

Nesnelerin İnterneti Ağlarında Çekişmeli Saldırıların Tespiti

Elif Değirmenci

**DOKTORA TEZİ**

Bilgisayar Mühendisliği Anabilim Dalı

Aralık 2024



Adversarial Attack Detection on Internet of Things Networks

Elif Değirmenci

**DOCTORAL DISSERTATION**

Department of Computer Engineering

December 2024

# Adversarial Attack Detection on Internet of Things Networks

A thesis is submitted to the Eskisehir Osmangazi University  
Graduate School of Natural and Applied Sciences in partial  
fulfillment of the requirements for the degree of Doctoral Dissertation  
in Discipline of Computer Software of the Department of Computer Engineering  
by  
Elif Değirmenci

Supervisor: Prof. Dr. Ahmet YAZICI  
Co-Supervisor: Asst. Prof. Dr. İlker ÖZÇELİK

This thesis is supported by the VALU3S project that has received funding from the ECSEL  
Joint Undertaking under grant agreement No 876852 and the Scientific and Technical  
Research Council of Turkey (TUBITAK) with contract no 120N800: “Verification and  
Validation of Automated System’ Safety and Security”

December 2024

## APPROVAL OF THE THESIS

The thesis titled “Adversarial Attack Detection on Internet of Things Networks” and submitted by Elif Değirmenci has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Doctoral Dissertation in the Department of Computer Engineering.

Supervisor : Prof. Dr. Ahmet YAZICI

Co-Supervisor : Asst. Prof. Dr. İlker ÖZÇELİK

Examining Committee Members:

Member: Prof. Dr. Ahmet YAZICI

Member: Dr. Öğr. Üyesi Esra N. YOLAÇAN

Member: Doç. Dr. Uğur YAYAN

Member: Dr. Öğr. Üyesi Mehmet SEVRİ

Member: Doç. Dr. Ahmet ARSLAN

Graduation of Elif Değirmenci was approved by the Graduate School Board Decision on ..... with the decision number of .....

Prof. Dr. ....

Director of the Institute

## ETHICAL STATEMENT

I hereby declare that this thesis study titled “Adversarial Attack Detection on Internet of Things Networks” has been prepared in accordance with the thesis writing rules of Eskisehir Osmangazi University Graduate School of Natural and Applied Sciences under the academic consultancy of my supervisors Prof. Dr. Ahmet Yazıcı and Asst. Prof. Dr. İlker Özçelik. I hereby declare that the work presented in this thesis is original. I also declare that I have respected scientific ethical principles and rules in all stages of my thesis study, all information and data presented in this thesis have been obtained within the scope of scientific and academic ethical principles and rules, and all materials used in this thesis which are not original to this work have been fully cited and referenced, and all knowledge, documents, and results have been presented in accordance with scientific ethical principles and rules.

16/12/2024

Elif Değirmenci

Sign

## ÖZET

Çekişmeli saldırılar, yapay zeka tabanlı modellerindeki savunmasızlıkları sömürerek, bu sistemlerin farklı sektörlerde yaygınlaşmasıyla büyüyen bir tehdit oluşturur. Bu saldırılar, verilere küçük ve genellikle fark edilemeyen bozulmalar ekleyerek modelleri yanıltır. Bu saldırılar, özellikle Nesnelerin İnterneti (IoT) için Saldırı Tespit Sistemleri (IDS) gibi güvenlik açısından kritik uygulamalarda giderek daha fazla kullanılan derin öğrenme modelleri için de önemli bir zorluk oluşturmaktadır.

Bu tez, ağ trafiği üzerindeki çekişmeli saldırıların tespit edilmesini incelemektedir. Bu saldırılara karşı savunmasızlık yüzeyi, mevcut saldırı yöntemleri ve bu saldırıların tespit mekanizmaları üzerine odaklanmaktadır. İlk olarak, makine öğrenimi yaşam döngüsündeki savunmasızlık yüzeyleri tanımlanarak, literatürdeki örnekler ve potansiyel zayıflıklar değerlendirilmiştir. İkinci olarak, ROSIDS23 veri seti sunulmuştur. ROSIDS23 veri seti, ROS tabanlı robotik sistemlerdeki siber güvenlik araştırmaları için kapsamlı bir trafik verisi sunmakta ve çoklu saldırı türlerini içeren detaylı analizler için güçlü bir kaynak oluşturmaktadır. Üçüncü olarak, Yeniden Yapılandırma Tabanlı Çekişmeli Saldırı Tespiti (READ) yaklaşımı önerilmiştir.

Bu tez kapsamında tanımlanan savunmasızlık yüzeyi (vulnerability surface), geleneksel saldırı yüzeyi (attack surface) kavramını genişleterek makine öğrenimi yaşam döngüsü boyunca hem bilinen hem de potansiyel savunmasızlıkları kapsar. Zaafiyet yüzeyi saldırganların zayıflıkları nasıl istismar edebileceğine dair kapsamlı bir bakış açısı sunar. Veri toplama aşamasından yeniden eğitim aşamasına kadar yapay zeka sistemlerinin bütünlüğünü ve dayanıklılığında olabilecek noktalar incelenmiş, mevcut ve gelecekteki zorluklara yönelik savunmalar geliştirilebilecek noktalar sunulmuştur.

Bu tezin bir diğer katkısı, özellikle ROS tabanlı siber saldırılara odaklanarak, robotik sistemlerin siber güvenlik araştırmaları için özel olarak tasarlanmış kapsamlı bir veri kümesi olan ROSIDS23 veri kümesidir. Bu veri kümesi, tcpdump ağ protokolü analizörü kullanılarak ROS arakatmanından elde edilen zararsız (benign) ve çeşitli saldırı trafiği

verilerini içerir. CICFlowMeter aracıyla çıkarılan bu veri setinde seksen iki tane ağ trafiği özelliği içermektedir. Veri kümesi, ham ve işlenmiş verileri içerir. Unauthorized publish, unauthorized subscribe, subscriber flood ve DoS saldırı türlerinin yanı sıra zararsız örneklerin ayrıntılı kayıtlarını kapsar. Düşmanca ve zararsız trafik modellerinin sağlam bir şekilde analiz edilmesini destekleyen ROSIDS23, robotik sistemlerde saldırı tespit sistemlerini geliştirmek için değerlidir.

Önerilen Hata Tespit Değeri tabanlı Çekişmeli Tespiti (READ) tespit yöntemi, çekişmeli saldırıları belirlemek için dört metriği bir arada kullanarak hem mevcut metrikleri hem de yeniden yapılandırma hata oranı metriğini birleştirir ve yüksek tespit oranlarına ulaşır. Sonuçlar, READ'in IDS performansını artırarak, çekişmeli etkileri önemli ölçüde azalttığını göstermekte ve özellikle küçük pertürbasyon seviyelerinde de etkili olduğunu kanıtlamaktadır. Önerilen yöntem çekişmeli saldırıları 92-100%, başarı oranıyla tespit etmektedir. Deneysel sonuçlar, çekişmeli saldırıların meydana geldiği senaryolarda IDS'nin doğruluğu %51 seviyelerine kadar düşebilirken, READ'in IDS ile entegrasyonunun doğruluğu %98'e kadar artırdığını ve sistem güvenilirliğini önemli ölçüde güçlendirdiğini göstermektedir. Bu durum, READ'in çekişmeli saldırıların olumsuz etkisini etkili bir şekilde azalttığını ve IDS'nin tespit doğruluğunu ve genel güvenilirliğini önemli ölçüde artırdığını göstermektedir.

**Anahtar Kelimeler:** Çekişmeli Saldırıları, Derin Öğrenme, Saldırı Tespit Sistemleri, Yeniden Yapılandırma Hatası, Güvenlik, Makine Öğrenimi Dayanıklılığı

## SUMMARY

Adversarial attacks pose a significant challenge to deep learning (DL) models, which are increasingly deployed in security-critical applications such as Intrusion Detection Systems (IDS) for the Internet of Things (IoT). These attacks also present a significant challenge to deep learning models, which are increasingly used in security-critical applications such as Intrusion Detection Systems (IDS) for the Internet of Things (IoT). This thesis examines adversarial attacks on network traffic and focuses on vulnerabilities, attack methods, and detection mechanisms. First, vulnerability surfaces in the machine learning lifecycle are identified, and examples and potential weaknesses in the literature are evaluated. Second, existing attacks are reviewed, classified, and the ROSIDS23 dataset is introduced. The ROSIDS23 dataset provides comprehensive traffic data for cybersecurity research in ROS-based robotic systems and constitutes a valuable resource for detailed analyses involving multiple attack types.

The proposed Reconstruction Error-based Adversarial Detection (READ) method uses four metrics together to detect adversarial attacks by combining existing metrics and a reconstruction error rate metric, achieving high detection rates. Results show that READ enhances IDS performance by significantly reducing adversarial effects and is particularly effective at low perturbation levels. FGSM attacks, due to their simplicity, are more easily detected compared to iterative attacks such as PGD and BIM. The proposed method detects adversarial attacks with a success rate of 92-100%. Experimental results show that integrating READ into IDS increases accuracy by up to 98% and significantly enhances system reliability.

**Keywords:** Adversarial Attacks, Deep Learning, Intrusion Detection Systems, Reconstruction Error, Security, Machine Learning Resilience

## ACKNOWLEDGEMENT

I would like to thank my family for their unwavering support throughout my graduate education and thesis process.

I would also like to thank my advisors, Prof. Dr. Ahmet Yazıcı and Assoc. Prof. Dr. İlker Özçelik, with whom I have worked for years, for their guidance, advice, and assistance.

This work was supported by the VALU3S project that has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876852. The JU receives support from the European Union's Horizon 2020 research and innovation program and Austria, Czech Republic, Germany, Ireland, Italy, Portugal, Spain, Sweden, and Turkey.

This work is supported by the Scientific and Technical Research Council of Turkey (TUBITAK), Contract No 120N800, project title: "Verification and Validation of Automated Systems' Safety and Security".

## LIST OF CONTENTS

|                                                                                                                         | <u>Page</u> |
|-------------------------------------------------------------------------------------------------------------------------|-------------|
| <b>ÖZET .....</b>                                                                                                       | <b>vi</b>   |
| <b>SUMMARY .....</b>                                                                                                    | <b>viii</b> |
| <b>ACKNOWLEDGEMENT .....</b>                                                                                            | <b>ix</b>   |
| <b>LIST OF CONTENTS .....</b>                                                                                           | <b>x</b>    |
| <b>LIST OF FIGURES.....</b>                                                                                             | <b>xii</b>  |
| <b>LIST OF TABLES.....</b>                                                                                              | <b>xiii</b> |
| <b>LIST OF ABBREVIATIONS AND SYMBOLS.....</b>                                                                           | <b>xiv</b>  |
| <b>1. INTRODUCTION AND PURPOSE .....</b>                                                                                | <b>1</b>    |
| <b>2. LITERATURE REVIEW .....</b>                                                                                       | <b>5</b>    |
| 2.1. Security Studies for ROS-based IoT Systems .....                                                                   | 5           |
| 2.2. Datasets that are Commonly used in Intrusion Detection Systems .....                                               | 7           |
| 2.2.1. CSE-CIC-IDS-2018 Dataset.....                                                                                    | 8           |
| 2.2.2. CIC-IDS 2017 Dataset.....                                                                                        | 8           |
| 2.2.3. KDDCup99 Dataset.....                                                                                            | 8           |
| 2.3. Adversarial Attacks Classification .....                                                                           | 9           |
| 2.4. Adversarial Attack Methods .....                                                                                   | 13          |
| 2.4.1. Fast Gradient Sign Method (FGSM) .....                                                                           | 13          |
| 2.4.2. Basic Iterative Method (BIM) .....                                                                               | 14          |
| 2.4.3. Projected Gradient Descent (PGD) Method .....                                                                    | 14          |
| 2.5. Adversarial Attacks Vulnerability Surface .....                                                                    | 15          |
| 2.6. Adversarial Attacks Detection .....                                                                                | 25          |
| 2.7. Metrics for Adversarial Attack Detection.....                                                                      | 27          |
| <b>3. MATERIALS AND METHODS .....</b>                                                                                   | <b>30</b>   |
| 3.1. ROSIDS23 Dataset .....                                                                                             | 30          |
| 3.2. Proposed Adversarial Attack Detection Approach.....                                                                | 38          |
| 3.2.1. Hypothesis .....                                                                                                 | 38          |
| 3.2.2. The Proposed System Architecture: Reconstruction Error-based Adversarial<br>Detection (READ) System Design ..... | 40          |
| <b>4. RESULTS AND DISCUSSION .....</b>                                                                                  | <b>44</b>   |
| 4.1. The Effect of an Adversarial Attack on Various Deep Learning Methods .....                                         | 44          |
| 4.2. The Effect of Various Adversarial Attacks on Deep Learning Methods.....                                            | 49          |

## LIST OF CONTENTS

|                                                                                                                        | <b><u>Page</u></b> |
|------------------------------------------------------------------------------------------------------------------------|--------------------|
| 4.3. The Comparative Analysis of Adversarial Attack on Various Datasets: ROSIDS23, CSE-CIC-IDS2018, and KDDCup99 ..... | 53                 |
| 4.4. The Effects of Adversarial Attacks on Data Features .....                                                         | 55                 |
| 4.5. The Results of Adversarial Attack Detection using READ .....                                                      | 56                 |
| 4.5.1. Experiment Design .....                                                                                         | 56                 |
| 4.5.2. The Experimental Results for READ .....                                                                         | 58                 |
| <b>5. CONCLUSION AND RECOMMENDATIONS.....</b>                                                                          | <b>68</b>          |
| <b>REFERENCES .....</b>                                                                                                | <b>71</b>          |

## LIST OF FIGURES

| <b><u>Figure</u></b>                                                                                                                                     | <b><u>Page</u></b> |
|----------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|
| 2.1. Runtime verification architecture .....                                                                                                             | 7                  |
| 2.2. The adversarial threat model attack surface .....                                                                                                   | 10                 |
| 2.3. Adversarial attack threat model classification.....                                                                                                 | 11                 |
| 2.4. Vulnerability surface for adversarial attacks .....                                                                                                 | 16                 |
| 2.5. Machine learning adversarial attack surfaces .....                                                                                                  | 17                 |
| 2.6. Machine learning vulnerability surface .....                                                                                                        | 19                 |
| 3.1. Laboratory environment setup.....                                                                                                                   | 30                 |
| 3.2. System Design: Network traffic star collection design .....                                                                                         | 31                 |
| 3.3. Dataset generation methodology from robotic devices to ROSIDS23 dataset.....                                                                        | 33                 |
| 3.4. Graphical representation of the class distribution of the dataset .....                                                                             | 34                 |
| 3.5. The repository ROSIDS23 dataset file structure .....                                                                                                | 34                 |
| 3.6. Deep learning architecture includes the last hidden layers and output layer.....                                                                    | 40                 |
| 3.7. (a) Reconstruction architecture, (b) Reconstruction error calculation .....                                                                         | 41                 |
| 4.1. Experiment design for the effect of adversarial attack on DL architecture; (a) No<br>attack, (b) FGSM attack .....                                  | 45                 |
| 4.3. Classic DL methods architecture: (a) Autoencoder, (b) Multi-Layer Perceptron, (c)<br>Deep Neural Networks, (d) Convolutional Neural networks.....   | 46                 |
| 4.3. Preprocessing flow of the dataset .....                                                                                                             | 47                 |
| 4.4. Effects of adversarial attacks on different DL networks experiment design.....                                                                      | 49                 |
| 4.5. Feature importance weight result's graph of CSE-CIC-IDS2018 dataset.....                                                                            | 56                 |
| 4.6. The steps of training for READ system .....                                                                                                         | 57                 |
| 4.7. Experiment design for the test of the READ system.....                                                                                              | 58                 |
| 4.8. CSE-CIC-IDS2018 dataset reconstruction error results in original samples and<br>adversarial samples .....                                           | 59                 |
| 4.9. CSE-CIC-IDS2018 dataset reconstruction error results in graphical visualization<br>under FGSM, PGD, and BIM with various perturbation amounts ..... | 60                 |
| 4.10. Comparative evaluation of ROC curves for FGSM attack.....                                                                                          | 62                 |
| 4.11. ROC curve of CSE-CIC-IDS2018 dataset under FGSM attack with 0.04<br>perturbation amount .....                                                      | 63                 |
| 4.12. Performance comparison of the proposed method and Tuna et. al. (2022) at<br>adversarial attack detection ROC-AUC scores .....                      | 65                 |
| 4.13. Comparative evaluation of adversarial attack detection performance on FGSM,<br>PGD, and BIM for the proposed method and Tuna et al. (2022).....    | 66                 |

## LIST OF TABLES

| <b><u>Table</u></b>                                                                                                                                                   | <b><u>Page</u></b> |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|
| 3.1. The ROSIDS23 dataset public repository file summary .....                                                                                                        | 35                 |
| 3.2. Public datasets comparisons with proposed ROSIDS23 dataset .....                                                                                                 | 37                 |
| 4.1. The effect of FGSM Attack on Deep Learning Models .....                                                                                                          | 48                 |
| 4.2. Experimental results of adversarial attacks on deep learning architectures enhanced<br>with autoencoders .....                                                   | 48                 |
| 4.3. Effects of adversarial attacks on different DL networks test results .....                                                                                       | 52                 |
| 4.4. CSE-CIC-IDS2018, KDDCup99, and ROSIDS23 datasets accuracy result under<br>normal and adversarial attacks with various perturbation amounts .....                 | 54                 |
| 4.5. The top five important features of the CSE-CIC-IDS2018 dataset are original samples<br>and adversarial samples .....                                             | 55                 |
| 4.6. MSE and MedAE of reconstruction errors for different datasets under FGSM, PGD,<br>and BIM with various perturbation amounts and normal samples .....             | 59                 |
| 4.7. Proposed detection system ROC-AUC scores across CSE-CIC-IDS-18 and<br>KDDCup99 datasets under FGSM, BIM, and PGD attacks at varying perturbation<br>levels ..... | 61                 |
| 4.8. Proposed detection system comparison for FGSM and PGD attacks .....                                                                                              | 67                 |

## LIST OF ABBREVIATIONS AND SYMBOLS

| <u>Symbols</u>             | <u>Descriptions</u>                                                                                                                                              |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $X_{adv}$                  | The modified input is designed to fool a machine-learning model.                                                                                                 |
| $X$                        | The original input that is normally classified correctly.                                                                                                        |
| $\epsilon$                 | A small number that limits the amount of modification applied.                                                                                                   |
| $\text{sign}(\cdot)$       | Function returning the sign of the gradient.                                                                                                                     |
| $\nabla_x J(X, y)$         | The gradient of the loss function with respect to the input data.                                                                                                |
| $J(X, y)$                  | The loss function indicates model performance.                                                                                                                   |
| $y$                        | True label of the data.                                                                                                                                          |
| $X_{adv,n}$                | The adversarial data at the $n^{\text{th}}$ iteration.                                                                                                           |
| $X_{adv,n+1}$              | The adversarial data at the $(n+1)^{\text{th}}$ iteration is the updated adversarial data.                                                                       |
| $\alpha$                   | The step size or learning rate that scales the update.                                                                                                           |
| $\text{sign}(\cdot)$       | The sign of the gradient of the loss function with respect to the adversarial data at the $n^{\text{th}}$ iteration. This indicates the direction of the update. |
| $\nabla_x J(X_{adv,n}, y)$ | The gradient of the loss function with respect to the adversarial data at the $n^{\text{th}}$ iteration.                                                         |
| $J(X_{adv,n}, y)$          | Loss function, indicating model performance.                                                                                                                     |
| $X_{adv,0}$                | The initial adversarial data equals the original data.                                                                                                           |
| $P$                        | A projection function that ensures the adversarial data stays within a valid range (often within a certain distance from the original data).                     |
| $X_i$                      | The original data point.                                                                                                                                         |
| $f(X)$                     | Dimensionality reduction step that transforms the original data $X$ into a reduced form $Z$ .                                                                    |
| $g(Z)$                     | The reconstructed data point.                                                                                                                                    |

## LIST OF ABBREVIATIONS AND SYMBOLS

| <u>Symbols</u>                 | <u>Descriptions</u>                                                                        |
|--------------------------------|--------------------------------------------------------------------------------------------|
| $  X_i - g(f(X_i))  ^2$        | The squared error between the original and reconstructed data points.                      |
| $  g(f(X_i) - X_i)  $          | The absolute error between the original and reconstructed data points.                     |
| $median()_{i=1}^n$             | The median is taken over all n data points.                                                |
| $\Delta$                       | Reconstruction error in the standard scenario, typically negligible.                       |
| $A(X)$                         | An adversarial attack was applied to the original data, yielding modified data $X_{adv}$ . |
| $R(F(X))$                      | The reconstructed version of the original data.                                            |
| $R(F(X_{adv}))$                | The reconstructed version of the adversarial data.                                         |
| $\epsilon_{RE}(R(F(X)))$       | The error rate for the original data reconstruction.                                       |
| $\epsilon_{RE}(R(F(X_{adv})))$ | The error rate for the reconstruction of adversarial sample.                               |
| $H$                            | The last hidden layers in the model architecture.                                          |
| $\delta$                       | The difference between $F(X)$ and $F(X_{adv})$ is due to adversarial attack effects.       |

| <u>Abbreviations</u> | <u>Explanation</u>                                                 |
|----------------------|--------------------------------------------------------------------|
| <b>ML</b>            | <b>M</b> achine <b>L</b> earning                                   |
| <b>DL</b>            | <b>D</b> eep <b>L</b> earning                                      |
| <b>NIDS</b>          | <b>N</b> etwork <b>I</b> ntrusion <b>D</b> etection <b>S</b> ystem |
| <b>IDS</b>           | <b>I</b> ntrusion <b>D</b> etection <b>S</b> ystem                 |
| <b>FGSM</b>          | <b>F</b> ast <b>G</b> radient <b>S</b> ign <b>M</b> ethod          |
| <b>PGD</b>           | <b>P</b> rojected <b>G</b> radient <b>D</b> escent                 |
| <b>BIM</b>           | <b>B</b> asic <b>I</b> terative <b>M</b> ethod                     |
| <b>C&amp;W</b>       | <b>C</b> arlini & <b>W</b> agner <b>A</b> ttack                    |

## LIST OF ABBREVIATIONS AND SYMBOLS

| <b><u>Abbreviations</u></b> | <b><u>Descriptions</u></b>                                                       |
|-----------------------------|----------------------------------------------------------------------------------|
| <b>IoT</b>                  | <b>I</b> nternet <b>o</b> f <b>T</b> hings                                       |
| <b>READ</b>                 | <b>R</b> econstruction <b>E</b> rror based <b>A</b> dversarial <b>D</b> etection |
| <b>AI</b>                   | <b>A</b> rtificial <b>I</b> ntelligence                                          |
| <b>ROS</b>                  | <b>R</b> obot <b>O</b> perating <b>S</b> ystem                                   |
| <b>DoS</b>                  | <b>D</b> enial <b>o</b> f <b>S</b> ervice                                        |
| <b>DDoS</b>                 | <b>D</b> istributed <b>D</b> enial <b>o</b> f <b>S</b> ervice                    |
| <b>RV</b>                   | <b>R</b> untime <b>V</b> erification                                             |
| <b>AE</b>                   | <b>A</b> utoencoder                                                              |
| <b>GAN</b>                  | <b>G</b> enerative <b>A</b> dversarial <b>N</b> etwork                           |
| <b>MSE</b>                  | <b>M</b> ean <b>S</b> quared <b>E</b> rror                                       |
| <b>MAE</b>                  | <b>M</b> ean <b>A</b> bsolute <b>E</b> rror                                      |
| <b>MedAE</b>                | <b>M</b> edian <b>A</b> bsolute <b>E</b> rror                                    |
| <b>RMSE</b>                 | <b>R</b> oot <b>M</b> ean <b>S</b> quared <b>E</b> rror                          |
| <b>CNN</b>                  | <b>C</b> onvolutional <b>N</b> eural <b>N</b> etwork                             |
| <b>DNN</b>                  | <b>D</b> eep <b>N</b> eural <b>N</b> etwork                                      |
| <b>MLP</b>                  | <b>M</b> ultilayer <b>P</b> erceptron                                            |
| <b>AEMLP</b>                | <b>A</b> utoencoder <b>M</b> ultilayer <b>P</b> erceptron                        |
| <b>AEDNN</b>                | <b>A</b> utoencoder <b>D</b> eep <b>N</b> eural <b>N</b> etwork                  |
| <b>AECNN</b>                | <b>A</b> utoencoder <b>C</b> onvolutional <b>N</b> eural <b>N</b> etwork         |

## 1. INTRODUCTION AND PURPOSE

Adversarial attacks, which exploit vulnerabilities in Artificial Intelligence (AI) models, present a growing threat as AI-based systems become integral across various industries. These attacks, where small, often imperceptible perturbations are introduced into data to mislead models, expose a critical flaw in the robustness of AI applications (Goodfellow et al., 2014a). As AI continues to advance fields like healthcare, finance, security, and manufacturing through its predictive and automation capabilities, the susceptibility of these models to adversarial manipulation has emerged as a key concern. Furthermore, AI is transforming businesses by reshaping operations and models, although adoption challenges persist (Lahlali et al., 2021). Ensuring the security and reliability of these AI-driven systems is essential to address potential vulnerabilities effectively.

