UNIVERSITY OF THE WEST OF ENGLAND

DOCTORAL THESIS

Methods for Improving Robustness against Adversarial Machine Learning Attacks

Author:

Andrew MCCARTHY

Supervisory Team:

Prof. Phil LEGG

Dr. Panagiotis ANDRIOTIS

Dr. Essam GHADAFI

Prof. Larry BULL

A thesis submitted in partial fulfilment of the requirements of the University of the West of England, Bristol for the degree of Doctor of Philosophy

in the

Computer Science Research Centre

School of Computing and Creative Technologies

This research programme was carried out in collaboration with Techmodal.

August 10, 2023

Declaration of Authorship

I, Andrew McCarthy, declare that this thesis titled, "Methods for Improving Robustness against Adversarial Machine Learning Attacks" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: A. McCarthy

Date: 10/8/2023

Word Count: 38,752

Publications

The work contained in this thesis is based on the following publications:

- A. McCarthy, P. Andriotis, E. Ghadafi and P. Legg, "Defending against Adversarial Machine Learning Attacks using Hierarchical Learning: A case study on Network Traffic Attack Classification". Journal of Information Security and Applications. 2023, 72, 103398. https://doi.org/10.1016/j.jisa.2022.103398

- A. McCarthy, P. Andriotis, E. Ghadafi and P. Legg, "Functionality-Preserving Adversarial Machine Learning for Robust Classification in Cybersecurity and Intrusion Detection Domains: A Survey". Journal of Cybersecurity and Privacy, 2022, 2(1), 154-190. https://doi.org/10.3390/jcp2010010

- A. McCarthy, P. Andriotis, E. Ghadafi and P. Legg, "Feature Vulnerability and Robustness Assessment against Adversarial Machine Learning Attacks". 2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), 2021, pp. 1-8. https://doi.org/10.1109/CyberSA52016.2021.9478199.

"The sky is not the limit ... There are footprints on the Moon!"

Dr. Buzz Aldrin

Abstract

Andrew McCarthy

Methods for Improving Robustness against Adversarial Machine Learning Attacks

Machine learning systems can improve the efficiency of real-world tasks, including in the cyber security domain; however, these models are susceptible to adversarial attacks; indeed, an arms race exists between adversaries and defenders. The benefits of these systems have been accepted without fully considering their vulnerabilities, resulting in the deployment of vulnerable machine learning models in adversarial environments. For example, intrusion detection systems are relied upon to accurately discern between malicious and benign traffic but can be fooled into allowing malware onto a networks. Robustness is the stability of performance in well-trained models facing adversarial examples. This thesis tackles the urgent problem of improving the robustness of machine learning models, enabling safer deployments in adversarial domains. The logical outputs of this research are countermeasures against adversarial examples. Original contributions to knowledge are: a survey of adversarial machine learning in the cyber security domain, a generalizable approach for feature vulnerability and robustness assessment, and a constraint-based method of generating transferable functionality-preserving adversarial examples in an intrusion detection domain. Novel defences against adversarial examples are presented: Feature selection with recursive feature elimination, and hierarchical classification. Machine learning classifiers can be used in both visual and non-visual domains. Most research in adversarial machine learning considers the visual domain. A primary focus of this work is how adversarial attacks can be effectively used in non-visual domains, such

as cyber security. For example, attackers may exploit weaknesses in an intrusion detection system classifier, enabling an intrusion to masquerade as benign traffic. Easily fooled systems are of limited use in critical areas such as cyber security. In future, more sophisticated adversarial attacks could be used by ransomware and malware authors to evade detection by machine learning Intrusion Detection Systems.

Experiments in this thesis focus on intrusion detection case studies and use Python code and Python libraries: the CleverHans API, and the Adversarial Robustness Toolkit libraries to generate adversarial examples, and the HiClass library to facilitate Hierarchical Classification. An adversarial arms race is playing out in cyber security. Every time defences are improved, adversaries find new ways to breach networks. Currently, one of the most critical holes in defences are adversarial examples. This thesis examines the problem of robustness against adversarial examples for machine learning systems and contributes novel countermeasures, aiming to enable the deployment of machine learning in critical domains.

Acknowledgements

I am fortunate to have received generous support throughout my research. I would first like to thank my supervisory team: Prof. Phil Legg, Dr. Panos Andriotis, and Dr. Essam Ghadafi, and of course Prof. Larry Bull. Thanks for your confidence and trust in me. Phil, thank you for encouraging me in the first instance to apply for the PhD, and for your continuous support and gentle nudges along the way. Panos, I wish you well in your new post at the University of Birmingham. Thank you for your guidance, your much appreciated support was unwavering throughout. Essam, I must also wish you well in your new post at the University of Newcastle. Your feedback has greatly improved the quality of my work. I am grateful to have had your input and support. Larry, although you were not regularly at my supervision meetings, you are always a friendly face around the Computer Science Research Centre. I feel truly fortunate to have worked with you. I must also thank and acknowledge Dr. Theodoros Spyridopoulos who encouraged me in the early stages of this work.

My devoted thanks to Emma for her encouragement, extraordinary support and understanding. I am very lucky to have such a wonderfully caring and supportive partner. I am also very grateful for my close-knit family who though far away are always supportive of my endeavours. Thanks to my mother, Monica, and my siblings Nicola, Rachel, and Simon. I am immensely grateful for your support throughout. I hope to see you all for many more happy get-togethers.

I appreciate all my friends. Thank you for your patience while I have needed to focus on my academic work. I promise that I will soon be available for social events again.

I also wish to take a moment to thank my fellow researchers and friends who listened to me and suggested practical solutions. Particular thanks to: Gwyn Wilkinson, Ryan Fellows, James Barrett, Sadegh Bahmohabbat Chafjiri, Peter Mayhew, Nathan Duran, Nathan Renney, Harri Renney, and Sinclair-Emmanuel Smith.

I am grateful to all the staff at UWE, and particularly the vibrant research communities of the Computer Science Research Centre and the Cyber Security and Cyber Crime Group. I very much enjoyed all the thought-provoking discussions and free exchange of ideas. Special thanks also to Helen Frisby and the Research, Business, and Innovation Team for their excellent Research Skills Development Programme which has undoubtedly helped me develop as a researcher.

Much of the work on this thesis was completed during the global Covid-19 pandemic. I vividly recall near empty campus during the summer months. I wish to extend my gratitude to Craig Duffy and Ben Gaster whose friendly but brief and socially distanced conversations and encouragement helped anchor me during those especially difficult months.

I would like to express my thanks for the financial support from my sponsor Techmodal. Particular thanks are due to David Evans and Daniel Jones who provided feedback on my early research ideas.

Next, I would like to thank the many researchers who offered advice throughout the thesis. Dr. Arash Habibi Lashkar (Canadian Institute for Cybersecurity (CIC)) for answering my questions on CICFlowMeter. Thanks are also due to lecturers and colleagues for their support, and informative discussion. In particular: Richard McClatchey for his academic writing guidance, and Hisham Ihshaish for his lively discussions on machine learning.

During my full-time PhD I have also been fortunate to work on projects with skilled researchers. Thanks to the all the GRAIMATTER team and especially: Emily Jefferson; Jim Smith; Richard Preen; Felix Ritchie; Alberto Blanco Justicia Alba Crespi; Simon Rogers; Christian Cole; James Liley. Thanks also to the Trimetis team, in particular Craig Williams, Nicola Turner, and Alastair Vincent.

Thanks to all the members of the University of the West of England Doctoral Society (UWE Doc Soc). I am thankful to have the support of so many fellow postgraduate researchers.

A brief note of thanks to those researchers in the field whose tireless work has informed this research. I cite you all in the bibliography. If I have seen further, it is because I was standing on your collective shoulders.

Contents

List of Figures

List of Tables

List of Abbreviations

AD	Anomaly Detection
Adam	Adaptive Moments
AE	Adversarial Example
AI	Artificial Intelligence
AIA	Attribute Inference Attack
ANN	Artificial Neural Network
API	Application Programming Interface
APT	Advanced Persistent Threat
AUC	Area Under Curve
CFA	Cuttlefish Algorithm
CNN	Convolutional Neural Network
COCO	Common Objects in Context
CPS	Cyber Physical Systems
DDoS	Distributed Denial of Service
DL	Deep Learning
DoS	Denial of Service
DT	Decision Tree
FGSM	Fast Gradient Descent Method
FPR	False Positive Rate
GA	Genetic Algorithm
GAN	Generative Adversarial Network
HBBC	Histogram Based Boosting Classifier
ICS	Industrial Control System
IDS	Intrusion Detection System
IoT	Internet of Things

JSMA	Jacobian Saliency Map Attack
KNN	K-Nearest Neighbour
L-BFGS	Limited-memory Broyden-Fletcher-Goldfarb-Shanno
LCPL	Local Classifier Per Level
LCPN	Local Classifier Per Node
LCPPN	Local Classifier Per Parent Node
LGBM	Light Gradient Boosting Machine
LR	Logistic Regression
LSTM	Long Short Term Memory
MIA	Membership Inference Attack
ML	Machine Learning
MLP	Multi Layer Perceptron
MQTT	Message Queuing Telemetry Transport
NB	Naïve Bayes
NIDS	Network Intrusion Detection System
NLP	Natural Language Processing
PE	Portable Executable
PSO	Particle Swarm Optimization
QDA	Quadratic Discrimination Analysis
RF	Random Forest
RFE	Recursive Feature Elimination
RNN	Recurrent Neural Network
RMSProp	Root Mean Squared Propagation
ROC	Receiver Operator Characteristic
SGD	Stochastic Gradient Descent
SVC	Support Vector Classification
TPR	True Positive Rate
UAP	Universal Adversarial Perturbation
XGBoost	eXtreme Gradient Boosting
ZOO	Zeroth Order Optimization

List of Symbols

Symbol	Name
ϵ	Epsilon: FGSM parameter controlling perturbation size.
γ	Gamma: JSMA parameter controlling proportion of features.
θ	Theta: JSMA paramater controlling the perturbation size.
$L0$	Number of altered pixels.
$L2$	Euclidean distance (root-mean-square).
$L\infty$	Maximum change to any of the coordinates.

For all those who want to make

better decisions…

Chapter 1

Introduction

Cyber security and the protection of associated computer and network systems is fundamental for most organizations. The recent Cyber Security Breaches survey conducted by the UK Government in 2022 found that 39% businesses had experienced a cyber-attack in the last 12 months. Across organisations reporting material losses the average estimated cost of a cyber-attack was £4,200. The cost increases to £19,400 when considering only medium and large businesses. Moreover, the full cost of cyber-attacks may be under reported because organisations find it harder to consider indirect costs. [1]. The sheer scale and magnitude of modern cyber-attacks requires automated response and intervention. Machine learning (ML) is becoming widely used for the detection and classification of malicious network activity to aid the response to cyber-attacks [2][3][4][5], where a mathematical model is learned to relate input feature observations to a set of possible output classes. For the classification of network traffic attacks, input features may be derived from the observed network communications and packet header information, which may be indicative of either benign traffic, or a malicious attack such as a Denial of Service, a Remote Access Trojan, a BotNet, or other network-based attack.

Whilst machine learning can help manage this wealth of information, it is not without limitation. Recent years have seen a growing interest in the domain of adversarial machine learning [6] that seeks to identify well-crafted examples that knowingly force misclassification by the model. This has been particularly effective in the computer vision domain since the manipulation of few input features (i.e.,

image pixels) may inadvertently adjust the performance of the model without being noticeable to the human observer, due to small perturbations of pixel intensity values. A fundamental challenge in adversarial learning is to determine which features are most susceptible such that a minimal change can result in misclassification by the model, whilst the overall input to the model appears unchanged or unaltered to the human observer. For example, a parallel can be drawn to the challenge of network traffic classification, where a malicious attack should exhibit the same characteristics such that the activity is still deemed malicious, whilst identifying the minimal amount of perturbation in the derived features such that the model believes the observation is benign, hence resulting in misclassification. This characteristic is hence referred to as *functionality preservation*.

1.1 What is Machine Learning?

Machine learning is a branch of artificial intelligence that uses the statistical properties of a set of observations to learn a function that maps an input to the most probable or desirable output. This enables a computer model to learn without being specifically programmed. The field of machine learning is partitioned into four main types of machine learning: supervised, unsupervised, semi-supervised, and reinforcement learning. In supervised learning data values and labels are provided through finding patterns in the data a function is generated that maps the input values to the most probable label. In semi-supervised learning only some of the data values have labels. Unsupervised learning involves the analysis of data without predefined labels, where patterns are inferred to classify samples into distinct clusters. Reinforcement learning can model a set of states and actions. Given an input state the model learns the most beneficial action by maximising a reinforcement or reward. The major applications for each type of machine learning are shown in Table 1.1.

One popular definition of machine learning, by Tom Mitchell [7] follows:

> A computer program is said to learn from experience E with respect to some class of tasks T, and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Type of Machine Learning	Major Applications
Supervised	Intrusion Detection; Image and Object Recognition; Identifying and filtering Spam and Malware.
Semi-Supervised	Speech Recognition; Text and Document Classification; Self Training.
Unsupervised	Anomaly Detection; Clustering; Visualisation; Dimensionality Reduction.
Reinforcement Learning	Autonomous Robotics; Autonomous Agents; Recommendation Systems.

TABLE 1.1: Types of Machine Learning and their Major Applications.

Classification tasks examine the data values of previously unseen data and predict the most likely class. Machine learning classification has been applied to a range of tasks. For example, identifying handwritten digits, detecting fraudulent credit card activity, determining whether a tumour is benign or malignant based on medical imagery, the identification of spam e-mails and text messages. Machine learning has become widely adopted as a strategy for dealing with a variety of cyber security issues. Cyber security domains particularly suited to ML include: intrusion detection and prevention [8], network traffic analysis [9], malware analysis [10] [11], user behaviour analysis [12], insider threat detection [13], social engineering detection [14], spam detection [15], detection of malicious social media usage [16], health misinformation [17], climate misinformation [18], and more generally "Fake News" [19]. These are essentially classification problems.

The performance of a model can be evaluated through a range of metrics, common metrics are: accuracy, precision, recall, and F1-Score. Classification accuracy works well only if there are equal number of samples belonging to each class. Precision measures the number of class predictions that actually belong to that class. Recall measures the number of class predictions made out of all examples of the class in the dataset. F1-Score is the harmonic mean of Precision and Recall. The F1-score is the harmonic mean of precision and recall, taking both metrics into account. It is considered a more robust metric because it punishes extreme values. When considering classification tasks, it is important to consider imbalanced datasets and the problems of over-fitting to the training data.

There are many types of classification models ranging from simple to more complex. Here the most commonly used classification models are introduced. Logistic Regression uses a logistic function to model the probability of a binary outcome. Naive Bayes is a probabilistic model relying on Bayes' Theorem to predict the probability of a class given a set of features. Decision trees are a hierarchical model that splits the data into increasingly smaller subsets based on the features. Random Forest is an ensemble model composed of multiple decision trees. Support Vector Machines separate data into classes by finding the optimal hyperplane between two classes in high-dimensional space. k-Nearest Neighbours classifies data based on the majority class of its k nearest neighbours. Neural Networks use a set of interconnected neurons to learn underlying patterns in the data. Logistic Regression, and Support Vector Classifier (SVC) are typically suited to binary classification problems. Whereas Decision Trees, Random Forests, k-Nearest Neighbours, and Neural Networks (NN) are also suited to multi-class problems.

Dhar *et al.* [20] note that few studies analyze the complexity of models and the associated trade-offs between accuracy and complexity. The complexity of an algorithm is often expressed in *Big-O notation*. They review models, noting that the number of features and activations affects both memory usage and computational complexity. Moreover, they argue that accuracy alone cannot justify the choice of model type, particularly in regard to Deep Neural Networks (DNN); however, the risks involved for inaccurate predictions will vary across domains. In security domains, correct classification may be considered critical, possibly assuaging concerns regarding computational complexity of models.

1.2 What are Adversarial Examples and Adversarial Machine Learning?

In essence adversarial examples are carefully crafted to cause a machine learning model to make a mistake. Adversarial machine learning is a research field that aims to make machine learning models more robust and secure against adversarial

examples. Researchers in adversarial machine learning aim to understand how machine learning models can be attacked and develop defences against attacks. Despite society's reliance and trust in ML systems, the inherent nature of machine learning, learning to identify patterns, is in itself a potential attack vector for adversaries wishing to circumvent ML-based system detection processes.

