar value, this approach leverages the multi-dimensional nature of the error vector to provide a more granular view of the reconstruction process. This technique has shown promising results in improving detection accuracy for certain types of anomalies, especially in complex datasets, where traditional scalar-based methods might struggle.

1.4.6 Classification of Adversarial Attacks

Recent advances in adversarial defense have introduced various techniques to mitigate the impact of adversarial attacks. Mazda Moayeri and Soheil Feizi [13] propose a method that leverages self-supervised learning for adversarial detection. Their approach, known as SimCat, uses embeddings from a Simple Framework for Contrastive Learning of Visual Representations (SimCLR) encoder to classify and detect various adversarial attacks. This approach aligns with the trend towards embedding-based methods in AD and offers valuable insights into efficient adversarial detection.

1.5 Gaps in Knowledge

While significant progress has been made in the fields of adversarial attack detection and AD, several gaps remain that this study aims to address:

1. **Limited Focus on Adversarial Attacks in AD**: Traditional AD techniques often concentrate on identifying deviations from normal behavior without explicitly considering the impact of adversarial attacks. Existing methods typically classify data into normal or anomalous categories but do not differentiate between anomalies caused by natural deviations and those induced by adversarial manipulations. This limitation can reduce the effectiveness of AD systems in environments where adversarial threats are prevalent.

2. **Integration of Attack Classification**: Few studies have integrated the classification of data based on the type of attack [13] alongside traditional AD. There is a need for methodologies that not only identify anomalous data but also categorize it by the type of adversarial attack, providing a more detailed understanding of the threats. By classifying data into normal, anomalous, or attacked categories, and further identifying the specific attack techniques used (e.g., FGSM, PGD), models can offer more robust protection against a variety of adversarial strategies.

1.6 Theoretical Framework

Autoencoders in AD: Autoencoders are a type of neural network used primarily for unsupervised learning of data representations. They are structured with an encoder, which compresses the input into a lower-dimensional latent representation, and a decoder, which reconstructs the input from this representation. In AD, autoencoders are trained on normal data, allowing them to effectively reconstruct similar inputs. Anomalies are identified by their higher reconstruction error, as they deviate from the learned patterns of normal data.

Adversarial Attack Detection: Adversarial attacks exploit vulnerabilities in ML models by introducing small, carefully crafted perturbations to input data, resulting in incorrect outputs. Techniques like the FGSM[5] and PGD[10] are commonly used to generate adversarial examples. These methods leverage the model's gradients to create inputs that appear normal to human observers but cause significant errors in the model's predictions. Detecting such attacks involves identifying the patterns and features that these adversarial perturbations introduce.

Vector Reconstruction Error: Traditional approaches to AD using autoencoders focus on scalar reconstruction loss, which measures the overall difference between input and output. This approach, however, may not capture the full extent of deviations, particularly in the context of adversarial attacks. By utilizing vector reconstruction error[21], each element of the reconstruction loss is considered, providing a more detailed analysis of how the input and output differ. This technique offers enhanced sensitivity in distinguishing between normal, anomalous, and adversarially perturbed data, making it a valuable tool for improving AD and adversarial defense.

Integration in This Study: In this research, the theoretical concepts of autoencoders, adversarial attack detection, and vector reconstruction error are integrated to benchmark ML models' ability to detect adversarial attacks. Autoencoders are used to generate reconstruction errors from the dataset, while adversarial attacks are applied using FGSM and PGD methods. The vector reconstruction error approach is tested to differentiate between normal, anomalous, and adversarially attacked data, providing a comprehensive framework for evaluating model performance in the presence of adversarial threats.

1.7 Conclusion

The literature review highlights the growing complexity of adversarial attacks and the corresponding challenges in defending ML models. While significant advancements have been made in understanding and mitigating these threats, the review also uncovers several critical gaps in the current research. Notably, traditional AD methods often fail to differentiate between natural deviations and those induced by adversarial manipulations. Moreover, the integration of attack classification into AD remains an underexplored area. Addressing these gaps will require the development of more sophisticated detection mechanisms and the exploration of novel approaches to adversarial defense. Future research should focus on enhancing the robustness of ML models, particularly in the context of complex and evolving adversarial strategies.

CHAPTER 2

Environment Setup

2.1 Introduction

This chapter provides an overview of the development environment and tools used in this research. The setup includes programming languages, libraries, and version control systems essential for replicating the experiments and further developing the project.

2.2 Programming Language and Tools

2.2.1 Python

Python was chosen as the primary programming language due to its simplicity and extensive support for scientific computing [17]. It is particularly well-suited for machine learning and data analysis.

Figure 2.1: Python Logo

As shown in Figure 2.1, Python's logo represents its widespread adoption in the programming community.

2.2.2 GitHub

GitHub served as the version control platform, allowing for efficient management of code and collaboration. The repository contains all scripts, models, and results from the experiments. 2.2.

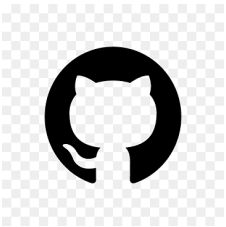


Figure 2.2: Github Logo

2.2.3 Libraries and Frameworks

PyTorch

PyTorch was the deep learning framework used for building and training neural networks [16]. It provides dynamic computation graphs and extensive support for GPU acceleration, making it ideal for research and development in machine learning.

Figure 2.3: Pytorch logo

Pandas

Pandas was used for data manipulation and analysis [11]. It offers data structures and functions needed to manipulate structured data seamlessly, which was crucial for preparing datasets and analyzing results.

Figure 2.4: Pandas logo

NumPy

NumPy provided support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays [6]. It is a foundational library for numerical computing in Python.

Figure 2.5: NumPy logo

2.3 Conclusion

Setting up the environment with these tools was crucial for the research, ensuring a robust and efficient workflow. The combination of Python with libraries like PyTorch, Pandas, and NumPy, along with version control via GitHub, provided a comprehensive platform for developing and testing machine learning models.

CHAPTER 3

Methodology

3.1 Introduction

This section outlines the pipeline developed for evaluating machine learning models' ability to detect adversarial attacks in an anomaly detection context. The methodology includes data preprocessing, model training, adversarial example generation, autoencoder training, error visualization, and dataset augmentation. The steps are detailed below.

3.2 Research Design

This study implements a multi-phase pipeline to assess the adversarial examples, analyzing reconstruction errors, and augmenting the dataset to benchmark various models. The pipeline consists of the following main phases:

1. Data Preprocessing
2. Model Training
3. Adversarial Example Generation
4. Autoencoder Training with Adversarial Examples
5. Error Visualization
6. Dataset Augmentation and Evaluation

The pipeline architecture is illustrated in Figures 3.2 and 3.1.

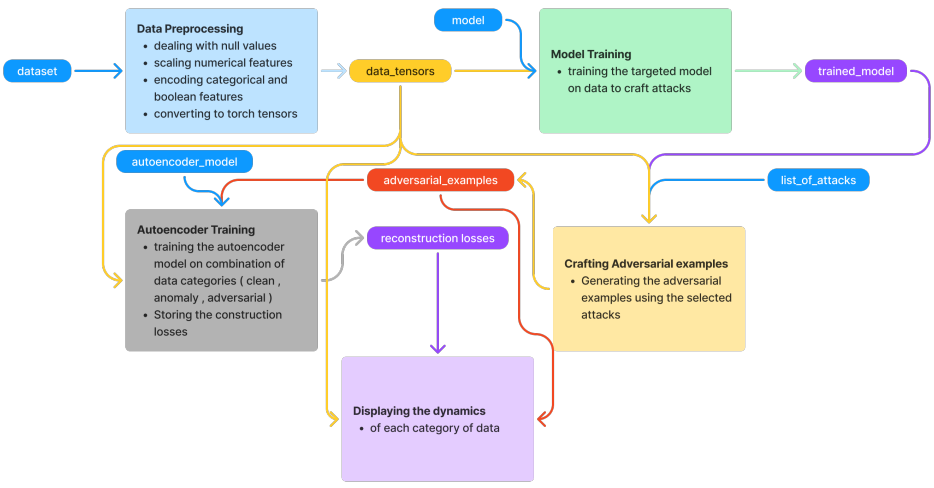


Figure 3.1: Pipeline architecture for the dynamics of the reconstruction loss

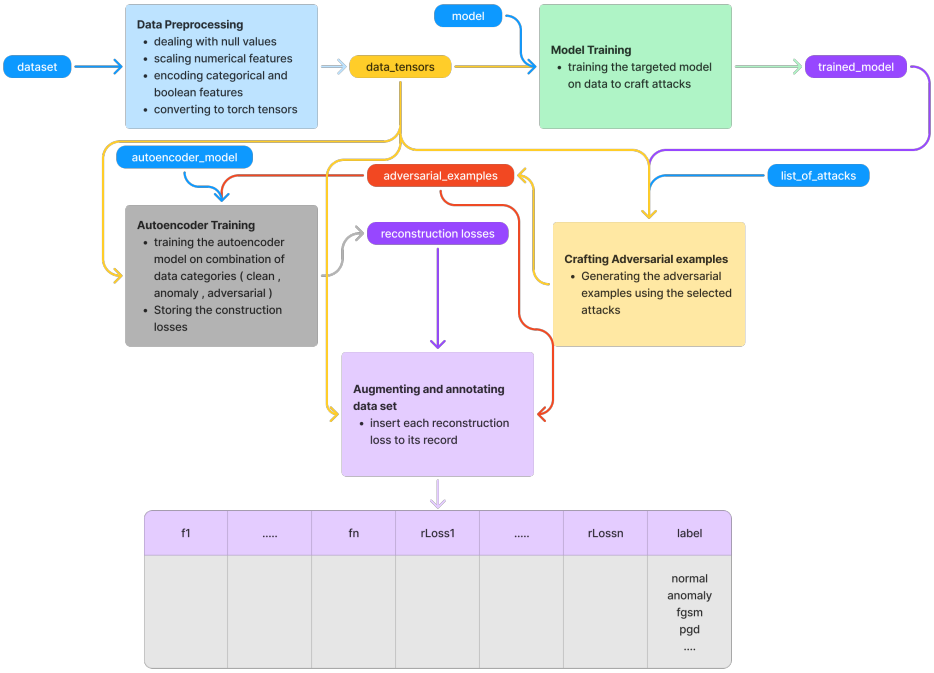


Figure 3.2: Pipeline architecture for benchmarking models on the augmented dataset