Deep Learning (DL), a subset of AI known for its proficiency in learning complex patterns, has experienced particular success in diverse applications due to its effectiveness and speed. However, this growth in DL applications has coincided with an increase in adversarial attacks targeting these systems. While early studies primarily addressed adversarial threats in image processing (Ma et al., 2021), recent research has expanded the focus to network-based intrusion detection systems (NIDS) (Alotaibi & Rassam, 2023; K. He et al., 2023). Attack strategies like the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2014a), Projected Gradient Descent (PGD) (Kurakin et al., 2016), Basic Iterative Method (BIM) (Madry et al., 2017a), DeepFool (Moosavi-Dezfooli et al., 2016), JSMA (Carlini & Wagner, 2017) and Carlini & Wagner (C&W) (Carlini & Wagner, 2017) demonstrate how adversarial attacks can compromise one-dimensional data processing in critical security systems. These attacks can disrupt operations by either suppressing legitimate threat detection or by overwhelming systems with false alerts, thereby undermining the trustworthiness of such models.

The rise of adversarial attacks, especially within security-sensitive applications like network intrusion detection systems (NIDS), underscores the urgent need for enhanced defense mechanisms in DL models. The growing prevalence and sophistication of

adversarial attacks call for robust methods to detect and mitigate these threats, ensuring the reliability of DL systems in high-stakes environments. Recent studies highlight the substantial impact of these attacks on learning-based systems, underlining the urgent need for robust detection and mitigation techniques (P. T. Duy et al., 2021; Zhao et al., 2021).

This thesis proposes a novel approach for detecting adversarial attacks by utilizing the reconstruction error metric and the commonly used metrics in the literature. This approach aims to enhance the detection capabilities of adversarial attack detection systems by leveraging the distinct characteristics of adversarial samples compared to original samples. The fundamental idea behind the proposed method is based on examining both adversarial and original samples. Despite the small perturbations introduced by adversarial attacks, these perturbations lead to significant changes in the features that best represent the data. Furthermore, it is observed that the error values in reconstructed data serve as a distinguishing characteristic, effectively differentiating original samples from adversarial ones.

During the data reduction and reconstruction, original samples exhibit small reconstruction errors between their initial and reconstructed versions. However, adversarial samples demonstrate a greater value of distortion between their initial and final versions compared to the original samples. This key insight forms the basis of the proposed detection method. The proposed method provides several advantages. First, the use of reconstruction error as a detection metric exploits the inherent differences in how original and adversarial samples are reconstructed, providing a robust basis for distinguishing between them. Second, even small perturbations introduced by adversarial attacks cause noticeable changes in reconstruction error, making the method highly sensitive to adversarial perturbations. Third, the method can be applied across various DL models and architectures, making it versatile and adaptable to different application scenarios. The proposed approach effectively detects adversarial attacks. This novel method enhances the security and robustness of DL models, particularly in the context of Intrusion Detection Systems (IDS) for Internet of Things (IoT) systems. This thesis makes several significant contributions to the field of adversarial attack detection. The key contributions of this study are outlined below:

- A comprehensive identification and literature review of adversarial attack surfaces are conducted. This work maps out various vulnerabilities within DL models, providing a thorough understanding of how adversarial attacks exploit these weaknesses. This foundational research is essential for developing effective countermeasures and sets the stage for the subsequent innovations presented in this thesis.
- A novel adversarial attack detection system is proposed. This system is specifically designed to identify various types of adversarial attacks as well as different network attacks. By leveraging a combination of metrics in the literature and the use of reconstruction error, the proposed system demonstrates enhanced detection capabilities. The system's effectiveness has been rigorously tested against various adversarial attack scenarios, improving its robustness and reliability.
- A new dataset tailored for IoT middleware and a corresponding attack detection system is developed. This dataset captures the intricacies of data transformation through the layers of DL models, allowing for a more granular analysis of adversarial attacks. The attack detection system built using this dataset can identify anomalies and adversarial perturbations with greater precision, contributing to the overall security of IoT systems.

In this thesis introduces a novel approach to adversarial attack detection using the reconstruction error metric, which evaluates differences between original and adversarial samples during data reduction and reconstruction. Training a reconstruction model on original samples ensures optimized weights, resulting in significantly higher reconstruction errors for adversarial samples during testing, providing a clear detection feature. This method addresses a critical gap in the literature by effectively identifying small distortions caused by adversarial attacks, enhancing the robustness of deep learning models, particularly in IoT intrusion detection systems. These contributions offer new insights, methodologies, and practical solutions for securing DL applications.

The second chapter, Literature Review begins by examining security challenges in ROS-based IoT systems, followed by an overview of commonly used intrusion detection datasets, including CSE-CIC-IDS-2018, CIC-IDS 2017, and KDDCup99. It then delves into

adversarial attack classifications, methods such as FGSM, BIM, and PGD, the vulnerability surfaces of such attacks, detection techniques, and evaluation metrics. The third chapter, Materials and Methods, outlines the research methodology. In this chapter, the ROSIDS23 dataset is introduced and details the Reconstruction Error-based Adversarial Detection (READ) system is proposed. The fourth chapter, Results and Discussion, presents the outcomes of the experiments conducted. The effects of adversarial attacks on various deep learning models and datasets are analyzed, their influence on data features is examined, and the effectiveness of the READ system is demonstrated through experimental results. The final chapter, Conclusion and Recommendations summarizes the study's findings and their implications. It connects the experimental results back to the research objectives and highlights the contributions and novelties of the study, particularly the significance of the READ system in advancing cybersecurity for IoT systems. The chapter also identifies opportunities for future research, such as refining detection methodologies, exploring emerging adversarial attack strategies, and translating the findings into practical security solutions for IoT environments.

## 2. LITERATURE REVIEW

This literature review provides the background of the proposed approach. Initially, the key aspects of security studies for ROS-based IoT systems and intrusion detection datasets are given. Then, lifecycle management in machine learning and deep learning models is given to define the vulnerability surface. Finally, various dimensions of adversarial attacks, including their classification, methods, vulnerability surfaces, detection techniques, and evaluation metrics.

### 2.1. Security Studies for ROS-based IoT Systems

The adoption of robotic systems in industrial settings is increasing by an average of 30% annually, while their use in non-industrial applications is growing even faster, at 52% per year (Siriweera & Naruse, 2021). However, with the expansion of their usage into diverse areas, safety and security have become primary concerns (Plósz et al., 2017). The rise in digital accessibility has led to an increase in security threats targeting these systems. The Robot Operating System (ROS) plays a critical role in integrating components within robotic platforms, but it also introduces significant security challenges. Solutions like SROS (White et al., 2016) and ROS2 (Maruyama et al., 2016) have been developed to address certain security issues, primarily focusing on encrypting and signing messages. However, these solutions lack robust countermeasures against volumetric attacks.

Current ROS-based security studies focus on data encryption and policy creation to enhance security (Balsa-Comerón et al., 2018; Fernández Muro, 2018; Huang et al., 2014; Staffa et al., 2018). Despite these measures, vulnerabilities remain, particularly to attacks such as Denial-of-Service (DoS), blackhole, and man-in-the-middle attacks. Recent research has concentrated on DoS attacks against ROS systems (Degirmenci et al., n.d.; Rivera, S. et al., 2019). For instance, the ROSPlloit tool (Rivera, S. et al., 2019) provides a framework for analyzing security attacks on ROS platforms. Ensuring the security of robotic systems is crucial during both the design phase and runtime. Threat modeling plays a pivotal role throughout the system's lifecycle, starting from the initial design phase and requiring

continuous assessment during runtime (Tseng et al., 2019). However, time constraints often lead to the omission of certain threat scenarios, which must then be identified through runtime by Intrusion Detection Systems (IDS).

Anomaly detection is increasingly applied to enhance security in robotic systems (Narayanan and Bobba 2018), serving as a critical component of IDS to identify and mitigate potential threats during runtime. For instance, Narayanan and Bobba (2018) investigated anomaly detection in an industrial robotic arm within a ROS-based environment, identifying anomalous behaviors as repeated arm movements. Similarly, another study focused on detecting anomalies in sensor data from autonomous racing vehicles operating on ROS (Afzal, A. et al., 2022). Anomaly detection also plays a significant role in securing IoT systems (Chatterjee & Ahmed, 2022). Despite these advancements, existing anomaly detection approaches face challenges, such as dependency on the quality of training data and the difficulty in distinguishing between anomalies and correct-but-unexpected behaviors (Afzal, A. et al., 2022). Consequently, IDS require verification to ensure reliability.

Runtime verification (RV) offers a promising approach to validate IDS performance in complex and dynamic systems. By continuously monitoring system behavior during runtime, RV ensures that IDS remains effective in identifying and mitigating security threats, even in evolving scenarios. The study by Kirca et al. (2023) proposed a Runtime Verification (RV) architecture for an anomaly detection algorithm to enhance the security of robotic systems (see Figure 2.1). This architecture comprises three main components: a robotic platform, a verification system, and an attacker system. The robotic platform consists of a ROS Master, a controller, and either a simulated or physical robot. In this setup, the robot control node transmits the robotic arm's control locations to the physical robotic platform. Each component in the robotic system is equipped with a packet loss tracker to detect unexpected events, while the ROS Master is responsible for performing anomaly detection.

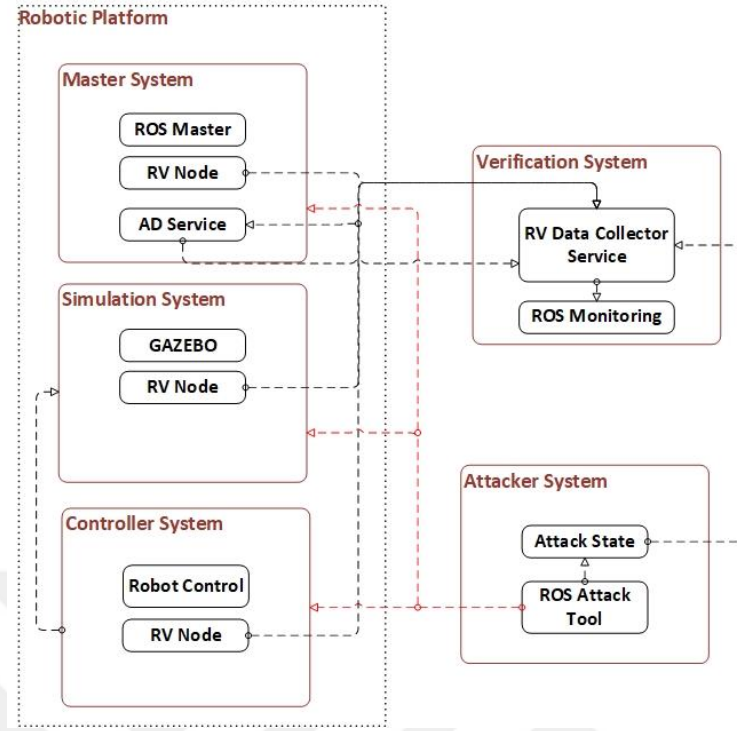


Figure 2.1. Runtime verification architecture

The verification system is key to the RV of the anomaly detection algorithm, comprising an RV data collector and ROSMonitoring. The RV data collector monitors the anomaly detection node for anomalies and tracks attack states in the attacker device, while ROSMonitoring evaluates verification using specifications or a Temporal Logic Oracle, logging alerts. The attacker system executes various attacks, including unauthorized publishing, DoS, and SYN flood attacks, targeting devices in the robotic platform, primarily the master device. This architecture and attack tool are also used to create the ROSIDS23 dataset (Değirmenci, Kırca, et al., 2023) used in this thesis. The details of this dataset are given in Section 3.1.

## 2.2. Datasets that are Commonly used in Intrusion Detection Systems

This section provides an overview of the datasets commonly utilized in developing and evaluating intrusion detection systems (IDS). For this thesis, three widely recognized datasets were selected: the CSE-CIC-IDS-2018 Dataset, the CICIDS2017 Dataset, and the KDDCup99 Dataset. The KDDCup99 dataset was chosen for literature comparisons. The CICIDS2017 and CSE-CIC-IDS-2018 datasets are chosen for attack diversity and realistic traffic.

### **2.2.1. CSE-CIC-IDS-2018 Dataset**

The CSE-CIC-IDS2018 dataset is a collection of real-world network traffic data containing both normal and malicious activities, collected in 2018 by the Canadian Institute for Cybersecurity (CIC) in collaboration with the Communications Security Establishment (CSE) (Sharafaldin et al., 2018). It comprises seven distinct attack scenarios: Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and network infiltration from inside. The attacking infrastructure includes 50 machines. The target organization comprises five departments, with a total of 420 machines and 30 servers. Feature extraction was performed individually for each machine on the network with CICFlowmeter-v3 (Lashkari, Seo, et al., 2017). The dataset contains 80 features.

### **2.2.2. CIC-IDS 2017 Dataset**

The dataset contains normal and malicious network traffic collected from a real environment by the Canadian Institute for Cybersecurity in 2017 (Sharafaldin et al., 2018). The dataset was gathered by monitoring network traffic and saved as pcap files, which were then converted into a labeled dataset format using the CICFlowMeter tool. It includes six different types of attacks: Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet, and DDoS. The dataset contains 84 features.

### **2.2.3. KDDCup99 Dataset**

The KDDCup 99 dataset (1999), originally derived from the DARPA 1998 dataset used in a network intrusion detection program at MIT's Lincoln Laboratory, is widely recognized for evaluating NIDS. This dataset was generated by converting network traffic data into 41 specific features per connection, making it suitable for ML applications in intrusion detection. Each connection is classified as either normal or as one of four primary attack types: DoS, User to Root (U2R), Remote to Local (R2L), and Probing. Both training and testing data are included, with the test set containing unique attack types absent from the training set, which enhances the dataset's complexity and real-world applicability. However, researchers have highlighted certain drawbacks, including a high percentage of duplicate

records, which can cause algorithms to favor common patterns and overlook rare but critical attacks. To address these issues, improved versions like NSL-KDD were developed, reducing bias and duplication, thus providing a more effective dataset for studies in network security and anomaly detection.

### **2.3. Adversarial Attacks Classification**

Adversarial attacks are intentional efforts by malicious actors to deceive and manipulate learning-based systems. According to the formal definition, these attacks aim to mislead such systems by altering input data (P. Duy et al., 2021). A seminal study by Goodfellow et al. (2014a) highlighted adversarial vulnerabilities within ML-based paradigms. This pioneering work unveiled that minor alterations to an image could significantly change its classification. Specifically, adding small perturbations to an image of a panda caused the system to classify it as a gibbon. As the application scope of learning-based approaches has expanded, adversarial attack research has diversified accordingly.

Another example is the compromised accuracy of stop sign detection in autonomous vehicles. By placing a sticker on a stop sign, attackers caused the detection system to misinterpret it (S.-T. Chen et al., 2019). Similarly, in medical image analysis, adversarial manipulations misled a medical system into classify a benign image as malignant (Ma et al., 2021). In automatic speech recognition systems, adversarial attacks have been used to generate targeted or untargeted adversarial samples that, while perceptually normal to human listeners, rendered the content unrecognizable to the detection system (Cisse et al., 2017). In natural language processing, adversarial manipulations have been employed to bypass spam detection methods to enable inappropriate content to appear normal and thus pass through spam filtering systems (H. Zhang et al., 2021). Additionally, adversarial attacks on graph algorithms have been utilized to distort the creditworthiness evaluation systems, making an unqualified loan applicant appear trustworthy (Ghamizi et al., 2020).

Although image-related examples play a crucial role in the academic discourse, the burgeoning employment of ML and DL in autonomous attack detection within network traffic has unveiled novel attack surfaces for adversarial exploits. IDSs aim to identify security threats before malicious traffic infiltrates the system, mitigating attacks on IoT-

based infrastructures (Li et al., 2018; Yao et al., 2018). The pursuit of reliable, robust, and efficient IDSs remains an open area of research. Since signature-based IDS approaches struggle to adapt to novel attacks, the adoption of learning-based algorithms has surged. However, these systems also exhibit vulnerabilities to adversarial attacks.

Threat modeling plays an essential role in constructing robust and secure systems. When constructing a threat model for adversarial attacks against learning-based systems, in this thesis, learning-based systems are identified as assets. These systems can be identified as potential targets when constructing a threat model for adversarial attacks. Despite inheriting all vulnerabilities associated with learning-based systems, it is essential to implement preventative measures within the threat model against attack vectors utilizing these known vulnerabilities.

The asset in this context is a learning-based system, representing an attack surface comprised of vulnerabilities susceptible to exploitation by specific attack vectors. However, the system remains vulnerable to attacks even if precautions against known vulnerabilities are implemented on the vulnerability surface. Without preventive measures, these systems remain susceptible to adversarial attacks such as targeted and untargeted attacks, confidence reduction, or misclassification. The threat model used in this study is depicted in Figure 2.2.



Figure 2.2. The adversarial threat model attack surface

In this study, the focus is placed on adversarial threat models concerning network traffic analysis and IDSs. The classification of adversarial attack vectors from various perspectives is presented in Figure 2.3. In this classification, adversarial attacks were

investigated across five distinct dimensions: Model deployment timeline, Attacker knowledge, Attacker specificity, Attack computation, and Attacker goal. The details are given below.

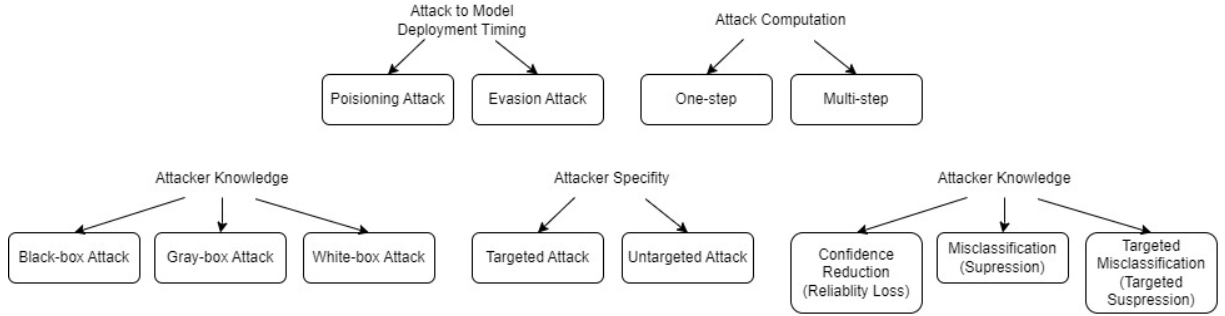


Figure 2.3. Adversarial attack threat model classification

*Attacks to Model Deployment Timing:* Based on the timing of adversarial sample insertion, attacks are classified as poisoning and evasion. In poisoning attacks, adversarial samples are generated to disrupt the learning phase. Poisoning attacks occur before the model's deployment, which could potentially lead to errors in the testing phase. Various stages of the learning process exhibit vulnerabilities to poisoning attacks. These attacks may occur at the data collection, labeling, and preprocessing stages. In the literature, the terms Trojan or backdoor attacks are often associated with poisoning attacks (Xiang et al., 2021). On the other hand, evasion attacks target the model after deployment. In evasion attacks, the learned model is assumed to be well-trained without being attacked or corrupted. During the testing phase, adversarial samples are introduced to the model. This type of attack could be utilized to decrease the model's robustness or fool the decision systems.

*Attacker Knowledge:* Attacker knowledge is segmented into three categories: black-box, gray-box, and white-box, based on the extent of the attacker's understanding of the target model, including aspects such as hyperparameters and the architectures of learning-based approaches. In black-box attacks, the attacker has no information about the system's model, hyperparameters, or learning algorithm. In this scenario, the attacker inputs data into the system and conducts the attack based on the received outputs. In a gray-box attack, the malicious user has limited information about the system. Additional knowledge about the system can be obtained through exploratory or gradient-based attacks. White-box attacks

require attackers to have all the necessary information about the system. They possess the model, understand the learning algorithm, and are aware of hyperparameters or features, etc.

*Attacker Specificity:* The categorization of attacks is divided into two groups based on their targets: targeted and untargeted. The main purpose of targeted attacks is to ensure that the generated adversarial samples yield the desired output. In contrast, untargeted attacks aim to induce wrong decisions by the model without targeting a specific outcome.

*Attack Computation:* The computation dimension of attacks categorizes the attack based on the complexity of its methodology. Adversarial attack methods can be classified into two categories depending on the number of iterations needed to achieve the objective: one-step and multi-step attacks. In one-step attacks, adversarial samples are generated once and then employed to manipulate the model. After creation, the adversarial sample remains unchanged, regardless of the model's output. The choice between executing a one-step attack before or after the model's deployment hinges on the specific goals of the attacker. On the other hand, multi-step attacks involve a more dynamic and iterative approach where attackers adjust their methods based on the model's responses. In this strategy, the attacker engages the model repeatedly, either until a successful misclassification occurs or until a predetermined maximum number of iterations is reached. This distinction underscores the variance in complexity and adaptability between one-step and multi-step adversarial attacks, each presenting unique challenges and considerations for both the attackers and the defenders.

*Attacker Goal:* The dimension of attacker's goal assesses the fundamental objective of the attack. An attacker's objectives vary based on the timeline—specifically, before and after the deployment of the model. Before deployment, the primary goal is often confidence reduction, which aims to undermine a model's reliability during its training phase. After deployment, the attacker's goals shift towards misclassification and targeted misclassification. Misclassification seeks to elevate the number of false positives and false negatives, thereby disrupting the model's accuracy without targeting specific outputs. Conversely, targeted misclassification is more precise in its approach, striving to achieve the intended result instead of inducing random misclassification of samples.

## 2.4. Adversarial Attack Methods

This section explores various adversarial attack methods designed to manipulate deep learning models. It focuses on the Fast Gradient Sign Method (FGSM), Basic Iterative Method (BIM), and Projected Gradient Descent (PGD) techniques. The details of their mechanisms and mathematical formulations are given.

### 2.4.1. Fast Gradient Sign Method (FGSM)

Goodfellow et. al. (2014a) introduced the FGSM as a strategy to deceive DL models, with its initial application primarily in image processing. Over time, FGSM has been adapted for broader domains such as network traffic analysis (Fu et al., 2021) and NLP (Fu et al., 2021). FGSM offers a one-step adversarial attack mechanism applicable to targeted and untargeted attacks. The method integrates a user-determined proportion ( $\epsilon$ ) of the utmost loss to the primary data, as denoted in Equation 2.1. Initially, it discerns the prediction, original, and either targeted or nearest decision boundaries then supplements the genuine data with amplified loss through a multiplier termed epsilon. FGSM attack sample generation function is:

$$X_{adv} = X + \epsilon * \text{sign}(\nabla_x J(X, y)) \quad (2.1)$$

where;

- $X_{adv}$ : The modified input is designed to fool a machine-learning model.
- $X$ : The original input that is normally classified correctly.
- $\epsilon$ : A small number that limits the amount of modification applied.
- $\text{sign}(\cdot)$ : Function returning the sign of the gradient.
- $\nabla_x J(X, y)$ : Gradient of the loss function with respect to the input data.
- $J(X, y)$ : The loss function indicates model performance.
- $y$ : True label of the data.