In order to introduce the field more fully, a mathematical definition of machine learning models is now presented. Papernot *et al.* [21] state that most ML models can be described mathematically as functions $h_0(x)$ with an input x and parameterized by a vector $\theta \in \Theta$. Here Θ refers to the set of all possible configurations of a given model. For example, consider a linear regression model where Θ is all possible real numbers for the slope and intercept of the linear equation. When considering a specific neural network with a fixed number of layers and neurons Θ describes the range of possible weights. It should be noted that some models such as K Nearest Neighbor are non-parametric. The output of the function $h_0(x)$ is the model's prediction of some property of interest for the given input x. The input x is usually represented as a vector of values called features. The space of functions $h(x) = x \mapsto h_0(x)|\theta \in \Theta$ defines the set of candidate hypotheses. In supervised learning, the parameters are adjusted to align the model predictions $h_0(x)$ with the expected output y. This is achieved by minimizing a loss function that captures the dissimilarity of $h_0(x)$ and the corresponding y. Model performance must be validated against a separate validation dataset to confirm if the model also generalizes well for unseen data. Classification ML systems find a function (f) that matches a vector ($\vec{x}$) to its corresponding class (y).

Szegedy *et al.* [22] discovered naturally occurring and intriguing properties of neural networks, discovering that imperceptible perturbations of the input values can cause differences in the output of the neural network. Machine learning systems are therefore vulnerable to carefully crafted noise, known as adversarial examples [23]. Adversarial examples are problematic for many ML algorithms and models including Random Forests (RF) and Naive Bayes (NB) classifiers.

Adversarial examples were first discovered on Artificial Neural Networks (ANN) which are known to be highly susceptible to adversarial examples. Deep Neural Networks (DNN) are a type of Artificial Neural Network with a number of hidden layers. DNNs are particularly effective at handling complex classification tasks, such as intrusion detection. Artificial Neural Networks, especially Deep Neural Networks are a primary focus of this work because they are used for cyber security tasks and are highly susceptible to adversarial examples. Improving robustness in these models could have significant impact. Artificial Neural Networks are inspired by the network of neurons in the human brain. ANNs are useful because they can generalize from a finite set of examples, essentially mapping a large input space (infinite for continuous inputs) to a range of discrete outputs. Unfortunately, in common with other ML algorithms, neural networks are vulnerable to attacks using carefully crafted perturbations to inputs, including evasion and poisoning attacks. In recent work, carefully crafted inputs described as "adversarial examples" are considered possible in ANN because of these inherent properties that exist within neural networks [22], such as:

1. The semantic information of the model is held across the model and not localised to specific neurons.

2. Neural Networks learn input-output mappings that are discontinuous (and discontiguous).

These properties mean that even extremely small perturbations of an input could cause a neural network to provide a misclassified output. Given that neural networks have these properties, it is reasonable to expect that biological neural networks will also suffer misclassifications and/or to have evolved mitigations. Human brains are more complex than current artificial neural networks yet suffer a type of misclassification (illusory perception), in the form of face pareidolia [24] [25]. This strengthens the case that the properties of neural networks are the source of adversarial examples (AE) in neural networks. In cyber security related domains, it has been seen how adversaries exploit adversarial examples, using carefully-crafted noise to evade detection through misclassification [26] [27].

1.3 Trustworthiness of Machine Learning Systems

The previous sections have introduced machine learning and the problem of adversarial examples. As machine learning systems are used in increasingly diverse areas including defence, health, and government. It is important that these systems can be trusted. Hankin and Barrèere [28] note that there are many aspects to trustworthiness. Their work focuses on cyber security of industrial control systems (ICS). The focus of this work is similarly on cyber security; that is that a system should be protected against adversaries. Other interrelated aspects of trustworthiness include safety, reliability, and correctness. Correctness is determined by verifying the systems against formal specifications. Although these aspects can be considered independently there is a large intersection. In relation to computer science, safety and security are distinct considerations. Safety is often established via rigorous formal methods and validation processes; however, a system designed to be safe may notwithstanding be *insecure* and susceptible to security vulnerabilities. Trustworthiness encompasses many aspects, as shown in Figure 1.1 adapted from [28].

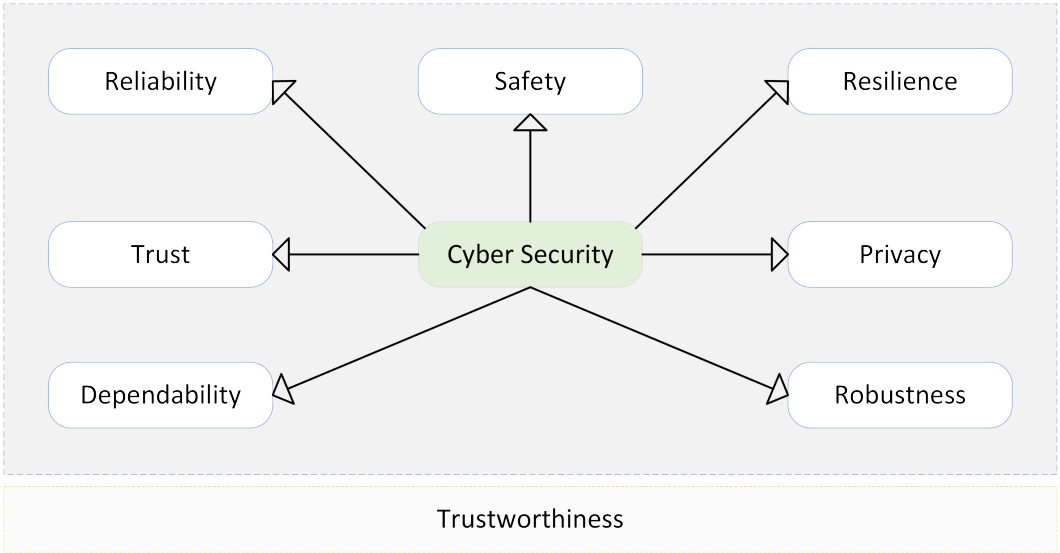


FIGURE 1.1: The role of cyber security for trustworthy systems.

If machine learning systems are to be useful, they must be able to be trusted. Unfortunately, adversaries ranging from solo hackers to state-sponsored Advanced Persistent Threats (APTs) have interest in attacking these systems. Successful

attacks against neural networks mean that unprotected systems are vulnerable and therefore dangerously deployed in application domains.

Incorrect classification of network traffic could allow malware onto public and private networks. Moreover, the increasing size of data being processed by machine learning systems enlarges the attack surface available to attackers while obfuscating the attack to humans. If these vulnerabilities are unaddressed future mature attack methods will facilitate more destructive attacks. There is an urgent need for research in this area. One key area of research is that of robustness. The robustness of a well-trained model relates to its performance when facing adversarial examples. That is, the ability of a model to continue to make accurate predictions in the presence of adversarial examples. This thesis explores the robustness of machine learning systems, and in particular neural networks, aiming to understand the principles behind successful attacks and consider mitigations in key domains of network intrusion detection. Improving robustness of machine learning systems enables safer deployment across a wider range of domains.

1.4 Definitions

It is appropriate to clarify some of the concepts and terms by defining some key terms in Table 1.2 (adapted from [29]).

1.5 Aim of the Research

The aim of this research is to investigate to what extent machine learning systems can be compromised and which sensible methods can improve robustness in machine learning systems. This thesis demonstrates vulnerabilities of machine learning systems and how they might be mitigated. A recognized trade-off exists between accuracy and robustness of machine learning systems. This research aims to discover ways to improve the robustness of machine learning models whilst

Metric	Definition	Source
Dependability	The ability to avoid service failures that are more frequent and more severe than is acceptable.	[30]
Privacy	Assurance that the confidentiality of, and access to, certain information about an entity is protected.	[31]
Safety	Absence of catastrophic consequences on the user(s) and the environment.	[30]
Security	Making all aspects of a computing system, including both physical and cyber systems, free from danger or threats by preserving availability, integrity, and confidentiality.	[32]
Reliability	Continuity of correct service.	[30]
Resilience	Ability to withstand system degradation by reducing the duration and magnitude of disruptions and by recovering a normal, functional system state persistently within acceptable delay and cost.	[33]
Robustness	The performance of well-trained models facing adversarial examples.	[34]
Trust	Subjective belief that a trustee will behave as a trustor expected when taking risk under uncertainty based on the cognitive assessment of past interactions with the trustee.	[35][36]

TABLE 1.2: Definitions of Key Terms in Cyber security for Trustworthy Systems.

retaining acceptable accuracy. Sensible and scalable defences are considered that could be used to mitigate the effects of adversarial examples.

1.5.1 Research Questions

The research questions are as follows:

- RQ1: To what extent can adversarial attacks influence the output of machine learning systems for intrusion detection systems?
- RQ2: To what extent can countermeasures and defensive approaches mitigate the effects of adversarial examples for intrusion detection systems?
- RQ3: To what extent can attacks and defences be generalized to other scenarios, datasets, and data types?

1.5.2 Research Objectives

The Research Objectives are:

- RO1a: Demonstrate that adversarial examples are applicable to domains other than the visual domain and in particular can be applied to intrusion detection domain. The effect of adversarial examples, and proposed countermeasures will be principally determined through measuring and comparing the generalisation-error robustness metric using F1-Score, and accuracy of the machine learning models under normal conditions and under adversarial example conditions. This relates to RQ1.

- RO1b: Demonstrate that the effect of adversarial examples can be mitigated through the use of feature selection. The effect of mitigations is determined principally through measuring the unmitigated accuracy of a model under adversarial example conditions and examining the difference between the accuracy achieved by a model with applied mitigations. This relates to RQ1.

- RO2 Evaluate current research and provide future research directions or research challenges. This will be measured and achieved through the collection of relevant papers, and the production of a survey paper discussing research challenges for adversarial example attacks and defences including the application in the cyber security domain. Through the analysis of the literature, the aim is to find answers and information relevant to RQ1 and RQ2.

- RO3: Assess the robustness of neural networks used in intrusion detection domains. This will be measured through analysis of a range of experiments using appropriate intrusion detection datasets. These experiments will help answer RQ1 and RQ2.

- RO4: Assess how well attacks and defences generalize to other scenarios, datasets and datatypes. This relates to RQ3

- RO5: Formulate generalizable recommendations to improve robustness against adversarial attacks. This will be measured by the production of a list

of recommendations drawn from this research. This objective relates to all the research Questions.

1.6 Research Strategy

This section details the research strategy. The research questions are addressed by designing and executing experiments, in order to create new knowledge. The experiments examine the effect on robustness metrics, typically F1-Score. The hypothesized conditions are isolated, and subsequently the results are compared with the experimental control conditions and results. Experiments and case studies are used to effect the research objectives and ultimately answer the Research Questions.

1.6.1 Experiments

A selection of experiments is executed to determine the robustness of machine learning models against adversarial examples. All experiments are developed in Python with the following libraries: CleverHans, Adversarial Robustness Toolbox, HiClass.

The experiments and hypotheses are:

- Feature Vulnerability and Robustness Assessment against Adversarial Machine Learning Attacks using Recursive Feature Elimination (RFE) in Chapter 3.

  H_0: Deselecting features using RFE does not improve accuracy under adversarial examples.

  H_1: Deselecting features using RFE improves accuracy under adversarial examples.

- Defending against adversarial machine learning attacks using hierarchical learning: A case study on network traffic attack classification in Chapter 4

  H_0: Hierarchical Classifiers do not improve robustness as measured by F1-Score.

  H_1: Hierarchical Classifiers improve robustness as measured by F1-Score.

1.6.2 Case Study

This thesis employs the use of intrusion detection case studies. The author's choice to use intrusion detection case studies is logical and appropriate. It is considered that an intrusion detection system is likely the initial entry point to a system and the first barrier that adversaries will encounter. The research uses the CICIDS2017 [37] dataset and derivatives [38] to evaluate the use of machine learning models for Intrusion Detection.

1.7 Value of the Research

Machine learning systems are impressive and are improving the efficiency of real-world tasks. Machine learning is increasingly supporting much technological progress. Indeed, machine learning algorithms are present in almost every aspect of modern living. Slowly but surely, machine learning systems are increasingly used to assist in making decisions in critical areas including hospitals [39] and the criminal justice system. To what extent should the output of a machine learning system be trusted? Unfortunately, as previously discussed machine learning systems are vulnerable to adversarial examples. There is an urgent need for research on the susceptibility of machine learning systems to adversarial attacks. Researchers and policy makers cannot neglect the risks and implications of expanding use of machine learning systems into adversarial domains. This chapter introduced how machine learning systems are vulnerable. Attackers wish to exploit these vulnerabilities while defenders aim to secure against the vulnerabilities. In this way, an adversarial arms race exists between adversaries and defenders. The recent SolarWinds supply chain attack [40] [41] identified in December 2020 indicates the reliance that organisations have on intrusion detection software, and the presence of Advanced Persistent Threats with the expertise and resources to attack organisations' network defences. Adversarial machine learning is a critical area of research. If not addressed, there is increasing potential for novel attack strategies that seek to exploit the inherent weaknesses present within machine learning models. For this reason, this research addresses the issues related to the

robustness of machine learning models against adversarial attacks across the cyber security domain, where problems of functionality-preservation are recognized. Although a case study of a network-based intrusion detection system (NIDS) is used, these issues might be applicable in other areas where ML systems are used such as industrial control systems and cyber-physical systems.

In the domain of network traffic analysis, it is recognized that adversaries need to evade detection methods. A suitable network firewall will reject adversarial traffic and malformed packets while accepting legitimate traffic. Therefore, successful adversarial examples should be crafted to comply with domain constraints such as those related to the transmission control protocol/internet protocol (TCP-IP) stack. Moreover, adversaries wish to preserve the functionality of their attacks. A successful attack should not lose functionality at the expense of evading a classifier.

There is an arms race in cyber security where adversaries, including well-resourced Advanced Persistent Threats may eventually overcome most security systems. This thesis contributes to the arms race, forcing adversaries to expend more time and resources to attack machine learning systems.

1.8 Research Context and Contributions

The motivations of this research and the resultant thesis stem back to my Master's thesis as a requisite part of completing my M.Sc. in Cyber Security at UWE in 2019. In this research I explore the realm of adversarial machine learning where *carefully-crafted* noise is added to inputs. Such inputs are known as 'adversarial examples' and can cause misclassification. Thus began my deep dive into both the broad concepts and the precise technical details. The fact that machine learning systems are susceptible to adversarial examples raises broad concerns about the security, trustworthiness, and suitability of machine learning in adversarial environments. These concerns will increase with greater and wider deployment of

machine learning systems. This thesis explores the implications of adversarial machine learning, the main contributions are:

- A survey of adversarial machine learning in the cyber security domain.
- A generalizable approach for assessing feature vulnerability and robustness.
- A constraint-based method of generating transferable functionality-preserving adversarial examples in an intrusion detection domain.
- A novel defence against adversarial examples employing feature selection and Recursive Feature Elimination.
- A novel defence against adversarial examples employing Hierarchical classification.

1.9 Thesis Outline

The thesis is structured as follows:

- Chapter 2 presents a systematic literature survey of adversarial machine learning with a focus on the relatively new research topic of functionality-preserving adversarial examples in the cyber security and intrusion detection domains. Attacks and current defences are examined. The contents of this chapter form the work published in the Journal of Cyber security and Privacy [42].
- Chapter 3 presents my ablation study research toward identifying and mitigating the vulnerability of machine learning systems. I demonstrate a generalizable approach to assess the vulnerability and robustness of features in a machine learning context. The chosen approach systematically eliminates vulnerable features whilst maintaining acceptable classification accuracy. The contents of this chapter are published and were presented at the 2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA) [43].

- Chapter 4 presents research mitigating the effects of adversarial examples and proposes a hierarchical learning approach as a defensive strategy independent of attack algorithm. The results reveal that hierarchies can help models perform better under adversarial conditions, in comparison to their equivalent flat model. The contents of this chapter are published in the Journal of Information Security And Applications [44].

- Chapter 5 further explores adversarial machine learning and how this work generalizes to other scenarios, conditions, settings, and situations.

- Chapter 6 concludes this thesis, providing: A description of new knowledge gained, an evaluation of how far each research question has been answered, appropriate recommendations. The research challenges and future work are discussed, and finally an overall evaluation of the thesis is provided.

Chapter 2

Literature Review

Following the introduction of the security and robustness of machine learning in Chapter 1, this chapter now presents a review of the existing literature. This literature review examines the main issues surrounding adversarial machine learning for robust classification in cyber security and intrusion detection domains. The possible attacks and current defences are examined. The literature in this area is large and rapidly increasing. Special care was taken to define the scope and eligibility criteria for inclusion. By exploring the literature, a significant contribution is made to this research, informing the experiments detailed in the following chapters.

2.1 Introduction

Machine Learning has become widely adopted as a strategy for dealing with a variety of cyber security issues. Cyber security domains particularly suited to ML include: intrusion detection and prevention [8], network traffic analysis [9], malware analysis [10] [11], user behaviour analytics [12], insider threat detection [13], social engineering detection [14], spam detection [15], detection of malicious social media usage [16], health misinformation [17], climate misinformation [18], and more generally "Fake News" [19]. These are essentially classification problems. Recall from Chapter 1 that adversaries wish to exploit adversarial examples by using carefully-crafted noise to evade detection through misclassification. Defenders wish to deny or detect adversarial examples, while adversaries wish to remain undetected and evade defences.