3.2.1 Data Preprocessing

- **Objective:** Clean and preprocess the input data to ensure it is suitable for model training.
- **Steps:**
  - **Cleaning:** Delete rows and columns that exceed a threshold in terms of the portion of missing values.
  - **Data Imputation:** Impute the remaining missing values according to a selected method (Mean, Mode, and Median).
  - **Normalization:** Standardize features to a common scale to enhance model training.
  - **Method:** Normalize each feature to zero mean and unit variance or min max stadarization.
  - **Feature Encoding:** Convert categorical variables into numerical values where necessary.
  - **Method:** Use one-hot encoding or label encoding as appropriate.
  - **Tools:** Python libraries such as pandas and numpy for data manipulation and preprocessing.

The class diagram of the DataPreprocessor class is shown in Figure 3.3.

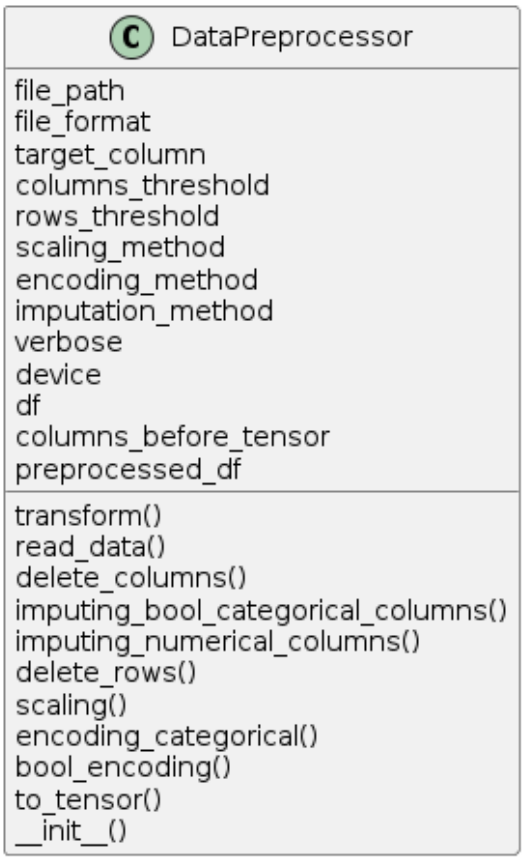


Figure 3.3: DataPreprocessor class diagram

3.2.2 Model Training

- **Objective:** Train a neural network model on the preprocessed data to be utilized for crafting white-box adversarial examples.
- **Steps:**
  - **Neural Network Architecture:** Design and train a neural network model.
  - **Architecture:** Use a Multilayer Perceptron (MLP). It has to extend an abstract class that I have created in order to implement some specific methods such as resetting input size because the initial input might be incompatible with the initial dataset in case of deleting a column during the preprocessing phase.
  - **Training:** Train the model using the training data, optimizing for accuracy in classification tasks.
  - **Optimizer:** Use Adaptive Moment Estimation (Adam) or Stochastic Gradient Descent (SGD) optimizer.
  - **Loss Function:** Utilize cross-entropy loss for classification.
  - **Tools:** PyTorch for neural network implementation.

The class diagram of the ModelTrainer class is shown in Figure 3.4.

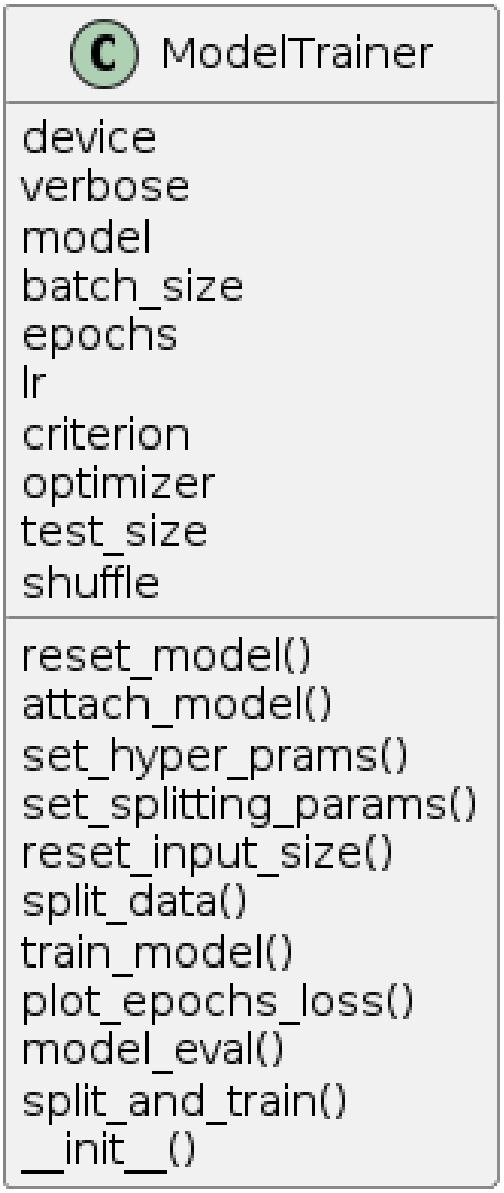


Figure 3.4: ModelTrainer class diagram

3.2.3 Adversarial Example Generation

- **Objective:** Craft adversarial examples to evaluate the robustness of the trained model.
- **Steps:**
  - **FGSM Attack:** Generate adversarial examples using the Fast Gradient Sign Method.
  - **Formula:** $x' = x + \epsilon \cdot \text{sign}(\nabla_x J(x, y))$
  - **Parameters:** ϵ (epsilon) is the perturbation size, e.g., 0.1.

- **PGD Attack:** Generate adversarial examples using Projected Gradient Descent.
- **Formula:** Apply FGSM iteratively with projection back to the feasible data space.
- **Parameters:** Use a step size and number of iterations tailored to the model and data.
- **Tools:** From scratch implementation.

The class diagram of the AdversarialCrafter class is shown in Figure 3.5.

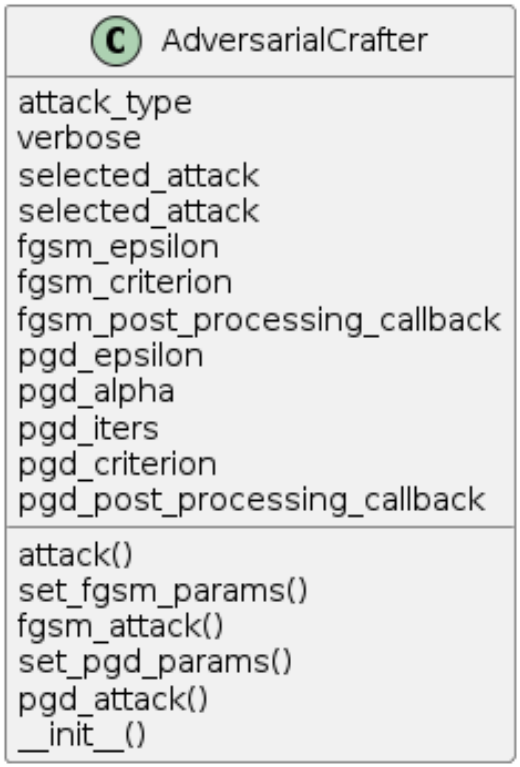


Figure 3.5: AdversarialCrafter class diagram

3.2.4 Autoencoder Training with Adversarial Examples

- **Objective:** Train an autoencoder on both clean and adversarial examples, recording the reconstruction loss for each data point.
- **Steps:**
  - **Autoencoder Architecture:** Design and implement an autoencoder.
  - **Components:** Include an encoder for compression and a decoder for reconstruction.
  - **Training:** Train the autoencoder on a combined dataset of clean and adversarial examples.
  - **Loss Function:** Use Mean Squared Error (MSE) for reconstruction loss.