### 2.4.2. Basic Iterative Method (BIM)

Kurakin et al. (2016) proposed an iterative adversarial attack method known as the BIM. Using the BIM attack, adversarial samples can be generated for both targeted and untargeted purposes. The BIM formula detailed in Equation 2.2 is also known as Iterative-FGSM. This approach first computes the loss for the relevant targeted or untargeted classes. The step size or perturbation magnitude per iteration, denoted as alpha ( $\alpha$ ), along with the maximum number of iterations, constitutes its key hyperparameters. The adversarial sample generation process using BIM proceeds as follows:

$$X_{adv,0} = X, \quad X_{adv,n+1} = X_{adv,n} + \alpha * \text{sign}(\nabla_x J(X_{adv,n}, y)) \quad (2.2)$$

where;

- $X_{adv,n}$ : The adversarial data at the  $n^{\text{th}}$  iteration.
- $X_{adv,n+1}$ : The adversarial data at the  $(n+1)^{\text{th}}$  iteration, which is the updated adversarial data.
- $\alpha$ : The step size or learning rate that scales the update.
- $\text{sign}(\cdot)$ : The sign of the gradient of the loss function with respect to the adversarial data at the  $n^{\text{th}}$  iteration. This indicates the direction of the update.
- $\nabla_x J(X_{adv,n}, y)$ : Gradient of the loss function with respect to the adversarial data at the  $n^{\text{th}}$  iteration.
- $J(X_{adv,n}, y)$ : Loss function, indicating model performance.
- $y$ : True label of the data.

### 2.4.3. Projected Gradient Descent (PGD) Method

Madry et al. (2017a) introduced the PGD method to generate adversarial samples by solving a core maximization problem. The maximization is achieved by iteratively adjusting the input to maximize the loss function, which leads to the generation of adversarial samples. It stands compatible with both FGSM and PGD methods, whether targeted or untargeted. Unlike BIM, where iterations invariably commence from the same point, PGD starts arbitrarily from varied initial positions within the neighboring vicinity but identifies the optimal direction via the BIM approach. The PGD equation is outlined in Equation 2.3.

$$X_{adv,0} = X, \quad X_{adv,n+1} = P_e(X_{adv,n} + a * \text{sign}(D_x J(X_{adv,n}, y))) \quad (2.3)$$

where;

- $X_{adv,0}$ : The initial adversarial data equals the original data.
- $X_{adv,n+1}$ : The adversarial data after the  $(n+1)^{\text{th}}$  update.
- $P$ : A projection function that ensures the adversarial data stays within a valid range (often within a certain distance from the original data).
- $a$ : The learning rate or step size for the update.
- $\text{sign}(\cdot)$ : The sign of the gradient of the loss function with respect to the adversarial data at the  $n^{\text{th}}$  iteration, which directs the adversarial update.
- $\nabla_x J(X_{adv,n}, y)$ : Gradient of the loss function with respect to the adversarial data at the  $n^{\text{th}}$  iteration.
- $J(X_{adv,n}, y)$ : Loss function, indicating model performance.
- $y$ : True label of the data.

## 2.5. Adversarial Attacks Vulnerability Surface

The vulnerability surface represents the aggregate of vulnerabilities inherent to a system and the potential attack vectors that can be exploited against it (see Figure 2.4). The attack surface refers to the possible vulnerability points of a cyber system that can be compromised. Usually, the attack surface is defined in relation to an existing attack vector. However, in theory, the number of vulnerabilities is much greater than the practical attack surface. This thesis defines attack surfaces as known system vulnerabilities that can be exploited using existing or known attack vectors. In this thesis, the attack surface concept expanded and referred to known and possible vulnerabilities as the vulnerability surface.

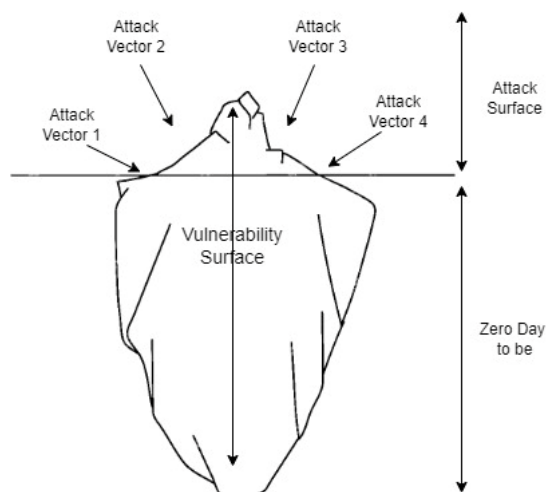


Figure 2.4. Vulnerability surface for adversarial attacks

Figure 2.5 illustrates a high-level overview of the machine learning (ML) lifecycle and its interaction with the model deployment timeline and associated attack surfaces. The life cycle of a learning-based approach comprises several key phases: data preparation, model training, model evaluation, parameter tuning, and retraining (Ashmore et al., 2021) (see Figure 2.5). Data preparation begins with data collection, where raw data is gathered from various sources such as databases, sensors, or online platforms. Following this, data preprocessing is performed—a critical step that significantly impacts the success of ML models (Huang et al., 2015; S. Zhang et al., 2003). Depending on the data type, preprocessing may involve handling missing values, scaling, feature selection, or encoding categorical variables. In some cases, feature extraction is also performed to create or identify relevant features that improve the model's learning ability. Once the data is preprocessed, the model is developed during the model training phase, where the algorithm learns patterns and relationships from the prepared data. After this, parameter tuning is conducted to optimize hyperparameters, such as learning rate or tree depth, to enhance model performance. The model is then evaluated in the model evaluation phase using metrics like accuracy, precision, and recall to measure its robustness and generalization to unseen data. The final stages include inference, where the trained model is deployed to make predictions on new data, and retraining, where the model is updated with fresh data or adjusted based on performance insights to maintain or improve effectiveness. The lifecycle is iterative, with feedback loops between training, tuning, and retraining, ensuring the model remains robust, accurate, and adaptive to new challenges.

The ML lifecycle can be split temporally into 'before deployment' and 'after deployment' phases (see Figure 2.5). This distinction is crucial as it denotes the varying privileges an adversary requires before executing an attack. 'Before deployment' phase, an attacker should have access to the data collection or training phases, and they target the model itself. On the other hand, in the “after deployment” phase, attackers do not need privileged access and aim to steal the learning model or cause misclassification by disturbing input data.

On the other hand, the attack surfaces include backdoor, poisoning, evasion, and model stealing (see Figure 2.5). This study maps existing attacks in the literature before or after the deployment phase, demonstrating how they exploit various vulnerabilities in the learning life cycle. These include backdoor and poisoning attacks for the 'before deployment' and evasion and model stealing attacks for the 'after deployment' phases. The backdoor poisoning attack perturbs data at the initial collection stage. Poisoning attacks aim to corrupt the dataset through adversarial manipulations before model training. Evasion attacks, like poisoning attacks, create adversarial samples to disturb/test model robustness during the testing phase. Model stealing attacks explore the model's boundaries by submitting a high number of queries to the system, enabling the replication of the model.

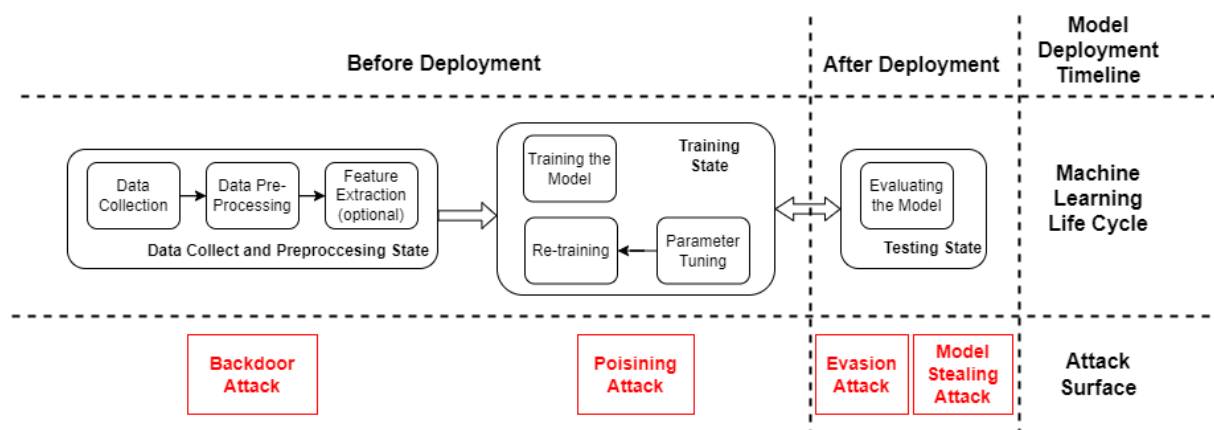


Figure 2.5. Machine learning adversarial attack surfaces

The scope of the attack surface is expanded to the vulnerability surface by examining known and potential vulnerabilities. Established vulnerabilities and potential weak points are considered, broadening the understanding of the attack surface into a comprehensive vulnerability surface (see Figure 2.6). The following sections aim to present a detailed discussion of this expanded perspective.



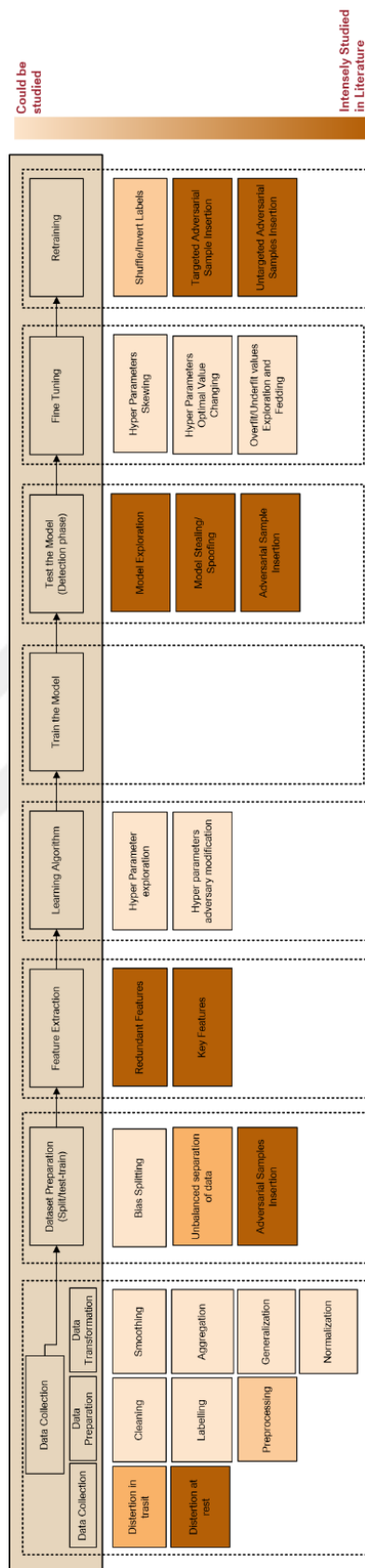


Figure 2.6. Machine learning vulnerability surface

Figure 2.6 represents a comprehensive flowchart for the "Machine Learning Vulnerability Surface." This flowchart details the various stages in the ML workflow where vulnerabilities might arise and specific processes at each stage that could be exploited. Each stage is associated with types of known or potential vulnerabilities, some of which have been extensively studied, as indicated by the color-coding. Areas that have been extensively researched and discussed in academic and professional literature, highlighted in dark brown and called "Intensely Studied in Literature". Lighter color areas that have not been studied may require more research.

Figure 2.6 identifies eight points with potential vulnerabilities in the learning life cycle. Within this vulnerability surface, the literature has primarily focused on dataset preparation and model testing stages. It was assumed that there was no vulnerability during the model training phase. The initial point pertains to data collection, encompassing the steps of gathering, preparing, and transforming data. In this ML life cycle, literature studies have focused on the data collection stage.

In the data collection phase, if an attacker successfully breaches the network, they can manipulate the data collected from diverse, high-traffic networks. Three vulnerability surfaces were identified on this point: data collection, data preparation, and data transformation. Data collection can have vulnerabilities that can emerge in both the "data in transit" and "data at rest" stages, making them susceptible to adversarial attacks. In the "data in transit" phase, when data is transferred between systems, attackers can intercept and manipulate data packets, introducing harmful inputs that may mislead the model's learning process. This interference compromises the model's performance and reliability. On the other hand, in the "data at rest" stage, where data is stored and not actively processed, attacks on storage systems can directly alter or corrupt data, weakening the model's accuracy and trustworthiness. The study targeted the data collection stage by jamming the data during the transmission (Sagduyu et al., 2019). In data preparation, there are vulnerability surfaces like data cleaning. First, in the data cleaning phase, there's a risk where the handling of null or missing values can be manipulated, creating an avenue for adversarial samples. Second, it's possible to introduce distortions when constructing a dataset, especially when integrating multiple data sources. This can lead to inconsistencies or biases. Third, the methods used for

dataset creation can be explored and exploited; for instance, some techniques might be more susceptible to generating adversarial samples than others. Fourth, even data considered "normal" or "standard" can be tweaked or stretched to the boundaries of what's acceptable, making them seem like outliers. This can distort the overall picture of the dataset. Lastly, there's the risk associated with label distortion. Incorrectly tagging or labeling data, either intentionally or unintentionally, can misrepresent normal samples as adversarial or the other way around. This misrepresentation can have significant implications for ML models trained on such data. Data transformation in the context of scaling methods involves the manipulation or conversion of data to facilitate analysis or model training, often enhancing the effectiveness and efficiency of large-scale computations. Adversarial attacks in this domain aim to disrupt or explore the vulnerabilities of these scaling methods, seeking to either degrade performance or uncover weaknesses in how data is processed and interpreted.

A second point is the dataset preparation phase. With learning-based methods, datasets are typically split into two primary subsets. In this phase, the dataset might be infiltrated by adversarial samples, as highlighted previously. When segregating data into test/train sets, the robustness of the learning-based model could be compromised if malicious samples are grouped together. This might cause the model to overfit, especially if training and testing datasets are too similar or identical. For classification or clustering, the decision boundaries may become ambiguous. Another prospective vulnerability at this juncture is related to outlier boundaries within conventional data, potentially corrupting the dataset. During the tagging, label assignments might become a focal point for attacks. Mislabeling can occur by altering existing labels or introducing shifts in the labeling window. As outlined, vulnerabilities during the dataset preparation phase can impact model integrity: Han et al. (Han et al., 2021) focus on feature set preparation, which should be done carefully. They use a Generative Adversarial Network (GAN) for features in the model's low-confidence region. Then, with particle swarm optimization, most similar features similar to adversarial features are selected, and they manipulate the malicious traffic automatically. Yan et al. (Yan et al., 2019) divided the feature set into four parts: basic features, content features, traffic features of a two-second time window, and traffic features of a hundred connections window. Changing the features intended to identify the DoS attack within each group created a similar sample of the malicious traffic.

A third vulnerable point is feature extraction. If the tool used for dataset creation is identified, a new dataset populated with adversarial samples might be formulated. Alternatively, these adversarial samples might be incorporated during the subsequent feature extraction. This is a pivotal step for potential model identification. Methods used for compression techniques like Autoencoders (AE) also create vulnerability surfaces in situations like adding deceptive noise. During feature extraction, the extraction algorithm itself can be vulnerable to detection-based attacks. Both important and non-important features might become targets. A strategic poisoning attack might compromise the dataset by focusing solely on important or non-important features. The research landscape concerning the vulnerability and resilience of ML and DL methods is vast and evolving. Kuppa and Le-Khac (Kuppa & Le-Khac, 2021) have investigated ML algorithms for adversarial attacks. The intricacies of these attacks, such as evasion, poisoning, inference, and exploratory attacks, are evaluated, and their complexity, accuracy, and performance are analyzed. The feature extraction phase is critical for the entire learning process. Attack or redundant features that can be used as a target from malicious users can be identified during this phase. For instance, Bao et. al. (2018) analyzed effects of feature selection on attack impact and explored to balance between minimizing the feature space and a high accuracy model. Ahmed and Kim (2017) used an information gain algorithm to find most k-discriminant features and deceived SVM algorithm-based detection methods with minimum feature change. Hartl et al. (2020) examined the effect of the feature sensitivity concept on the robustness of RNNs. Authors showed that sensitive features are more vulnerable to attack.

During the testing phase, the model becomes susceptible to adversarial samples. These adversarial samples could have objectives such as model theft, deceiving detection systems, or revealing model outputs. Model deployment is one of the most studied phases. These studies mainly focus on creating adversarial samples, stealing models, and discovering model output. By applying high queries to the model, its operation can be facilitated. To reduce the model's reliability, the outputs of the model can be manipulated to be perceived as normal during an attack and as under attack when there is none. Shi et al. (2019) proposed a penetration testing framework that includes adversarial attacks to test the security. They highlight the Deep Neural Network (DNN) based IDS and ML methods. Their

comprehensive framework consists of exploratory and evasion, and causative attacks. Various known adversarial attacks have been studied in the literature to deceive DL models. Some of the prominent examples of these attacks are the FGSM (Goodfellow et al., 2014b), DeepFool (Moosavi-Dezfooli et al., 2016), JSMA (Carlini & Wagner, 2017), PGD (Madry et al., 2017b), BIM (Madry et al., 2017b), and C&W (Carlini & Wagner, 2017). Given the ubiquity of these attack types, they are used for many research purposes. The study focuses on the attack effects on different network traffic datasets (Sauka et al., 2022), and they are also used for creating adversarial samples for adversarial training (Abou Khamis & Matrawy, 2020). Moreover, Warzynski and Kolaczek (2018) created adversarial samples with an FGSM attack and defeated the neural network-based anomaly detection method. Building on this, Piplai, Chukkapalli, and Joshi (2020) applied the FGSM adversarial attack to GAN-based IDSs. Similarly, Ayub et. al. (2020) performed a JSMA adversarial white-box evasion attack on Multi-layer Perceptron (MLP) based IDSs. In the literature, GAN-based adversarial attacks are targeting the testing phase. Sarkar, Vinod, and Yerima (2020) leverage GAN to create an adversarial attack. It's observed that GAN-based adversarial ML attacks can defeat the black-box IDSs (Chaitou et al., 2021; Qayyum et al., 2020). Wasserstein GAN (WGAN), known for its convergence in training, not only decreases the detection rate in black-box IDSs but also successfully deceives the IDS (P. Duy et al., 2021). There are some specialized adaptations. Yan et al. (2019) delved into WGAN as DoS-WGAN to defeat the gradient-based classification algorithm. Zhao et al. (2021) gave a twist to the Wasserstein GAN by adding a feedback system named attack GAN to deceive black-box IDS. Sun et al. (2017) adopted a distinct approach by creating mimicking attacks using Least Squares Generative Adversarial Networks (LSGAN) for adversarial attacks. Their methodology succeeded in deceiving the enhanced Random Forest algorithm that detects Distributed Denial of Service (DDoS) and Flash Crowds attacks. To ensure robustness against attacks, adversarial samples (also known as adversarial attacks) have been tailor-made to mislead the model, especially during its testing phase. A notable method in this arena is the use of AE. In this approach, the AE-based created model learns the distribution of normal features and, using this knowledge crafts the adversarial network traffic flow to bypass IDSs (J. Chen et al., 2021). Building on this, Apruzzese, Colajanni, and Marchetti (2019) generate adversarial samples with a twist: rather than altering the entire feature set, they handpick a few features, perturb them with slight random values, and thereby diminish the performance of the IDS.

The continuous learning-based approaches have a retraining phase. This phase has vulnerabilities against three points. In the retraining stage, attackers can aim to disrupt the decision boundaries of the model by changing the labels incorrectly. The stability of the model can be targeted by giving adversarial samples of the decision limits. This phase also vulnerable against poisoning attacks that adversarial samples fed to the models. Shi et al. (2019) causative attacks can be applied at retraining stage in different ways; falsified label directly through the original samples or attacker can be falsified the data only for those data that have high classification rates. They propose a penetration framework for analyzing the error probability under this attack. Chauhan and Heydari (2020) craft polymorphic DDoS attack samples to mutate attack profiles. They employed GAN to generate synthetic DDoS samples, and as time progressed, by integrating these into the IDSs, they could destabilize the detection system model. In a novel approach, the DL model was stolen with augmented training data, which was synthetically spawned using a SMOTE-based specialized oversampling algorithm and subsequently trained with the DNN model. Furthermore, DNN-based network traffic prediction models, which play a crucial role in contemporary networks, are not exempt from adversarial attacks (M. Wang et al., 2020). Their methodology was intricate, leveraging Deep Reinforcement Learning and the GAN model to supervise and interact with software-defined IP over elastic optical networks. When adversarial attacks were injected, they resulted in disruptions like light path congestions, bandwidth allocations, and network reconfigurations.

When examining the ML lifecycle, the areas of learning algorithms and fine-tuning can also create a surface of vulnerability, but these points have not been studied in the literature. For these attacks to be evaluated, an insider attacker must have access to the system. There are few studies on these areas.

In the training phase, before model formulation, the hyperparameters of the learning algorithm might be compromised or detected. The methods used to optimize hyperparameters, such as grid search, random search, bayesian optimization, genetic algorithm, particle swarm optimization, etc., can also create a vulnerability surface. In cases where these algorithms are misled, the model's ability to produce accurate results can be compromised.

After testing, the hyperparameters might also be recalibrated in the fine-tuning phase to enhance model resilience. At this stage, the vulnerabilities of fine-tuning methods also create a vulnerability surface for this phase. During this fine-tuning phase, the range values for testing hyperparameters can become attack targets. Those hyperparameter values that optimize the model's accuracy might be manipulated. The model could then be flooded with hyperparameter values, driving it to either overfit or underfit. As a result, the progression of the model towards an enhanced version might be hindered.

## 2.6. Adversarial Attacks Detection

Adversarial attack detection refers to the methodologies and techniques employed to identify and mitigate the impact of adversarial samples that can deceive learning-based approaches, particularly in security-critical applications such as IDS (Dhinakaran & Anto, 2023; H. Xu et al., 2020). Prominent studies in this field have explored various approaches, including the use of GANs for generating adversarial instances, as well as the development of robust detection mechanisms that utilize deep learning and statistical methods to enhance the resilience of IDS against such attacks (Droos, 2023; Saha et al., 2019).

Adversarial attack detection methods are categorized in the literature under four main headings: secondary classifier, projection-based detection, statistics-based detection, and mutation-based detection (K. He et al., 2023). The secondary classifier method uses an additional classification algorithm to classify adversarial samples. Adversarial and original examples are classified as binary and trained with a model that classifies them as either adversarial or binary (Gong & Wang, 2023). In projection-based detection, attack detection is done by converting to a lower-dimensional space vector. In the study using PCA, adversarial images were found to have higher coefficients. Statistical-based detection methods use statistical tests to detect differences in distributions between adversarial and clean examples. Gao et al. (2020) used the Maximum Mean Discrepancy test, and Feinman et al. (2017a) modeled output layers as Gaussian Mixtures. These methods also face limitations against more advanced attacks. Mutation-based detection randomly alters decision boundaries and measures sensitivity to these changes. Adversarial samples, close

to decision boundaries, show inconsistent classifications. Wang et al. (2022) concentrated on understanding the nature of adversarial attacks. Wang et al.'s study shows that adversarial attacks tend to be close to both the original class and decision boundary. Wang et al. (2022) study leveraged the proximity of adversarial samples (AEs) to their original manifold and decision boundary for accurate AE detection.