In this way, an adversarial arms race exists between adversaries and defenders. The recent SolarWinds supply chain attack [40] [41] identified in December 2020 indicates the reliance that organisations have on intrusion detection software, and the presence of Advanced Persistent Threats (APTs) with the expertise and resources to attack organisations' network defences. Adversarial machine learning is a critical area of research. If not addressed, there is increasing potential for novel attack strategies that seek to exploit the inherent weaknesses that exist within machine learning models. For this reason, this survey addresses the issues related to the robustness of machine learning models against adversarial attacks across the cyber security domain, where problems of functionality-preservation are recognized. While a case study of a network-based intrusion detection system was used, these issues might be applicable in other areas where ML systems are used. The focus is on papers detailing adversarial attacks and defences. Attacks are further classified by attack type, attack objective, domain, model, knowledge required, and constraints. Defences are further categorised by defence type, domain, and model. In the domain of network traffic analysis, adversaries need to evade detection methods. A suitable network firewall will reject adversarial traffic and malformed packets while accepting legitimate traffic. Therefore, successful adversarial examples must be crafted to comply with domain constraints such as those related to the transmission control protocol/internet protocol (TCP-IP) stack. Moreover, adversaries wish to preserve the functionality of their attacks. A successful attack must not lose functionality at the expense of evading a classifier. The essence of a simple adversarial attack is that a malicious payload evades detection by masquerading as benign. This thesis refers to this characteristic as *functionality-preserving*.

Compared to domains such as computer vision whereby the image modification is only to fool human vision sensors, adversarial attacks in other domains are significantly more challenging to fool both a human and/or system-based sensor. Critically ML systems are increasingly trusted within cyber physical systems [45] such as power stations, factories, and oil and gas industries. In such complex physical environments, the potential damage that could be caused by a vulnerable system might even be life threatening [46]. Despite ML systems

being relied upon and trusted, the inherent nature of machine learning - learning to identify patterns - is in itself a potential attack vector for adversaries wishing to circumvent ML-based system detection processes.

The major contributions of this chapter are:

- A survey of the literature is conducted to identify the trends and characteristics of published works on adversarial learning in relation to cyber security, addressing both attack vectors and defensive strategies.
- The issue of functionality-preservation in adversarial learning is addressed in contrast to domains such as computer vision. To preserve functionality a malformed input must suitably fool a system process as well as a human user such that the original functionality is maintained despite some modification.
- This relatively new research domain is summarised to address the future research challenges associated with adversarial machine learning across the cyber security domain.

The remainder of this chapter is structured as follows: Section 2.2 provides an overview of other important surveys; Section 2.3 discusses background material; Section 2.4 details the literature survey; Section 2.5 details the results; Section 2.6 provides a discussion and conclusion, summarises the findings, and identifies research challenges.

2.2 Related Works

Corona *et al.* [47] provide a useful overview of intrusion detection systems. They predict greater use of machine learning for intrusion detection and call for further investigation into adversarial machine learning. This chapter now considers a number of related academic surveys that have been presented in the last 5 years with a focus on adversarial examples, security, and intrusion detection.

2.2.1 Secure and Trustworthy Systems

Machine learning systems are used in increasingly diverse areas including those of cyber security. Trust in these systems is essential. Hankin and Barrèere [28] note that there are many aspects to trustworthiness: reliability, trust, dependability, privacy, resilience, and safety. Adversaries ranging from solo hackers to state-sponsored APTs have an interest in attacking these systems. Successful attacks against machine learning models mean that systems are vulnerable and therefore potentially dangerously deployed in cyber security domains. Cho *et al.* [29] propose a framework considering the security, trust, reliability and agility metrics of computer systems; however, they do not specifically consider adversarial machine learning, or robustness to adversarial examples.

2.2.2 Adversarial ML in General

Papernot *et al.* [21] note that the security and privacy of ML is an active but nascent area of research. In this early work, they systematize their findings on security and privacy in machine learning. They note that a science for understanding many of the vulnerabilities of ML and countermeasures is slowly emerging. They analyse ML systems using the classical confidentiality, integrity, and availability (CIA) model. They analyse: training in adversarial settings; inferring adversarial settings; robust, fair, accountable, and private ML models. Through their analysis, they identify a total of 8 key takeaways that point towards two related notions of sensitivity. The sensitivity of learning models to their training data is essential to privacy preserving ML, and similarly the sensitivity to inference data is essential to secure ML. Central to both notions of sensitivity is the generalization error (i.e., the gap between performance on training and test data). They focus on attacks and defences for machine learning systems and hope that understanding the sensitivity of modern ML algorithms to the data they analyse will foster a science of security and privacy in machine learning. They argue that the generalization error of models is key to secure and privacy-preserving ML.

Zhang and Li [48] discuss opportunities and challenges arising from adversarial examples. They introduce adversarial examples and survey state-of-the-art adversarial example generation methods, and defences before raising future research opportunities and challenges. They note three challenges for the construction of adversarial examples:

1. The difficulty of building a generalizable method.
2. The difficulty in controlling the size of perturbation (too small will not result in adversarial examples, and too large can easily be perceived).
3. Difficulty in maintaining adversarial stability in real-world applications (some adversarial examples do not hold for transformations such as blurring).

They identify two challenges for defence against adversarial examples. Firstly, black-box attacks do not require knowledge of the model architecture and therefore cannot be easily resisted by modifying the model architecture or parameters. Secondly, defences are often specific to an attack method and are less suitable as a general defence. Defences against one attack method do not easily defend against adversarial examples based on other methods for generating adversarial examples. They subsequently identify three opportunities:

1. Construction of adversarial examples with high transferability (high confidence).
2. Construction of adversarial examples without perturbing the target image, they suggest that perturbation size will affect the success rate and transferability of adversarial examples.
3. Considering and modelling physical transformations (translation, rotation, brightness, and contrast).

Their focus is on the visual domain, and they do not specifically discuss IDS or functionality-preserving adversarial attacks.

Apruzzese *et al.* [49] examine adversarial examples and consider realistic attacks, highlighting that most literature considers adversaries with complete knowledge about the classifier and are free to interact with the target systems. They further emphasize that few works consider 'realizable' perturbations that take account of domain and/or real-world constraints. There is perhaps a perception that the threat from adversarial attacks is low based on the assumption that much prior knowledge of the system is required. This approach has some merit; however, this could be an over-confident position to take. Their idea that realistically the adversary has less knowledge of the system conflicts with Shannon's maxim [50] and Kerckhoff's second cryptographic principle [51] which states that the fewer secrets the system contains, the higher its safety. The pessimistic 'complete knowledge' position is often used in cryptographic studies, in cryptographic applications it is considered safe because it is a bleak expectation. This expectation is also realistic since well-resourced adversaries may be expected to eventually discover or acquire all details of the system. Many adversarial example papers assume complete knowledge, this is however unlikely to always be the case. Perhaps leading some to believe models are more secure against adversarial examples. However, the transferability property of adversarial examples means that complete knowledge is not required for successful attacks, and black-box attacks are possible with no prior knowledge of machine learning models. An adversary may only learn through interacting with the model. Therefore, the level of knowledge required by an adversary must be accounted for, including white-box, black-box, and gray-box knowledge paradigms.

2.2.3 Intrusion Detection

Wu *et al.* [52] consider several types of deep learning systems for network attack detection, including supervised and unsupervised models, and they compare the efficiency and effectiveness of different attack detection methods using two intrusion detection datasets: "KDD Cup 99" dataset and an improved version known as NSL-KDD [53] [54]. These two datasets have been used widely in the past by academic researchers; however, they do not fairly represent modern network traffic analysis problems due to concept-drift. Networks have increasing

numbers of connected devices, increasing communications per second, and new applications using the network. The use of computer networks and the internet has changed substantially in twenty years. The continued introduction of IPv6, Network address Translation, Wi-Fi, mobile 5G networks, and cloud providers has changed network infrastructure [55]. Furthermore, the internet is now increasingly used for financial services. Akamai [56] report financial services now see millions or tens of millions of attacks each day. These attacks were less common twenty years ago. Furthermore, social media now constitutes much internet traffic and most social media platforms were founded after the KDD Cup 99 and NSL-KDD datasets were introduced. For example, Facebook, YouTube, and Twitter were founded in 2004, 2005, and 2006 respectively. This limits the validity of some research using outdated datasets. Therefore, research should use modern datasets that represent modern network traffic.

Kok *et al.* [57] analyse intrusion detection systems (IDS) that use a machine learning approach. They specifically consider the datasets used, the ML algorithms, and the evaluation metrics. They warn that some researchers are still using datasets first introduced decades ago (e.g., KDD Cup 99, NSL-KDD). They warn that this trend could result in no or insufficient progress on IDS. This would ultimately lead to the untenable position of obsolete IDS while intrusion attacks continue to evolve along with user behaviour and the introduction of new technologies. Their paper does not consider adversarial examples or robustness of ML models. Alatwi and Morisset [58] tabulate a list of Network Intrusion datasets in the literature that is extended in Table 2.1.

Work	Dataset	Network	Year	Attack Categories
[53]	KDD Cup 99	Traditional	1999	DoS, Probe, User 2 Root and Remote to User
[59]	NSL-KDD	Traditional	2009	DoS, Probe, User 2 Root and Remote to User
[60]	DARPA	Traditional	2009	DDoS, Malware, Spambots, Scans, Phishing
[61]	CTU-13	Traditional	2011	Botnet
[62]	Kyoto	Traditional	2015	Botnet
[63]	UNSW-NB15	Traditional	2016	Backdoors, Fuzzers, DoS, Generic, Shell code, Reconnaissance, Worms, Exploits, Analysis
[64]	WSN-D5	Wireless	2016	Greyhole, Blackhole, Scheduling, Flooding.
[65]	SDN Traffic	SDN	2016	DDoS
[66]	CICIDS2017	Traditional	2017	DoS, DDoS, SSH-Patator, Web, PortScan, FTP-Patator, Bot.
[67]	Mirai	IoT	2017	Botnet
[66]	CICIDS2018	Traditional	2018	Bruteforce Web, DoS, DDoS, Botnet, Infiltration.
[66]	CICDDoS2019	Traditional	2019	DDoS
[68]	Bot-IOT	IoT	2018	DDoS, DoS, OS Service Scan, Keylogging, Data exfiltration
[69]	Kitsune	IoT	2018	Recon, Man in the Middle, DoS, Botnet Malware
[70]	IEEE BigData Cup	Traditional	2019	N/A
[71]	MQTT-IOT-IDS2020	IoT	2020	Aggressive scan, UDP scan, Sparta SSH brute-force, and MQTT brute-force attack.
[72]	HIKARI-2021	IoT	2021	Brute force attack, Brute force attack (XMLRPC), Vulnerability, probing, Synthetic Traffic

TABLE 2.1: Datasets used in the literature.

2.3 Background

Here, further background is provided on some key concepts that are related to adversarial learning, to support the reader of this survey. This background material covers the topics of model training, robustness, common adversarial example algorithms, adversary capabilities, goals, and attack methods.

2.3.1 Model Training

It is important to consider the dataset on which models are trained, because the trustworthiness and quality of a model is impacted by the distribution, quality, quantity, and complexity of dataset training samples [73]. Biased models are more susceptible to adversarial examples. Therefore, models must be trained on unbiased training data; although Johnson *et al.* consider the *absolute* number of training samples may be more important than the ratio of class imbalance [74]. For example, a small percentage of a large number of samples is sufficient to train a model regardless of high class imbalance (e.g., 1% malicious samples in 1 million network flows yields 10,000 samples). Unfortunately, cyber security datasets are often prone to bias, in part because of limited samples of some malicious traffic (e.g., *zero-day* attacks) and large amounts of benign traffic. Sheatsley *et al.* [75] state biased distributions enable successful adversarial examples with the modification of very few features.

Resampling

Three common data-level techniques tackle biased datasets by resampling:

- Oversampling: Random samples of minority classes are duplicated until the bias of majority classes is compensated.
- Undersampling: Random samples from the majority class are discarded until the bias of majority classes is compensated.
- Hybrid Sampling: Combines modest oversampling of minority classes and modest undersampling of majority classes aiming to give better model performance than applying either technique alone.

Algorithm-level techniques tackling dataset bias commonly employ cost-sensitive learning where a class penalty or weight is considered, or decision thresholds are shifted to reduce bias [74].

Loss Functions

When training a model, the goal is to minimize the loss function through use of an optimizer which adjusts the weights at each training step. Common optimizers include stocastic gradient descent (SGD), Adaptive Moments (Adam), and Root Mean Squared Propagation (RMSProp). Commonly a regularizer is employed during training to ensure the model generalizes well to new data. A dropout layer is often employed as a regularizer.

The loss function must be chosen carefully: for binary classification binary_crossentropy (Eq. 2.1) is usual; for multiclass classification problems categorical_crossentropy (Eq. 2.2) or mean_squared_error (Eq. 2.3) are suitable.

$$f_{\text{binary\_crossentropy}}(y) = -y_i^{label}\, log\, y_i i^{prediction} - \left(1 - y_i^{label}\right)\, log\, \left(1 - y_i i^{prediction}\right) \tag{2.1}$$

$$f_{\text{categorical\_crossentropy}}(y) = -\sum_{i=1}^{categories} y_i^{label} log\, y_i^{prediction} \tag{2.2}$$

$$f_{\text{mean\_squared\_error}}(y) = \frac{1}{categories} \sum_{i=1}^{categories} \left(y_i^{label} - y_i^{prediction}\right)^2 \tag{2.3}$$

Cross-Validation

Cross-Validation [76] is widely used data resampling methods to assess generalizability of a model and to prevent over-fitting. Cross-validation often involves stratified random sampling meaning the sampling method retains the class proportions in the learning set. In Leave One Out Cross-Validation each sample is used in turn as the validation set. The test error approximates the true prediction error; however, has high variance. Moreover, its computational cost can

be high for large datasets. k-fold cross-validation aims to optimise the bias/variance trade-off. In k-fold cross-validation the dataset is randomly split into k equal size partitions. A single partition is retained for test data, and the remaining $k-1$ partitions are used for training. The cross-validation steps are reiterated until each partition is used once for validation as shown in Figure 2.1. The results are averaged across all iterations to produce an estimation of the performance of the model (Eq. 2.4). Refaelzadeh *et al.* highlight risks of elevated Type I errors (False Positives). With larger values of k variance is reduced. Moreover, bias also reduces because the model is trained on more of the dataset. This thesis posits that resampling techniques could be used to improve robustness against adversarial examples.

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i \tag{2.4}$$

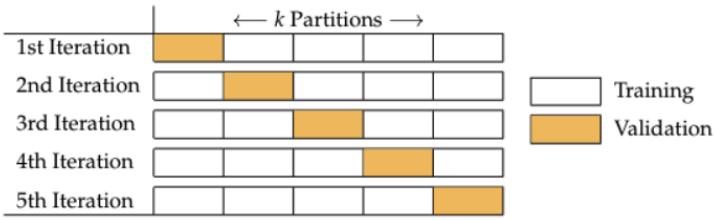


FIGURE 2.1: k-fold Cross-Validation.

Martins *et al.* [5] consider adversarial machine learning for intrusion detection and malware scenarios, noting that IDS are typically signature-based, and that machine learning approaches are being widely employed for intrusion detection. They describe five 'tribes' of ML algorithms before detailing some fundamentals of adversarial machine learning, including commonly used distance metrics: L_∞, L_0, and L_2. They subsequently describe common white-box methods to generate adversarial examples including: Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS), Fast Gradient Sign Method (FGSM), Jacobian-based Saliency Map Attack (JSMA), DeepFool, and Carlini & Wagner attacks (C&W). They also consider black-box methods using Generative Adversarial Networks (GANS). Traditional GANS sometimes suffer problems of mode collapse. Wasserstein Generative Adversarial Networks (WGANS) solve some of these problems. They introduce

Zeroth-order optimization attack (ZOO) as a black-box method. ZOO *estimates* the gradient and optimizes an attack by iteratively adding perturbations to features. They note that most attacks have been initially tested in the image domain but can be applied to other types of data which poses a security threat. Furthermore, they consider there is a trade-off when choosing an adversarial attack. For example, JSMA is more computationally intensive than FGSM but modifies fewer features. They consider JSMA to be the most realistic attack because it perturbs fewer features. When considering defences, they tabulate advantages and disadvantages of common defences. For example, feature squeezing is effective in image scenarios, but unsuitable for other applications because compression methods would result in data loss for tabular data. They note that GANS are a very powerful technique that can result in effective adversarial attacks where the samples follow a similar distribution to the original data but cause misclassification.