- **Training Procedure:** Train until reconstruction loss stabilizes.
- **Recording Loss:** Record the reconstruction loss for each data point in the dataset.
- **Tools:** TensorFlow or PyTorch for autoencoder implementation.

The class diagram of the AutoencoderTrainer class is shown in Figure 3.6.

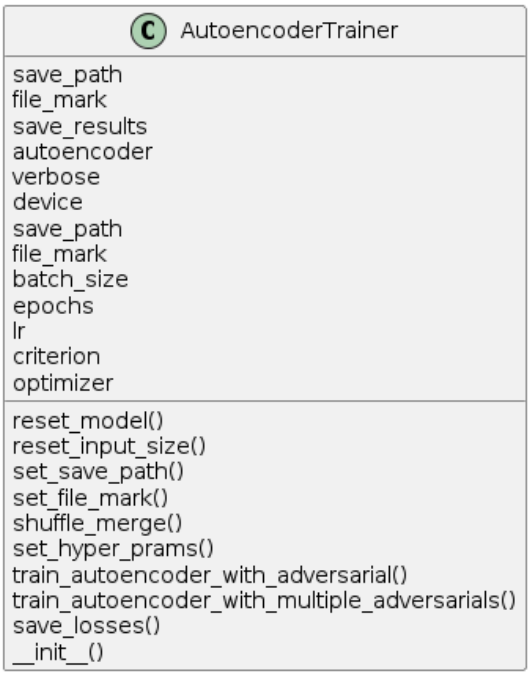


Figure 3.6: AutoencoderTrainer class diagram

3.2.5 Error Visualization

- **Objective:** Plot the dynamics of the reconstruction error to analyze differences between normal, anomalous, and adversarial data.
- **Steps:**
  - **Mean and Standard Deviation:** Calculate and plot the mean and standard deviation of the reconstruction error for each data category.
  - **Method:** Use Python libraries such as matplotlib or seaborn for plotting.
  - **Tools:** Python plotting libraries for visualization.

The class diagram of the ErrorVisualizer class is shown in Figure 3.7.

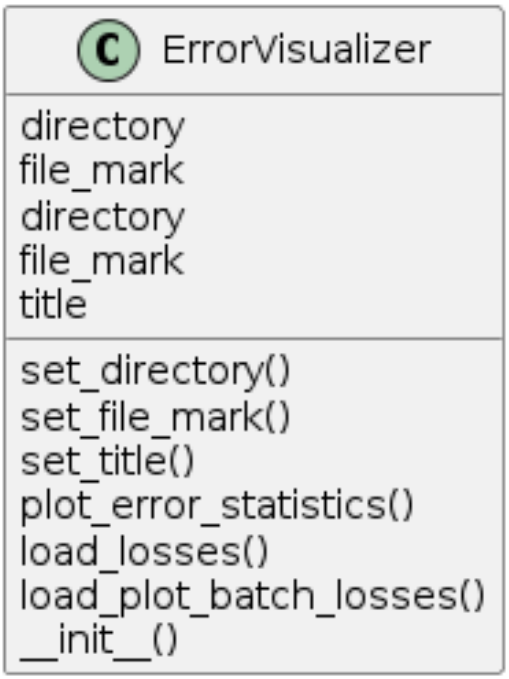


Figure 3.7: ErrorVisualizer class diagram

3.2.6 Dataset Augmentation and Evaluation

- **Objective:** Create an augmented dataset using the adversarial examples and reconstruction loss, and evaluate candidate models.
- **Steps:**
  - **Dataset Augmentation:** Augment the dataset both vertically (by adding adversarial examples) and horizontally (by adding reconstruction error as a vector).
  - **Augmentation Strategy:** Create three datasets:
    - **FGSM Only:** Inject adversarial examples generated by FGSM.
    - **PGD Only:** Inject adversarial examples generated by PGD.
    - **Both FGSM and PGD:** Inject examples from both methods.
  - **Poisoning Portion:** Use a 10% poisoning portion of adversarial examples in each variant.
  - **Variants Creation:** Develop three variants for each augmented dataset:
    - **Variant 1:** Contains only the reconstruction loss.
    - **Variant 2:** Contains only the original dataset features.
    - **Variant 3:** Contains both the reconstruction loss and the original dataset features.
  - **Model Training and Evaluation:** Train candidate models on each variant and evaluate their performance.
  - **Candidate Models:** Use Support Vector Machines (SVM), Decision Trees, and Naive Bayes.

- **Evaluation Metrics:** Use accuracy, precision, recall, and F1 score to assess model performance.
- **Tools:** scikit-learn for model implementation and evaluation.

The class diagram of the DataAugmenter class is shown in Figure 3.8.

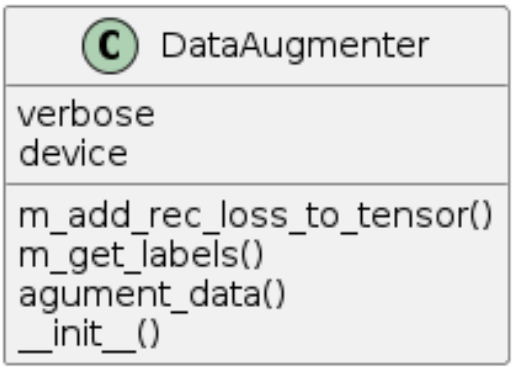


Figure 3.8: DataAugmenter class diagram

3.3 Conclusion

The developed methodology presents a flexible and adaptable pipeline for evaluating machine learning models' ability to detect adversarial attacks in anomaly detection contexts. With its modular design encompassing data preprocessing, model training, adversarial example generation, and autoencoder training, the pipeline supports comprehensive evaluation of model robustness. The integration of error visualization and dataset augmentation allows for detailed analysis and benchmarking.

This plug-and-play pipeline is easily generalizable to different datasets and accommodates the addition of various attack types and models. It offers a valuable tool for enhancing machine learning security through rigorous testing and exploration of new defense strategies.

CHAPTER 4

Results

4.1 Introduction

In this chapter, we delve into the intricate dynamics of reconstruction loss across various data types—normal, anomalous, and adversarially crafted using FGSM and PGD attacks. By examining the mean and variance of reconstruction loss, we aim to understand its potential as a distinguishing metric between these different data categories. This analysis is essential for advancing adversarial detection techniques, particularly in understanding how adversarial examples mimic the statistical properties of anomalous data. Furthermore, we evaluate the effectiveness of a novel feature engineering approach, which treats reconstruction error as a vector rather than a scalar, and explore its impact on adversarial attack detection performance. The chapter also covers our data augmentation strategies and their implications for model robustness, ultimately providing insights that align with and contribute to existing literature on adversarial detection.

4.2 Dynamics of Reconstruction Loss

In the initial phase of our study, we focused on analyzing the dynamics of reconstruction loss between normal, anomalous, and adversarially crafted data (FGSM and PGD) in terms of mean and variance. This analysis is crucial to understand how reconstruction loss behaves across different types of data, and whether these dynamics can aid in distinguishing between them.

We present three plots to illustrate these dynamics:

- **Plot 1: Normal vs. Anomalous Data** 4.1 - This plot shows the distribution of reconstruction loss for normal and anomalous data, highlighting the differences in mean and variance.
- **Plot 2: Normal, Anomalous, and FGSM Data** 4.2 - This plot compares the reconstruction loss for normal, anomalous, and FGSM adversarial data, demonstrating the similarity in variance between anomalous and FGSM data.
- **Plot 3: Normal, Anomalous, and PGD Data** 4.3 - This plot examines the reconstruction loss for normal, anomalous, and PGD adversarial data, again showing similar variance between anomalous and PGD data.

The results from these plots indicate that:

- Normal vs. Anomalous Data: There is a clear distinction between normal and anomalous data in terms of both the mean and variance of the reconstruction loss. Normal data exhibits lower mean and variance compared to anomalous data.

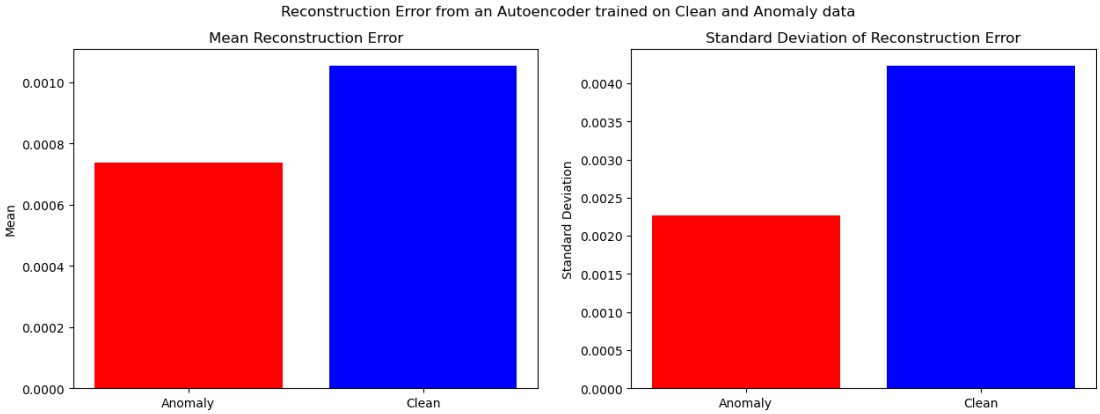


Figure 4.1: Reconstruction Loss Dynamics: Normal vs. Anomalous Data

- Normal, Anomalous, and FGSM Data: The variance of reconstruction loss for FGSM data is similar to that of anomalous data, suggesting that FGSM adversarial examples mimic the statistical properties of anomalous data.