Another method for detecting adversarial attacks is using hybrid classification methods to determine if samples are adversarial or original. Pawlicki et al. (2020) propose a detection approach based on adversarial machine learning with five pattern recognition algorithms. Uelwer et al. (2021) analyze the detection performance of various attacks, including CW, BIM, FGSM, boundary attack, and their combinations using class scores.

A distinct approach, the EsPADA multi-model framework, uses feature extraction via N-gram, commonly used in natural language processing problems (Vidal et al., 2020). The features are stored in Counting Bloom Filters for further usage. Deep learning models' uncertainty metrics produce different results under adversarial attacks compared to original examples (Feinman et al., 2017b). Tuna et al. (2022) also investigated gradient-based attacks, employing five different metrics, including epistemic uncertainty, aleatoric uncertainty, scibilic uncertainty, entropy, and closeness score, to detect adversarial attack features. Another approach is the feature-squeezing method for detecting adversarial attacks on deep neural networks (W. Xu et al., 2017). Ye and Liu (2022) explore using feature autoencoders to detect adversarial samples. Peng et al. (2020) leverage a bidirectional generative adversarial network to learn the distribution of normal data and identify adversarial samples, thereby improving the robustness and accuracy of network IDS in adversarial environments.

These studies above highlight the importance of developing effective adversarial detection methods for network intrusion detection systems. Despite pioneer studies in literature, detecting adversarial attacks is still an open research area. It requires investigation of various methods including data reconstruction, dimensionality reduction, uncertainty metrics and adversarial attack methods.

## 2.7. Metrics for Adversarial Attack Detection

Various methods are used to detect adversarial attacks, including defensive distillation, statistical methods, GAN-based detection, uncertainty metrics, and closeness metrics. The proposed thesis evaluated uncertainty metrics such as aleatoric, epistemic, and entropy. Additionally, in this thesis, reconstruction error metrics are applied as part of the detection process.

In machine learning, uncertainties have been evaluated using statistical methods (Der Kiureghian & Ditlevsen, 2009; Hora, 1996). Aleatoric uncertainty is based on the randomness of the training data (Hüllermeier & Waegeman, 2021). Tossing a coin can be given as an example. There is a possibility that the coin will land heads or tails, but the certainty is not sure. Or it is the unpredictability of natural events such as the arrival of a hurricane (Bjarnadottir et al., 2019). Aleatoric uncertainty is a measurable metric that cannot be manipulated (W. Zhang et al., 2024). High aleatoric uncertainty means that the model cannot learn stably, and the noise or variability in the data. Therefore, an adversarial sample might purposefully change features in the testing. This results to high aleatoric uncertainty. Addressing aleatoric uncertainty is crucial for improving the robustness and reliability of machine learning models, particularly in tasks where the data exhibits significant variability and unpredictability (Kendall & Gal, 2017).

On the other hand, epistemic uncertainty is based on the model's lack of knowledge (Hüllermeier & Waegeman, 2021). Compared to other uncertainty metrics, this metric can be overcome with more information. In the context of machine learning and predictive modeling, epistemic uncertainty can arise when the training data is not representative of the true distribution of the data, leading to a lack of generalization in the model's predictions (Kendall & Gal, 2017). In adversarial attacks, it has been observed that small perturbations in the data cause differences in both of these metrics (Bjarnadottir et al., 2019; Tuna et al., 2022). Epistemic uncertainty is particularly relevant because adversarial samples often fall outside the model's learned distribution. High epistemic uncertainty suggests the model is less certain about its predictions due to lack of training on similar data, making it an indicator of potentially adversarial samples.

Another approach for detecting adversarial attacks is entropy as a metric. It measures how decisively data is used when making decisions during the testing phase. A high entropy value indicates a higher level of uncertainty and randomness; this suggests that the data may not provide clear distinctions for decision-making. Conversely, a low entropy value implies a more stable and informative dataset that can contribute to more robust and reliable decision-making processes. Understanding the entropy of data is necessary to evaluate the quality and reliability of information used in decision-making processes, especially in machine learning and statistical modeling (Shannon, 1948). High entropy in predictions indicates low confidence and uncertainty, which may be due to an adversarial sample attempt. One pixel adversarial attack can be detected using the entropy metric by analyzing the entropy level as low or high (Su et al., 2019). Adversarial attacks like PGD, CW, Spatial, and Hopskip for images can be detected by analyzing entropy-changing image processing methods (Pedraza et al., 2020). In both image and attack detection systems, the entropy value is used for detection along with many metrics.

The reconstruction error measures the difference between the original data and its reconstruction. There are several methods to calculate reconstruction error, like mean square error (MSE), root mean squared error (RMSE), mean absolute error (MAE), and median absolute error (MedAE) (Cort J. Willmott & Kenji Matsuura, 2005; Sánchez-González et al., 2020; W. Wang & Lu, 2018). The reconstruction success can be calculated using the error calculation formulas. A low error rate indicates the correctness of the selected reproduction method. The reconstruction error formula of MSE is defined in Equation 2.4:

$$MSE = \sum_{i=1}^n ||X_i - g(f(X_i))||^2 \quad (2.4)$$

where:

- $n$  is the number of data points,
- $X_i$  is the original data point,
- $f(d_i)$  is the reduced representation of the data point,
- $g(f(X_i))$  is the reconstructed data point,
- $||X_i - g(f(X_i))||^2$  is the squared error between the original and reconstructed data points.

Another reconstruction error metric MedAE mathematical formula is defined in Equation 2.5:

$$MedAE = median(||g(f(X_i) - X_i)||)_{i=1}^n \quad (2.5)$$

where:

- $||g(f(X_i) - X_i)||$  is the absolute error between the original and reconstructed data points,
- The median is taken over all  $n$  data points.

These metrics provide insights into how closely the reconstructed data approximates the original data. In practical applications, choosing the appropriate metric depends on the specific requirements and characteristics of the data and the reconstruction method. For instance, MSE is sensitive to outliers, whereas MedAE provides a robust measure that is less affected by extreme values.

### 3. MATERIALS AND METHODS

This section describes the materials and methods used in this thesis. Firstly, the generated dataset, including the experimental setup, is explained. Then, the proposed adversarial attack detection method is given.

#### 3.1. ROSIDS23 Dataset

ROSIDS23 is a dataset designed for intrusion detection studies in ROS-based environments. The dataset was acquired from an experimental system employing ROS, a widely used middleware for robotic IoT applications. To collect the dataset, IFARLab-DIH laboratory environment is used with runtime verification architecture (Kirca et al., 2023) given in Figure 2.1 (see section 2.1). The IFARLab-DIH includes various TRL5 test platforms for industrial applications (see Figure 3.1).



Figure 3.1. Laboratory environment setup

The network architecture for the dataset is given in Figure 3.2. It comprises essential ROS components such as the Controller, robotic arm devices, and Master. Additionally, a Logger device was utilized to capture and record traffic, and an Attacker device was included to simulate various security threats. For attack data collection, an attacker device, operating on Ubuntu 20.04 LTS, was incorporated into the framework. Both benign and malicious data were recorded via the Logger unit.

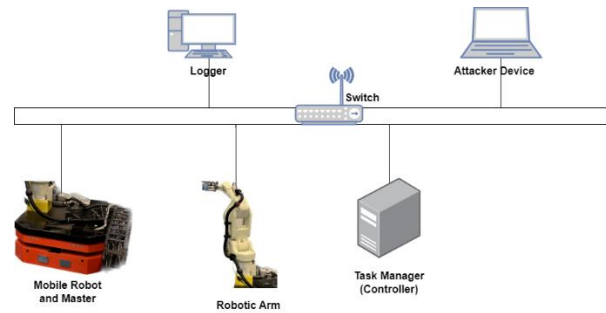


Figure 3.2. System Design: Network traffic star collection design

During the data collection process, tcpdump was used to capture network packets, and CICFlowMeter was utilized to extract features from these packets. Tcpdump is a tool for real-time packet capture and network analysis (Goyal & Goyal, 2017), while CICFlowMeter, an open-source traffic analyzer, extracts network flow characteristics relevant to ROS middleware traffic (Lashkari, Zang, et al., 2017). All packets passing through the network switch were mirrored to the Logger device throughout the process. The Logger could monitor all device communications within the ROS system, even without direct communication targeting it. The standard protocol in ROS for communication between nodes typically utilizes either TCP or UDP. The Master device, however, primarily utilizes the HTTP protocol to enable data exchange in XML format. Essential functions, including starting and stopping nodes, establishing and ending inter-node communication, and retrieving information about nodes or topics, depend on this protocol. While ROS permits the use of alternative protocols through customization, this study is limited to the default configurations of ROS. The dataset mainly focused on four distinct security attacks such as DoS, Unauthorized Publish, Unauthorized Subscribe, and Subscriber Flood. These are potential threats in robotic environments. These attacks are detailed below.

- **DoS Attack:** A DoS attack occurs when network resources are exploited to deny legitimate users access to a system. Attackers may achieve this by altering the configuration files of compromised resources, causing physical damage to network components, or excessively consuming system resources (Özçelik & Brooks, 2020).
- **Unauthorized Publish Attack:** This attack involves injecting malicious data into the ROS environment. By targeting specific, critical areas, an attacker can seize control of a robot, cause malfunctions, or disrupt intended operations (Değirmenci, Kirca, et

al., 2023). The attack mimics legitimate ROS communications using ROS libraries. As the number of attackers increases, critical nodes responsible for controlling the robot may be misled. Additionally, this attack's high volume of data can produce a DoS-like effect.

- **Unauthorized Subscribe Attack:** In this attack, the attacker gains access to all ROS communications, including sensitive sensor data. This data can be exploited for malicious purposes, such as stealing personal or confidential company information. ROS's lack of built-in authentication and authorization mechanisms makes this attack easy to execute. Simply installing ROS on the attacker's device is sufficient to carry out the attack, without requiring external scripts. However, in the dataset used for this study, a method was developed to realistically simulate this attack by listening to all topics published in ROS simultaneously. If the system includes controller software, extracted data can be manipulated to deliberately cause operational accidents (Değirmenci, Kirca, et al., 2023).
- **Subscriber Flood Attack:** This attack is a form of DoS attack where the attacker uses multiple fake identities to repeatedly subscribe to a ROS-published topic. The robotic system attempts to handle each incoming request, consuming its bandwidth and resources. The attacker only communicates with the ROS Master to appear as a legitimate ROS node and avoids sending additional message packets to conserve their resources. However, the excessive demand depletes the bandwidth and exhausts the resources of both the ROS Master and the target publisher device (Kirca et al., 2023).

To carry out these attacks, both ROS-specific tools like ROSAttackTool (Kirca et al., 2022) and general network attack tools were utilized. The process of creating the dataset starts with monitoring ROS communication between robotic system devices during normal traffic conditions. At the same time, the Attacker system launches its attacks. Logger devices intercept network packets through the switch's mirror port and capture ROS traffic, storing the data using the tcpdump tool. The generated .pcap files provide a comprehensive record of all network communication between ROS devices. The workflow for dataset generation is depicted in Figure 3.3. Both benign background traffic and background traffic with active attacks are included in the recordings. Following data capture, the CICFlowMeter tool (Lashkari, Zang, et al., 2017) is used to extract traffic flow features from the .pcap files. This

process generates over 80 traffic features, which are then converted into samples and stored as .csv files.

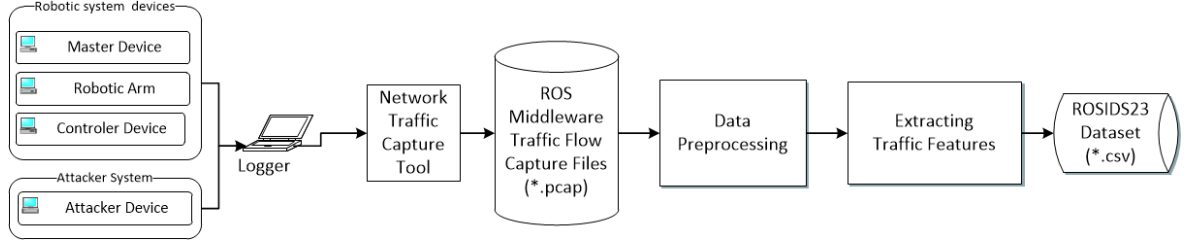


Figure 3.3. Dataset generation methodology from robotic devices to ROSIDS23 dataset

The benign traffic and attack scenarios are saved in separate .pcap files. The benign traffic, recorded first, serves as the baseline and is referred to as the benign profile. Each attack scenario is executed individually, with features extracted and labeled separately for each scenario.

ROSIDS23 is a multi-class dataset in which each row contains a timestamp, 83 features, and a label field. The label field specifies the type of traffic, with possible values including benign, DoS, unauthorized publish, unauthorized subscribe, and subscriber flood. The dataset includes a large number of records across various categories. The largest among these is the benign category, representing non-malicious instances, comprising 62,511 records. Denial of Service (DoS) attacks account for 31,000 records, closely followed by subscriber flood attacks with 30,064 records. Two additional categories address specific security concerns: unauthorized subscribe and unauthorized publish. The unauthorized subscribe category consists of 5,289 records, representing cases where unauthorized entities subscribe to or intercept network communications. In contrast, the unauthorized publish category includes 7,817 records, reflecting instances where unauthorized entities publish or inject data. Figure 3.4 shows the distribution of these record types, providing a clear overview of the dataset's composition.

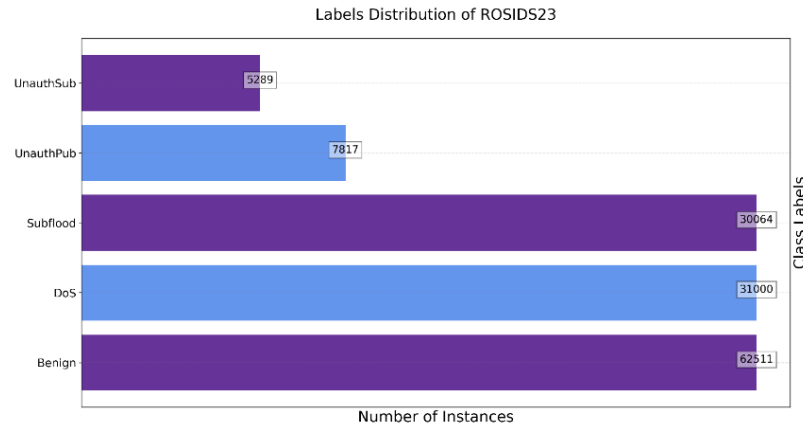


Figure 3.4. Graphical representation of the class distribution of the dataset

The dataset includes both raw and processed files. For each network traffic logging session involving an attack, the start times of the relevant attacks were recorded and saved in the \*\_attacktimes.txt file, which is located within each corresponding Raw folder. These files, along with the dataset, are published in the Zenodo repository (Degirmenci, 2023). The file structure of the repository is illustrated in Figure 3.5.

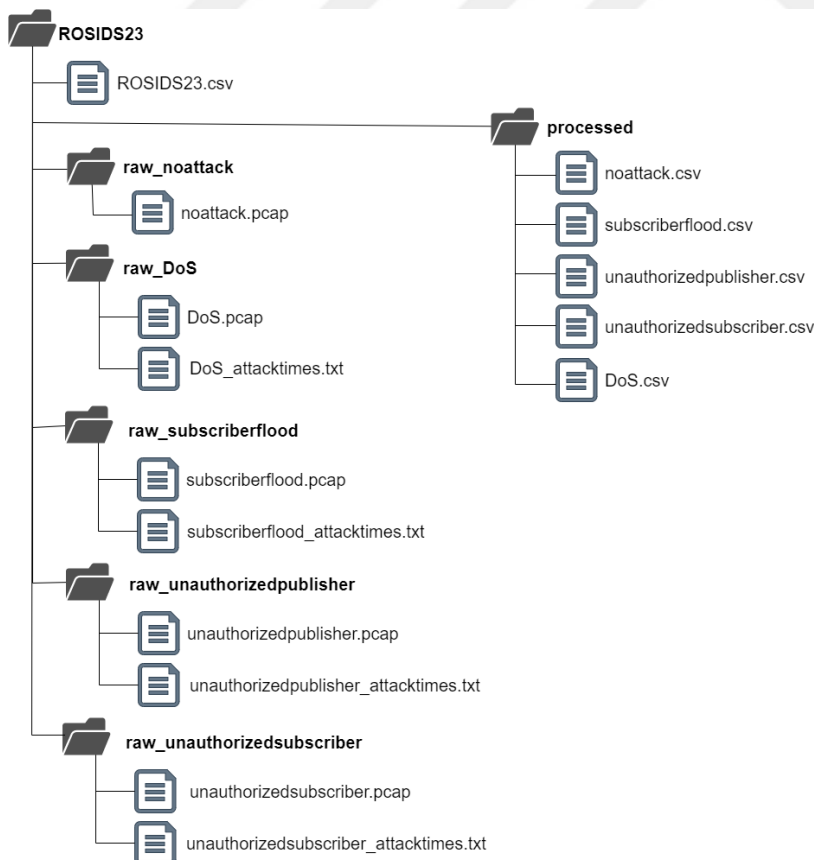


Figure 3.5. The repository ROSIDS23 dataset file structure

Table 3.1 provides detailed information about the structure of the ROSIDS23 dataset, including the file names, paths, descriptions, and sizes of the dataset components.

Table 3.1. The ROSIDS23 dataset public repository file summary

| File Path                             | File Name                       | Description                                                                                                                                                                               | Size     |
|---------------------------------------|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|
| ./                                    | ROSIDS23.csv                    | This file contains a feature-based multi-class labeled dataset.                                                                                                                           | 72.5 MB  |
| ./Raw_<br>noattack/                   | noattack.pcap                   | This file contains benign network traffic captured over 4 hours with no security attacks. It provides a reference for benign communication between ROS components.                        | 3.2 GB   |
| ./Raw_<br>Subscriber<br>flood/        | subscriberflood.pcap            | This file captures network traffic over 60 minutes. A Subscriber attack was evaluated at 1-minute time intervals for a total of 9 minutes (a ROS-specific security attack).               | 3.2 GB   |
| ./Raw_<br>unauthorize<br>dsubscriber/ | unauthorized<br>subscriber.pcap | This file captures network traffic over 60 minutes. An Unauthorized Subscriber attack was evaluated at 1-minute time intervals for a total of 9 minutes (a ROS-specific security attack). | 1.9 GB   |
| ./Raw_<br>unauthorize<br>dpublisher/  | unauthorized<br>publisher.pcap  | This file captures network traffic over 60 minutes. An Unauthorized Publisher attack was evaluated at 1-minute time intervals for a total of 9 minutes (a ROS-specific security attack).  | 1.5 GB   |
| ./Raw_DoS/                            | DoS.pcap                        | This file captures network traffic over 60 minutes. A DoS attack was evaluated at 1-minute time intervals for a total of 9 minutes.                                                       | 4.3 GB   |
| ./Processed/                          | benign.csv                      | This file contains only benign traffic features extracted from benign.pcap file. (Not-labelled)                                                                                           | 8.7 MB   |
| ./Processed/                          | DoS.csv                         | This file contains benign and DoS attack traffic features extracted from the DoS.pcap file. (Not-labelled)                                                                                | 318.4 MB |
| ./Processed/                          | unauthorized<br>subscriber.csv  | This file contains benign and unauthorized subscriber attack traffic features extracted from the unauthorizedsubscriber.pcap file. (Not-labelled)                                         | 6.3 MB   |

| File Path    | File Name                  | Description                                                                                                                                     | Size   |
|--------------|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| ./Processed/ | unauthorized publisher.csv | This file contains benign and unauthorized publisher attack traffic features extracted from the unauthorizedpublisher.pcap file. (Not-labelled) | 7.1 MB |
| ./Processed/ | subscriberflood.csv        | This file contains benign and subscriber flood attack traffic features extracted from the subscriberflood.pcap file. (Not-labelled)             | 236 MB |

The dataset consists of 84 features, with the final column serving as the label. These features are generated using the CICFlowMeter tool (Lashkari, Zang, et al., 2017). Below is a breakdown of the feature structure:

- Basic Network Connection Features (1st to 7th): These include Flow ID, Source IP, Source Port, Destination IP, Destination Port, Protocol, and Timestamp.
- Packet-Based Features (8th to 20th): These describe packet-level details such as total packets and their lengths (forward and backward directions), as well as the maximum, minimum, mean, and standard deviation of packet lengths in both directions.
- Statistical Flow Features (21st to 36th): These provide statistical insights into each network flow, including flow rate in bytes and packets per second, and the mean, standard deviation, maximum, and minimum of inter-arrival times for packets in both directions.
- Flag Counts and Header Length Information (37th to 57th): These features include counts of TCP flags (e.g., PSH, URG, FIN, SYN, RST, ACK, CWE, ECE), header lengths in both directions, and the number of packets per second in each direction.
- Content-Based Features (58th to 66th): These describe content-level statistics such as the ratio of download to upload traffic, average packet size, average segment size, average bytes and packets per bulk, and the average bulk rate for both directions.
- Subflow Features (67th to 71st): These detail the number of packets and bytes in a subflow for both directions, along with the number of bytes in the initial window for each direction.

- **Additional Flow Features (74th and 75th):** The 74th feature represents the number of packets containing actual data in the forward direction, while the 75th feature records the minimum segment size observed in the forward direction.
- **Active and Idle Period Features (76th to 83rd):** These capture flow behavior during 'active' and 'idle' periods, including the mean, standard deviation, maximum, and minimum time the flow was active or idle.
- **Label (84th):** This feature serves as a classification marker for supervised learning models, categorizing traffic as benign or one of the attack types.

The proposed ROSIDS23 dataset is compared with widely used public datasets (Kilincer et al., 2021) that are introduced in Section 2.2. All the datasets in the table are based on realistic network environments and are labeled. The results of the comparison are given in Table 3.2. The table compares various datasets based on several key attributes. Attack Types list the specific security threats covered, such as DoS, R2L, or ROS-specific attacks. Attributes indicate the number of features or variables used for analysis in each dataset. The ROS-based attacks column identifies whether the dataset includes attacks specific to the ROS environment, and the ROS Environment confirms if the dataset is designed to operate within a ROS system.

Table 3.2. Public datasets comparisons with proposed ROSIDS23 dataset

| Dataset            | Attack Types                                                                                                               | Attributes | ROS-based attacks | ROS Environment |
|--------------------|----------------------------------------------------------------------------------------------------------------------------|------------|-------------------|-----------------|
| <b>KDDCup99</b>    | DoS, Remote to Local (R2L), User to Root(U2R), Probing                                                                     | 43         | No                | No              |
| <b>Darpa</b>       | DoS, Remote to Local (R2L), User to Root(U2R), Probing                                                                     | 41         | No                | No              |
| <b>NSL-KDD 99</b>  | Normal, DoS, Remote to Local (R2L), User to Root(U2R), Probing                                                             | 43         | No                | No              |
| <b>CICIDS-2017</b> | DoS Golden Eye, Bening, DoS hulk, DoS Slow HTTP, DoS Slowloris, DDoS-LOIC HTTP, DDoS-LOIC-UDP, DDoS-HOIC, SSH-Patator, FTP | 84         | No                | No              |

| Dataset                 | Attack Types                                                                                                                                                          | Attributes | ROS-based attacks | ROS Environment |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|-------------------|-----------------|
|                         | Patator, Brute force, XSS, Botnet, infiltration, SQL injection                                                                                                        |            |                   |                 |
| <b>CSE-CIC-IDS 2018</b> | DoS Golden Eye, Heartbleed, DoS hulk, DoS Slow HTTP, DoS Slowloris, DDoS, SSH-Patator, FP, Patator, Brute force, XSS, Botnet, infiltration, PortScann, SQL injection. | 84         | No                | No              |
| <b>UNSW-NB15</b>        | Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms                                                                                | 49         | No                | No              |
| <b>ROSIDS23</b>         | Subscribing Flood, DoS, Unauthorized Publish, Unauthorized Subscribe                                                                                                  | 83         | Yes               | Yes             |

### 3.2. Proposed Adversarial Attack Detection Approach

The proposed adversarial attack detection approach is introduced with the foundational hypothesis. The designed system architecture for the Reconstruction Error-based Adversarial Detection (READ) system is also explained.