Bootstrapping

Bootstrapping is resampling with replacement and is often used to statistically quantify the performance of a model, to determine if a model is statistically significantly better than other models.

2.3.2 Cyber-Physical Systems

Cyber-Physical Systems (CPS) rely on computational systems to create actuation of physical devices. The range of devices is increasing from factory operations to power stations, autonomous vehicles to healthcare operations. Shafique *et al.* [77] consider such smart cyber-physical systems. They discuss reliability and security vulnerabilities of machine learning systems, including: hardware trojans, side channel attacks, and adversarial machine learning. This is important, because system aging and harsh operating environments mean CPS are vulnerable to numerous security and reliability concerns. Advanced persistent threats could compromise the training or deployment of CPSs through stealthy supply-chain attacks. A single vulnerability is sufficient for an adversary to cause a misclassification which could lead to drastic effects in a CPS (e.g., an incorrect steering decision of an autonomous vehicle could cause a collision). It is considered

that vulnerabilities in ML could lead to a range of unwanted effects in CPSs including those that could lead to life-threatening consequences [46]. The Stuxnet worm is an example of malware with dire consequences.

2.3.3 Contributions of this survey

The main objectives of this survey are:

- Collect and collate current knowledge regarding robustness and functionality-preserving attacks in cyber security domains.
- Formulate key takeaways based on the presented information, aiming to assist understanding of the field.

This survey aims to complement existing work while addressing clear differences, by also studying the robustness of adversarial examples, specifically functionality-preserving use cases. Most previous work aims to improve the accuracy of models or examine the effect of adversarial examples. Instead, the robustness of models to adversarial examples is considered.

Machine learning systems are already widely adopted in cyber security. Indeed, with increasing network traffic, automated network monitoring using ML is becoming essential. Modern computer networks carry private personal and corporate data including financial transactions. These data are an attractive lure to cyber-criminals. Adversaries may wish to steal or disturb data. Malware, spyware, and ransomware threats are endemic on many computer networks. IDS help keep networks safe; however, an adversarial arms race exists, and it is likely that adversaries, including advanced persistent threats are developing new ways to evade network defences. Some research has evaded intrusion detection classifiers using adversarial examples.

While adversarial examples in the visual domain are well understood, less work has focused on how adversarial examples can be applied to network traffic analysis and other non-visual domains. Similarly with machine learning models used for image and object recognition. For example, Convolutional Neural Networks

(CNNs) are well researched, whereas other model types used for intrusion detection, e.g., Recurrent Neural Networks (RNNs) receive less attention. The generation of adversarial examples to fool IDS is more complicated than visual domains because the features include discrete and non-continuous values [78]. Compounding the defence against adversarial examples is the overconfident assumption that successful adversarial examples require 'complete knowledge' of the model and parameters. On the contrary black-box attacks are possible with no or limited knowledge of the model. Most defences so far proposed consider the visual domain and most are ineffective against strong and black-box attacks. This survey addresses the problem of adversarial machine learning across cyber security domains. Further research is required to head off future mature attack methods that could facilitate more complex and destructive attacks.

2.3.4 Robustness

Robustness can be defined as the performance of well-trained models facing adversarial examples [79]. Essentially, robustness considers how sensitive a model's output is to a change in the input. The robustness of a model is related to the generalization-error of the model. There is a recognised trade-off between accuracy and robustness in machine learning. That is, highly accurate models are less robust to adversarial examples. Machine learning models in adversarial domains must be both highly accurate and robust. Therefore, improving the robustness of machine learning models enables safer deployment of ML systems across a wider range of domains.

To examine the robustness of a model it necessary to understand how the performance of a model can be evaluated. For a simple binary classifiers that is trained on a labelled training set, each of the model's predictions can be categorised as one of four categories:

- True Positive: A malicious sample that was correctly predicted as malicious.
- False Positive: A benign sample that was predicted as malicious.
- False Negative: A malicious sample that was predicted as benign.
- True Negative: A benign sample that was correctly predicted as benign.

A common way of of representing these categories is with a *confusion matrix* as shown in Figure 2.2.

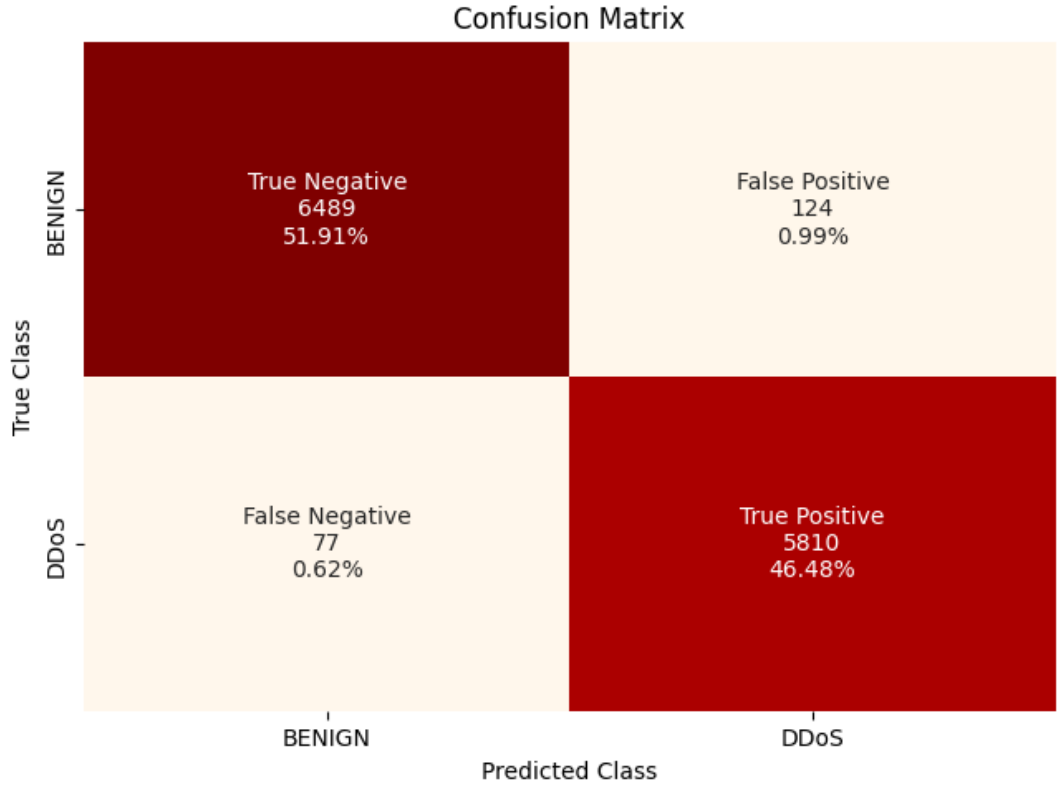


FIGURE 2.2: A confusion matrix showing the four distinct categories of True Positive, False Positive, True Negative, and False Negative.

In this example there are 5,887 malicious samples and 6,613 benign samples. There are 5,819 True Positives, 124 False Positives, 6,489 True Negatives, and 77 False Negatives. The values presented here can be used to compute the additional model performance statistics including Precision (Eq. 2.6), Recall (Eq. 2.7), Specificity 2.8), Sensitivity (Eq. 2.9), and F1 Score (Eq. 2.10). These metrics may have a relationship to robustness. A model with high precision has more cautious decision boundaries, reducing the likelihood that subtle adversarial examples will cause misclassification. A model with high Recall has more inclusive decision

boundaries and may therefore be more susceptible to adversarial examples. While Precision and Recall have some relation to robustness, they do not directly address the robustness of a model. On their own these metrics are not *robustness* metrics; however, the generalization-error can be calculated from F1 Scores.

$$Accuracy = \frac{TruePositives + TrueNegatives}{TruePositives + TrueNegatives + FalsePositives + FalseNegatives} \tag{2.5}$$

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{2.6}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{2.7}$$

$$Specificity = \frac{TrueNegatives}{TrueNegatives + FalsePositives} \tag{2.8}$$

$$Sensitivity = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{2.9}$$

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.10}$$

Other Possible useful metrics to evaluate robustness include: The Lipschitzian property which monitors the changes in the output with respect to small changes to inputs. CLEVER (Cross-Lipschitz Extreme Value for nEtwork Robustness) is an Extreme Value Theory (EVT) based robustness score for large-scale deep neural networks (DNNs). The proposed CLEVER score is attack-agnostic and computationally feasible for large neural networks improving on the Lipschitzian property metric [80].

Work	Metric	Advantages	Disadvantages
N/A	Generalisation error (F1-Score)	Commonly used by researchers.	Biased by the majority class.
[80]	CLEVER	Attack-agnostic and computationally feasible.	CLEVER is less suited to Black-box attacks and where gradient masking occurs [81]; however, extensions to CLEVER help mitigate these scenarios [82].
[83]	Empirical Robustness	Suitable for very deep neural networks and large datasets.	N/A.

TABLE 2.2: Robustness Metrics.

2.3.5 Common Adversarial Example Algorithms

There are numerous algorithms to produce adversarial examples Szegedy *et al.* [22] used a box-constrained limited memory L-BFGS. Other methods include FGSM [23] and iterative derivatives: Basic Iterative Method (BIM), Projected Gradient Descent (PGD). JSMA optimises for the minimal number of altered features (L_0). The DeepFool algorithm [83] optimizes for the root-mean-square (Euclidean distance, L_2). Carlini and Wagner [84] propose powerful C&W attacks optimizing for the L_0, L_2, L_∞ distance metrics. There are many algorithms to choose from. Furthermore, Papernot *et al.* [85] developed a software library for the easy generation of adversarial examples. There are now a number of similar libraries that can be used to generate adversarial examples as shown in Table 2.3.

Work	Library Name	Year	Advantages	Disadvantages
[85]	CleverHans	2016	Recently updated to v4.0.0, well used by the community. MIT License.	It can be complicated to configure.
[86]	Foolbox	2017	Fast Generation of Adversarial examples. MIT License.	Large number of open issues.
[87]	Adversarial Robustness Toolbox	2018	Well-maintained and supported. Supports most known machine learning frameworks. Extensive attacks and model robustness tools are supported.	It does not support every model.
[88]	Advertorch	2019	GNU Lesser Public License.	There are not many active contributors.

TABLE 2.3: Libraries for Generating Adversarial Examples.

Moreover, algorithms like FGSM that modify all features are unlikely to preserve functionality. Algorithms like JSMA that modify a small subset of features are not guaranteed to preserve functionality; although, with fewer modified features, the likelihood improves. Checking for and keeping only examples that preserve functionality is possible, although it is a time-consuming and inelegant solution. A potentially better solution could ensure only functionality-preserving adversarial examples are generated.

When considering the robustness of machine learning models, the threat model must be considered. For example, how much the adversary knows about the classifier, ranging from *no knowledge* to *perfect knowledge*. Adversaries may have a number of different goals:

1. Accuracy degradation (where the adversary wants to sabotage the effectiveness of the overall classifier accuracy).

2. Target misclassification (where the adversary wants to misclassify a particular instance as another given class),

3. Untargeted classification (where the adversary wants to misclassify a particular instance to any arbitrary class).

The attack surface should also be considered. In IDS, the attack surface can be considered as an end-to-end pipeline, with varying vulnerabilities and potential for compromise at each stage of the pipeline.

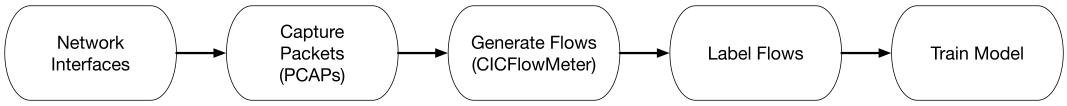


FIGURE 2.3: End to End Pipeline for Network Intrusion Detection System.

In one basic pipeline as shown in Figure 2.3 the raw network traffic on network interfaces is collected as packet capture files (PCAPs), which are then processed into network flows. There are different applications that could be used to process PCAPs into network flows. CICFlowMeter [89] is a network traffic flow generator and analyser that has been used in cyber security datasets [90] [91] and produces bidirectional flows with over 80 statistical network traffic features. The generated flows are unlabelled and so must be labelled manually with the traffic type, typically benign/malicious, although multiclasses could be labelled given sufficient information including attack type, IP source and destination dyad, duration, and start time. Finally, the labelled flows are used to train the model. Repetitive training cycles could enable detection of new attacks; however, the cyclic nature of the training means that an adversary could attack any iteration of training. Furthermore, an adversary could choose to attack any point in the pipeline. The training data used to train the model generally consists of feature-vectors and expected outputs; although, some researchers are considering unsupervised learning models. The collection and validation of these data offer an attack surface. Separately, the inference phase also offers an attack surface. It is interesting to note

that the size of the feature-set a machine learning model uses can be exploited as an attack surface. A fundamental issue is that each feature processed by a model may be modified by an adversary. Moreover, Sarker *et al.* [92] note that the computational complexity of a model can be reduced by reducing the feature dimensions. Large feature-sets include more features and hence provide more opportunities to an adversary for manipulation. Almomani *et al.* [93] indicate accuracy can be maintained with fewer features, and McCarthy *et al.* [43] indicate that more features tend to reduce the necessary size of perturbations. Therefore, larger feature-sets are more readily perturbed than smaller feature-sets which have fewer modifiable features and hence require larger perturbations.

2.3.6 Threat Model - Adversary Capabilities

Adversaries are constrained by their skills, knowledge, tools, and access to the system under attack. An insider threat might have access to the classification model and other associated knowledge, whereas an external threat might only be able to examine data packets. While the attack surface may be the same for both adversaries, the insider threat is potentially a much stronger adversary because they have greater knowledge and access. Adversary capabilities mean that attacks can be split into three scenarios: White-box, Black-box, and Gray-Box.

In white-box attacks, an adversary has access to all machine learning model parameters. In black-box attacks, the adversary has no access to the machine learning model's parameters. Adversaries in black-box scenarios may therefore use a different model, or no model at all to generate adversarial examples. The strategy depends on successfully transferring adversarial examples to the target model. Gray-box attacks consider scenarios where an adversary has some, but incomplete knowledge of the system. White-box and black-box are most commonly considered.

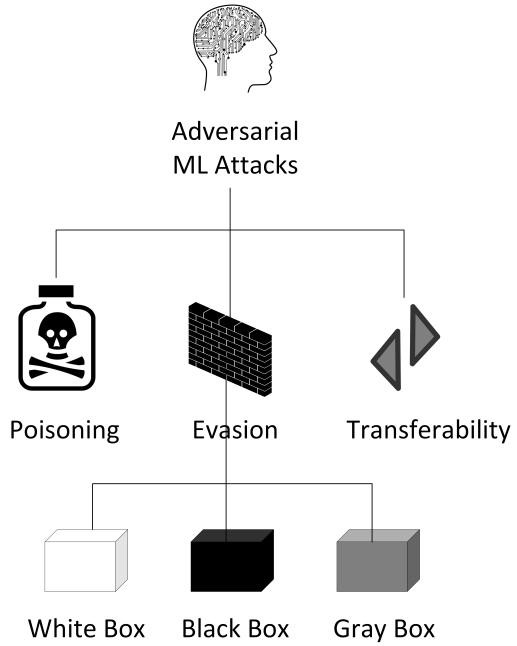


FIGURE 2.4: Common Adversarial Machine Learning Attacks.

2.3.7 Threat Model - Adversary Goals

Adversaries aim to subvert a model through attacking its confidentiality, integrity, or availability. Confidentiality attacks attempt to expose the model, or the data encapsulated within. Integrity attacks occur when an adversary attempts to control the output of the model. For example, to misclassify some adversarial traffic and therefore allow it to pass a detection process. Availability attacks could misclassify all traffic types, or deteriorate a model's confidence, consistency, performance, and access. In this way, an integrity attack resembles a subset of availability attack, since an incorrect response is similar in nature to a correct response being unavailable; however, the complete unavailability of a response would likely be more easily noticed than decreases in confidence, consistency, or performance. The goals of an adversary may be different but are often achieved with similar methods.

2.3.8 Threat Model - Common Attack Methods

Poisoning

In a poisoning attack, an adversary with access to the training data or procedure, manipulates it, implanting an attack during the training phase, when the model is trained on adversarial training data. This is achieved with carefully crafted noise or

sometimes random noise. Unused or dormant neurons in a trained Deep Neural Network (DNN) signify that a model can learn more, essentially an increased number of neurons allows for a greater set of distinct decision boundaries forming distinct classifications of data. The under-utilised degrees of freedom in the learned model could potentially be used for unexpected classification of inputs. That is, the model could learn to provide selected outputs based on adversarial inputs. These neurons have very small weights and biases. However, the existence of such neurons allows successful poisoning attacks through training the model to behave differently for poisoned data. This suggests that distillation [94] could be effective at preventing poisoning attacks, because smaller models have lower knowledge capacity and likely fewer unused neurons. Distillation reduces the number of neurons that contribute to a model by transferring knowledge from a large model to a smaller model. Despite initial analysis indicating reduction in the success of adversarial attacks, Carlini [84] experimented with three powerful adversarial attacks and a high confidence adversarial example in a transferability attack and found that distillation does not eliminate adversarial examples and provides little security benefit over undistilled networks in relation to powerful attacks. Unfortunately, they did not specifically consider poisoning attacks. Additional experiments could determine whether distillation is an effective defence against poisoning attacks.