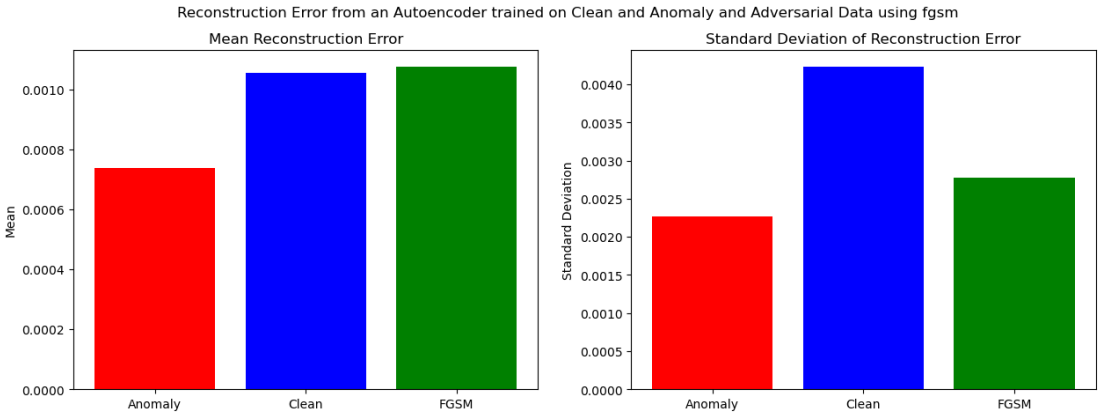


Figure 4.2: Reconstruction Loss Dynamics: Normal, Anomalous, and FGSM Data

- Normal, Anomalous, and PGD Data: Similarly, PGD adversarial data shows variance in reconstruction loss that is comparable to anomalous data, indicating that PGD attacks also create examples that resemble anomalous patterns in terms of variance.

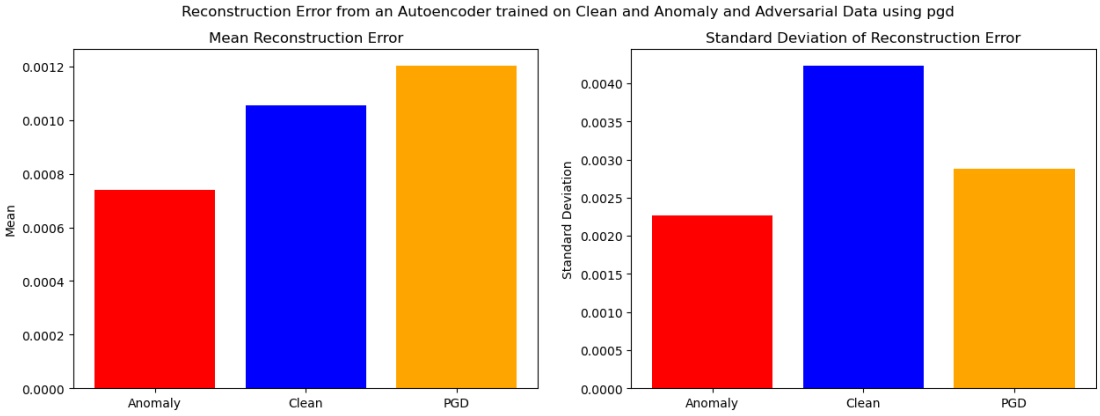


Figure 4.3: Reconstruction Loss Dynamics: Normal, Anomalous, and PGD Data

4.3 Evaluation of Feature Engineering Technique

The study investigated the effectiveness of a novel feature engineering technique that treats the reconstruction error as a vector[21] rather than a scalar in the context of adversarial attack detection. Unfortunately, this approach did not yield significant improvements in the model's performance. Specifically:

- **Reconstruction Error as a Vector:** The feature engineering technique of considering the reconstruction loss as a vector was tested in different combinations of experiments when we included it with the original features. The results indicated that the vectorized reconstruction error did not significantly enhance the detection capabilities as expected.
- **Comparative Analysis:** The performance metrics was analyzed using various candidate models (SVM, Decision Trees, Naive Bayes, etc.)

4.3.1 Data Augmentation and Experimental Setup

This subsection details the data augmentation strategies employed for evaluating the performance of machine learning models in detecting adversarial attacks. Three distinct cases were considered: augmentation with FGSM only fig4.5 and fig4.6, PGD only fig4.7 and fig4.8, and a combination of both FGSM and PGD fig4.9 and fig4.10. Each case was further divided into three experimental variants based on the features used: reconstruction loss vector only, original dataset features only, and a combination of both.

Data Augmentation Process: Adversarial examples were generated using the Fast Gradient Sign Method and Projected Gradient Descent to augment the dataset. A perturbation size (ϵ) of 0.1 was used for FGSM, and PGD was applied with an iterative process and specific step size parameters.

Poisoning Portion: A 10% poisoning portion was maintained in each case, ensuring that 10% of the dataset consisted of adversarial examples generated by either FGSM, PGD, or both.

The experimental setup plan is shown in Figure 4.4.

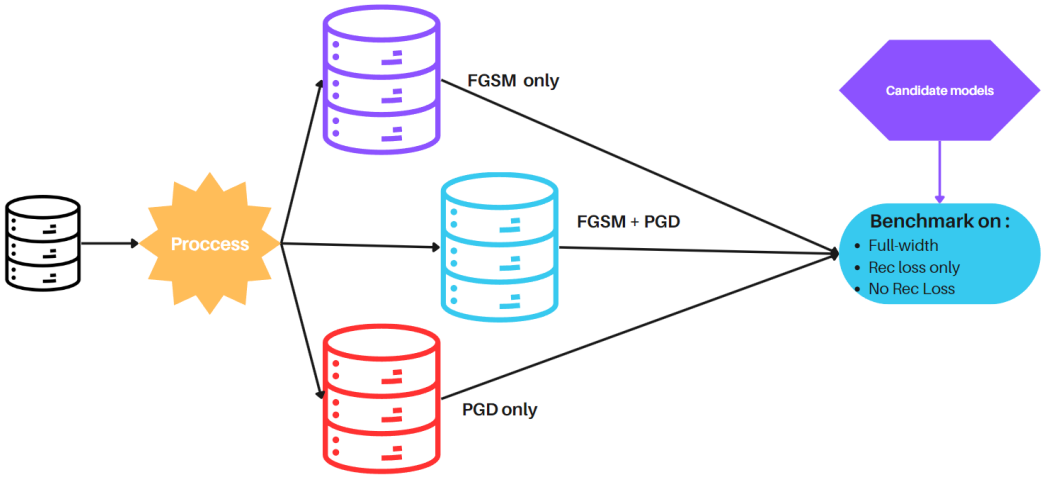


Figure 4.4: Illustration of the data augmentation process for FGSM, PGD, and both.

4.3.2 Mention in Supervisor's Paper

My contributions to the development and evaluation of adversarial detection techniques have been recognized by my supervisors, Pierre-Martin Tardif,Jordan Felicien Masakuna and DJEFF KANDA NKASHAMA in a paper currently under review at NeurIPS. The paper proposes a novel preprocessing approach for anomaly detection that focuses on data cleaning without relying on contamination ratio information. The approach is based on dynamically analyzing the cosine similarity between each data point and its reconstruction during autoencoder training. This method has shown significant improvements in data quality and model performance across multiple benchmarks, and my work on adversarial detection and autoencoder models played a key role in these findings.

4.3.3 Experimental Results:

The results of the experiments are presented below, showcasing the evaluation metrics depicted in the confusion matrices and the boxplots for accuracy, recall, F1 score, and precision, as illustrated in Figure 4.4.