#### 3.2.1. Hypothesis

Adversarial attacks perturb the most critical features of data, causing significant changes in data representation. It is proposed that a reconstruction-based methodology can effectively and accurately detect these attacks. While these attacks subtly alter data to mislead classification methods, the attacks also cause notable shifts in the data's representation in lower dimensions, which can be leveraged for detection.

This hypothesis initially considers the normal scenario of data processing where  $X$  represents the original data. The process begins with a dimensionality reduction step, denoted as  $F(X)$  in Equation 3.1, which transforms the original data  $X$  into a reduced form  $Z$ . Following this, a reconstruction operation,  $R(Z)$ , is applied, attempting to revert  $Z$  back

to its original form, resulting in  $\hat{X}$  in Equation 3.2. The reconstruction error in this case,  $\Delta$ , is typically negligible, as shown in Equation 3.3.

$$Z = F(X) \quad (3.1)$$

$$\hat{X} = R(Z) \quad (3.2)$$

$$\Delta = |X - \hat{X}| \quad (3.3)$$

Then, an adversarial attack  $A(X)$  is applied to the original data, yielding a modified data  $X_{adv}$  in Equation 3.4. This modified data undergoes the same dimensionality reduction and reconstruction process given in Equation 3.5, yielding a significantly larger reconstruction error, denoted as  $\bar{\Delta}$ , which is significantly larger than in the standard scenario in Equation 3.6.

$$A(X) \rightarrow X_{adv} \quad (3.4)$$

$$R(F(X_{adv})) \rightarrow X + \bar{\Delta} \quad (3.5)$$

$$\bar{\Delta} \gg \Delta \quad (3.6)$$

For the reconstruction model, the reconstructed version of original data is  $R(F(X))$  and of the adversarial attacked data is  $R(F(X_{adv}))$ . The error rates for original and adversarially attacked data are denoted  $\epsilon_{RE}(R(F(X)))$  in Equation 3.7 and  $\epsilon_{RE}(R(F(X_{adv})))$  is given Equation 3.8 respectively. Based on proposed hypothesis:

$$\begin{aligned} R(F(X)) &\approx X \\ \epsilon_{RE}(R(F(X))) &\rightarrow \text{minimal} \end{aligned} \quad (3.7)$$

$$\begin{aligned} R(F(X_{adv})) &\neq X_{adv} \\ \epsilon_{RE}(R(F(X_{adv}))) &\rightarrow \text{high} \end{aligned} \quad (3.8)$$

The model architecture includes the last hidden layers,  $H$  and a reconstruction architecture,  $R(X)$ , which would provide a robust mechanism to capture and model the nuanced relationships and patterns in the transformed data representation, potentially introduced due to adversarial attacks. If  $A(X) \rightarrow X_{adv}$  and  $F(X) \rightarrow F(X_{adv}) + \delta$  then

difference between  $X_{adv}$  and  $R(F(X_{adv}))$  gauged by reconstruction error calculation  $\epsilon_{RE}$ , can serve as a mechanism to detect adversarial attacks.

### 3.2.2. The Proposed System Architecture: Reconstruction Error-based Adversarial Detection (READ) System Design

The methodology of this study has a detailed hierarchical structure, as shown in Figure 3.6 and Figure 3.7. Figure 3.6 focuses on the learning phase, which utilizes a deep neural network (DNN) model. This model includes the last hidden and output layers, representing the core of the learning architecture. It is important to highlight that the outputs from the last hidden layer can be extracted from any DL architecture, not just the DNN employed in this study. This versatility enables the integration of various architectures depending on the research needs.

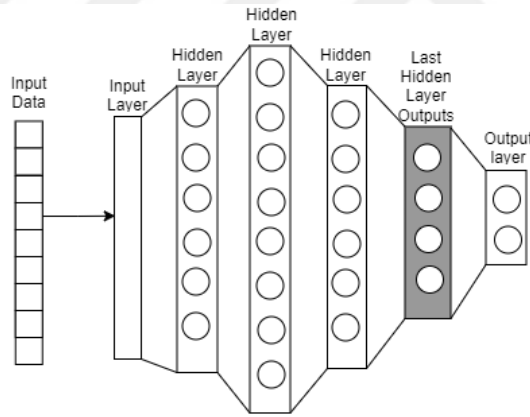


Figure 3.6. Deep learning architecture includes the last hidden layers and the output layer

In Figure 3.7 (a), the reconstruction model is depicted. This model plays a critical role in reconstructing the input data ( $X_i$ ) from the final layer's vector representation ( $H_i$ ), which is shown in Figure 3.6, to its original size. The reconstruction process employs Algorithm 1, specifically designed for this purpose. Mathematically, the reconstruction is expressed in Equation 3.9.

$$\hat{X}_i = R(H_i) \quad (3.9)$$

where  $R$  represents the regressor model trained on original input samples  $X_i$  to minimize the reconstruction error. When adversarial samples  $X_{adv}$  are passed through this model their reconstruction  $\hat{X}_{adv}$  is inherently influenced by the learned weights  $R$ , which are based on the original data distribution. This results in a notable divergence showed in Equation 3.10 and also mentioned as hypothesized in Section 3.2.1.

$$||X_{adv} - \hat{X}_{adv}|| > ||X_i - \hat{X}_i|| \quad (3.10)$$

This divergence underscores the robustness of the method, as it highlights the discrepancy between adversarial inputs and their reconstructions, effectively distinguishing adversarial patterns from benign ones.

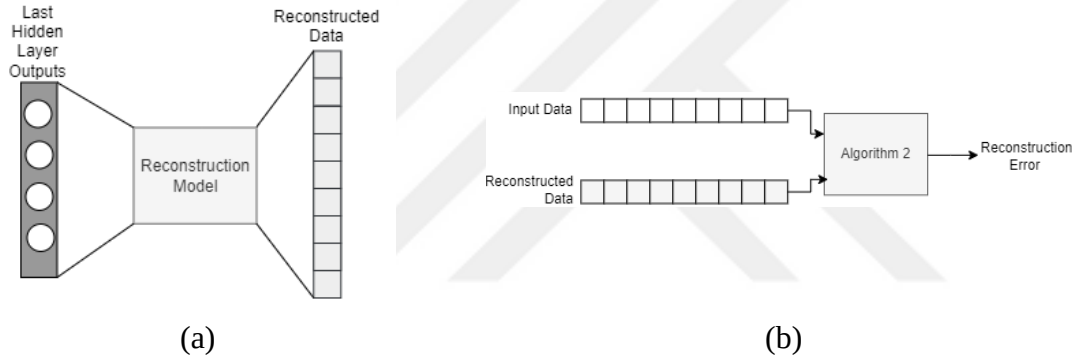


Figure 3.7. (a) Reconstruction architecture, (b) Reconstruction error calculation

---

#### Algorithm 1

---

**function:** Reconstruct\_Input\_Data ( $R$ ,  $H_i$ ):

**input:** A regressor model ( $R$ ), last hidden layer outputs from IDSs model ( $H_i$ ), input data ( $X_i$ )

**output:** Reconstructed data ( $\hat{X}_i$ )

$\hat{X}_i \leftarrow R.predict(H_i)$

**return**  $RD_i$

**end function**

---

In Figure 3.7 (b), the reconstruction error is calculated using both reconstructed data and input data in algorithm 2. Algorithm 2 formalizes the reconstruction error calculation

process. It ensures consistency by verifying the dimensionality of the input and reconstructed data before computing the chosen metric—either Mean Squared Error (MSE) or Median Absolute Error (MedAE). This flexibility enables tailored assessments based on the specific requirements of the research.

---

**Algorithm 2**


---

**function:** Calculate\_Reconstruction\_Error ( $X_i, \hat{X}_i, Str$ )

**input:** input data ( $X_i$ ), reconstructud data ( $\hat{X}_i$ ), error metric ( $Str$ ) (either 'mse' or 'medae')

**output:** Average Mean Squared Error ( $MSE_i$ ), Average Median Absolute Error ( $MedAE_i$ ), Error ( $E$ )

**if** size( $X_i$ )  $\neq$  size( $X_i''$ ) **then**

raise an error "InputData and ReconstructedData must have the same shape"

**else if**  $Str = 'mse'$  **then**

$MSE_i \leftarrow \text{mean}((\hat{X}_i - X_i)^2)$

$E \leftarrow MSE_i$

**else if** Metric = 'medae' **then**

$MedAE_i \leftarrow \text{median}(|\hat{X}_i - X_i|)$

$E \leftarrow MedAE_i$

**Else** \*raise an error\* "Invalid metric. Supported metrics are 'mse' and 'medae'."

**return**  $E$

**end function**

---

There are several metrics for the calculation of reconstruction error, such as MSE and MedAE that are defined in Equation 3.11 and Equation 3.12, respectively.

$$MSE_i = \frac{1}{n} \sum_{k=0}^n (X_i - \hat{X}_i)^2, \text{ for all } i \in [1, n] \quad (3.11)$$

$$MedAE_i = \text{median} (|\hat{X}_1 - X_1|, |\hat{X}_2 - X_2|, \dots, |\hat{X}_n - X_n|) \quad (3.12)$$

where  $X_i$  refers to each element of the input data vector  $X = \{x_1, x_2, \dots, x_n\}$ .  $\hat{X}_i$  refers to each element of the reconstructed data vector  $\hat{X} = \{\hat{x}_1, \hat{x}_2, \dots, \hat{x}_n\}$ .  $n$  refers to the total

number of elements in the vectors  $X$  and  $X''$ .  $|\hat{X}_i - X_i|$  indicates the absolute difference between corresponding elements of the input and reconstructed data.  $(X_i - \hat{X}_i)^2$  indicates the squared difference between corresponding elements of the input and reconstructed data. MSE quantifies the squared differences between the input and reconstructed data vectors, while MedAE measures their absolute differences. These metrics effectively quantify the original's reconstruction error value and the adversarial's reconstruction error value, providing critical insights to detect adversarial manipulations.



## 4. RESULTS AND DISCUSSION

In this thesis, five types of experiments have been conducted. Initially, the effect of an adversarial attack was observed for some DL architectures (see Section 4.1). In the second experiment, the number of adversarial attacks increased (see Section 4.2). In the third experiment, the effects of adversarial attacks on data samples were assessed by analyzing both original and adversarial samples (see Section 4.3). In Section 4.4, the impact of various adversarial attacks was examined in addition to the studies conducted in previous subsections. The effects of the attacks were analyzed on the ROSIDS23 dataset, specifically designed for IoT environments, and on other datasets. Based on these observations, the proposed READ architecture was tested for different types of adversarial attacks (see Section 4.5).

### 4.1. The Effect of an Adversarial Attack on Various Deep Learning Methods

In this experiment, the impact of the FGSM adversarial attack on a deep learning (DL)-based IDS was analyzed for various DL models. The experiment design is given in Figure 4.1. Figure 4.1 (a) illustrates a standard IDS, where a network traffic dataset is processed through data preprocessing and dataset splitting steps. The resulting original samples (train data) are used to develop a DL model during the training phase. During testing, original samples (test data) are fed into the DL model, classifying the traffic as benign or malicious. Figure 4.1 (b) highlights the scenario of the IDS against adversarial attacks, specifically using the FGSM Attack. After dataset splitting, adversarial samples are generated by applying the FGSM attack (Goodfellow et al., 2014a) to the original samples (test data). These adversarial samples are added to the testing phase, creating inputs that attempt to trick the DL model into misclassification. This reveals the system's vulnerability to adversarial samples. To address this, in this thesis, the READ Model is introduced to mitigate the effects of adversarial attacks and improve the robustness of the IDS (see Subsection 4.5).

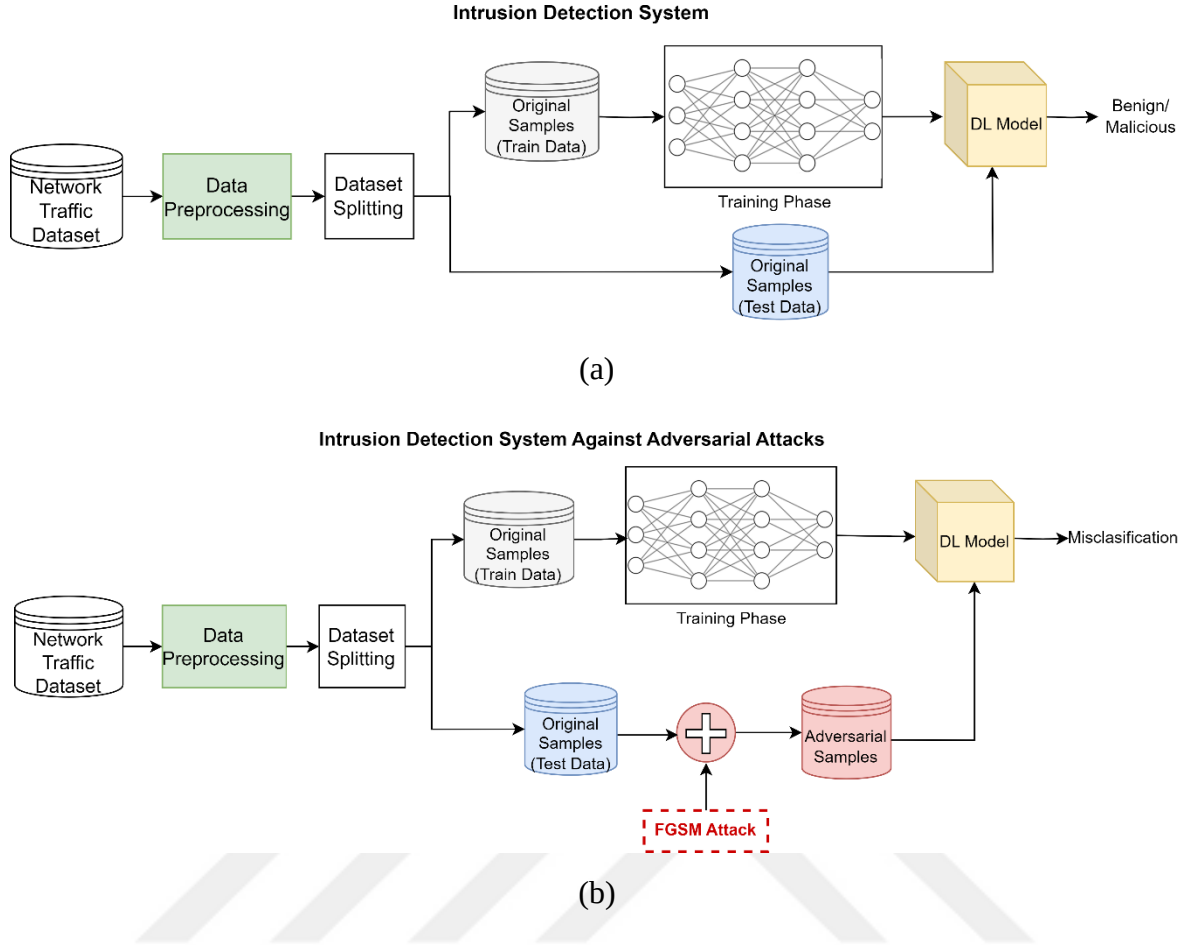


Figure 4.1. Experiment design for the effect of adversarial attack on DL architecture; (a) No attack, (b) FGSM attack

In this experiment design, seven deep learning (DL) models were tested, including Autoencoders (AE), Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), and Multi-Layer Perceptron (MLP). To enhance their robustness, AE was integrated as a feature reduction layer into DNN, CNN, and MLP, creating hybrid models. The hybrid architectures were compared to their standard counterparts under adversarial conditions. The architectures behind the DL models are presented in Figure 4.3. Brief descriptions of the architectures behind the DL models are provided below:

- Autoencoders (AE) are specialized networks that convert input data into vectors, commonly used for feature reduction. Frequently applied in unsupervised deep networks for denoising, feature extraction, and dimensionality reduction, AE consists of two symmetrical networks: a compression layer and a reconstruction layer. The compression layer compresses the data to the hidden layer size, while the

reconstruction layer restores the data back to its original dimensions. Minimal loss between these two layers indicates good data representation.

- Multi-Layer Perceptron (MLP), developed as feedforward Artificial Neural Networks in 1992 by Pal et al. (1992), are fully connected, multi-layer neural networks. Typically composed of three layers—the input layer receiving data, the hidden layer as a hyperparameter of the MLP, and the output layer as the classification layer—the MLP's learning improves with additional hidden layers, though excessive layers may lead to gradient problems.
- Deep Neural Networks (DNN) are inspired by the human brain's ability to learn through experiences. They serve as an umbrella model encompassing MLPs and consist of input, hidden, and output layers. Unlike MLPs, DNNs can employ a significantly higher number of hidden layers.
- Convolutional Neural Networks (CNN) are composed of fully connected layers capable of feature extraction and classification. The architecture comprises five main layers: the input layer, which receives data; the convolutional layer, which extracts features from input data using multiple filters; the pooling layer, which reduces the input size for the next layer; fully connected layers as middle layers, which recognize extracted features; and, finally, the classification layer.

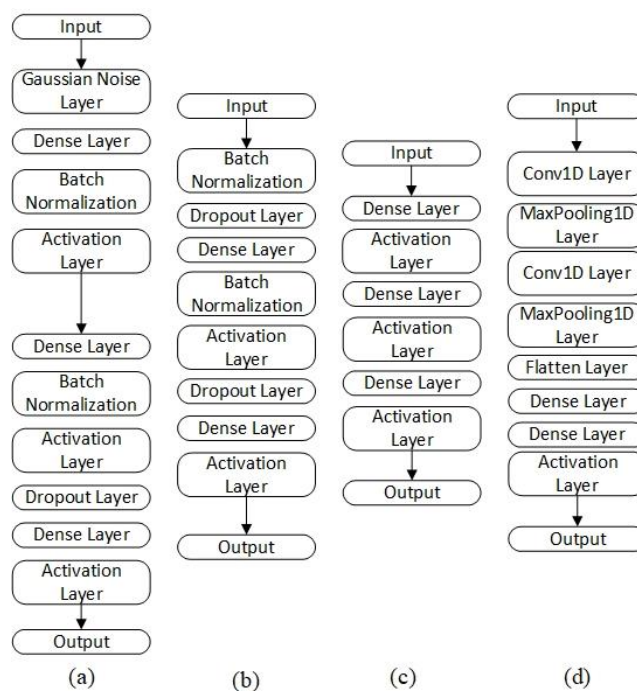


Figure 4.2. Classic DL methods architecture: (a) Autoencoder, (b) Multi-Layer Perceptron, (c) Deep Neural Networks, (d) Convolutional Neural networks

The DL model in Figure 4.7 is designed for the detection of Distributed Denial of Service (DDoS) attacks using the CICIDS2017 dataset (Sharafaldin et al., 2018). In the experiments, 79 commonly used features are focused (Abdulhammed et al., 2019; Ghurab et al., 2021; Siddiqi & Pak, 2020). Excluded features were Flow ID, Source IP, Destination IP, Timestamp, and External IP. The dataset was preprocessed by removing rows with missing values, and continuous features were scaled using Min-Max normalization to mitigate boundary effects (see Figure 4.3). Following preprocessing, the dataset contained 225,745 samples, with 183,910 representing normal traffic and 41,835 as attack traffic. It was divided into training and testing sets, with 180,596 samples in the training set, which were further split into 80% training and 20% validation subsets. All the DL models mentioned above are trained using this dataset. Each model was trained for up to 50 epochs with early stopping. All experiments were conducted in Google Colab with GPU support, using Python and the TensorFlow framework. Adversarial samples for the FGSM attack were generated with the Cleverhans library (Papernot et al., 2016).

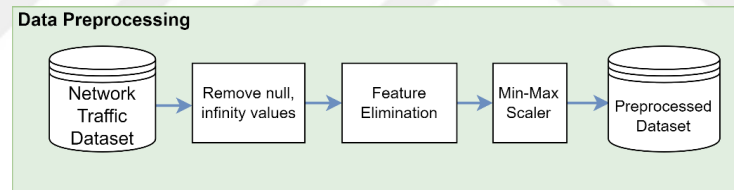


Figure 4.3. Preprocessing flow of the dataset

The system was initially tested under no-attack conditions, as depicted in Figure 4.7 (a), where the DL model effectively identified malicious traffic. When adversarial attacks were applied, as shown in Figure 4.7 (b), adversarial samples were generated using the Fast Gradient Sign Method (FGSM), testing the model's robustness. The DL models are tested under FGSM attacks and results are given in Table 4.1 and Table 4.2. In both tables, the first row shows the accuracy of the DL models in the no-attack scenario. The second row highlights the impact of the FGSM attack on the performance of these models. Table 4.1 presents the effects of an adversarial attack on the accuracy value of DL models (AE, MLP, DNN, CNN). While learning performance was consistent across models in the no-attack scenario, CNN demonstrated better robustness against FGSM adversarial samples, whereas AE showed the greatest drop in performance.

Table 4.1. The effect of FGSM Attack on Deep Learning Models

| Deep Learning Models |     |     |     |            |
|----------------------|-----|-----|-----|------------|
| Accuracy             | AE  | MLP | DNN | CNN        |
| <b>No Attack</b>     | 89% | 89% | 90% | 90%        |
| <b>FGSM</b>          | 73% | 80% | 76% | <b>81%</b> |

Table 4.2 presents the results of DL models enhanced with AE (AEMLP, AEDNN, AECNN). Before the FGSM attack, the hybrid models achieved similar accuracy around 90%. Under the adversarial attack, DNN (AEDNN) and MLP (AEMLP) architectures augmented with AE show improved performance compared to the standalone DNN and MLP models. Although the learning performance of the AECNN architecture is similar to that of CNN, the addition of the Autoencoder layer has made it more sensitive to adversarial attacks.

Table 4.2. Experimental results of adversarial attacks on deep learning architectures enhanced with autoencoders

| Deep Learning (Hybrid) Models |     |            |       |       |
|-------------------------------|-----|------------|-------|-------|
| Accuracy                      | AE  | AEMLP      | AEDNN | AECNN |
| <b>No Attack</b>              | 89% | 89%        | 90%   | 90%   |
| <b>FGSM</b>                   | 73% | <b>81%</b> | 78%   | 79%   |

The performance of the CNN classifier among DL models is comparable to other models but shows superior results when exposed to adversarial attacks. CNN's resilience to adversarial attacks can be attributed to its ability to address overfitting issues through convolutional layers, enabling better feature recognition. This robustness is presumed to enhance CNN's resistance to adversarial samples. DNN's higher sensitivity to adversarial attacks compared to MLP can be linked to the stabilizing effect of MLP's higher number of hidden units and the presence of batch normalization layers, which enhance network stability.

In hybrid models, AE is employed for dimensionality reduction and is configured with only a single dense layer. Consequently, AE alone appears more sensitive to adversarial attacks than other models. When combined with the CNN model, the performance of CNN with AE falls slightly below that of the standalone CNN in adversarial scenarios, likely due to information loss during the dimensional reduction stage in the Autoencoder layer. It has been observed that MLP (AEMLP) and DNN (AEDNN) models combined with AE perform better against adversarial samples than their non-Autoencoder counterparts. This improvement is likely due to the vector representations generated after the compression layer, which enable DL methods to create more effective data representation models.

Accuracy declined across all models (AE, MLP, DNN, CNN, AEMLP, AEDNN, and AECNN) when tested with adversarial samples generated via FGSM attacks. The findings indicate that DL architectures exhibit vulnerabilities in adversarial detection. However, the impact of adversarial attacks can be mitigated by utilizing AE's dimensional reduction capability. The combination of AE with MLP was particularly effective, producing the most robust models against adversarial attacks.

#### 4.2. The Effect of Various Adversarial Attacks on Deep Learning Methods

In this experiment, the robustness of deep learning-based systems is analyzed for various adversarial attacks such as FGSM, PGD, and BIM. Unlike the previous section that focused on FGSM, this analysis aimed to compare the effects of different attacks on system performance. As shown in the experimental setup depicted in Figure 4.4, the DL models were tested with adversarial samples generated through these attacks.