Evasion

In evasion attacks, the adversary is often assumed to have no access to the training data. Instead, adversaries exploit their knowledge of the model and its parameters, aiming to minimise the cost function of adversarial noise, which when combined with the input causes changes in the model output. Untargeted attacks lead to an arbitrary incorrect output, targeted attacks lead to a specific incorrect output, and an attack may disrupt the model by changing the confidence of the output class. In the visual domain, the added noise is often imperceptible to humans. In non-visual domains such as intrusion detection, this problem may be much more challenging since even small modifications may corrupt network packets and may cause

firewalls to drop these malformed packets. This highlights the need for functionality preservation in adversarial learning, as a clear distinction from vision-based attacks that exploit the human visual system.

Transferability

The transferability property of adversarial examples means that adversarial examples generated against one model will likely also work against other models trained for the same purpose. The second model need not have the same architecture or underlying model as the first and need not be trained on the same data. The transferability property of adversarial examples can form the basis for some black-box attacks where a surrogate model is used to generate adversarial examples that are subsequently presented to the target model.

2.4 Methodology

This section describes the chosen approach for surveying the literature, conducting an effective and meaningful survey of the literature.

Eligibility Criteria: Search queries were determined, leading to the most relevant articles. The chosen search terms are presented in Table 2.4.

Topic	**Search Query**
Cyber Security / Intrusion Detection	("Cyber Security" OR "Intrusion detection" OR IDS)
Adversarial Machine Learning Attacks and Defences	("adversarial machine learning" or "machine learning" or "adversarial example") and (attack or defence)
Robustness / Functionality Preservation	((robustness or generalization error or accuracy or F1-score or f-score or TPR or FPR) or ((functionality or payload) and preservation)))

TABLE 2.4: Topics and associated search terms used in this survey.

It is expected for these search queries to result in good coverage of the relevant literature. Each database was searched using the identified search terms. The literature search was conducted up to September 2021. Generally, works not yet peer-reviewed are excluded, such as those appearing on arXiv. However, occasionally a choice was made to include a significant paper which makes a clear contribution to the subject domain. The search results were collated, and any subsequent duplicates were removed. Each paper was screened by reading the title and abstract to determine the relevance. Inclusion criteria were: the article is related to functionality preservation in adversarial machine learning for cyber security or intrusion detection with insight into robust classification.

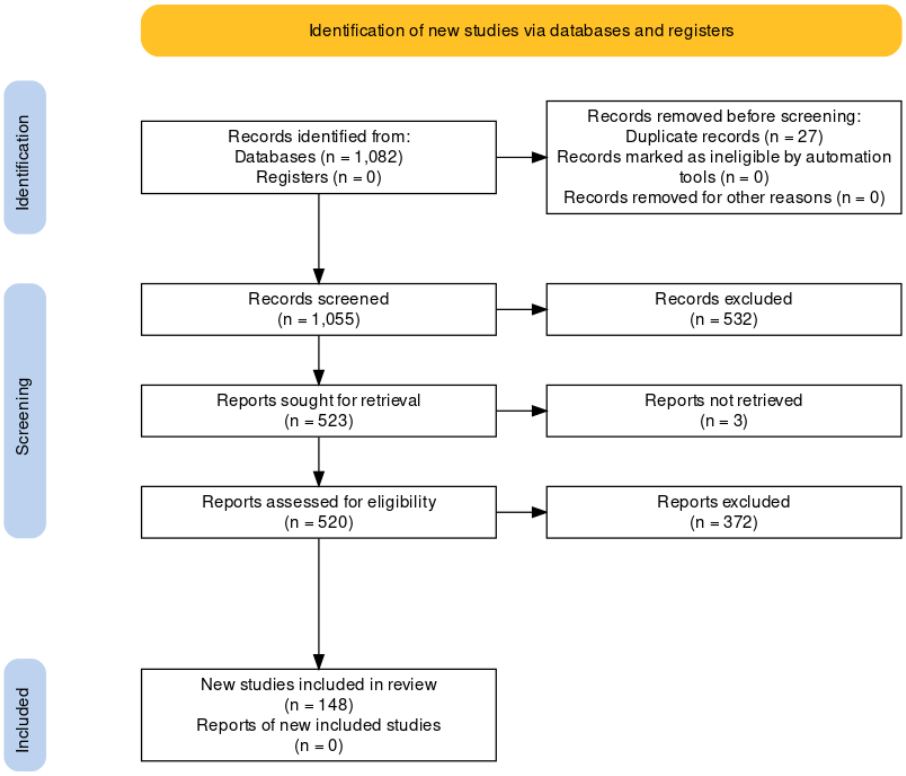


FIGURE 2.5: Preferred Reporting Items for Systematic Meta-Analysis.

From this large list, a specific focus on adversarial machine learning attacks and defences was adopted, narrowing the literature down to relevant papers. The selection process is roughly based on the Preferred Reporting Items for Systematic Meta-Analysis (PRISMA) framework [95]. Figure 2.5 details the selection process.

Information Sources: To search and retrieve relevant literature, the following databases were chosen for their sizeable computer science elements: **IEEE Xplore**, **ACM Digital Library**, **ScienceDirect**, **Scopus**, **SpringerLink**, **Google Scholar**.

2.5 Results

This section describes the results of the search and selection process. The classification scheme is described, and the findings tabulated and discussed, including: Adversarial Attacks in traditional and cyber security domains of malware, IDS, and CPS. There are 146 relevant papers included in this survey.

2.5.1 Classification Scheme

The attacks are classified by attack type, attack objective (targeted/untargeted), domain, model, knowledge required, and whether any constraints are placed on the adversarial examples. Defences are classified by type, domain, and model. To summarise the attacks and defences, three tables are produced. Attacks are detailed in Table 2.5. Defences are detailed in Table 2.8. Functionality-preserving attacks are detailed in 2.7.

2.5.2 Adversarial Example Attacks

This work focuses on attacks that exploit adversarial examples that cause differences in the output of neural networks. Adversarial examples were discovered by Szegedy *et al.* [22]. Adversarial examples are possible in ANN as a consequence of the properties of neural networks; however, they are possible for other ML models. This complicates mitigation efforts, and adversarial examples can be found for networks explicitly trained on adversarial examples [121]. Furthermore, adversarial examples can be algorithmically generated, e.g., using gradient descent. Moreover, adversarial examples are often transferable, that is, an adversarial example presented to a second machine learning model trained for the same task, perhaps on a subset of the original dataset, may also cause the second model to misclassify the adversarial example.

Work	Year	Attack	Type			Obj		Domain			Model			Knowledge			Constraint		
			AE	Sequence of AEs	Transferability	Targeted	Untargeted	Visual	Cybersecurity	Text	MLP	CNN	RNN	White-Box	Black-box	Gray-Box	Box	Sparse	Func-Preserving
[22]	2014	L-BFGS	✓			✓		✓				✓		✓			✓		
[96]	2013	GradientDescent	✓			✓		✓				✓		✓		✓			
[97]	2016	Adversarial Sequences		✓			✓			✓			✓	✓					✓
[98]	2016	JSMA	✓			✓	✓	✓				✓		✓				✓	
[83]	2016	DeepFool	✓				✓	✓				✓		✓					
[99]	2017	AddSent,AddOneSent	✓	✓			✓			✓			✓		✓				
[100]	2018	GAN	✓				✓	✓				✓			✓				
[101]	2017	EnchantingAttack		✓		✓		✓			✓			✓					
[101]	2017	StrategicAttack	✓			✓		✓			✓			✓					
[84]	2017	C&W, L_0, L_2,L_∞	✓				✓	✓				✓		✓					
[102]	2017	FGSM,JSMA	✓				✓		✓		✓			✓					
[103]	2018	Generative RNN	✓				✓		✓				✓		✓				✓
[104]	2018	NPBO	✓				✓		✓		✓			✓					✓
[105]	2018	GADGET	✓				✓		✓				✓		✓				✓
[106]	2018	JSMA,FGSM,DeepFool,CW	✓			✓			✓		✓			✓					
[107]	2018	FGSM	✓				✓		✓		✓			✓					
[108]	2018	IDS-GAN	✓				✓		✓		✓				✓				
[109]	2018	ZOO, GAN	✓				✓		✓		✓				✓				
[110]	2019	One Pixel Attack	✓			✓		✓				✓		✓				✓	
[111]	2019	ManifoldApproximation	✓				✓		✓		✓				✓				✓
[112]	2019	FGSM,BIM,PGD	✓				✓		✓		✓			✓		✓			
[113]	2019	GAN Attack	✓				✓		✓		✓				✓				✓
[114]	2020	PWPSA	✓	✓	✓		✓		✓				✓	✓					✓
[114]	2020	GA	✓	✓	✓		✓		✓				✓	✓					✓
[115]	2020	One Pixel Attack	✓				✓		✓			✓		✓				✓	
[116]	2020	Opt Attack,GAN Attack	✓				✓		✓		✓				✓				✓
[117]	2021	GAMMA	✓				✓		✓						✓				✓
[118]	2021	UAP	✓				✓		✓		✓			✓		✓			✓
[119]	2020	Variational Auto Encoder	✓				✓		✓		✓			✓					✓
[120]	2021	Best-Effort Search	✓				✓		✓		✓	✓		✓	✓	✓	✓		✓

TABLE 2.5: Chronologically ordered summary of adversarial example attacks.

Adversarial Examples - Similarity Metrics

In the visual domain, distance metrics are well used to judge how similar two inputs are, and therefore how easy the differences might be perceived. The following metrics are commonly used to describe the difference between normal and adversarial inputs:

- Number of altered pixels (L_0)
- Euclidean distance (L_2, root-mean-square)
- Maximum change to any of the co-ordinates. (L_∞)

Human perception may not be the best criterion to judge a successful adversarial input. A successful attack in a vision ML task may be to fool a human. Success in

an ML-based system is to fool some other detection routine, while conforming to the expected inputs of the system. For example, a malicious packet must remain malicious after any perturbation has been applied. If a perturbed packet is very close to the original packet, this would only be considered successful if it also retained its malicious properties, and hence its intended function.

Adversarial Examples - Types of Attack

White-box Attacks: Most white-box attacks are commonly achieved through gradient descent to increase the loss function of the target model. The algorithmic generation of adversarial examples is possible. Moreover, Papernot *et al.* [85] developed a software library for the easy generation of adversarial examples and other libraries are now available. An early gradient descent approach was proposed by Szegedy *et al.* [22] using a box-constrained limited memory L-BFGS. Given an original image, this method finds a different image that is classified differently, whilst remaining similar to the original image. Gradient descent is used by many different algorithms; however, algorithms have been designed to be optimized for different distance metrics. There are numerous gradient descent algorithms that produce adversarial examples; they can differ in their optimization and computational complexity. The relative computational complexity of common adversarial example algorithms is noted in Table 2.6 (adapted from [48]). High success rates correlate with high computational complexity. This correlation is expected to be more pronounced for functionality preserving attacks. FGSM [23] was improved by Kurakin *et al.* [122] who refined the fast gradient sign by taking multiple smaller steps. This iterative granular approach improves on FGSM by limiting the difference between the original and adversarial inputs. This often results in adversarial inputs with a predictably smaller L_∞ metric. However, FGSM modifies all parameters. This is problematic for features that must remain unchanged or for discrete features such as Application Programming Interface (API) calls. JSMA differs from FGSM in that it optimizes to minimize the total number of modified features (L_0 metric). In this greedy algorithm, individual features are chosen with the aim of step-wise increasing the target classification in each iteration. The gradient is used to generate a saliency map, modelling each

Method	Computational Complexity	Success Rate
L-BFGS	High	High
FGSM	Low	Low
JSMA	High	High
DeepFool	Low	Low
One-pixel	Low	Low
C&W Attack	High	High

TABLE 2.6: Computational Complexity of Common Adversarial Example Algorithms.

feature's impact towards the resulting classification. Large values significantly increase the likelihood of classification as the target class. Thus, the most important feature is modified at each stage. This process continues until the input is successfully classified as the target class, or a threshold number of pixels is reached. This algorithm results in adversarial inputs with fewer modified features. The DeepFool algorithm [83] similarly uses gradient descent but optimizes for the root-mean-square also known as Euclidean distance (L_2). This technique simplifies the task of shifting an input over a decision boundary by assuming a linear hyper-plane separates each class. The optimal solution is derived through analysis and subsequently an adversarial example is constructed; however, neural network decision boundaries are not truly linear. Therefore, subsequent repetitions may be required until a true adversarial image is found.

The optimizations for different distance metrics are types of constraint: Maximum change to any feature (L_∞); minimal root-mean-square (L_2); minimal number of altered features (L_0). Constrained adversarial examples are important for functionality-preserving attacks. Additional constraints for specific domains are likely required, and this remains an open avenue for further research.

Most gradient descent algorithms were originally presented in the visual domain and used on images and pixel values. The pixel values of images are often presented as continuous values ($0 - 255$). The use of adversarial examples with discrete data values is less well explored and remains an interesting avenue for further research.

Black-box Attacks: Researchers have also considered black-box attacks that need not rely on gradient descent. Some black-box techniques commonly rely on the transferability of adversarial examples. Table 2.5 shows that few researchers employ the transferability of adversarial examples. Other common black-box techniques include GANS and genetic algorithms (GAs). Sharif *et al.* [123] propose a way of attacking DNN with a general framework to train an attack generator or generative adversarial network (GAN). GANs can be trained to produce new, robust, and inconspicuous adversarial examples. Attacks like Biggio *et al.* [96] are more suitable for the security domain when assessing the security of algorithms and systems under worst-case attacks [6] [124].

An important consideration in attacks against intrusion detection systems is that attackers cannot perform simple oracle queries against an intrusion detection system and must minimize the number of queries to decrease the likelihood of detection. Apruzzese *et al.* [125] further note that the output of the target model is not directly observable by the attacker; however, exceptions occur where detected malicious traffic is automatically stopped or dropped, or where the attacker gains access to/or knowledge of the system.

Gray-box attacks consider scenarios where an adversary has only partial knowledge of the system. Biggio *et al.* [96] highlight the threat from skilled adversaries with limited knowledge; More recently Gray-box attacks are receiving some attention: Kuppa *et al.* [111] consider malicious users of the system with knowledge of the features and architecture of the system, recognizing that attackers may differ in their level of knowledge of the system. Labaca-Castro *et al.* [118] use universal adversarial perturbations showing that unprotected systems remain vulnerable even under limited knowledge scenarios. Li *et al.* [120] consider limited knowledge attacks against cyber-physical systems and successfully deploy universal adversarial perturbations where attackers have incomplete knowledge of measurements across all sensors.

Building on Simple Adversarial Examples: Table 2.5 shows much research considers simple adversarial examples, although less research considers sequences of adversarial examples or transferability. This survey chooses to classify attacks as either a simple adversarial example, a sequence of adversarial examples, or a transferable adversarial example. A simple adversarial example is sufficient to alter the output of a simple classifier. Lin *et al.* [101] suggest using adversarial examples strategically could affect the specific critical outputs of a machine learning system. Sequences of adversarial examples consist of two or more adversarial examples. Sequences of adversarial examples are more challenging than simple adversarial examples. Lin *et al.* [101] further suggest an *enchanting* attack to lure a machine learning system to a target state by crafting a series of adversarial examples. Table 2.5 shows that most research considers simple adversarial examples. Researchers are starting to consider sequences of adversarial examples and to consider the transferability of adversarial examples. Classifying attacks in this way clarifies the level of complexity in attack types. Furthermore, the table shows that sequences of adversarial examples and the transferability of adversarial examples is under-represented, providing opportunities for further research.

Adversarial Examples - Attack Objectives

There is a distinction between the objectives of attacks: targeted or untargeted. An attack objective might be to cause a classifier to misclassify an input as *any* other class (untargeted) or to misclassify an input as a *specific* class (targeted). In the cyber security domain, IDS often focus on binary classification: malicious or benign. For binary classification the effect of targeted and untargeted attacks is the same. More complex multi-class IDS can help network analysts triage or prioritise different types of intrusions. Network analysts would certainly treat a Distributed Denial of Service (DDoS) attack differently than a BotNet or infiltration attempt. Adversaries could gain significant advantage through *targeted* attacks. For example, by camouflaging an infiltration attack as a comparatively less serious network intrusion.