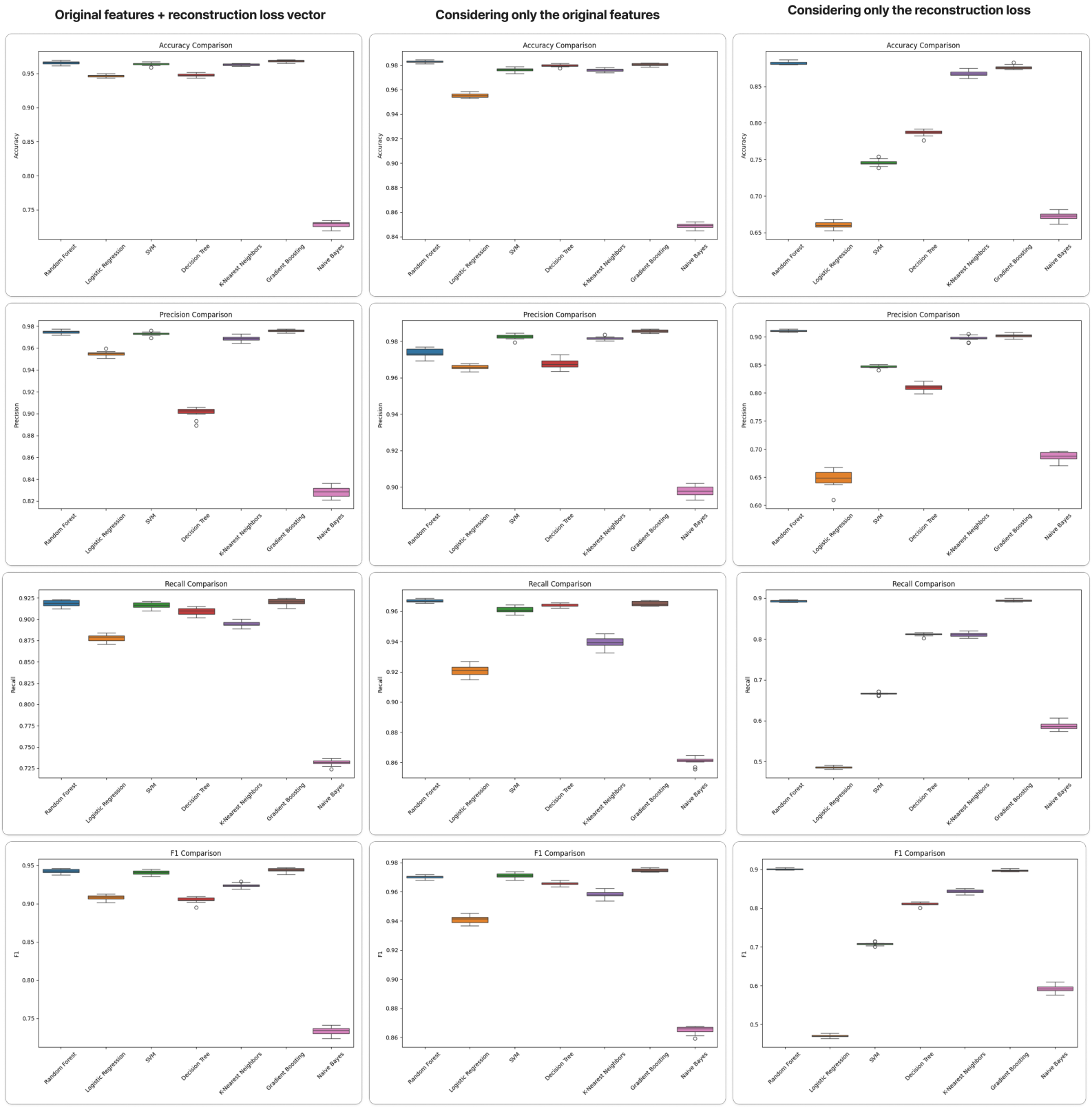


Figure 4.5: FGSM score

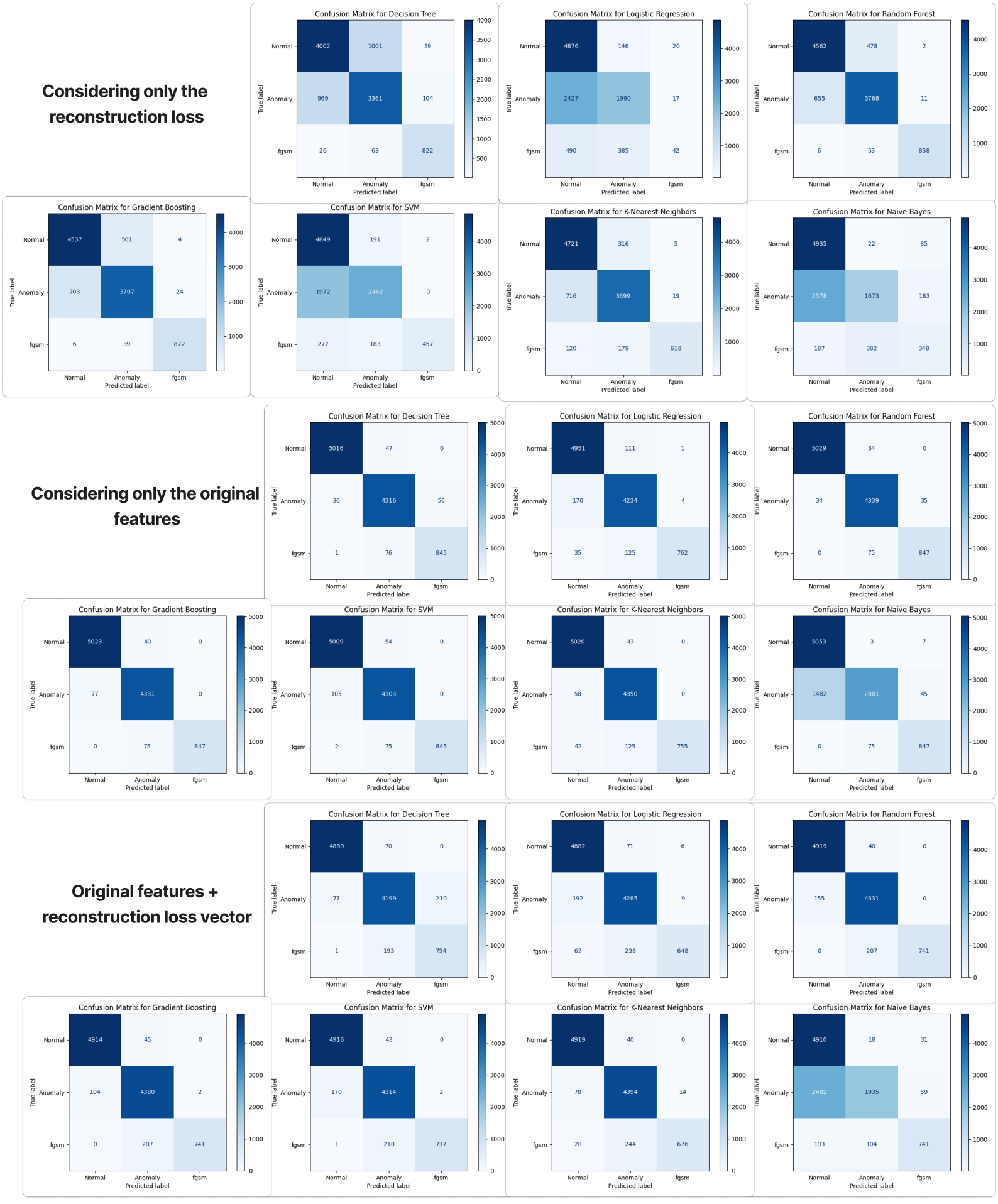


Figure 4.6: Confusion matrices for data poisoned only with FGSM

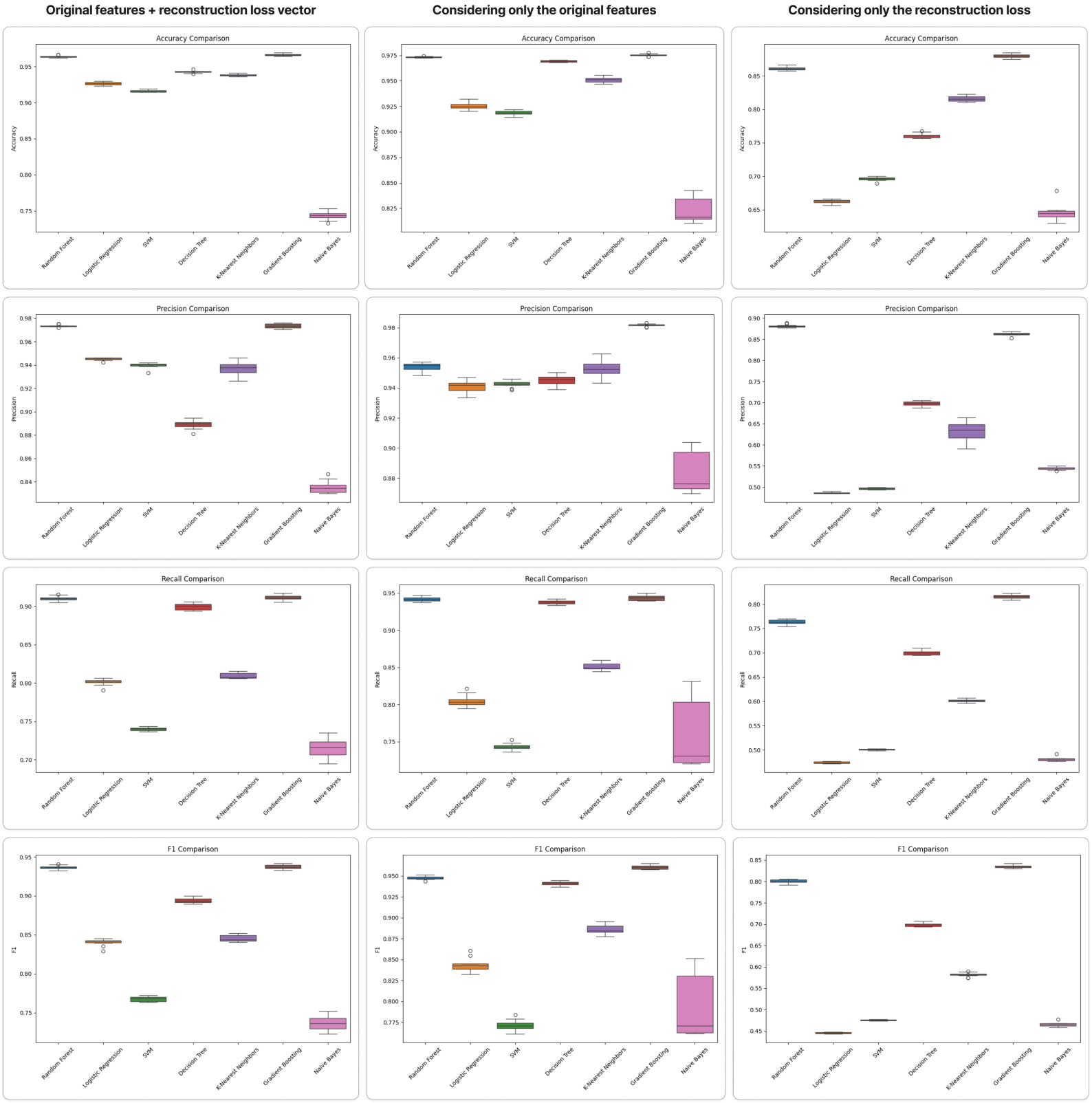


Figure 4.7: PGD score

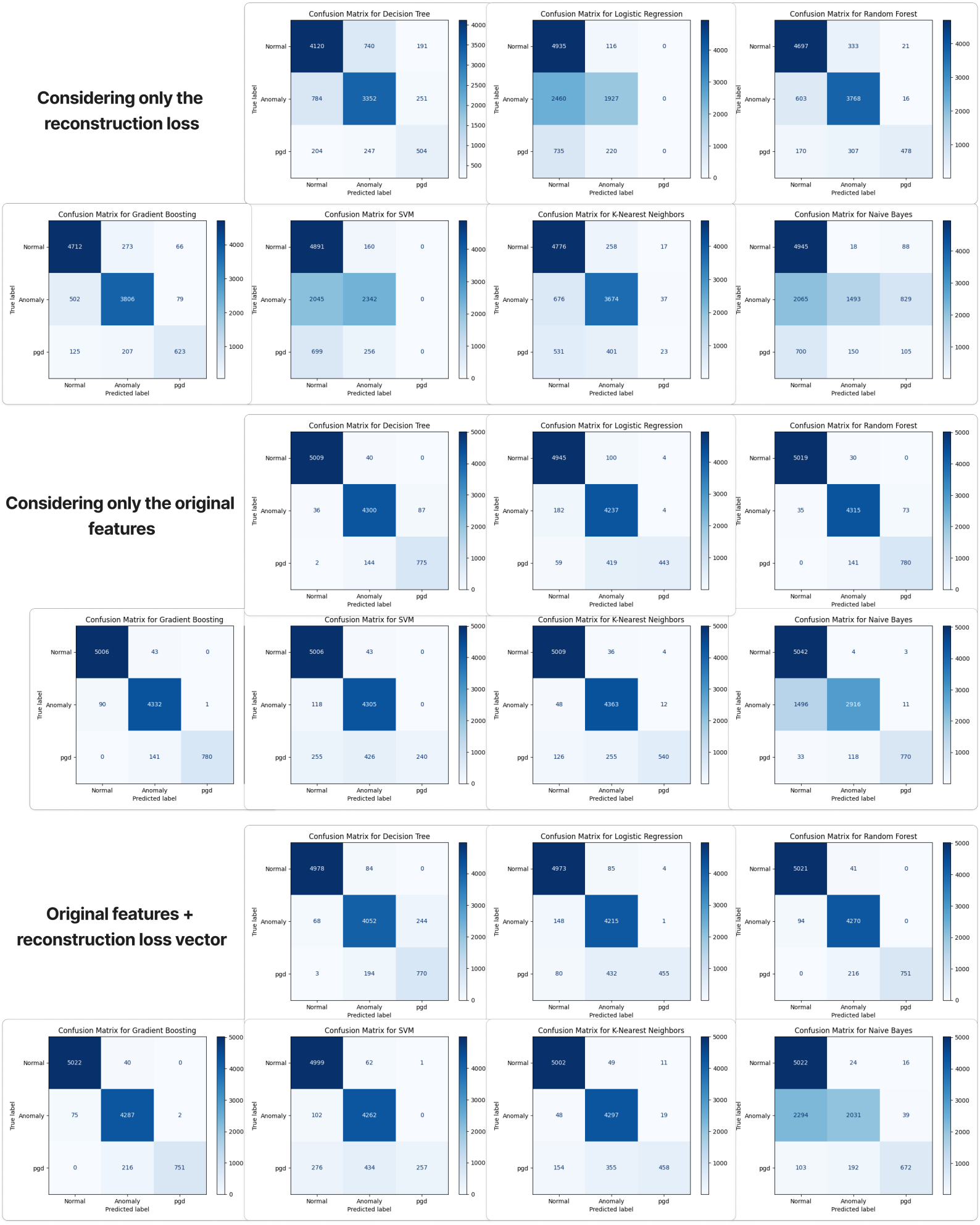


Figure 4.8: Confusion matrices for data poisoned only with PGD

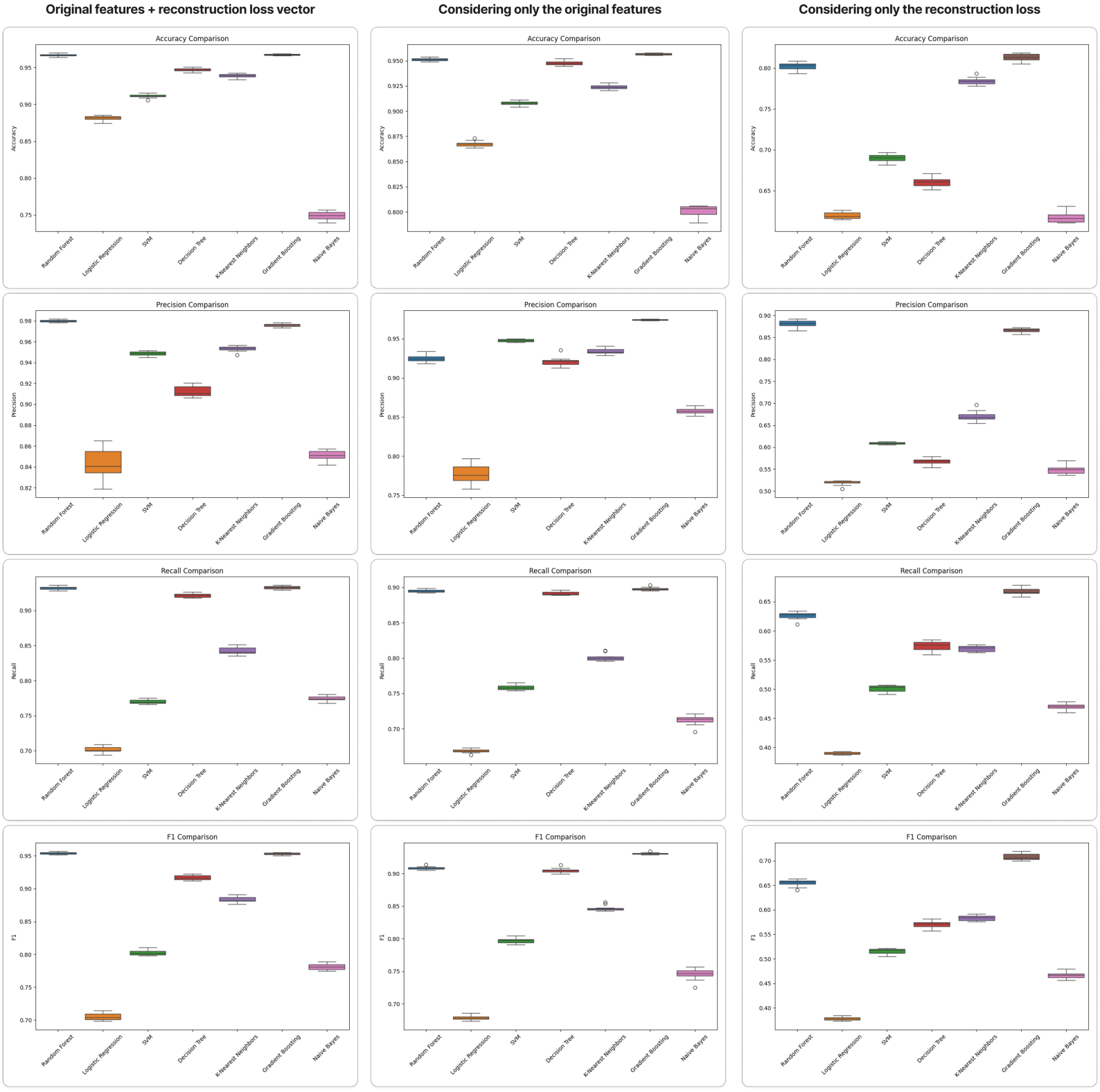
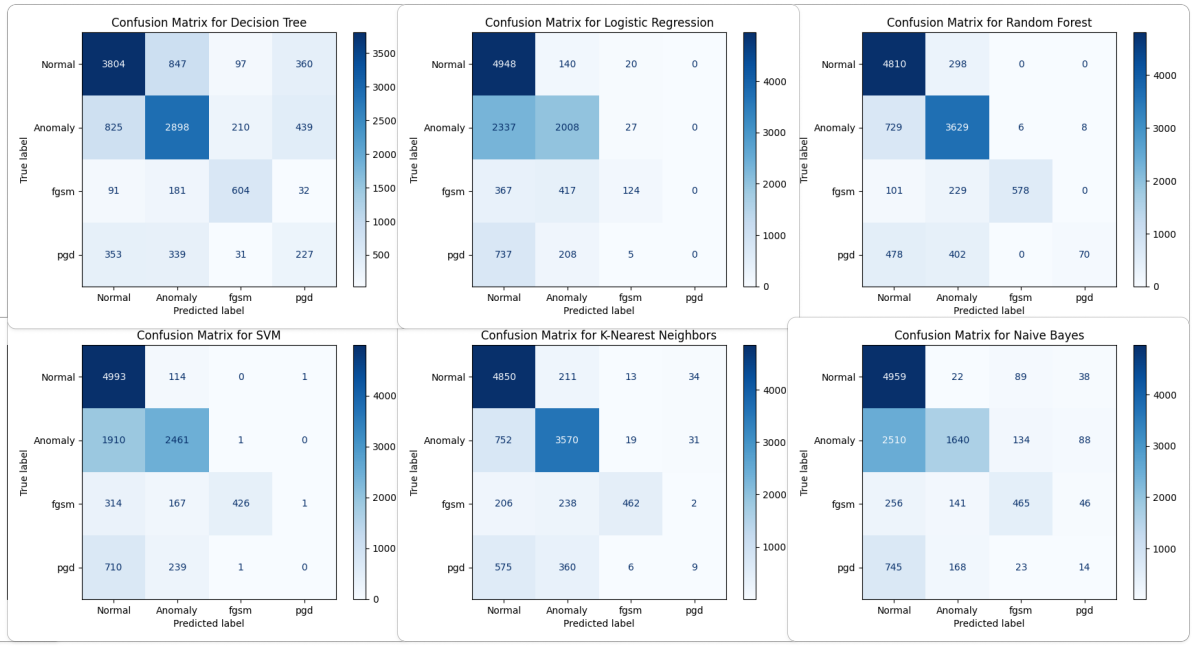
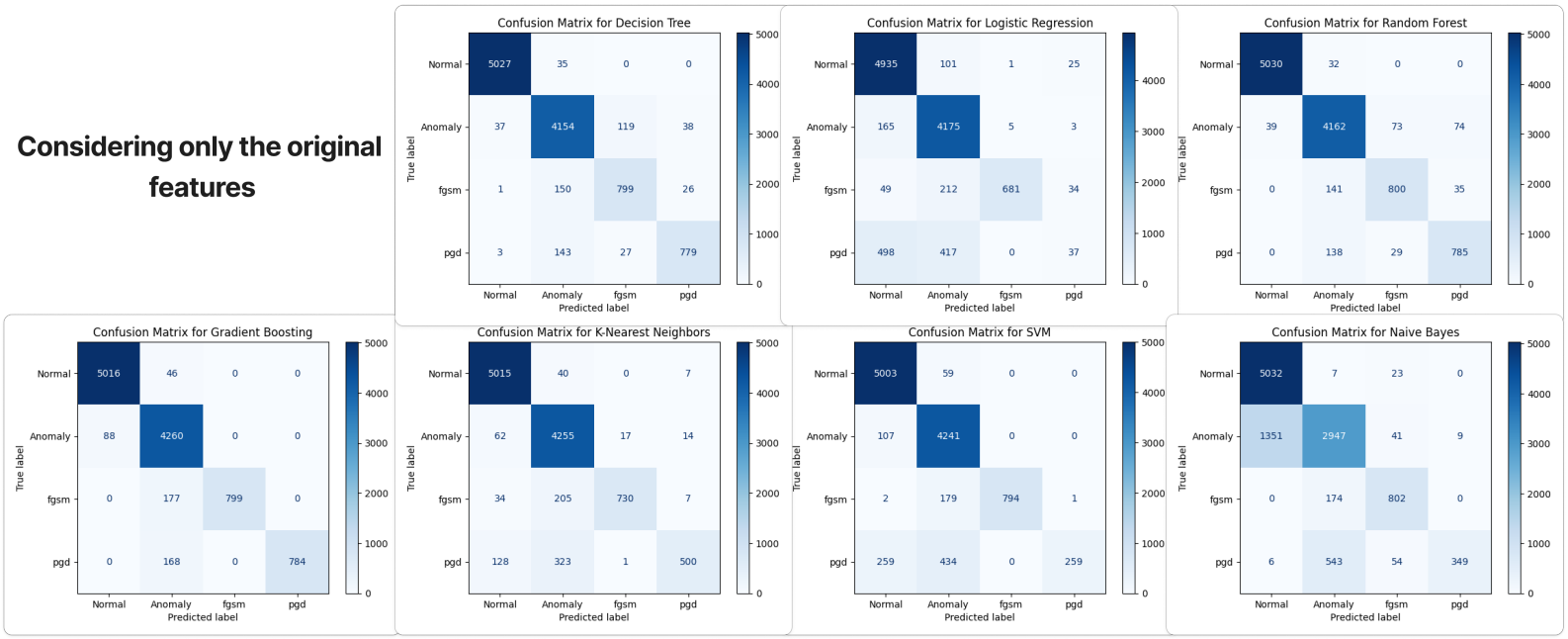


Figure 4.9: FGSM + PGD score

Considering only the reconstruction loss



Considering only the original features