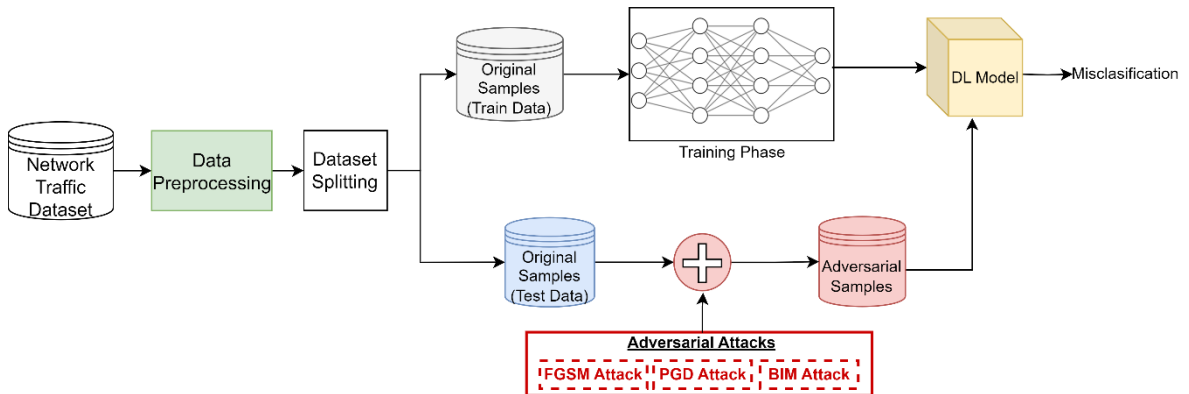


Figure 4.4. Effects of adversarial attacks on different DL networks experiment design

This subsection evaluated the robustness of DL models against adversarial attacks during the testing phase. Three attack methods—FGSM, PGD, and BIM—were tested on four DL models (Autoencoder, MLP, DNN, and CNN). The details of the architecture behind the DL models are given in Section 4.1. The experiments, conducted in a Google Colab environment with GPU support, utilized Python 3.7, TensorFlow 0.16, and NumPy 1.21. In all experiments, the CICIDS2017 dataset was used. The focus of this study was on DDoS attacks for binary classification. The dataset was preprocessed in all experiments as outlined in the flow diagram in Figure 4.3. To prepare the dataset, null and infinity values were removed, and 84 features were reduced to 79 by excluding Flow ID, Source IP, Destination IP, and Timestamp, as in the previous subsection. Min-Max Scaler normalization was applied to the preprocessed dataset, as shown in the diagram. The dataset consisted of 225,745 samples, with 183,910 normal traffic instances and 41,835 attack traffic instances, and was split into 80% training and 20% testing data. A 5-fold cross-validation with 50 epochs was applied, where the training set was divided into five parts, and the trained model was used to evaluate the test dataset.

The evaluation of the models in Table 4.3 was conducted using a 5-fold cross-validation approach with 50 epochs. In this method, the training dataset was divided into five equal parts, or folds, and during each iteration, one fold was used for validation while the remaining four were used for training. This process was repeated five times, ensuring that each fold was used for validation once. After training on each fold, the model was tested against the test dataset, and adversarial samples were generated to evaluate the robustness of the models under attack scenarios.

Table 4.3 shows the accuracy results of four deep learning models (AE, MLP, DNN, and CNN) under both non-attack conditions and three adversarial attack scenarios (FGSM, PGD, and BIM). The results for the non-attack scenario indicate that all DL methods achieved approximately 90% accuracy. As shown in Table 4.3, each adversarial attack led to a decrease in accuracy, with the "No attack" rows reflecting the original dataset accuracy. The average results reveal that DNN and CNN maintained higher accuracy during the attacks, whereas AE and MLP exhibited lower accuracy under the same conditions. Additionally, the impact of the PGD and BIM attacks was found to be more severe compared to FGSM.

The accuracy results across all folds, analyzed from two perspectives, are presented in Table 4.3. First, the effects of FGSM, PGD, and BIM attacks on the models were evaluated individually, followed by an analysis of fold-specific accuracy against these attacks. The FGSM attack led to an accuracy decrease across all models. Among them, MLP and AE demonstrated the lowest performance, while CNN performed better, and DNN achieved the highest accuracy, indicating that CNN and DNN are more resilient to FGSM attacks. For the PGD attack, all models experienced reduced accuracy, with CNN and AE showing similar decreases across folds. However, DNN was less affected compared to others, while MLP exhibited the greatest drop, highlighting its vulnerability. Similarly, the BIM attack showed effects comparable to PGD, with MLP being the most affected model, while DNN proved the most robust, maintaining 70% accuracy. AE and CNN displayed intermediate performance, falling between DNN and MLP. These findings identify MLP as the most vulnerable architecture to adversarial attacks, while DNN consistently demonstrated the highest robustness against all three attack types. AE and CNN offered average performance, with DNN emerging as the most effective model for detecting DDoS attacks. The second perspective involved analyzing the robustness of individual folds against adversarial attacks. Although all models achieved similar accuracy under “no attack” conditions, the results showed significant variability across folds. For AE models, the 1st and 3rd folds were more vulnerable to attacks, whereas the 4th and 5th folds displayed greater resilience, with the 2nd fold performing best overall. For MLP, the 3rd fold demonstrated better resistance to adversarial attacks, while the 1st and 2nd folds were the most affected. In the case of DNN models, the 1st, 2nd, and 3rd folds exhibited stronger performance against adversarial attacks compared to the 4th and 5th folds. CNN models showed better accuracy against adversarial attacks in the 1st, 2nd, and 5th folds than in the 3rd and 4th folds. These findings indicate that the robustness of the folds can vary based on preprocessing and that DNN models consistently exhibit superior resilience to adversarial attacks across all scenarios.

The experiment study findings indicate that MLP and AE models are more susceptible to adversarial attacks compared to DNN and CNN models. This vulnerability may stem from the inclusion of Batch Normalization, Gaussian Noise (in AE), and Dropout layers in their architectures. While these layers contribute to the uniqueness of MLP and AE

models, they also appear to make them more prone to adversarial manipulation. However, further research is required to understand this relationship better.

Table 4.3. Effects of adversarial attacks on different DL networks test results

| Accuracy               |                            | Deep Learning Methods               |                                     |                                     |                                     |
|------------------------|----------------------------|-------------------------------------|-------------------------------------|-------------------------------------|-------------------------------------|
| <i>Fold No</i>         | <i>Adversarial Attacks</i> | <i>AE</i>                           | <i>MLP</i>                          | <i>DNN</i>                          | <i>CNN</i>                          |
| <b>1st</b>             | <b>No Attack</b>           | 0,90                                | 0,90                                | 0,90                                | 0,90                                |
| <b>Fold</b>            | <b>FGSM</b>                | 0,60                                | 0,38                                | 0,72                                | 0,71                                |
|                        | <b>PGD</b>                 | 0,34                                | 0,32                                | 0,73                                | 0,69                                |
|                        | <b>BIM</b>                 | 0,34                                | 0,32                                | 0,73                                | 0,69                                |
|                        | <b>No Attack</b>           | 0,90                                | 0,90                                | 0,90                                | 0,91                                |
| <b>2nd Fold</b>        | <b>FGSM</b>                | 0,70                                | 0,28                                | 0,73                                | 0,71                                |
|                        | <b>PGD</b>                 | 0,63                                | 0,23                                | 0,72                                | 0,66                                |
|                        | <b>BIM</b>                 | 0,63                                | 0,23                                | 0,72                                | 0,66                                |
|                        | <b>No Attack</b>           | 0,90                                | 0,90                                | 0,91                                | 0,90                                |
| <b>3rd Fold</b>        | <b>FGSM</b>                | 0,54                                | 0,73                                | 0,74                                | 0,60                                |
|                        | <b>PGD</b>                 | 0,37                                | 0,69                                | 0,73                                | 0,48                                |
|                        | <b>BIM</b>                 | 0,37                                | 0,69                                | 0,73                                | 0,48                                |
|                        | <b>No Attack</b>           | 0,90                                | 0,90                                | 0,90                                | 0,91                                |
| <b>4th Fold</b>        | <b>FGSM</b>                | 0,70                                | 0,60                                | 0,70                                | 0,66                                |
|                        | <b>PGD</b>                 | 0,60                                | 0,53                                | 0,67                                | 0,61                                |
|                        | <b>BIM</b>                 | 0,60                                | 0,53                                | 0,67                                | 0,61                                |
|                        | <b>No Attack</b>           | 0,90                                | 0,90                                | 0,90                                | 0,91                                |
| <b>5th Fold</b>        | <b>FGSM</b>                | 0,70                                | 0,62                                | 0,70                                | 0,68                                |
|                        | <b>PGD</b>                 | 0,58                                | 0,46                                | 0,70                                | 0,64                                |
|                        | <b>BIM</b>                 | 0,58                                | 0,46                                | 0,70                                | 0,64                                |
|                        | <b>No Attack</b>           | 0,90                                | 0,90                                | 0,90                                | 0,91                                |
| <b>Average Results</b> | <b>No Attack</b>           | <b>0,90</b>                         | <b>0,90</b>                         | <b>0,90</b>                         | <b>0,91</b>                         |
|                        | <b>FGSM</b>                | <b>0,65 (<math>\pm 0,06</math>)</b> | <b>0,52 (<math>\pm 0,16</math>)</b> | <b>0,72 (<math>\pm 0,02</math>)</b> | <b>0,67 (<math>\pm 0,04</math>)</b> |
|                        | <b>PGD</b>                 | <b>0,50 (<math>\pm 0,12</math>)</b> | <b>0,45 (<math>\pm 0,16</math>)</b> | <b>0,71 (<math>\pm 0,02</math>)</b> | <b>0,62 (<math>\pm 0,07</math>)</b> |
|                        | <b>BIM</b>                 | <b>0,50 (<math>\pm 0,12</math>)</b> | <b>0,45 (<math>\pm 0,16</math>)</b> | <b>0,71 (<math>\pm 0,02</math>)</b> | <b>0,62 (<math>\pm 0,07</math>)</b> |

Among the tested attack methods, FGSM was found to be less effective on DL models compared to PGD and BIM attacks. The iterative nature of PGD and BIM makes them more impactful, as opposed to FGSM, which has a fixed direction of loss. PGD incorporates random perturbations at the outset, enhancing its potency, as noted by (Jiang et al. 2021). While PGD is generally considered more powerful than BIM and FGSM, this study observed that both PGD and BIM resulted in similar accuracy reductions. This outcome may be attributed to the binary classification setup of the experiment. The use of K-fold cross-validation divided the dataset into multiple folds. Although accuracy on the test dataset was similar across folds in "no attack" scenarios, the folds exhibited varied resilience to adversarial attacks. Testing folds with adversarial samples during training revealed differences in their robustness. These insights could inform the selection of more resilient folds for future training and testing processes.

#### **4.3. The Comparative Analysis of Adversarial Attack on Various Datasets: ROSIDS23, CSE-CIC-IDS2018, and KDDCup99**

This experimental study focuses on assessing the effects of adversarial attacks on intrusion detection systems (IDSs) using the ROSIDS23 dataset (Degirmenci, 2023; Degirmenci, Kirca, et al., 2023) alongside the CSE-CIC-IDS2018 (Sharafaldin et al., 2018) and KDDCup99 (KDD Cup 1999, 2007) datasets. Unlike previous sections, which examined individual attacks or general DL model robustness, this analysis highlights the practical implications of adversarial attacks on IDS datasets. The experiments evaluated IDS performance under both no adversarial attack and under adversarial attacks such as FGSM, PGD, BIM. This comprehensive approach provides a deeper understanding of IDS resilience across diverse datasets and attack methods. These three datasets—KDDCup99 (145,586 samples), CSE-CIC-IDS2018 (508,248 samples), and ROSIDS23 (136,681 samples)—were preprocessed identically. This process included normalization with a Min-Max Scaler and removing null or infinity values. For each dataset, 80% of the data was split for training, with the remaining 20% for testing.

This subsection examines the impact of three adversarial attacks—FGSM, BIM, and PGD—applied at varying intensities (0.01 to 0.10). Under normal conditions, all IDS models demonstrated high classification accuracy, as shown in Table 4.4. KDDCup99 achieved the

highest accuracy of 99.66%, outperforming ROSIDS23 at 96.64% and CSE-CIC-IDS2018 at 93.71%. However, all systems' accuracy was significantly compromised under adversarial attack conditions. Under adversarial attacks, the IDS models exhibited substantial declines in classification accuracy across all datasets and perturbation levels. For instance, with the FGSM attack at a perturbation amount of 0.01, CSE-CIC-IDS2018's accuracy dropped to 76.27%, KDDCup99 slightly decreased to 99.13%, and ROSIDS23 fell to 58.98%. As the perturbation increased to 0.04 and 0.10, the accuracies further plummeted to 28.01% and 1.74% for CSE-CIC-IDS2018, 57.91% and 39.08% for KDDCup99, and 35.64% and 12.61% for ROSIDS23, respectively. Similarly, the PGD attack caused CSE-CIC-IDS2018's accuracy to decline from 65.53% at a perturbation of 0.01 to a mere 0.30% at 0.10. KDDCup99 maintained relatively higher accuracy under PGD, decreasing from 99.11% to 36.68%, while ROSIDS23's accuracy decreased from 59.96% to 2.89%. The BIM attack showed a comparable trend, with CSE-CIC-IDS2018's accuracy dropping from 54.65% at 0.01 perturbation to 0.23% at 0.10, KDDCup99 from 99.03% to 36.96%, and ROSIDS23 from 58.98% to 3.63%.

Table 4.4. CSE-CIC-IDS2018, KDDCup99, and ROSIDS23 datasets accuracy result under normal and adversarial attacks with various perturbation amounts

|                               |                            | CSE-CIC-IDS2018 | KDDCup99 | ROSIDS23 |
|-------------------------------|----------------------------|-----------------|----------|----------|
| <b>Standard IDSs Accuracy</b> |                            | 93,71           | 99,66    | 96,64    |
| <b>Attack Type</b>            | <b>Perturbation Amount</b> |                 |          |          |
| FGSM                          | 0.01                       | 76,27           | 99,13    | 58,98    |
|                               | 0.04                       | 28,01           | 57,91    | 35,64    |
|                               | 0.10                       | 1,74            | 39,08    | 12,61    |
| PGD                           | 0.01                       | 65,53           | 99,11    | 59,96    |
|                               | 0.04                       | 4,06            | 51,24    | 29,94    |
|                               | 0.10                       | 0,3             | 36,68    | 2,89     |
| BIM                           | 0.01                       | 54,65           | 99,03    | 58,98    |
|                               | 0.04                       | 3,25            | 49,95    | 24,21    |
|                               | 0.10                       | 0,23            | 36,96    | 3,63     |

#### 4.4. The Effects of Adversarial Attacks on Data Features

In this experiment, the effects of adversarial attacks on the features of IDS are examined. During the preliminary stage of the proposed adversarial attack detection approach, the focus was on examining how adversarial attacks alter a small representation of the dataset. The key idea is that, while adversarial modifications may appear minor, they result in significant shifts in the optimal data representation, making the presence of an attack detectable. The primary finding, as detailed in this section, demonstrates that adversarial attacks specifically affect the relative importance of critical features within the data. Table 4.4 highlights this phenomenon by comparing original and adversarial samples from two perspectives. For instance, a FGSM attack with an intensity of 0.01 causes a notable change in the ranking of the top five most important features. This subtle yet impactful shift illustrates how adversarial attacks alter data representation in ways that can be harnessed for enhanced detection methods.

Table 4.5. The top five important features of the CSE-CIC-IDS2018 dataset are original samples and adversarial samples

| Rank | Original Samples | Score   | Adversarial Sample | Score   |
|------|------------------|---------|--------------------|---------|
| 1    | Feature: 30      | 3.07548 | Feature: 30        | 8.36310 |
| 2    | Feature: 8       | 2.80700 | Feature: 8         | 4.94248 |
| 3    | Feature: 24      | 1.42428 | Feature: 28        | 4.48721 |
| 4    | Feature: 6       | 1.02446 | Feature: 24        | 1.46055 |
| 5    | Feature: 28      | 0.38068 | Feature: 22        | 1.35351 |

Building on the initial exploration, the analysis progresses to examine the weight of feature importance in greater depth, as visually depicted in Figure 4.5. This step not only confirms the findings from the first phase but also offers a detailed perspective on how data distribution and characteristics are affected by adversarial attacks, even when minor perturbations are introduced.

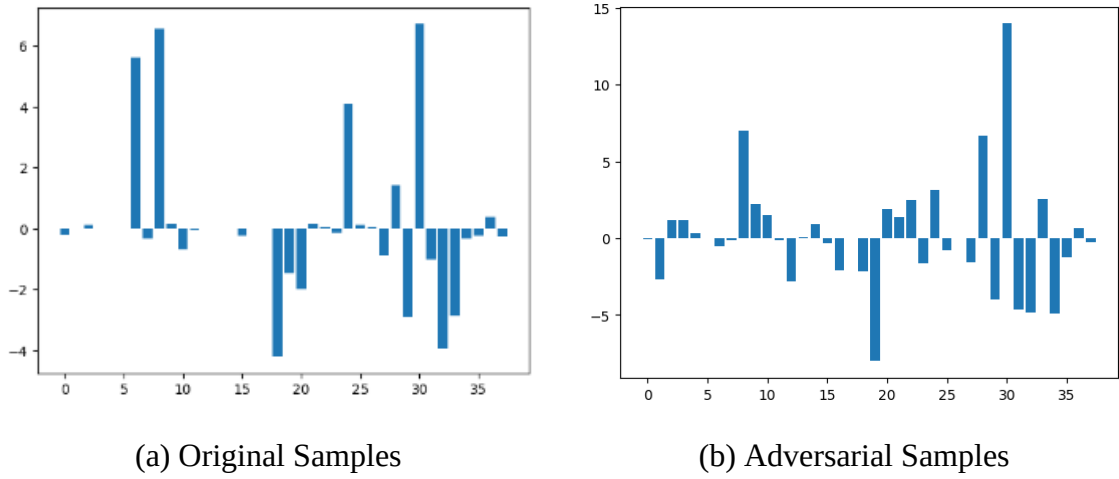


Figure 4.5. Feature importance weight result's graph of CSE-CIC-IDS2018 dataset

#### 4.5. The Results of Adversarial Attack Detection using READ

The final experiment focuses on assessment the effectiveness of the proposed Reconstruction Error-based Adversarial Detection (READ) approach. This section differs from previous analyses by focusing not only on the impact of adversarial attacks but also on the ability of a detection method specifically designed to counteract them. The READ system targets feature-based network traffic and utilizes reconstruction error as its primary detection mechanism. In addition to reconstruction error, three supplementary metrics—aleatoric uncertainty, epistemic uncertainty, and entropy—were employed to distinguish between original and adversarial data.

To comprehensively evaluate READ's performance, tests were conducted against three widely known adversarial attack methods: FGSM, BIM, and PGD. These experiments reveal how READ performs as a specialized adversarial detection system, emphasizing its robustness in mitigating the effects of adversarial attacks on IDS. This section provides critical insights into the practical application of READ and its ability to enhance the resilience of intrusion detection systems against adversarial manipulation.

##### 4.5.1. Experiment Design

The experiment design follows the machine learning lifecycle, which consists of training and testing phases. The training of the READ system is divided into five key steps,

which are given in Figure 4.6. The first step involves two stages of training. In the first step, two models were developed: a DL model for IDS and a reconstruction model. The IDS model utilized a CNN-based deep learning framework and was trained on original samples. In the second step, the reconstruction model was trained to reconstruct the original samples as closely as possible by leveraging the outputs from the last hidden layer of the IDS model. For the reconstruction, a Linear Regression model was used to map the relationship between the last hidden layer's outputs and the input data. This model was initialized using the `LinearRegression()` function from the scikit-learn library. The second step focused on generating adversarial samples. Adversarial attacks, including FGSM, BIM, and PGD, were applied to the original samples. In the third step, reconstruction errors, entropy, and uncertainty metrics (aleatoric and epistemic) (details in Section 2.7) were calculated for both the original and adversarial samples. Two algorithms, referred to as Algorithm 1 and Algorithm 2, were used to compute the reconstruction errors (see Section 3.2.2). Among the two metrics evaluated—MSE and MedAE—MedAE was selected for use in the experimental results due to its higher reliability. The fourth step involved training the READ model. This model was built using the calculated metrics, including reconstruction error (MedAE), entropy, and both aleatoric and epistemic uncertainties (see Section 3.2.2) for detecting adversarial attacks. The final step involved evaluating the performance of the adversarial attack detection model. This model is designed to differentiate adversarial samples from original ones. The four metrics derived from the test data were used for classification, and the resulting predictions were analyzed to determine whether each sample was original or adversarial.

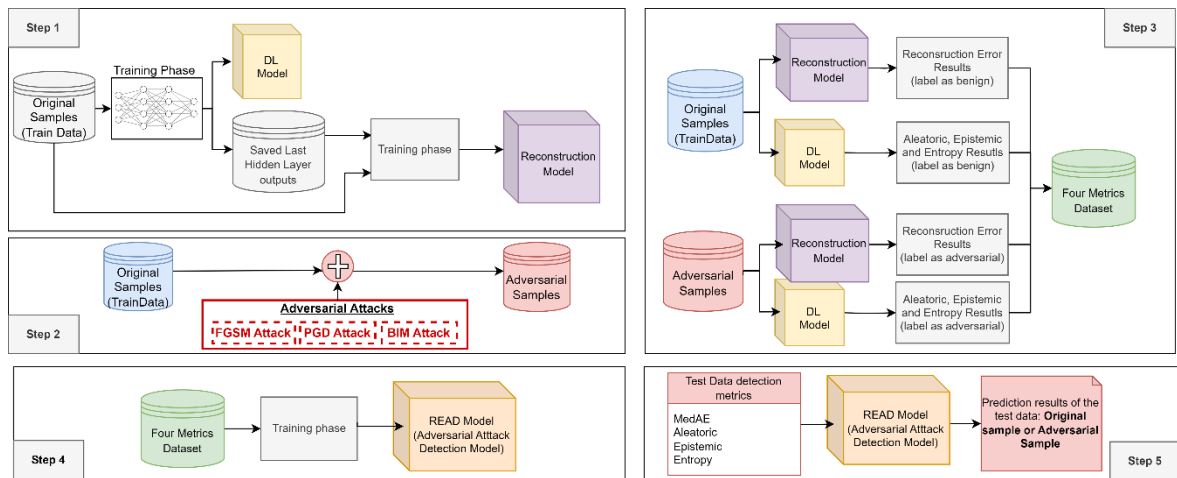


Figure 4.6. The steps of training for READ system

Figure 4.7 shows the testing phase of the READ model, which has been integrated to enhance the performance of the IDS model. Adversarial samples, which may be subjected to adversarial attacks such as FGSM, BIM, or PGD, are converted into "Adversarial Samples." Both the original and adversarial samples are subsequently processed by the READ model. The READ model evaluates each sample to determine if it is adversarial. Samples identified as adversarial are labeled as "Adversarial", while non-adversarial samples are passed to the IDS model for further analysis. This workflow, as illustrated in the figure, highlights the critical role of the READ model in detecting and filtering adversarial samples, thereby safeguarding the IDS model from potential disruptions.