Recent research goes beyond adversarial examples causing misclassification of a single input. Moosavi-Dezfooli *et al.* [126] further show the existence of untargeted *universal* adversarial perturbation (UAP) vectors for images, and venture this is problematic for classifiers deployed in real-world and hostile environments. In the cyber security domain, Labaca *et al.* [118] demonstrate UAPs in the feature space of malware detection. They show that UAPs have similar effectiveness as adversarial examples generated for specific inputs. Sheatsley *et al.* [75] look at UAP in the constrained domain of intrusion detection. Adversaries need only calculate one UAP that could be applied to multiple inputs. Pre-calculation of a UAP could enable faster network attacks (DDoS) that would otherwise require too much calculation time. Table 2.5 shows most research considers untargeted attacks. Targeted attacks are less represented in the literature. Furthermore, UAPs are a more recent avenue for research.

Adversarial Examples in Traditional Domains

Table 2.5 shows attacks in the visual domain were the subject of much early research, and the visual domain continues to attract researchers; although, researchers are beginning to consider attacks against other DNN systems such as machine learning models for natural language processing, with some considering semantic preserving attacks.

In visual domains, features are generally continuous. For example, pixel values range from $0 - 255$. A consensus exists in the visual domain that adversarial examples are undetectable to humans. Moreover, the application domain is clearly interrelated with the choice of machine learning model. Models such as CNNs are appropriate for visual-based tasks, whereas RNNs are appropriate for sequence-based tasks. Model types are discussed in section 2.5.2.

Some models such as recurrent neural networks are less effectively attacked using traditional attack algorithms; however, some research aims to discover new methods to attack these systems. Papernot *et al.* [97] note that because RNNs handle time sequences by introducing cycles to their computational graphs. The

presence of these computation cycles means that applying traditional adversarial example algorithms is challenging because cycles prevent direct computation of the gradients. They adapt adversarial example algorithms for RNNs and evaluate the performance of their adversarial samples. If the model is differential, FGSM can be applied even to RNN models. They use a case study of a binary classifier (positive or negative) for movie reviews. They define an algorithm that iteratively modifies words in the input sentence to produce an adversarial sequence that is misclassified by a well-trained model. They note that their attacks are white-box attacks, requiring access to, or knowledge of, the model parameters. Szegedy [22] discovered the transferability of adversarial examples, noting the same perturbation can cause a different network that was trained on a different subset of the dataset, to misclassify the same adversarial input. This property of adversarial examples has serious implications because it means gaining access to a model is unnecessary to attack it. An adversary can employ the transferability of adversarial examples, where adversarial examples generated against a model under the adversary's control can be successfully used to attack the target model. The transferability of adversarial examples implies that an adversary does not need full access to a model to attack it (Black-box).

Adversarial Examples in Cyber Security Domains

Adversarial examples (AE) have been shown to exist in many domains. Indeed, no domain identified (so far) is immune to adversarial examples [75]. Researchers are beginning to consider cyber security domains where features are often a mixture of categorical, continuous and discrete. Some research focuses on adversarial example attacks against IDS; although few specifically consider functionality-preserving attacks.

Recall that for the visual domain there is a consensus that adversarial examples are undetectable to humans. However, it is unclear how this idea should be translated to other domains. Carlini [84] holds that, strictly speaking, adversarial examples must be similar to the original input. However, Sheatsley *et al.* [75] note that research in non-visual domains provide domain specific definitions: perturbed

Work	Year	Domain	Generation Method	Realistic Constraints	Findings
[78]	2019	Malware	Gradient-based	Minimal Content additions/modification	Experiments showed the ability to use that information to find optimal sequences of transformations without rendering the malware sample corrupt.
[113]	2019	IDS	GAN	Preserve functionality	The proposed adversarial attack successfully evades the IDS while ensuring preservation of functional behaviour and network traffic features.
[127]	2019	IDS	Gradient-based	Respect mathematical dependencies and domain constraints.	Evasion attacks achieved by inserting a dozen network connections.
[128]	2019	IDS	Random Modification $\leq$ 4 features: flow duration, sent bytes, received bytes, exchanged packets.	Retain internal logic	Feature removal is insufficient defence against functionality-preserving attacks which may are possible by modifying very few features.
[129]	2019	IDS	Legitimate transformations: Split, Delay, Inject	Packets must maintain malicious intent, transformations hold to underlying protocols.	Detection rate of packet-level features dropped by up to 70% and flow-level features dropped by up to 68%.
[75]	2020	IDS - Flows	Jacobian Method (JSMA)	Obey TCP/IP constraints	Biased distributions with low dimensionality enable constrained adversarial examples. Constrained to five random features, $\approx$ 50% adversarial examples succeed.
[114]	2020	IDS - packet	Valid packet	Minimal modification/insertion of packets	Experimental results show powerful and effective functionality preserving attacks. More accurate models are more susceptible to adversarial examples.
[117]	2021	Malware	Injected unexecuted benign content	Minimal Injected Content	Section-injection attack can decrease the detection rate. Their analysis highlights that commercial products can be evaded via transfer attacks.
[120]	2021	CPS	Best-effort search	Real-world linear inequality	Best-effort search algorithms effectively generate adversarial examples meeting linear constraints. Their evaluation shows constrained adversarial examples significantly decrease detection accuracy.
[130]	2021	IDS	Minimal perturbation of each feature	FGSM	Functionality is not reported, but is less likely to preserve functionality because all features are perturbed.
[131]	2021	CPS/ICS	JSMA	Minimal number of perturbed features	Functionality is not reported, but is more likely to preserve functionality because relatively few features are perturbed.
[132]	2021	IDS	PSO-based mutation	original traffic retained and packet order is unchanged	Measured attack effect, malicious behaviour and attack efficiency
[133]	2021	IDS	GAN	preserving functional features of attack traffic	F1-score drops to zero from around 99% DIGFuPAS adversarial examples.
[134]	2021	IDS	PSO/GA/GAN	Only modify features where network functionality is retained	In the network traffic data, it is unrealistic to assume an adversary can alter all traffic features - Constraints on features that do not break functionality
[135]	2020	IDS	GAN/PSO	Original traffic and packet order is retained.	Detection performance and robustness should both be considered in feature extraction systems.

TABLE 2.7: Functionality-Preservation in Cyber Security and Intrusion Detection.

malware must preserve its malware functionality [75], perturbations in audio must be *nearly* inaudible [75], perturbed text must preserve its meaning. Sheatsley *et al.* further offer a definition for adversarial examples in intrusion detection: perturbed network flows must maintain their attack behaviour. Human perception may not be the best criterion for defining adversarial examples in cyber security domains. Indeed, human perception in some domains might be immaterial. For example, only very skilled engineers could *perceive* network packets in any meaningful way even with the use of network analysis tools. Furthermore, users likely cannot perceive a difference between the execution of benign or malicious software. After malware is executed, the effects are clear; however, during malware execution users often suspect nothing wrong. Therefore, it is considered that while fooling human perception remains a valid ambition. It is critical that adversarial perturbations in cyber security domains preserve functionality and behaviour.

In the cyber security domain, traditional gradient descent algorithms may be insufficient. Algorithms that preserve functionality are required. Moreover, some models used in the cyber security domain are distinct from those used for purely visual problems. For example, RNNs are useful for time sequences of network traffic analysis. Recent functionality-preserving attacks in the cyber security domains of Malware, Intrusion Detection, and CPS are now considered. As illustrated in Figure 2.6.

FIGURE 2.6: Common Machine Learning Tasks in Cyber Security.

Malware: Hu and Tan [103] propose a novel algorithm to generate adversarial sequences to attack a RNN based malware detection system. They claim that algorithms adapted for RNNs are limited because they are not truly sequential. They consider a system to detect malicious API sequences. Generating adversarial examples effective against such systems is non-trivial because API sequences are

discrete values. There is a discrete set of API calls; changing any single letter in an API call will create an invalid API call and cause that API call to fail. This will result in a program crash. Therefore, any perturbation of an API call must result in a set of valid API calls. They propose an algorithm based around a generative RNN and a substitute RNN. The generative RNN takes an API sequence as input and generates an adversarial API sequence. The substitute RNN is trained on benign sequences and the outputs of the generative RNN. The generative model aims to minimize the predicted malicious probability. Subsequently, adversarial sequences are presented to six different models. Following adversarial perturbation, the majority of the malware was not detected by any victim RNNs. The authors of the paper note that even when the adversarial generation algorithm and the victim RNN are implemented with different models and trained on different training sets, the majority of the adversarial examples successfully attack the victim RNN through the *transferability* property of adversarial examples. In MLP, they report a TPR of 94.89% which falls to 0.00% under adversarial perturbations.

Demetrio *et al.* [117] preserve the functionality of malware while evading static Windows malware detectors. Their attacks exploit the structure of portable executable (PE) file format. Their framework has three categories of functionality-preserving manipulations: Structural, Behavioural, and Padding. Some of their attacks work by injecting unexecuted (benign) content in new sections in the PE file, or at the end of the malware file. The attacks are a constrained minimization problem optimizing the trade-off between the probability of evading detection and the size of injected content. Their experiments successfully evade two Windows malware detectors with few queries and small payload size. Furthermore, they discover their attacks transfer to other Windows malware products. Note that the creation of new sections provides a larger attack surface that may be populated with adversarial content. They report that their section-injection attack is able to drastically decrease the detection rate (e.g., from an original detection rate of 93.5% to 30.5% also significantly outperforming their random attack at 85.5%).

Labaca-Castro *et al.* [78] present a gradient-based method to generate valid executable files that preserve their intended malicious functionality. They note that malware evasion is a current area of adversarial learning research. Evading the classifier is often the foremost objective; however, the perturbations must also be carefully crafted to preserve the functionality of malware. They note that removing objects from a PE file often leads to corrupt files. Therefore, they only implement additive or modifying perturbations. Their gradient-based attack relies on *complete-knowledge* of the system with the advantage that the likelihood of evasion can be calculated and maximised. Furthermore, they state that their system only generates valid executable malware files.

Intrusion Detection: Usama *et al.* [113] use a Generative Adversarial Network (GAN) to generate functionality-preserving adversarial examples. They note that adversarial examples aiming to evade IDS should not invalidate network traffic features. A typical GAN composed of two neural networks: a generator G and discriminator D is used to construct adversarial examples that masquerade as benign but functionally probe the network. Their attack is able to evade an IDS while preserving the intended behaviour. They suggest that adversarial training using GAN generated adversarial examples improves the robustness of their model. They report F1-Scores of 89.03 (original), 40.86 (After attack), 78.49 (after adversarial training), and an improved 83.56 after GAN-based adversarial training.

Wang *et al.* [136] note that relatively few researchers are addressing adversarial examples against IDS. They propose an ensemble defence for network intrusion detection that integrates GANS and adversarial retraining. Their training framework improves robustness while maintaining accuracy of unperturbed samples. Unfortunately, they evaluate their defences against traditional attack algorithms: FGSM, Basic Iterative Method (BIM), DeepFool, JSMA. However, they do not specifically consider functionality-preserving adversarial examples. They further recognise the importance of using recent datasets for intrusion detection. They report F1-Scores for three classifiers and a range of adversarial example algorithms. For example, F1-Score for an ensemble classifier tested on clean data is

0.998 compared to 0.746 for JSMA. Among all classifiers, the ensemble classifier achieved superior F1-Scores under all conditions.

Huang *et al.* [114] note that it is more challenging to generate DDoS adversarial examples because of their discrete properties. They note that work in the visual domain cannot be directly applied to adversarial examples for intrusion detection of DDoS. The input to their algorithm is a series of packets. This makes it difficult to optimize the distance between the original and adversarial sample while guaranteeing the validity of each packet. They propose two black-box methods to generate DDoS adversarial examples against LSTM-based intrusion detection system: Genetic Algorithm (GA) and Probability Weighted Packet Saliency Attack (PWPSA). Each method modifies the original input, either inserting or modifying packets. The GA method evolves a population of DDoS samples and selects adversarial examples from the population. In PWPSA the most important packet in the sequence is found and replaced with a different 'best packet' for this position. Both methods produce adversarial examples that can successfully evade their DDoS intrusion detection model. They report success rates for their different attacks against different detectors. For example, success results for detector D: GA-Replace 91.37 % GA-Insert 74.5%, PWPSA-Replace 88.9%, PWPSA-Insert 67.17%.

Cyber-Physical Systems: Cai *et al.* [119] warn that adversarial examples have consequences for system safety because they can cause systems to provide incorrect outputs. They present a detection method for adversarial examples in CPS. They use a case study of an Advanced Emergency Braking System where a DNN estimates the distance to an obstacle. Their adversarial example detection method uses a variational auto-encoder to predict a target variable (distance) and compare it with a new input. Any anomalies are considered adversarial. Furthermore, adversarial example detectors for CPS must function efficiently in a real-time monitoring environment and maintain low false alarm rates. They report since the p-values for the adversarial examples are almost 0, the number of false alarms is very small, and the detection delay is smaller than 10 frames or 0.5 s.

CPS include critical national infrastructure such as power grids, water treatment plants, and transportation. Li *et al.* [120] assert that adversarial examples could exploit vulnerabilities in CPS with terrible consequences; however, such adversarial examples must satisfy real-world constraints (commonly linear inequality constraints). For example, meter readings downstream may never be larger than meter readings upstream. Adversarial examples breaking constraints are noticeably anomalous. Risks to CPS arising from adversarial examples are not yet fully understood. Furthermore, algorithms and models from other domains may not readily apply because of distributed sensors and inherent real-world constraints. However, generating adversarial examples that meet such linear constraints were successfully applied to power grids and water treatment system case studies. The evaluation results show that even with constraints imposed by the physical systems, their approach still effectively generates adversarial examples, significantly decreasing the detection accuracy. For example, they report the accuracy under adversarial conditions to be as low as 0%.

Adversarial Examples and Model Type

In this survey models are classified based on their architecture in four broad types: Multi-Layer Perceptron (MLP), CNN, RNN, and RF. Ali *et al.* [137] observed that different deep learning architectures are more robust than others. They note that CNN and RNN detectors are more robust than MLP and hybrid detectors, based on low attack success rate and high query counts. Architecture plays a role in the accuracy of these models because CNNs can learn contextual features due to their structure, and RNNs are temporally deeper, and thus demonstrate greater robustness.

Unsurprisingly, research on CNNs coincides with research in the visual domain as shown in Table 2.5. The majority of adversarial example research on RNNs has until recently focused on the text or natural language domain; however, RNNs are also useful in the cyber security domain and researchers have recently considered

adversarial example attacks against RNN-based IDS.

Other promising research shows that radial basis function neural networks (RBFNN) are more robust to adversarial examples [138]. RBFNNs fit a non-linear curve during training, as opposed to fitting linear decision boundaries. Commonly RBFNNs transform the input such that when it is fed into the network it gives a linear separation. The non-linear nature of RBFNNs could be one potential direction for adversarial example research. Powerful attacks able to subvert RBFNNs would improve the understanding of decision boundaries. Goodfellow *et al.* [23] argue the primary cause of neural networks' vulnerability to adversarial perturbation is their linear nature. However, RBFNNs are less commonly deployed and are therefore not further discussed.

Adversarial Examples and Knowledge Requirement

The majority of the research focus is on white-box attacks as shown in Table 2.5, perhaps because such attacks are known to be efficient and effective. Less research focuses on black-box attacks, and few recognise gray-box attacks that need only partial model knowledge. Gray-box attacks will likely have advantages over black-box attacks. Adversaries will undoubtedly use *any* and *all* information available to them.

The attacks are classified based on the knowledge required by the adversary. White-box attacks are likely the most effective and efficient method of attack, because the adversary has *complete-knowledge* of the model architecture, and information on how the model was trained. However, access to this knowledge is harder to attain, although it might also be gained through insider threats [139] or model extraction attacks [140]. Extracted models might be a feasible proxy on which to develop and test adversarial examples.

Notwithstanding the efficiency of white-box attacks, effective black-box attacks are possible. Black-box (or oracle) attacks require no knowledge of the model. Adversaries only need the ability to query the model and receive its output.

Adversaries generate inputs and receive the output of the model. Typical black-box attacks include GA [114], and GANs [108] [116].

Gray-box attacks require only limited model knowledge, perhaps including knowledge of the features used by the model. This is a realistic prospect as adversaries will likely have or gain at least partial knowledge of the model.

Adversarial Example Constraints

Table 2.5 shows little research considering constraints of any sort. Much research on IDS ignores constraints; however, network traffic is highly constrained by protocols, and some network firewalls may drop malformed packets. Furthermore, it is insufficient that well-formed adversarial examples progress past firewalls. They must also retain their intended functionality. Stringent constraints exist in the cyber security domain. Extreme care must be taken to create valid adversarial examples. For example, in IDS, adversaries must conform the protocol specification of the TCP/IP stack.