Original features + reconstruction loss vector

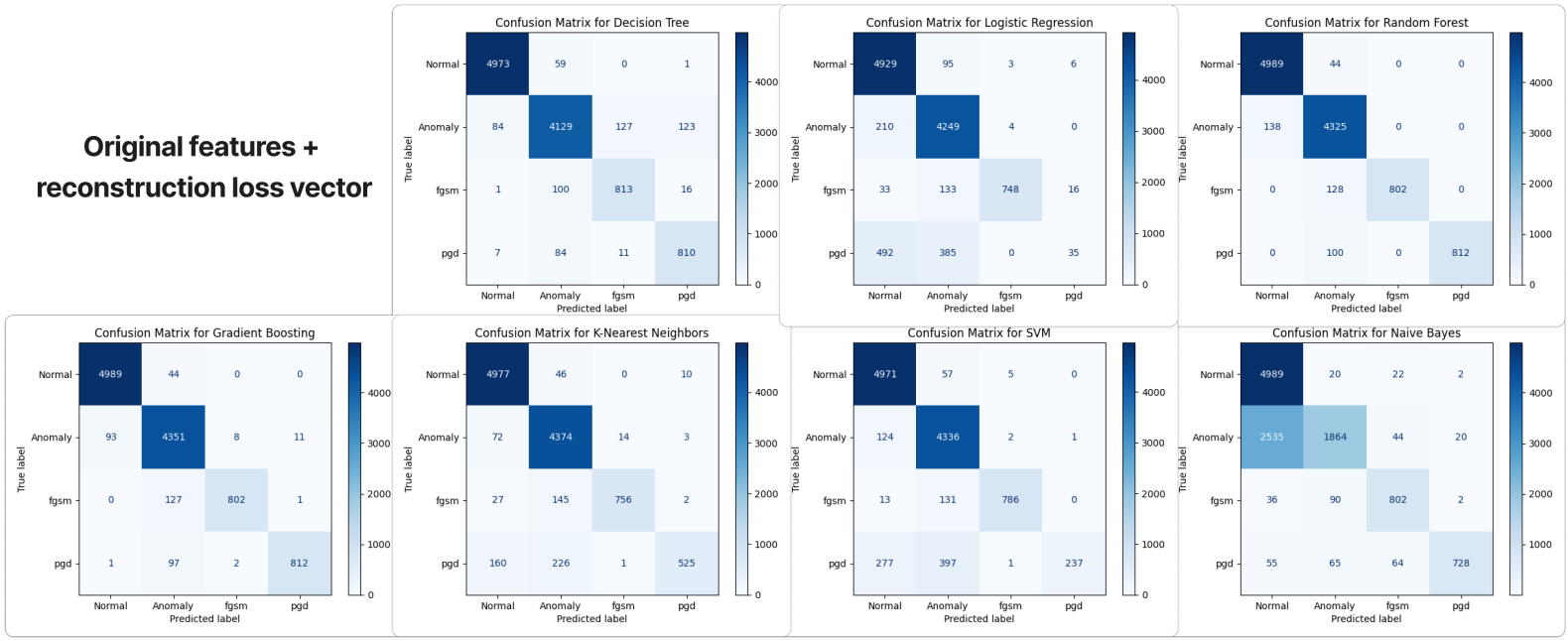


Figure 4.10: Confusion matrices for data poisoned with both of PGD and FGSM

4.4 Discussion

4.4.1 Interpretation of Results

The evaluation of machine learning models for detecting adversarial attacks provided several insights into the dynamics of autoencoder reconstruction loss and the performance of various detection models. Key observations include:

- **Feature Engineering Technique**: The use of reconstruction error as a vector did not enhance the models' ability to detect adversarial attacks significantly. Despite theoretical advantages, the empirical results suggested that treating the reconstruction loss as a vector did not capture additional discriminative features necessary for improving detection accuracy.
- **Difficulty in Detecting PGD Attacks**: The comparative analysis revealed that models found it more challenging to detect PGD attacks than FGSM attacks. This aligns with existing literature that identifies PGD as a more potent adversarial technique due to its iterative nature and refined attack strategies [10].