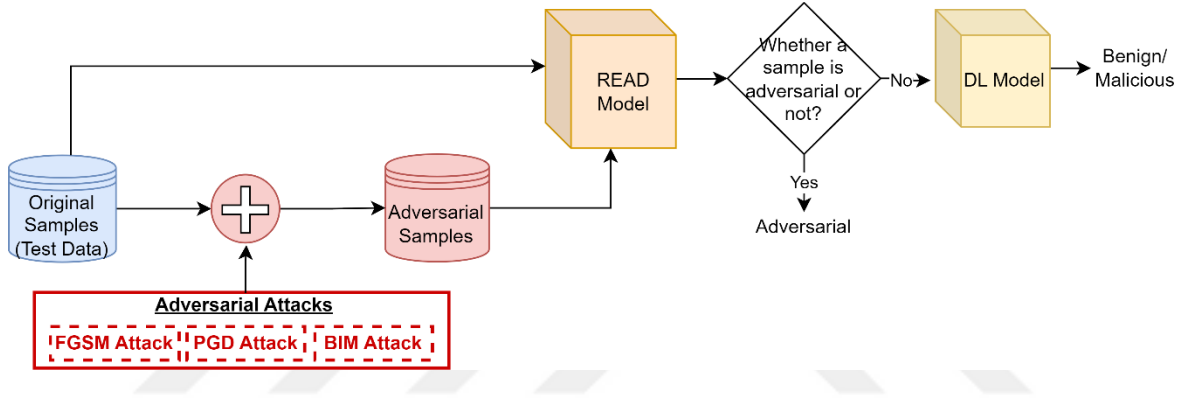


Figure 4.7. Experiment design for the test of the READ system

#### 4.5.2. The Experimental Results for READ

The proposed methodology for detecting adversarial samples employs multiple metrics, one of which involves calculating reconstruction errors. Figure 4.8 presents the reconstruction error results for both original and adversarial samples from the CSE-CIC-IDS2018 dataset. The findings demonstrate a significant disparity in reconstruction error rates between adversarial and original samples, even at a perturbation level of 0.04 under the FGSM attack.

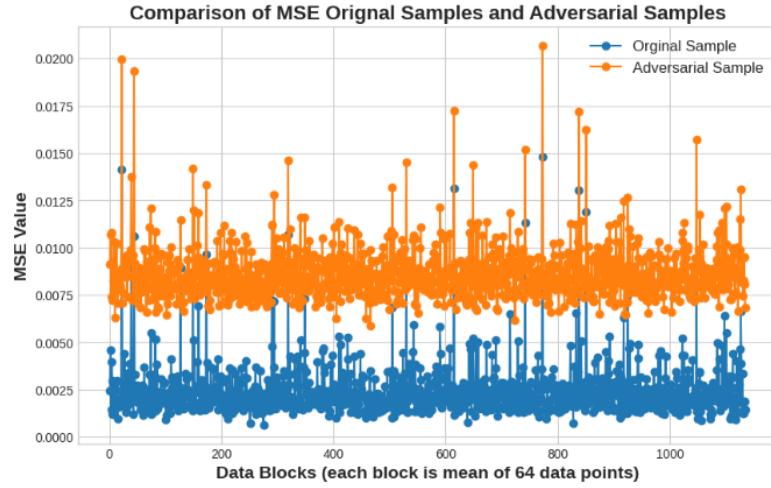


Figure 4.8. CSE-CIC-IDS2018 dataset reconstruction error results in original samples and adversarial samples

Table 4.6 lists the MSE and MedAE reconstruction errors for the CSE-CIC-IDS2018, KDDCup99, and ROSIDS23 datasets under normal conditions and various adversarial attacks (FGSM, PGD, and BIM) with varying perturbation levels.

Table 4.6. MSE and MedAE of reconstruction errors for different datasets under FGSM, PGD, and BIM with various perturbation amounts and normal samples

| Attack Type | Epsilon | CSE-CIC-IDS2018 |         | KDDCup99 |         | ROSIDS23 |         |
|-------------|---------|-----------------|---------|----------|---------|----------|---------|
|             |         | MSE             | MedAE   | MSE      | MedAE   | MSE      | MedAE   |
| <b>None</b> | 0.00    | 0.00251         | 0.00411 | 0.00293  | 0.00505 | 0.00657  | 0.01351 |
| <b>FGSM</b> | 0.01    | 0.03291         | 0.03355 | 0.00644  | 0.01223 | 0.01596  | 0.03157 |
|             | 0.04    | 0.03773         | 0.04524 | 0.02748  | 0.04484 | 0.12169  | 0.10562 |
|             | 0.10    | 0.37858         | 0.12729 | 0.06046  | 0.09771 | 0.16085  | 0.11215 |
| <b>PGD</b>  | 0.01    | 0.03701         | 0.03392 | 0.00643  | 0.01223 | 0.01841  | 0.03086 |
|             | 0.04    | 0.29255         | 0.06455 | 0.01894  | 0.04484 | 0.0997   | 0.08672 |
|             | 0.10    | 7.97365         | 0.16187 | 0.03792  | 0.09771 | 0.07323  | 0.13217 |
| <b>BIM</b>  | 0.01    | 0.03543         | 0.03207 | 0.00648  | 0.01229 | 0.01738  | 0.03072 |
|             | 0.04    | 0.48575         | 0.06191 | 0.01815  | 0.03802 | 0.14822  | 0.10492 |
|             | 0.10    | 9.06683         | 0.15634 | 0.0394   | 0.07434 | 0.08148  | 0.16986 |

For original samples, the baseline reconstruction errors without adversarial perturbations are as follows: the CSE-CIC-IDS2018 dataset has an MSE of 0.00251 and a MedAE of 0.00411, the KDDCup99 dataset has an MSE of 0.00293 and a MedAE of 0.00505, and the ROSIDS23 dataset exhibits an MSE of 0.00657 and a MedAE of 0.01351. Analysis reveals that both MSE and MedAE scores increase under adversarial attack conditions across all datasets, highlighting the substantial impact of these attacks. This trend signifies greater average deviations from true values, emphasizing the pronounced effects of adversarial perturbations on the datasets. MSE and MedAE are effective metrics for distinguishing between original and adversarial data. While most studies rely on MSE for adversarial attack detection, MedAE proves to be more effective in the 0-1 value range due to its reliance on absolute errors instead of squared differences. In this thesis, MedAE was particularly adept at highlighting reconstruction errors for small perturbations, whereas MSE became more effective as the attack intensity increased. However, the computational overhead introduced by MedAE is higher compared to MSE. To balance accuracy and efficiency, the proposed approach prioritizes the use of MedAE during the initial stage to accurately detect small perturbations.

Figure 4.9 provides a graphical representation of reconstruction error outcomes for the CSE-CIC-IDS2018 dataset subjected to FGSM, PGD, and BIM attacks across various perturbation levels. The results indicate that perturbation rates deviating from 0.1 are more distinguishable from the original dataset. Additionally, as the perturbation rate increases, the reconstruction errors for each attack exhibit a greater degree of divergence, further illustrating the impact of adversarial perturbations.

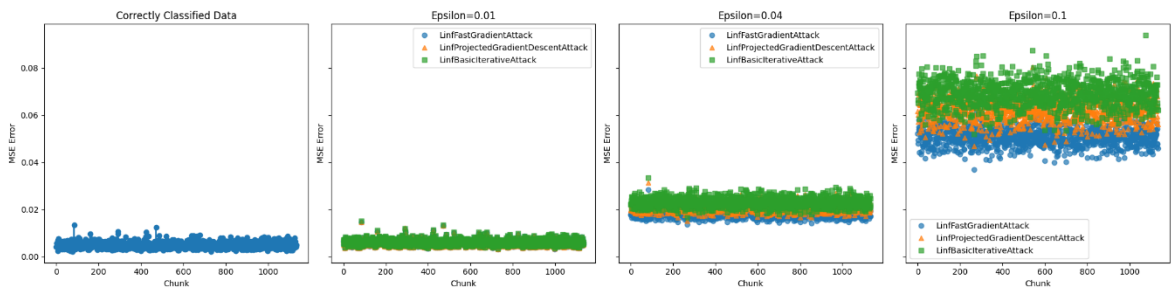


Figure 4.9. CSE-CIC-IDS2018 dataset reconstruction error results in graphical visualization under FGSM, PGD, and BIM with various perturbation amounts

The effects of adversarial attacks vary across datasets even for the same epsilon values (see Table 4.7). Consequently, the proposed approach is compared with studies in the literature. Priority is given to evaluating the extent of performance loss experienced by the system under adversarial attack. This performance loss served as the primary metric for assessing the success of detection methods.

The experimental results revealed that the impact of adversarial attacks differs across datasets depending on the epsilon values (see Table 4.7). To ensure meaningful comparisons with the literature, the experiments were structured around the performance loss caused by the attacks on the system, as detailed in Table 4.8.

Table 4.7. Proposed detection system ROC-AUC scores across CSE-CIC-IDS-18 and KDDCup99 datasets under FGSM, BIM, and PGD attacks at varying perturbation levels

| Epsilon | Attack Type | CSE-CIC-IDS2018     |                         | KDDCup99            |                         | ROSIDS23            |                         |
|---------|-------------|---------------------|-------------------------|---------------------|-------------------------|---------------------|-------------------------|
|         |             | Attack success rate | Roc-Auc detection score | Attack success rate | Roc-Auc detection score | Attack success rate | Roc-Auc detection score |
| 0.01    | FGSM        | 24%                 | 96%                     | 1%                  | 93%                     | 41%                 | 84%                     |
|         | BIM         | 35%                 | 96%                     | 1%                  | 92%                     | 40%                 | 84%                     |
|         | PGD         | 45%                 | 96%                     | 1%                  | 93%                     | 41%                 | 84%                     |
| 0.04    | FGSM        | 72%                 | 100%                    | 42%                 | 100%                    | 64%                 | 97%                     |
|         | BIM         | 95%                 | 99%                     | 49%                 | 100%                    | 70%                 | 97%                     |
|         | PGD         | 96%                 | 100%                    | 50%                 | 99%                     | 76%                 | 97%                     |
| 0.1     | FGSM        | 98%                 | 100%                    | 61%                 | 100%                    | 87%                 | 100%                    |
|         | BIM         | 99%                 | 100%                    | 63%                 | 100%                    | 97%                 | 99%                     |
|         | PGD         | 99%                 | 100%                    | 63%                 | 100%                    | 96%                 | 99%                     |

The same adversarial attack with the same perturbation may affect various datasets differently. Wang et al. (2022) study examined the differences and detection of adversarial examples for cases where decision boundaries are completely disjoint, close, or intertwined. In our study, the reason for the varying success of adversarial attacks detected on different datasets could be similar.

Figure 4.10 presents the ROC curves for the FGSM attack on three datasets—CSE-CIC-IDS2018, KDDCup99, and ROSIDS23—with an epsilon value of 0.01. These curves compare the performance of various detection methods: Aleatoric, Epistemic, Entropy, MedAE, and a combined method (All). The results highlight the effectiveness of these methods in distinguishing between true and false positives under FGSM attacks.

For the CSE-CIC-IDS2018 dataset, the combined method (All) achieved the highest AUC of 0.98, demonstrating exceptional performance. The MedAE method also performed strongly, achieving an AUC of 0.95. In the KDDCup99 dataset, the MedAE method reached an AUC of 0.92, while the combined method (All) slightly outperformed it with an AUC of 0.93. The Aleatoric, Epistemic, and Entropy methods, however, showed significantly lower performance, with an AUC of 0.61. For the ROSIDS23 dataset, both the MedAE method and the combined method (All) achieved AUC values of 0.84, indicating comparable performance. The Aleatoric and Entropy methods showed weaker results, each with an AUC of 0.59, while the Epistemic method achieved a slightly higher AUC of 0.62.

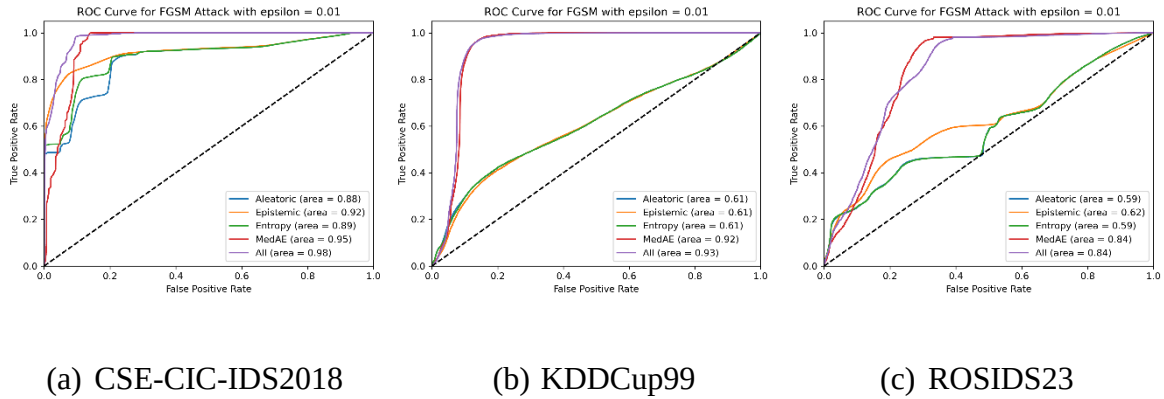


Figure 4.10. Comparative evaluation of ROC curves for FGSM attack

These experiments highlight that the combined method (All) and MedAE are highly effective in detecting adversarial samples, particularly for the CSE-CIC-IDS2018 dataset. For the KDDCup99 dataset, both methods also demonstrate strong detection performance, reaffirming their effectiveness. Similarly, in the case of the ROSIDS23 dataset, the MedAE and combined methods show good results, emphasizing their reliability and adaptability across various datasets.

Figure 4.11 shows the ROC curve performance of various detection methods under an FGSM attack with an epsilon value of 0.04 on the CSE-CIC-IDS2018 dataset. The graph plots the True Positive Rate (TPR) against the False Positive Rate (FPR) for the Aleatoric, Epistemic, Entropy, MedAE, and Combined (All) methods. Among the methods, the combined method (All) achieves the best performance with an area under the curve (AUC) of 1.00, indicating perfect detection capability. The MedAE method also performs exceptionally well, attaining an AUC of 0.99. The Epistemic and Entropy methods follow with AUC values of 0.96 and 0.94, respectively, while the Aleatoric method has the lowest AUC of 0.93. This analysis highlights the superior performance of the combined and MedAE methods in distinguishing adversarial samples from normal ones under FGSM attack conditions.

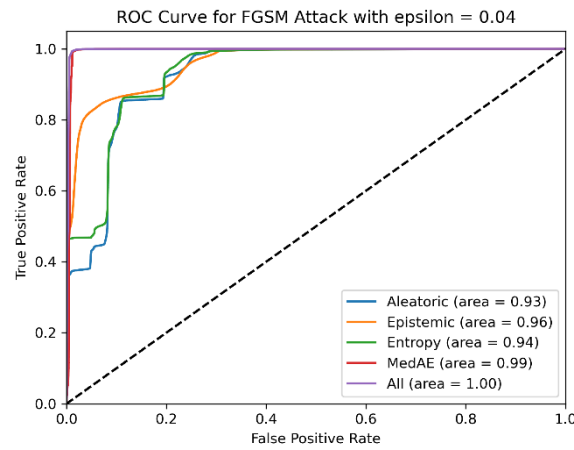


Figure 4.11. ROC curve of CSE-CIC-IDS2018 dataset under FGSM attack with 0.04 perturbation amount

Although adversarial attacks may share the same epsilon value, their impact can differ significantly. This variation depends on whether the target classes are similar or distinct, as highlighted in the study by Wang et al. (2022). Consequently, the attack's success rate was used as a baseline metric when assessing the effects of attacks. A performance loss caused by the attacks in the intrusion detection system (IDS) is quantified as the attack's success rate.

As the success of the attack increases, its detectability also improves. At an epsilon value of 0.04, the FGSM attack demonstrated slightly higher detection rates compared to the BIM and PGD attacks, although their detection outcomes were closely matched. This

difference arises because iterative attacks like BIM and PGD generate higher impact using smaller perturbations over multiple iterations, whereas FGSM applies all perturbations in a single step, potentially making it slightly easier to detect.

Table 4.7 presents the ROC-AUC scores of the proposed detection system when tested on the CSE-CIC-IDS2018, KDDCup99, and ROSIDS23 datasets under FGSM, BIM, and PGD attacks across various perturbation levels. For the CSE-CIC-IDS2018 dataset, attack success rates ranged from 24% to 99%, while detection scores consistently remained between 96% and 100%. In contrast, for the KDDCup99 dataset, attack success rates were lower at lower perturbation levels (1% to 50%) but increased to 63% at higher levels, with detection scores remaining steady between 92% and 100%. The ROSIDS23 dataset exhibited a broader range of attack success rates (40% to 97%), with detection scores ranging from 84% to 100%.

Higher perturbation levels generally correspond to increased attack success rates. For example, in the CSE-CIC-IDS2018 dataset, FGSM attack success rate from 24% at a perturbation level of 0.01 to 98% at 0.1. Similarly, in the ROSIDS23 dataset, FGSM success rates increased from 41% to 87% within the same range. Despite these increases in attack success rates, detection scores remained robust across all datasets. Notably, the KDDCup99 dataset showed comparatively lower attack success rates at smaller perturbations (1% to 50%) but aligned with the other datasets at higher perturbations, achieving up to 63%. Despite the rising success rates of adversarial attacks, the detection system maintained high ROC-AUC scores across all datasets, demonstrating its resilience and effectiveness in detecting attacks across varying levels of intensity. These findings underscore the robustness of the detection system, which consistently achieves high detection scores even as attack success rates rise with increasing perturbation levels.

Figures 4.12 and 4.13 show the outcomes of the three adversarial attacks, presenting detection scores relative to attack success rates. The figures compare the performance of the proposed detection method with the approach by Tuna et al. (2022) in terms of attack success rate and ROC-AUC scores across different epsilon levels. For both FGSM and PGD attacks, the proposed method consistently achieves higher ROC-AUC scores, indicating better detection capability.

Figure 4.12 shows the successful detection of adversarial attacks and their impact on the IDS comparative analysis with the study Tuna et.al. (2022). The proposed method starts above 90% detection performance for all three adversarial attacks and quickly approaches near 100%, even at lower attack success rates, demonstrating successful detection. Tuna et al.'s method also shows high performance, closely matching the proposed approach. The proposed method achieves higher scores, particularly at lower attack accuracies.

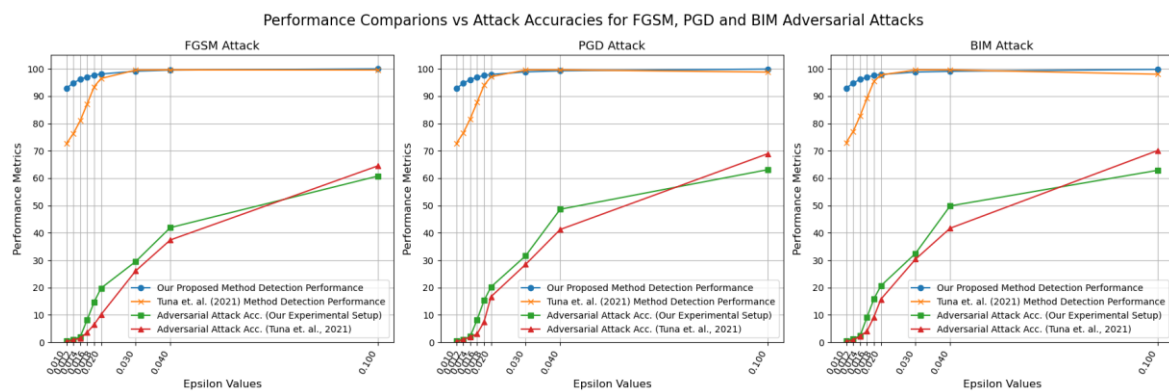


Figure 4.12. Performance comparison of the proposed method and Tuna et. al. (2022) at adversarial attack detection ROC-AUC scores

Figure 4.13 shows the success rate of adversarial attacks on the x-axis and the detection rate of the models on the y-axis. The figure highlights the relationship between the epsilon values used in adversarial attacks and the effectiveness of detection methods, categorized by each type of attack. Both models exhibit similar trends in the attack success rates of deceiving IDSs. The graph demonstrates that as the perturbation level increases, the attacks' impact becomes more significant. Notably, even with minimal perturbations, the proposed detection methods outperform those of Tuna et al. (2022), achieving detection rates exceeding 90%. Furthermore, as the attack severity escalates, the detection accuracy of both methods approaches nearly 100%.

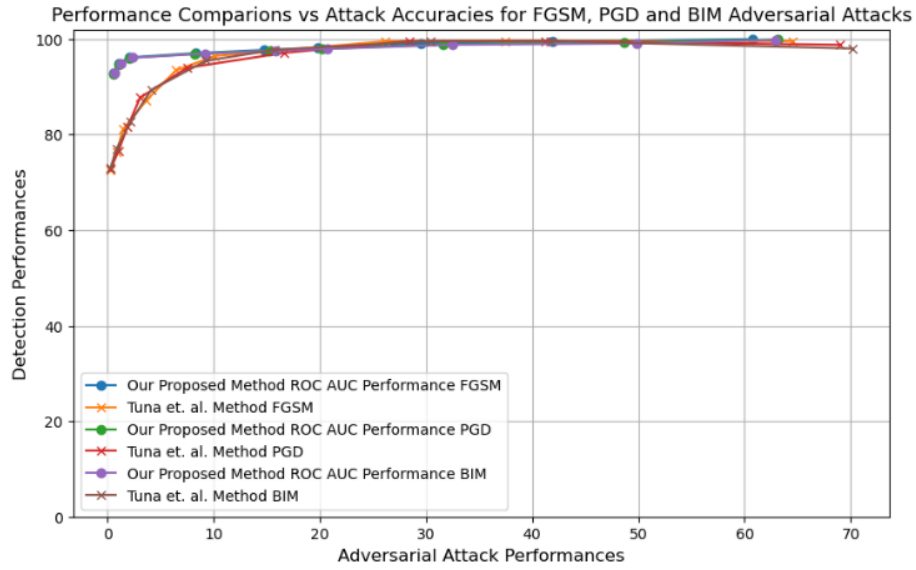


Figure 4.13. Comparative evaluation of adversarial attack detection performance on FGSM, PGD, and BIM for the proposed method and Tuna et al. (2022)

The READ system's performance is compared with the adversarial sample detection method proposed by Peng et al. (2020) referred to as ASD, as shown in Table 4.8. The evaluation metrics used for this comparison are Accuracy and F1-Score. Under FGSM attacks, the ASD method enhances system performance by improving accuracy from 61.56% to 73.41%. In contrast, this thesis's results demonstrate a significant improvement with the READ method, achieving an accuracy increase from 57.84% to 99.69%. This highlights READ's superior performance compared to ASD under FGSM attacks.

Similarly, for PGD attacks, ASD increases system accuracy from 55.81% to 82.2%. However, the READ method achieves a more substantial improvement, with accuracy rising from 51.17% to 98.49%. Once again, READ outperforms ASD under PGD attacks. Overall, the READ system demonstrates its effectiveness by consistently outperforming ASD in both accuracy and F1-Score for detecting adversarial attacks of both types. Adversarial attacks with low-level perturbations pose significant challenges for ASD detection. The ASD approach, which depends on analyzing reconstruction errors and matching losses, struggles to identify such attacks effectively. Minimal perturbations, especially during the initial phases of adversarial attacks, create only slight modifications intended to mislead the IDS's decision-making process. This results in reduced discrepancies between the original input data and the reconstructed data, limiting ASD's ability to detect adversarial samples accurately. In contrast, the proposed READ method leverages MedAE for calculating

reconstruction errors, providing a more effective solution for detecting adversarial samples, even in cases of minimal perturbation. However, this enhanced detection accuracy comes at the cost of increased computational overhead for the detection system.

Table 4.8. Proposed detection system comparison for FGSM and PGD attacks

| <b>Attack Type</b> | <b>Adversarial<br/>Detection Model</b> | <b>Accuracy</b> | <b>F1-Score</b> |
|--------------------|----------------------------------------|-----------------|-----------------|
| <b>FGSM</b>        | <b>None (Peng et. al. (2020))</b>      | 61,56%          | 59,10%          |
|                    | <b>ASD (Peng et. al. (2020))</b>       | 73,41%          | 84,46%          |
|                    | <b>None</b>                            | 57,84%          | 25,06 %         |
|                    | <b>READ</b>                            | <b>99,69%</b>   | <b>99,66%</b>   |
| <b>PGD</b>         | <b>None (Peng et. al. (2020))</b>      | 55,81%          | 55,69%          |
|                    | <b>ASD (Peng et. al. (2020))</b>       | 82,27%          | 89,52%          |
|                    | <b>None</b>                            | 51,17%          | 22,81%          |
|                    | <b>READ</b>                            | <b>98,49%</b>   | <b>98,71%</b>   |

## 5. CONCLUSION AND RECOMMENDATIONS

In this thesis, the focus has been on adversarial attack detection on IDS. The subject has been examined from three distinct perspectives. (1) Vulnerability surfaces against adversarial attacks in the life cycle of ML have been studied, considering both examples from the literature and potential vulnerabilities. (2) Attack applications in the literature have been reviewed according to their placement within the cycle and the ROSIDS23 dataset is proposed. (3) The Reconstruction-based adversarial attack detection (READ) approach is proposed.