Constraints on adversarial examples can be classified into three groups: 1) box constraints, simple constraints where values must remain within certain values. 2) Sparse constraints, where a maximum number of features can be modified, the most extreme version being where only one feature can be modified. . 3) functionality-preserving constraints, where adversarial examples must retain their original functionality. For example, malware must function as malware when perturbed to evade a malware detector, and DDoS attacks must function as DDoS attacks when perturbed to evade detection. Functionality-preserving adversarial examples are an interesting avenue for further research.

FIGURE 2.7: Common Defence Types against Adversarial Machine Learning.

2.5.3 Defences Against Adversarial Examples

Work	Year	Defence	Type								Domain				Model		
			Pre-Process	Detection	Adv-Training	Testing	Architectural	Distillation	Ensemble	Game Theory	Visual	Cybersecurity	Text	MLP	CNN	RNN	RF
[23]	2014	Adversarial Training			✓						✓				✓		
[94]	2016	Distillation as defence						✓			✓				✓		
[141]	2016	Feedback Alignment					✓				✓				✓		
[142]	2016	Assessing Threat	✓								✓				✓		
[143]	2017	Statistical Test		✓							✓	✓		✓	✓		
[144]	2017	Detector SubNetwork		✓							✓				✓		
[145]	2017	Artifacts		✓							✓				✓		
[146]	2017	MagNet		✓							✓				✓		
[147]	2017	Feature Squeezing		✓							✓				✓		
[148]	2017	GAT			✓						✓				✓		
[121]	2018	EAT			✓						✓				✓		
[149]	2018	Defense-GAN			✓						✓				✓		
[122]	2018	Assessing Threat	✓								✓				✓		
[150]	2018	Stochastic Activation Pruning					✓				✓			✓	✓		
[151]	2018	DeepTest				✓					✓				✓		
[152]	2018	DeepRoad				✓					✓				✓		
[153]	2018	Defensive Dropout					✓				✓				✓		
[153]	2018	Def-IDS			✓							✓		✓			
[6]	2018	Multi-Classifier System							✓		✓				✓		
[154]	2019	Weight Map Layers					✓				✓				✓		
[155]	2019	Sequence Squeezing		✓								✓				✓	
[128]	2019	Feature Removal	✓									✓		✓			
[156]	2020	Adversarial Training			✓							✓					✓
[157]	2020	Adversarial Training			✓							✓					✓
[158]	2019	Game Theory								✓		✓		✓			
[159]	2020	Hardening			✓					✓		✓					✓
[160]	2021	Variational Auto-encoder		✓							✓				✓		
[161]	2021	MANDA		✓								✓		✓			

TABLE 2.8: Chronologically ordered summary of defences against adversarial examples.

It is hard to defend against adversarial examples. People expect ML models to give correct outputs for all possible inputs. Because the range of possible inputs is so large, it is difficult to guarantee correct model behaviour for every input. Some researchers explore the possibility of exercising all neurons during training [151]. Furthermore, consideration must be given to how adversaries might react when faced with a defence. Researchers in secure machine learning must evaluate whether defences remain secure against adversaries with knowledge of model defences. The suggested defences against adversarial examples are classified into the following groups: Pre-processing, Adversarial Training, Architectural, Detection, Distillation, Testing, Game Theory, and Ensembles as illustrated in Figure 2.7.

Pre-Processing as a Defence against Adversarial Examples

Some promising research considers transformations such as: translation, additive noise, blurring, cropping, resizing. These often occur with cameras and scanners in the visual domain. Translations have shown initial success in the visual domain. Initial successes have prompted some researchers to discount security concerns. For example, Graese [142] overreaches by declaring adversarial examples an 'academic curiosity', not a security threat. This position misunderstands the threat from adversarial examples which remain a concern for cyber security researchers.

Eykolt *et al.* [162] note the creation of perturbations in physical space that survive more challenging physical conditions (distance, pose, and lighting). Transformations are appropriate for images; however, such translations may make little sense in cyber security domains. For example, what would it mean to rotate or blur a network packet? Nevertheless, inspiration could be taken from pre-processing methods in the visual domain. Adapting pre-processing methods to cyber security and other nonvisual domains is an interesting avenue for research.

Adversarial Training as a Defence against Adversarial Examples

Szegedy *et al.* [22] found robustness to adversarial examples can be improved by training a model on a mixture of adversarial examples and unperturbed samples. Specific vulnerabilities in the training data can be identified through exploring UAPs. Identified vulnerabilities could potentially be addressed with adversarial training. Adversarial training is recognised as a simple method aiming to improve robustness; however, it is potentially a cosmetic solution: the problem of adversarial examples cannot be solved only through ever greater amounts of adversarial examples in the training data. Tramér *et al.* [121] found adversarial training is imperfect and can be bypassed. Moreover, black-box attacks have been shown to evade models subject to adversarial training. Adversarial training has some merit because it is a simple method to improve robustness. It is unfortunately not a panacea and should be bolstered by other defences. Research avenues could combine adversarial training with other techniques. Models used in cyber security or other critical domains should not rely solely on adversarial training.

Architectural Defences against Adversarial Examples

Some research, rather than modifying a model's training data investigate defences through hardening the architecture of the model. This could involve changing model parameters or adding new layers. In Table 2.8 such defences are classified as architectural.

Many white-box attacks rely on the quality of the gradient. Some research considers how the model's weights can be used to disrupt adversarial examples. Amer and Maul [154] modify Convolutional Neural Networks (CNN) adding a weight map layer. Their proposed layer easily integrates into existing CNNs. A Weight Mapping layer may be inserted between other CNN layers; thus, increasing the network's robustness to both noise and gradient-based adversarial attacks.

Other research aims to block algorithms from using weight transport and back-propagation to generate adversarial examples. Lillicrap *et al.* [141] propose a

mechanism called 'feedback alignment' which introduces a separate feedback path via random fixed synaptic weights. Feedback alignment blocks the generation of adversarial examples that rely on the gradient because it uses the separate feedback path rather than weight transport.

Techniques to improve accuracy could similarly help harden models. For example, dropout can improve accuracy when used during training. It is particularly useful where there is limited training data and over-fitting is more likely to occur. Wang *et al.* [153] propose hardening DNN using defensive dropout at test time. Unfortunately, there is inherently a trade-off between defensive dropout and test accuracy; however, a relatively small decrease in test accuracy can significantly reduce the success rate of attacks. Such hardening techniques force successful attacks to use larger perturbations, which in turn may be more readily recognized as adversarial.

Defences that block gradient-based attacks complicate the generation of adversarial examples; however, like adversarial training these defences could be bypassed. In particular black-box attacks and transferability-based attacks are not blocked by such defences. A more promising defence "Defensive dropout" [153] can block both black-box and transferability-based attacks.

Detecting Adversarial Examples

Much research has considered the best way to detect adversarial examples. If adversarial examples can be detected they could be more easily deflected, and perhaps even the original input could be salvaged and correctly classified. Grosse *et al*. [143] propose a statistical test to detect adversarial examples before they are input into machine learning models. They observe adversarial examples are unrepresentative of the distribution and lie in unexpected regions of a model's output surface. Their proposed outlier detection system relies on the statistical separation of adversarial examples. They subsequently evaluate their model against adaptive strategies and strong black-box strategies.

Metzen *et al.* [144] propose a binary classifier "Detector Subnetwork" aiming to distinguish between genuine data and adversarial examples. The detection of adversarial examples does not unequivocally lead to correct classification; however, the effect of adversarial examples could perhaps be mitigated through fallback solutions. For example, by requesting human intervention. After successfully detecting adversarial examples in their experiments, they later bypassed their own defences by generating adversarial examples that fool both detector and classifier. They further propose a training procedure called 'dynamic adversary training' as a countermeasure to their attack against the detector.

Feiman *et al.* [145] also detect adversarial examples by considering which artifacts of adversarial examples could help detection. They consider two complementary features used to detect adversarial examples: Density estimates and Bayesian uncertainty estimates. They evaluate these features on CNNs trained on MNIST and CIFAR-10 datasets. They effectively detect adversarial examples with ROC-AUC of 92.6%. They further suggest that their method could be used in RNNs. This suggestion is bolstered by Gal and Ghahramani's [163] assertion that Bayesian approximation using dropout can be applied to RNN networks.

Meng *et al.* [146] propose a framework 'MagNET' to detect adversarial examples. This framework precedes the classifier it defends. The framework has two components: 1) A detector finds and discards any out-of-distribution examples (those significantly far from the manifold boundary). 2) A reformer that aims to find close approximations to inputs before forwarding the approximations to the classifier. Their system generalizes well because it learns to detect adversarial examples without knowledge of how they were generated. They propose a defence against gray-box attacks where the adversary has knowledge of the deployed defences. The proposed defence trains a number of auto-encoders (or *reformers*). At test-time a single auto-encoder is selected at random.

Xu *et al.* [147] propose 'Feature Squeezing' as a strategy to detect adversarial examples by squeezing out unnecessary features in the input. Through comparing

predictions of the original and feature squeezed inputs, adversarial examples are identified if the difference between the two predictions meets a threshold. Two feature-squeezing methods are used: 1) Reducing the colour bit-depth of the image. 2) Spatial smoothing. An adversary may adapt and circumvent this defence; however, the defence may frustrate the adversary because it changes the problem the adversary must overcome.

Rosenberg *et al.* [155] consider the feature squeezing defence designed for CNNs and propose 'Sequence Squeezing' which is adapted for RNNs. Adversarial examples are similarly detected by running the classifier twice: once on the original sequence, and once for the sequence-squeezed input. An input is identified as adversarial if the difference in the confidence scores meets a threshold value.

Zhang *et al.* [160] propose an image classifier based on a variational auto-encoder. They train two models each on half the dataset: a target model and a surrogate model. On the surrogate model they generate three types of strong transfer-based adversarial examples: L_0, L_2, and L_∞. Analysis of their model using the CIFAR-10, MNIST, and Fashion-MNIST datasets found their model achieves state-of-the-art accuracy with significantly better robustness. Their work is in the visual domain; however, perhaps their ideas can be applied to other domains such as intrusion detection.

Some architectural defences against adversarial examples have been discussed. In particular, methods for detecting adversarial examples. Carlini & Wagner [164] show Adversarial examples are harder to detect and that adversarial examples do not exhibit intrinsic properties. Moreover, many detection methods can be broken by choosing good attacker-loss functions. Grosse *et al.* [143] note adversarial defences exist within an arms race. Therefore, guarantees against future attacks are difficult because adversaries may adapt to the defences by adopting new strategies. Meng *et al.* [146] advocate that defences against adversarial examples should be independent of any particular attack. Note that human-in-the-loop solutions could be useful where few cases need human intervention; however repeated requests

might quickly overwhelm human operators given large numbers of adversarial examples. For example, as might be seen in network traffic analysis.

Defensive Testing

Adversarial examples cause unexpected behaviour. Recent research considers testing deep learning systems. Pei *et al.* [165] aim to discover unusual or unexpected behaviour of a neural network through systematic testing. They produce test data by solving a joint optimization problem. Their tests aim to trigger different behaviours and activate a high proportion of neurons in a neural network. Their method finds corner-cases where incorrect behaviour is exhibited. For example, malware masquerading as benign. They claim to expose more inputs and types of unexpected behaviour than adversarial examples. They further use the generated inputs to perform adversarial training. As a defence the practicability of triggering all neurons in larger neural networks is questioned; however, as an attack, their method could produce different types of adversarial inputs.

Other researchers are considering similar techniques to generate test data. Tian *et al.* [151] evaluate a tool for automatically detecting erroneous behaviour, generating test inputs designed to maximise the number of activated neurons using realistic driving conditions including: blurring, rain, and fog. Zhang *et al.* [152] propose a system to automatically synthesize large amounts of diverse driving scenes, including weather conditions using GANs. GANs may be useful for generating adversarial inputs and may even implicitly learn domain constraints.

Multi-Classifier Systems

Biggio *et al.* [6] Highlight that robustness against adversarial examples can be improved by careful use of ensemble classifiers. For example, by using rejection-based mechanisms. Indeed, Biggio *et al.* had implemented a multi-classifier system (MCS) [166] which was hardened using randomisation. Randomising the decision boundary makes a classifier harder to evade. Since the attacker has less information on the exact position of a decision boundary, they

must make too conservative or too risky choices when generating adversarial examples.

Game Theory

Zhou *et al.* [158] consider game theoretic modelling of adversarial machine learning problem. Many different models have been proposed. Some aim to optimise the feature set using a set of high-quality features, thus making adversarial attacks more difficult. Game theoretic models are proposed to address more complex situations with many adversaries of different types. Equilibrium strategies are acceptable to both players and neither has an incentive to change. Therefore, assuming rational adversaries, game theory-based approaches allowing a Nash equilibrium could potentially end the evolutionary arms race.

Adversarial Example defences in Cyber Security Domains

Domains were discussed in sections 2.5.2 and 2.5.2. Most research on defences against adversarial examples has focused on the visual domain. Comparatively little research has so far considered defences in cyber security domains such as intrusion detection and malware analysis. Applying current defences in the visual domain to other domains might efficiently kick-start research into defences for other domains. Effective defences against adversarial examples could help enable the use of ML models in cyber security and other adversarial environments.

Different model types are more suited to specific domains. It might be that different model types may require different defences, although a defence suitable for a range of different models would be useful and convenient. Again, the models are classified into four types: MLP, CNN, RNN, and RF.

2.6 Discussion and Conclusion

ML systems are deployed in complex environments including cyber security and critical national infrastructure. Such systems attract the interest of powerful advanced persistent threats who may target them. Crucially, robustness against

functionality-preserving adversarial examples must be addressed before novel attack strategies exploit inherent weaknesses in critical ML models.

Machine learning and adversarial learning are becoming increasingly recognised by the research community, given the rapid uptake of ML models in a whole host of application domains. To put this in context, 2,975 papers were published on arXiv in the last 12 months (October 2020 - September 2021) related to machine learning and adversarial learning. Over recent years, the number of papers being published on this topic has grown substantially. According to Carlini who maintains a blog post 'A Complete List of All (arXiv) Adversarial Example Papers' [167] the cumulative number of adversarial example papers nears 4000 in the year 2021. It is therefore evident that there is a lot of interest and many researchers active in this area. Not all papers in this list are useful or relevant, no judgement is passed of their quality, the aim is merely to clarify the research landscape and draw important research to the fore. The majority of prior research has been applied to the visual domain. Seminal contributions have been made by Szegedy *et al.* [22], Goodfellow *et al.* [23], Carlini *et al.* [84], Papernot [98]. It is clear that the visual domain continues to be well researched.

An extensive survey of the academic literature in relation to functionality-preservation in adversarial machine learning was conducted. A classification based on both attack and defence was derived. Possible robustness metrics were considered. Moreover, model training and data-level techniques were considered, to determine how they could help improve robustness through tackling biased datasets.

Analysis of functionality-preservation methods finds gradient-based methods may be less suitable for functionality-preservation and other constraints. Methods modifying large numbers of features are less likely to preserve functionality. It was found that GANS and Genetic algorithms are suitable for functionality-preserving attacks. Defence strategies against functionality-preserving adversarial examples were discussed. The survey found that preserving functionality in adversarial

machine learning is an open research topic. Finally, some key future directions and research challenges were identified for functionality-preserving machine learning.

The survey of relevant literature revealed that adversarial machine learning is a complex and highly dynamic landscape. The literature shows that most machine learning models are susceptible to adversarial examples. Indeed, the review of the literature stressed that the robustness of models to adversarial examples is a key concern. The survey identified that the distribution, quality, quantity, and complexity of dataset samples on which a model is trained can impact the trustworthiness and robustness of the model. Simple resampling techniques and cross-validation may have a role to play in improving robustness. The survey found that adversarial examples in constrained domains is still an open problem. Moreover, although there are many proposed defences, with the exception of dropout most can be evaded and are not scalable. The clear need for effective defences remains an open problem. Furthermore, relatively few researchers address the key challenge of blocking transferability.

Chapter 3

Feature Vulnerability and Robustness Assessment

Following the review of the literature, improving robustness was identified as an important research avenue. This chapter considers how feature selection could help identify vulnerable features and improve robustness in machine learning. This section begins to examine the forms of adversarial learning that are used, and methods for assessing feature vulnerability and robustness in machine learning models.

3.1 Introduction

Computerised systems are a feature of everyday life in all sectors, globally, including defence, energy, finance, health, and government. Adversaries such as criminals and advanced persistent threats deliberately explore network vulnerabilities, often gaining access to systems over computer networks and causing unwanted events. The European Union Agency for Cybersecurity [168] list common network attack scenarios including: Web-based attacks, Denial-of-Service, and Botnets. Public and private organisations must inhabit this threat landscape, The recent SolarWinds supply chain attack identified in December 2020 [40] [41] indicates organisation's reliance on intrusion detection software, and the impact of successful attacks. Current intrusion detection systems utilize machine learning techniques using labelled datasets to block such malicious traffic.