4.4.2 Comparison with Existing Literature

Our findings align with recent advancements in adversarial detection:

- **Adversarial Detection Techniques**: Similar to Moayeri and Feizi in their work on self-supervised embeddings, this study highlights the challenges in detecting sophisticated attacks like PGD compared to simpler ones like FGSM. Moayeri's method, which leverages SimCLR embeddings, also emphasizes the difficulty of generalizing across various attack types, which is consistent with the observed difficulty in detecting PGD in our study [13].
- **Autoencoder-Based Detection**: Hasan Torabi's findings indicate that using vectorized reconstruction error can improve performance, particularly in hierarchical classification settings. This suggests that while vectorized errors may not show broad practical benefits in all adversarial scenarios, they offer specific advantages in hierarchical classification contexts.[21].
- **Domain-Specific Defenses**: The study's results also reflect the challenges seen in Jiahao Shao et al. where domain-specific knowledge was necessary to improve detection robustness in ECG data against adversarial attacks. The general trend suggests that more sophisticated or context-aware approaches are required for effective adversarial detection [18].

4.4.3 Implications

The findings of this study have several implications:

- **Model Robustness**: The difficulty in detecting PGD attacks suggests that current models and detection strategies may not be sufficiently robust against iterative adversarial attacks. Enhancing model robustness requires

exploring advanced techniques that can handle the subtleties of such sophisticated attacks.

- **Feature Engineering**: The lack of improvement using vectorized reconstruction error highlights the need for innovative feature engineering techniques. Traditional approaches may not suffice, and more context-aware or domain-specific features might be necessary to improve detection accuracy.

4.4.4 Limitations

Several limitations in this study could impact the interpretation of the results:

- **Dataset and Attack Specificity**: The study was conducted on a specific dataset (NSL-KDD) and focused on FGSM and PGD attacks. Results may vary with different datasets or adversarial techniques.
- **Feature Engineering Technique**: The implementation of the vectorized reconstruction error might not have fully captured the potential benefits, possibly due to the simplicity of the method or the specific characteristics of the dataset.
- **Model Selection**: The study used a predefined set of models (SVM, Decision Trees, Naive Bayes , Random forests , Gradient Boosting , KNN and Logistic Regression). Different models or more advanced architectures might yield different results.

4.4.5 Future Research

Based on the findings and limitations, several avenues for future research are suggested:

- **Advanced Feature Engineering**: Investigate more sophisticated feature engineering techniques, possibly involving domain-specific knowledge or higher-dimensional feature representations, to improve detection accuracy.
- **Adversarial Training**: Explore the integration of adversarial training into detection models to enhance robustness against iterative and more sophisticated adversarial attacks.
- **Broader Evaluation**: Conduct studies across various datasets and adversarial attack types to generalize the findings and develop more versatile detection methods.
- **Hybrid Approaches**: Develop hybrid approaches that combine multiple detection strategies, such as embedding-based methods and traditional anomaly detection, to leverage their complementary strengths.

4.5 Conclusion

The comprehensive analysis presented in this chapter sheds light on the behavior of reconstruction loss across normal, anomalous, and adversarial data, revealing that adversarial examples, particularly those crafted using PGD, closely mimic the statistical characteristics of anomalous data. Despite the potential of the vectorized reconstruction error as a feature engineering technique, it did not yield significant improvements in detection accuracy. Our findings underscore the complexity of detecting sophisticated adversarial attacks like PGD, highlighting the limitations of current detection models and the need for advanced, context-aware approaches. These insights not only contribute to the ongoing research in adversarial detection but also pave the way for future studies aimed at enhancing model robustness and developing more effective detection strategies.

CHAPTER 5

Conclusion

This study aimed to evaluate the capability of machine learning models in detecting adversarial attacks on a given dataset, specifically focusing on the dynamics of autoencoder reconstruction loss and the effectiveness of a novel feature engineering technique. The primary findings can be summarized as follows:

- **Feature Engineering Technique**: The use of reconstruction error as a vector did not significantly enhance the detection performance of the models. Despite its theoretical advantages, the empirical results indicated that the idea of taking this approach as a featue engineering technique did not provide a significant additional discriminative power.
- **Detection of Adversarial Attacks**: The study highlighted the significant challenge in detecting PGD attacks compared to FGSM. Models consistently exhibited higher accuracy, recall, precision, and F1 scores for FGSM attacks, while struggling with the more sophisticated and iterative nature of PGD attacks.
- **Model Performance**: The detailed analysis using boxplots and confusion matrices provided insights into the strengths and weaknesses of various candidate models. The results underscored the need for advanced detection mechanisms to handle sophisticated adversarial attacks effectively.

5.1 Key Takeaways

- **Robustness of Detection Models**: The difficulty in detecting PGD attacks underscores the necessity for more robust and sophisticated adversarial detection techniques. Current models and feature engineering approaches may not be sufficient to handle advanced attack strategies.
- **Future Directions**: The findings suggest several avenues for future research, including the exploration of advanced feature engineering techniques, integration of adversarial training, broader evaluations across different datasets and attack types, and the development of hybrid detection approaches.

5.2 Implications for Practice

- **Practical Applications**: The insights from this study are relevant for practitioners seeking to enhance the robustness of anomaly detection systems in the presence of adversarial attacks. The findings can inform the development of more effective defense mechanisms against sophisticated adversarial techniques.
- **Research and Development**: The study highlights the need for continued research into adversarial detection, encouraging the exploration of innovative methods and the refinement of existing approaches to improve detection accuracy and robustness.

In conclusion, while the feature engineering technique of using reconstruction error as a vector did not yield significant improvements, the study provides valuable insights into the challenges and potential directions for enhancing adversarial detection. Addressing these challenges requires ongoing research and the development of more advanced and robust detection methods to effectively counter sophisticated adversarial attacks.

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS'16. ACM, October 2016.

[2] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[3] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection. *ACM Computing Surveys*, 41(3):1–58, July 2009.

[4] Ankan Dash, Junyi Ye, and Guiling Wang. A review of generative adversarial networks (gans) and its applications in a wide variety of disciplines: From medical to remote sensing. *IEEE Access*, 12:18330–18357, 2024.

[5] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

[6] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[7] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[8] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015.

[9] Yuening Li, Daochen Zha, Praveen Venugopal, Na Zou, and Xia Hu. Pyodds: An end-to-end outlier detection system with automated machine learning. In *Companion Proceedings of the Web Conference 2020*, WWW '20, page 153–157, New York, NY, USA, 2020. Association for Computing Machinery.

[10] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

[11] Wes McKinney et al. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX, 2010.

[12] David Miller, Zhen Xiang, and George Kesidis. Adversarial learning targeting deep neural network classification: A comprehensive review of defenses against attacks this article provides a contemporary survey of adversarial learning (al), focused particularly on defenses against attacks on deep neural network classifiers. *Proceedings of the IEEE*, PP:1–1, 02 2020.

[13] Mazda Moayeri and Soheil Feizi. Sample efficient detection and classification of adversarial attacks via self-supervised embeddings, 2021.

[14] Akarsh K. Nair, Ebin Deni Raj, and Jayakrushna Sahoo. A robust analysis of adversarial attacks on federated learning environments. *Comput. Stand. Interfaces*, 86:103723, 2023.

[15] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity, 2019.

[16] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[17] Guido Rossum. Python reference manual. Technical report, NLD, 1995.

[18] Jiahao Shao, Shijia Geng, Zhaoji Fu, Weilun Xu, Tong Liu, and Shenda Hong. Cardiodefense: Defending against adversarial attack in ecg classification with adversarial distillation training. *Biomedical Signal Processing and Control*, 91:105922, 2024.

[19] David Silver, Aja Huang, Christopher Maddison, Arthur Guez, Laurent Sifre, George Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 01 2016.

[20] Christian Szegedy et al. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[21] Hasan Torabi, Seyedeh Leili Mirtaheri, and Sergio Greco. Practical autoencoder based anomaly detection by using vector reconstruction error. *Cybersecurity*, 6(1):1, 2023.

[22] Dimitris Tsipras et al. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[23] Mingfu Xue, Chengxiang Yuan, Heyi Wu, Yushu Zhang, and Weiqiang Liu. Machine learning security: Threats, countermeasures, and evaluations. *IEEE Access*, 8:74720–74742, 2020.