The vulnerability surface, as introduced in this thesis, expands the traditional concept of the attack surface to include both known and potential vulnerabilities across the entire machine learning lifecycle. This broader framework provides a comprehensive perspective on how adversarial attacks exploit weaknesses, from data collection to retraining, compromising the integrity and robustness of AI systems. By mapping these vulnerabilities and their associated attack vectors, the thesis contributes to a deeper understanding of adversarial threats and emphasizes the importance of developing defenses that address both existing and future challenges.

Another contribution of this thesis is the ROSIDS23 dataset, an extensive dataset specifically designed for robotic systems' cybersecurity research, particularly focusing on ROS-based cyber-attacks. The dataset includes benign and various attack traffic data obtained from ROS middleware using the tcpdump network protocol analyzer. With eighty-two network traffic features extracted using the CICFlowMeter tool, this dataset provides a rich, high-fidelity resource in CSV format that can aid in developing security countermeasures for robotic systems. The dataset includes both raw and processed data, capturing detailed records across multiple attack types: unauthorized publish, unauthorized subscribe, subscriber flood, and DoS, with benign instances. By supporting robust analysis of adversarial and benign traffic patterns, ROSIDS23 is valuable for advancing IDS in robotic systems. It serves as a critical resource for understanding attack behaviors and testing detection strategies, making it essential for the security of ROS-based infrastructure.

The READ system is a Reconstruction-based Adversarial Detection approach designed to identify adversarial attacks with high precision. READ enhances the ability to differentiate between adversarial and original samples by leveraging three metrics previously used in the literature for adversarial attack detection. These metrics are combined with the reconstruction error rate metric for improved detection accuracy. Although high success rates starting from 0.04 were achieved in the literature, success rates exceeding 92% were observed in detecting attacks in low-impact ranges. By adding the three metrics from the literature to this proposed metric, the success rate was increased to 100% with computational complexity. By integrating READ in front of the IDS, the robustness and effectiveness of the IDS have been significantly enhanced. In scenarios where adversarial attacks occur, the accuracy of the IDS can decrease, falling to as low as 51%. However, with the implementation of READ, the accuracy of the IDS is substantially improved, reaching levels as high as 98%. This demonstrates that READ effectively mitigates the negative impact of adversarial attacks, ensuring a much higher detection accuracy and overall reliability of the IDS. The experimental results highlight the importance of incorporating multiple metrics to enhance the detection capabilities of IDSs against adversarial threats. High detection success rates are achieved with READ at high perturbation levels. Furthermore, the novelty of the proposed method of detecting low-level perturbations is a significant advancement over existing methods in the literature.

The current detection system's dependency on the last hidden layer of the data model necessitates access to the data, which might not always be feasible or secure. Future studies, therefore, focus on creating detection systems that operate independently of the data layer, ensuring broader applicability and enhanced security. In the proposed approach, MedAE is used to detect smaller perturbations successfully. While MSE is computationally efficient to calculate, MedAE is more effective at capturing small perturbations. Therefore, future studies could explore the combined use of MSE and MedAE, which may enhance detection performance while optimizing computational efficiency. As a future study, the relationship between the effects of attacks on datasets and data distribution can be examined, and detection system updates can be suggested. Future studies could focus on further developing and expanding the concept of the vulnerability surface introduced in this thesis. In particular, examining weaknesses in the machine learning lifecycle. Additionally, analyzing next-

generation attack vectors targeting these vulnerabilities could provide insights into the limitations of current security measures and guide the development of defense strategies.



## REFERENCES

- Abdulhammed, R., Musafer, H., Alessa, A., Faezipour, M., & Abuzneid, A. (2019). Features dimensionality reduction approaches for machine learning based network intrusion detection. *Electronics*, 8(3), 322.
- Abou Khamis, R., & Matrawy, A. (2020). *Evaluation of adversarial training on different types of neural networks in deep learning-based idss*. 1–6.
- Afzal, A., Le Goues, C., & Timperley, C. S. (2022). Mithra: Anomaly Detection as an Oracle for Cyberphysical Systems. *IEEE Transactions on Software Engineering*, 48(11), 4535–4552.
- Ahmed, M. E., & Kim, H. (2017). *Poster: Adversarial examples for classifiers in high-dimensional network data*. 2467–2469.
- Alotaibi, A., & Rassam, M. A. (2023). Adversarial machine learning attacks against intrusion detection systems: A survey on strategies and defense. *Future Internet*, 15(2), 62.
- Ashmore, R., Calinescu, R., & Paterson, C. (2021). Assuring the machine learning lifecycle: Desiderata, methods, and challenges. *ACM Computing Surveys (CSUR)*, 54(5), 1–39.
- Ayub, M. A., Johnson, W. A., Talbert, D. A., & Siraj, A. (2020). *Model evasion attack on intrusion detection systems using adversarial machine learning*. 1–6.
- Balsa-Comerón, J., Guerrero-Higueras, Á. M., Rodríguez-Lera, F. J., Fernández-Llamas, C., & Matellán-Olivera, V. (2018). *Cybersecurity in autonomous systems: Hardening ROS using encrypted communications and semantic rules*. 67–78.
- Bao, Z., Muñoz-González, L., & Lupu, E. C. (2018). Mitigation of Adversarial Attacks through Embedded Feature Selection. *arXiv Preprint arXiv:1808.05705*.
- Bjarnadottir, S., Li, Y., & Stewart, M. G. (2019). Chapter Nine—Climate Adaptation for Housing in Hurricane Regions. In E. Bastidas-Arteaga & M. G. Stewart (Eds.), *Climate Adaptation Engineering* (pp. 271–299). Butterworth-Heinemann. <https://doi.org/10.1016/B978-0-12-816782-3.00009-7>
- Carlini, N., & Wagner, D. (2017). *Towards evaluating the robustness of neural networks*. 39–57.

## REFERENCES

- Chaitou, H., Robert, T., Leneutre, J., & Pautet, L. (2021). *Assessing adversarial training effect on IDSs and GANs*. 543–550.
- Chatterjee, A., & Ahmed, B. S. (2022). IoT anomaly detection methods and applications: A survey. *Internet of Things*, 19, 100568.
- Chauhan, R., & Heydari, S. S. (2020). *Polymorphic Adversarial DDoS attack on IDS using GAN*. 1–6.
- Chen, J., Wu, D., Zhao, Y., Sharma, N., Blumenstein, M., & Yu, S. (2021). Fooling intrusion detection systems using adversarially autoencoder. *Digital Communications and Networks*, 7(3), 453–460.  
<https://doi.org/10.1016/j.dcan.2020.11.001>
- Chen, S.-T., Cornelius, C., Martin, J., & Chau, D. H. (2019). *Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector*. 52–68.
- Cisse, M. M., Adi, Y., Neverova, N., & Keshet, J. (2017). Houdini: Fooling deep structured visual and speech recognition models with adversarial examples. *Advances in Neural Information Processing Systems*, 30.
- Cort J. Willmott & Kenji Matsuura. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, 30(1), 79–82.
- Cup, K. (1999). [Http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html](http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html). *The UCI KDD Archive*.
- Degirmenci, E. (2023). *ROSIDS23 [Dataset]*. Zenodo.  
<https://doi.org/10.5281/zenodo.10014434>
- Değirmenci, E., Kirca, Y. S., Yolaçan, E. N., & Yazıcı, A. (2023). An analysis of DoS attack on robot operating system. *Gazi University Journal of Science*, 1–1.
- Değirmenci, E., Kirca, Y. S., Özçelik, İ., & Yazıcı, A. (2023). ROSIDS23: Network intrusion detection dataset for robot operating system. *Data in Brief*, 51, 109739.

## REFERENCES

- Der Kiureghian, A., & Ditlevsen, O. (2009). Aleatory or epistemic? Does it matter? *Structural Safety*, 31(2), 105–112.
- Dhinakaran, N., & Anto, S. (2023). Defenses for Adversarial Attacks in Network Intrusion Detection System – A Survey. *International Journal of Computing and Digital Systems*, 13(1), 1287–1299. <https://doi.org/10.12785/ijcds/1301105>
- Droos, A. (2023). *Resilient Intrusion Detection System for Adversarial Attacks on Low-Rate DDoS*. <https://doi.org/10.21203/rs.3.rs-3314942/v1>
- Duy, P. T., Khoa, N. H., Nguyen, A. G.-T., & Pham, V.-H. (2021). DIGFuPAS: Deceive IDS with GAN and function-preserving on adversarial samples in SDN-enabled networks. *Computers & Security*, 109, 102367.
- Duy, P., Tien, L., Khoa, N., Hien, D., Nguyen, A., & Pham, V. (2021). DIGFuPAS: Deceive IDS with GAN and function-preserving on adversarial samples in SDN-enabled networks. *COMPUTERS & SECURITY*, 109. <https://doi.org/10.1016/j.cose.2021.102367>
- Feinman, R., Curtin, R. R., Shintre, S., & Gardner, A. B. (2017a). Detecting adversarial samples from artifacts. *arXiv Preprint arXiv:1703.00410*.
- Feinman, R., Curtin, R. R., Shintre, S., & Gardner, A. B. (2017b). Detecting adversarial samples from artifacts. *arXiv Preprint arXiv:1703.00410*.
- Fernández Muro, B. (2018). *Securing communications in surgery robots*.
- Fu, X., Zhou, N., Jiao, L., Li, H., & Zhang, J. (2021). The robust deep learning-based schemes for intrusion detection in internet of things environments. *Annals of Telecommunications*, 76(5), 273–285.
- G. Apruzzese, M. Colajanni, & M. Marchetti. (2019). Evaluating the effectiveness of Adversarial Attacks against Botnet Detectors. *2019 IEEE 18th International Symposium on Network Computing and Applications (NCA)*, 1–8. <https://doi.org/10.1109/NCA.2019.8935039>
- Gao, R., Liu, F., Zhang, J., Han, B., Liu, T., Niu, G., & Sugiyama, M. (2020). Maximum mean discrepancy is aware of adversarial attacks. *arXiv Preprint arXiv:2010.11415*.

## REFERENCES

- Ghamizi, S., Cordy, M., Gubri, M., Papadakis, M., Boystov, A., Le Traon, Y., & Goujon, A. (2020). *Search-Based Adversarial Testing and Improvement of Constrained Credit Scoring Systems* (P. Devanbu, M. Cohen, & T. Zimmermann, Eds.; WOS:000744432100090; pp. 1089–1100). <https://doi.org/10.1145/3368089.3409739>
- Ghurab, M., Gaphari, G., Alshami, F., Alshamy, R., & Othman, S. (2021). A detailed analysis of benchmark datasets for network intrusion detection system. *Asian Journal of Research in Computer Science*, 7(4), 14–33.
- Gong, Z., & Wang, W. (2023). *Adversarial and clean data are not twins*. 1–5.
- Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014a). Explaining and harnessing adversarial examples. *arXiv Preprint arXiv:1412.6572*.
- Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014b). Explaining and harnessing adversarial examples. *arXiv Preprint arXiv:1412.6572*.
- Goyal, P., & Goyal, A. (2017). *Comparative study of two most popular packet sniffing tools-Tcpdump and Wireshark*. 77–81.
- Han, K., Li, Y., & Xia, B. (2021). A Cascade Model-Aware Generative Adversarial Example Detection Method. *TSINGHUA SCIENCE AND TECHNOLOGY*, 26(6), 800–812. <https://doi.org/10.26599/TST.2020.9010038>
- Hartl, A., Bachl, M., Fabini, J., & Zseby, T. (2020). *Explainability and adversarial robustness for rnns*. 148–156.
- Hora, S. C. (1996). Aleatory and epistemic uncertainty in probability elicitation with an example from hazardous waste management. *Reliability Engineering & System Safety*, 54(2–3), 217–223.
- Huang, J., Erdogan, C., Zhang, Y., Moore, B., Luo, Q., Sundaresan, A., & Rosu, G. (2014). *ROSRV: Runtime verification for robots*. 247–254.

## REFERENCES

- Huang, J., Li, Y.-F., & Xie, M. (2015). An empirical analysis of data preprocessing for machine learning-based software cost estimation. *Information and Software Technology*, 67, 108–127.
- Hüllermeier, E., & Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3), 457–506.
- K. He, D. D. Kim, & M. R. Asghar. (2023). Adversarial Machine Learning for Network Intrusion Detection Systems: A Comprehensive Survey. *IEEE Communications Surveys & Tutorials*, 25(1), 538–566.  
<https://doi.org/10.1109/COMST.2022.3233793>
- KDD Cup 1999. (2007). [Dataset].  
<http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>.
- Kendall, A., & Gal, Y. (2017). What uncertainties do we need in bayesian deep learning for computer vision? *Advances in Neural Information Processing Systems*, 30.
- Kilincer, I. F., Ertam, F., & Sengur, A. (2021). Machine learning methods for cyber security intrusion detection: Datasets and comparative study. *Computer Networks*, 188, 107840.
- Kirca, Y. S., Degirmenci, E., Demirci, Z., Yazici, A., Ozkan, M., Ergun, S., & Kanak, A. (2023). Runtime verification for anomaly detection of robotic systems security. *Machines*, 11(2), 166.
- Kirca, Y. S., Değirmenci, E., Yazıcı, A., & Özkan, M. (2022). ROS based attack tool for verification of robotic system security. 15–18.
- Kuppa, A., & Le-Khac, N.-A. (2021). Adversarial xai methods in cybersecurity. *IEEE Transactions on Information Forensics and Security*, 16, 4924–4938.
- Kurakin, A., Goodfellow, I., & Bengio, S. (2016). Adversarial machine learning at scale. *arXiv Preprint arXiv:1611.01236*.
- Lahlali, M., Berbiche, N., & El Alami, J. (2021). How Enterprise must be Prepared to be “AI First”? *INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS*, 12(5), 346–351.

## REFERENCES

- Lashkari, A. H., Seo, A., Gil, G. D., & Ghorbani, A. (2017). *CIC-AB: Online ad blocker for browsers*. 1–7.
- Lashkari, A. H., Zang, Y., Owhuo, G., Mamun, M., & Gil, G. (2017). *CICFlowMeter*. *GitHub*. [Vid. 2021-08-10]. Dostupné z: <https://github.com/AhLashkari/CICFlowMeter/blob/master/ReadMe.txt>.
- Li, H., Ota, K., & Dong, M. (2018). Learning IoT in edge: Deep learning for the Internet of Things with edge computing. *IEEE Network*, 32(1), 96–101.
- Ma, X., Niu, Y., Gu, L., Wang, Y., Zhao, Y., Bailey, J., & Lu, F. (2021). Understanding adversarial attacks on deep learning based medical image analysis systems. *Pattern Recognition*, 110, 107332.
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017a). Towards deep learning models resistant to adversarial attacks. *arXiv Preprint arXiv:1706.06083*.
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017b). Towards deep learning models resistant to adversarial attacks. *arXiv Preprint arXiv:1706.06083*.
- Maruyama, Y., Kato, S., & Azumi, T. (2016). *Exploring the performance of ROS2*. 1–10.
- Moosavi-Dezfooli, S.-M., Fawzi, A., & Frossard, P. (2016). *Deepfool: A simple and accurate method to fool deep neural networks*. 2574–2582.
- Narayanan, V., & Bobba, R. B. (2018). *Learning based anomaly detection for industrial arm applications*. 13–23.
- Özçelik, İ., & Brooks, R. (2020). *Distributed denial of service attacks: Real-world detection and mitigation*. CRC Press.
- Pal, S. K., & Mitra, S. (1992). *Multilayer perceptron, fuzzy sets, classification*.
- Papernot, N., Faghri, F., Carlini, N., Goodfellow, I., Feinman, R., Kurakin, A., Xie, C., Sharma, Y., Brown, T., & Roy, A. (2016). Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv Preprint arXiv:1610.00768*.

## REFERENCES

- Pawlicki, M., Choraś, M., & Kozik, R. (2020). Defending network intrusion detection systems against adversarial evasion attacks. *Future Generation Computer Systems*, 110, 148–154.
- Pedraza, A., Deniz, O., & Bueno, G. (2020). Approaching adversarial example classification with chaos theory. *Entropy*, 22(11), 1201.
- Peng, Y., Fu, G., Luo, Y., Hu, J., Li, B., & Yan, Q. (2020). *Detecting adversarial examples for network intrusion detection system with gan*. 6–10.
- Piplai, A., Chukkapalli, S. S. L., & Joshi, A. (2020). *NAttack! Adversarial Attacks to bypass a GAN based classifier trained to detect Network intrusion*. 49–54.
- Plósz, S., Schmittner, C., & Varga, P. (2017). *Combining safety and security analysis for industrial collaborative automation systems*. 187–198.
- Qayyum, A., Usama, M., Qadir, J., & Al-Fuqaha, A. (2020). Securing connected & autonomous vehicles: Challenges posed by adversarial machine learning and the way forward. *IEEE Communications Surveys & Tutorials*, 22(2), 998–1026.
- Rivera, S., Lagraa, S., & State, R. (2019). ROSploit: Cybersecurity Tool for ROS. *2019 Third IEEE International Conference on Robotic Computing (IRC)*, 415–416. <https://doi.org/10.1109/IRC.2019.00077>
- Sagduyu, Y. E., Shi, Y., & Erpek, T. (2019). Adversarial deep learning for over-the-air spectrum poisoning attacks. *IEEE Transactions on Mobile Computing*, 20(2), 306–319.
- Saha, S., Kumar, A., Sahay, P., George, J., Kruthiventi, S. S. S., & Muralidhara, H. (2019). *Attack Agnostic Statistical Method for Adversarial Detection*. 798–802. <https://doi.org/10.1109/iccvw.2019.00107>
- Sánchez-González, J.-M., Rocha-de-Lossada, C., & Flikier, D. (2020). Median absolute error and interquartile range as criteria of success against the percentage of eyes within a refractive target in IOL surgery. *Journal of Cataract & Refractive Surgery*, 46(10), 1441.
- Sarkar, D., Vinod, P., & Yerima, S. Y. (2020). *Detection of Tor traffic using deep learning*. 1–8.

## REFERENCES

- Sauka, K., Shin, G.-Y., Kim, D.-W., & Han, M.-M. (2022). Adversarial robust and explainable network intrusion detection systems based on deep learning. *Applied Sciences*, 12(13), 6451.
- Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3), 379–423.
- Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1, 108–116.
- Shi, Y., Zeng, H., & Nguyen, T. T. (2019). *Adversarial Machine Learning for Network Security*. 1–7.
- Siddiqi, M. A., & Pak, W. (2020). Optimizing filter-based feature selection method flow for intrusion detection system. *Electronics*, 9(12), 2114.
- Siriweera, A., & Naruse, K. (2021). Survey on cloud robotics architecture and model-driven reference architecture for decentralized multicloud heterogeneous-robotics platform. *IEEE Access*, 9, 40521–40539.
- Staffa, M., Mazzeo, G., & Sgaglione, L. (2018). *Hardening ROS via hardware-assisted trusted execution environment*. 491–494.
- Su, J., Vargas, D. V., & Sakurai, K. (2019). One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5), 828–841.
- Sun, D., Yang, K., Shi, Z., & Chen, C. (2017). *A new mimicking attack by LSGAN*. 441–447.
- Tseng, T. W., Wu, C. T., & Lai, F. (2019). Threat analysis for wearable health devices and environment monitoring internet of things integration system. *IEEE Access*, 7, 144983–144994.
- Tuna, O. F., Catak, F. O., & Eskil, M. T. (2022). Closeness and uncertainty aware adversarial examples detection in adversarial machine learning. *Computers and Electrical Engineering*, 101, 107986.

## REFERENCES

- Uelwer, T., Michels, F., & De Candido, O. (2021). *Learning to Detect Adversarial Examples Based on Class Scores*. 233–240.
- Vidal, J. M., Monge, M. A. S., & Monterrubio, S. M. M. (2020). Espada: Enhanced payload analyzer for malware detection robust against adversarial threats. *Future Generation Computer Systems*, 104, 159–173.
- Wang, M., Lu, H., Liu, S., & Zhu, Z. (2020). How to mislead AI-assisted network automation in SD-IPoEONs: A comparison study of DRL-and GAN-based approaches. *Journal of Lightwave Technology*, 38(20), 5574–5585.
- Wang, N., Chen, Y., Xiao, Y., Hu, Y., Lou, W., & Hou, Y. T. (2022). Manda: On adversarial example detection for network intrusion detection system. *IEEE Transactions on Dependable and Secure Computing*, 20(2), 1139–1153.
- Wang, W., & Lu, Y. (2018). *Analysis of the mean absolute error (MAE) and the root mean square error (RMSE) in assessing rounding model*. 324, 012049.
- Warzyński, A., & Kołaczek, G. (2018). *Intrusion detection systems vulnerability on adversarial examples*. 1–4.
- White, R., Christensen, D., Henrik, I., & Quigley, D. (2016). SROS: Securing ROS over the wire, in the graph, and through the kernel. *arXiv Preprint arXiv:1611.07060*.
- Xiang, Z., Miller, J., & Kesidis, G. (2021). Reverse engineering imperceptible backdoor attacks on deep neural networks for detection and training set cleansing. *COMPUTERS & SECURITY*, 106. <https://doi.org/10.1016/j.cose.2021.102280>
- Xu, H., Ma, Y., Liu, H.-C., Deb, D., Liu, H., Tang, J., & Jain, A. K. (2020). Adversarial Attacks and Defenses in Images, Graphs and Text: A Review. *International Journal of Automation and Computing*, 17(2), 151–178. <https://doi.org/10.1007/s11633-019-1211-x>
- Xu, W., Evans, D., & Qi, Y. (2017). Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv Preprint arXiv:1704.01155*.
- Yan, Q., Wang, M., Huang, W., Luo, X., & Yu, F. R. (2019). Automatically synthesizing DoS attack traces using generative adversarial networks. *International Journal of Machine Learning and Cybernetics*, 10(12), 3387–3396.

## REFERENCES

- Yao, S., Zhao, Y., Zhang, A., Hu, S., Shao, H., Zhang, C., Su, L., & Abdelzaher, T. (2018). Deep learning for the internet of things. *Computer*, 51(5), 32–41.
- Ye, H., & Liu, X. (2022). Feature autoencoder for detecting adversarial examples. *International Journal of Intelligent Systems*, 37(10), 7459–7477.
- Zhang, H., Cheng, N., Zhang, Y., & Li, Z. (2021). Label flipping attacks against Naive Bayes on spam filtering systems. *Applied Intelligence*, 51(7), 4503–4514.
- Zhang, S., Zhang, C., & Yang, Q. (2003). Data preparation for data mining. *Applied Artificial Intelligence*, 17(5–6), 375–381.
- Zhang, W., Ma, Z. M., Das, S., Weng, T.-W. L., Megretski, A., Daniel, L., & Nguyen, L. M. (2024). One Step Closer to Unbiased Aleatoric Uncertainty Estimation. *Proceedings of the Aaai Conference on Artificial Intelligence*, 38(15), 16857–16864. <https://doi.org/10.1609/aaai.v38i15.29627>
- Zhao, S., Li, J., Wang, J., Zhang, Z., Zhu, L., & Zhang, Y. (2021). attackGAN: Adversarial Attack against Black-box IDS using Generative Adversarial Networks. *2020 International Conference on Identification, Information and Knowledge in the Internet of Things, IIKI2020*, 187, 128–133. <https://doi.org/10.1016/j.procs.2021.04.118>