A competent model must correctly identify malicious traffic, whilst not misclassifying benign traffic. This is related to the concepts of precision and recall. Precision measures the reliability of positive predictions, while recall measures the model's ability to accurately predict *all* positive examples. In practice, there is always a trade-off between precision and recall. The trade-off exists because adjusting the threshold for positive classification will inevitably result in the identification of additional negative samples as members of the positive class. Therefore, recall is increased at the expense of precision, and vice-versa. A model that predicts malicious traffic when it is mildly confident will likely have high recall but low precision, meaning some benign traffic will be flagged as malicious; alternatively, a model that predicts malicious traffic only when it is certain will have low recall and high precision, meaning that some malicious packets will not be identified.

As previously stated in Chapter 1, Szegedy *et al.* [22] discovered naturally occurring and intriguing properties of neural networks, discovering that imperceptible perturbations to input values can cause differences in the output of the neural network. Neural network based machine learning systems are therefore vulnerable to carefully crafted noise, known as adversarial examples [23]. Adversarial examples are a form of evasion attack that relies on small perturbations to the original data, often undetectable to human observers. Adversarial examples can be algorithmically generated. Indeed, there are a selection of algorithms that produce adversarial examples, including Fast Gradient Sign Method [23] and Jacobian Saliency Map Attack. Moreover, Papernot et al. [85] developed a software library for the easy generation of adversarial examples.

Much of the existing work on adversarial learning is applied to computer vision tasks, such as image classification [84, 96, 169], and even well-trained models such as Microsoft's Common Objects in Context (COCO) [170] can be susceptible to adversarial attacks [27]. A fundamental issue is that images contain a significantly large amount of data (i.e., pixels) that would be used by the classifier - for example, a single 1080p colour image would have over 6 million input values for a classifier.

Furthermore, subtle variations in such values would unlikely be noticeable to humans, due to colour perception issues [171]. A primary focus in this work is how adversarial attacks against a machine learning classifier can translate to other domains, such as cyber security or network traffic analysis, where an attacker may exploit weaknesses in a IDS classifier's performance, essentially masquerading an attack as benign activity.

In this work The CICIDS2017 dataset is used [66] [37] to explore the potential security risks introduced through the use of machine learning systems. In Chapter 2 the accuracy metric was defined (Eq. 2.5). In this Chapter the trade-off between accuracy and the number of features is explored. The focus is on adversarial examples and how feature selection can improve the robustness of machine learning systems. The main contributions of this work are:

- Considering countermeasures against algorithmically generated adversarial examples, and in particular FGSM.
- Demonstrating an inverse relationship between the number of features and robustness against adversarial examples.
- Identifying that applying systematic feature selection for model training can improve the robustness of the model against adversarial examples.

3.2 Related Work

This related works section addresses relevant topics of adversarial attacks, defences, and visual analytics. It considers both white-box and black-box. The defences considered include architectural defences and those that use feature selection. This section also briefly examines how visual analytics could in assessing classifier performance.

3.2.1 Adversarial Attacks

Adversarial attacks can be classified as either white-box or black-box. The former requires an attacker to have access to the target model's parameters whereas the

latter does not require such access. Black-box attacks may use a different model, or no model at all. Black-box strategies can employ the transferability of adversarial examples, where adversarial examples generated against one model can be successfully used to attack the main target model [172]. Ayub *et al.* [27] built a multi-layer perceptron supervised machine learning model to detect and classify benign and malicious traffic using two distinct network-based intrusion detection system (IDS) datasets (CICIDS2017 [66] and TRAbID [173]). They achieved accurate classification results; however, via using Jacobian-based Saliency Map Attack [98] they reduce the accuracy by 22.52% and 29.87% on the CICIDS2017 and TRAbID datasets, respectively. Thus, demonstrating the ease of evading network defence, and the importance of countermeasures.

The fast gradient sign method (which is a white-box method) is designed to be fast rather than optimal. Therefore, generated adversarial examples may be based on the minimal perturbations. For each feature the gradient of the loss function is used to determine whether increasing or decreasing the feature's intensity would minimize the loss function. All features are shifted simultaneously. Kurakin *et al.* [122] refined the fast gradient sign method by taking multiple smaller steps.

Quereshi *et al.* [174] aim to understand the impact of adversarial examples. They build a random neural network-based adversarial intrusion detection system, before training it on the NSL-KDD dataset. Subsequently they craft adversarial examples using JSMA using the Cleverhans Python library. Their benchmark was highly accurate (95.6% benign and 96.61% denial-of-service). Under JSMA attack their system accuracy fell to 47.58% for benign and a minimum of 28.10% for other attack classes. They note the poor results under JSMA were due to class imbalance in the benchmark data. Further they suggest that better feature extraction techniques could improve accuracy.

3.2.2 Architectural Defences

Lillicrap *et al.* [141] proposed a mechanism called feedback alignment that introduced a separate feedback path through random fixed synaptic weights. Gradient based attacks rely on the quality of the gradient to determine a possible attack; the use of feedback alignment which does not use weight transport thus increases robustness against adversarial examples.

Amer and Maul [154] proposed modifying the architecture of Convolutional Neural Networks (CNN) by adding a weight map layer. Their proposed layer can be easily integrated into existing CNNs. A weight map layer may be inserted between other CNN layers; thus, increasing the network's robustness to both noise and gradient-based adversarial attacks, whilst maintaining accuracy.

Dropout is often used during training to improve test accuracy, particularly where over-fitting is seen due to limited training data; however, Wang *et al.* [153] propose defensive dropout at test time to harden deep neural networks against adversarial attacks. There is an inherent trade-off between defensive use of dropout and the test accuracy; however, a relatively small decrease in test accuracy can significantly reduce the attack success rate. Furthermore, the added perturbations to the inputs might be significant and therefore recognized by humans.

3.2.3 Feature Selection

Hamed *et al.* [175] integrate feature selection into a intrusion detection system, aiming to select the most informative features to assist in the detection of "zero-day" attacks where few attack samples are available. They consider Recursive Feature Addition (RFA) and bigram technique (using two adjacent elements from a string) training their model on the ISCX 2012 dataset. Their objective was to find combinations of features that do not necessarily give good accuracy results independently but work very well as part of a set of selected features.

Farahani [176] uses an intrusion detection system case study to propose a novel cross-correlation-based feature selection and compare it against the cuttlefish algorithm (CFA), and mutual information-based feature selection (MIFS). The selected features are used with four classifiers: support vector machines, naive bayes, decision tree, and K-nearest neighbour. They use four datasets: KDD Cup 99, NSL-KDD, AWID, and CICIDS2017. Their results show their proposed method has better accuracy, precision, recall, and F1-score when compared against CFA and MIFS.

Almomani [93] proposes a feature selection model for network intrusion detection systems utilising genetic algorithm, parallel swarm optimisation, and other bio-inspired algorithms to improve the performance of network intrusion detection systems. They use the UNSW-NB15 dataset and evaluate their selection model on support vector machine and J48 classifiers. They show that accuracy can be maintained with fewer features. The best results of their study for F-measure, accuracy and sensitivity were achieved using with generated feature-sets of 30 and 13 features.

3.2.4 Visual Analytics

Legg *et al.* [177] studied visual analytics-based active learning as a means of assessing robustness in classifier performance with limited samples. Such visual analytic tools can also inform where genuine vulnerabilities in machine learning performance may be introduced due to weaknesses in the training data.

Yoo *et al.* [178] propose an interactive visual analytics tool to allow users to visually analyse the type, period, traffic, and frequency of attacks answering the challenge of handling and analysing vast number of logs. They argue that the tool can be useful and show how a Denial-of-Service attack was successfully identified and subsequently blocked.

3.3 This Work

Most previous work aims at understanding the impact of adversarial examples or improving accuracy under "normal conditions" through the use of feature selections. Within the realm of cyber security and adversarial machine learning, the existence of malicious classes of traffic is very common and even anticipated. Therefore, in this context, the concept of “normal conditions” pertains to conventional ML model inputs that are unmodified by adversarial machine learning techniques. In this work the author addresses a different problem of improving the robustness of machine learning models against adversarial machine learning attacks. This work utilises the relatively recent CICIDS2017 dataset. This dataset is examined using PCA, t-SNE, UMAP, and parallel co-ordinate plots. The focus is on feature selection using RFE as described in Algorithm 1. The algorithm is composed of two stages: first, the features with the largest perturbations are found; second, the features are recursively eliminated, starting with the feature with the largest mean perturbation, and the Mean-Squared Error is stored.

The aim is to improve robustness against FGSM attack. FGSM is a very common algorithm and was selected for its simplicity. FGSM uses gradient descent which is common across a range of other algorithms. The simplest form of FGSM modifies all features; however, the algorithm can be adapted to modify a subset of features through the use of a feature mask to exclude features that should not be perturbed. The model’s robustness against adversarial examples is measured using the accuracy metric. The accuracy metric is selected here because it is an easily understood simple metric. Later, in Chapter 4 the F1-Score metric is used to assess classifier performance more fully. In this chapter consideration is given to perturbation size, aiming to determine whether feature selection could force more overt adversarial examples, that could hopefully be more easily noticed by network operations engineers.

Algorithm 1: Recursive Feature Elimination

```
# Generate AdvX from DDoS Samples
X_adv = FGSM(model,ddos,epsilon)
for row in ddos do
    for feature in ddos do
        # Calculate and store the absolute difference per feature.
        diff[row, feature] = abs(ddos[row, feature] − adv_x[row, feature])
        # Calculate and store the mean feature difference
        mean_feature_diff = diff.mean()
    end
end

# Sort the array by the mean difference
features_sorted_biggest_difference = mean_feature_diff.sort()

# Do Recursive Feature Elimination

for feature_to_drop in features_sorted_biggest_difference do
    dataset = dataset.drop(columns=feature_to_drop)
    X = dataset
    scaler ← MinMaxScaler().fit(X)
    X_scaled ← np.array(scaler.transform(X))
    # Train the model
    kfold ← StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
    for i in range(len(validation_set)) do
        if current_class equals 'DDoS' then
            # Generate Adversarial example
            X_adv = FGSM(model,validation_set,epsilon)
        end
    end

    # Calculate Mean-Squared Error
    error = (validation_set − X_adv)
    mse = MSE(error)
    store features,accuracy,mse for plots
    Plot features(x) and accuracy (y)
    Plot features(x) and mse (y)
end
```

3.4 Method

A generalizable approach is proposed for examining the robustness of features against adversarial attacks in the context of a machine learning classifier. The characteristics of the derived features from the data are examined and assessed to determine how these are manipulated by adversarial learning attacks. Based on these observations, a feature selection approach is derived that seeks to maintain classifier accuracy whilst maximising the amount of feature perturbation required to manipulate a classifier, hence improving robustness since the attack can no longer be performed in a subtle and discrete manner.

3.4.1 Dataset

The CICIDS2017 dataset [66] is used. One advantage of this dataset is that statistical time-related statistics have been calculated for both forward flows (client to server) and backward flows (server to client). Typical features in each flow are: Destination Port, Protocol, Flow Duration, Packet Statistics, Flow Bytes/s, Flow Packets/s, IAT Statistics, Flags, Header Length, Down/Up Ratio, Bulk Statistics, Subflow Statistics, Init Win bytes, act data pkt fwd, Active Statistics, and Idle Statistics. The flows are labelled with fifteen discrete classifications of traffic as shown in Table 3.1.

Traffic Type	**Number of Samples**
BENIGN	20,000
Bot	1,500
DDoS	1,500
DoS GoldenEye	1,500
DoS Hulk	1,500
DoS Slowhttptest	1,500
DoS slowloris	1,500
FTP-Patator	1,500
Heartbleed	11
Infiltration	36
PortScan	1,500
SSH-Patator	1,500
Web Attack Brute Force	1,500
Web Attack SQL Injection	21
Web Attack XSS	652

TABLE 3.1: CICIDS2017: Traffic Types and Number of Samples

The DDoS class is now considered. To further understand the difference between benign and malicious data features, violin plots can be used for comparative analysis to examine the distribution of each feature for each class. A violin plot is a hybrid of a box plot and a kernel density plot that can depict summary statistics and the density of each feature. Where a white dot can be seen this represents the median value. The thick line represents the inter-quartile range (IQR) and the thinner line represents the distribution excluding any outliers as determined as a function of the IQR. Wider sections of the plot represent higher probability of those values and more slender sections represent lower probability. The violin plots for the benign and DDoS classes are shown in Figure 3.1 and Figure 3.2.

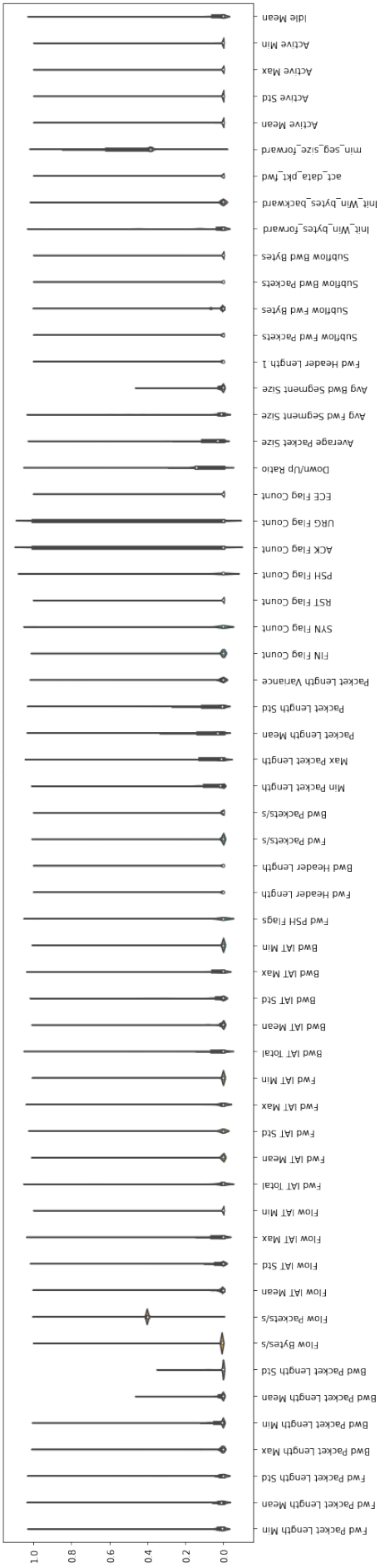


FIGURE 3.1: Benign - This series of violin plots shows the wide distribution of features in the benign class. Some feature values are statistically more likely as represented by the wider sections of the violin plots. The length of the violin plots illustrate the wide scope of features in the benign class.

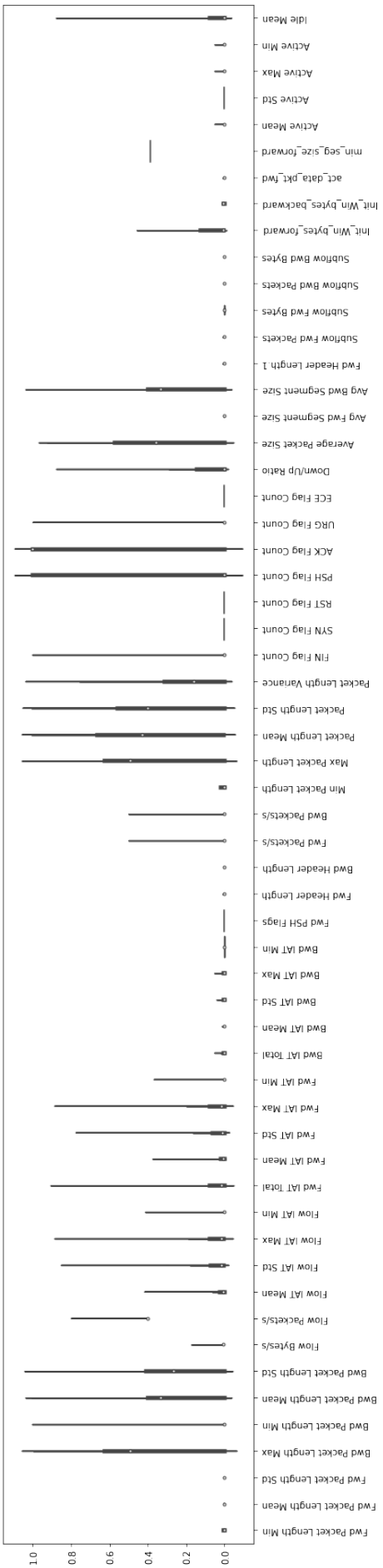
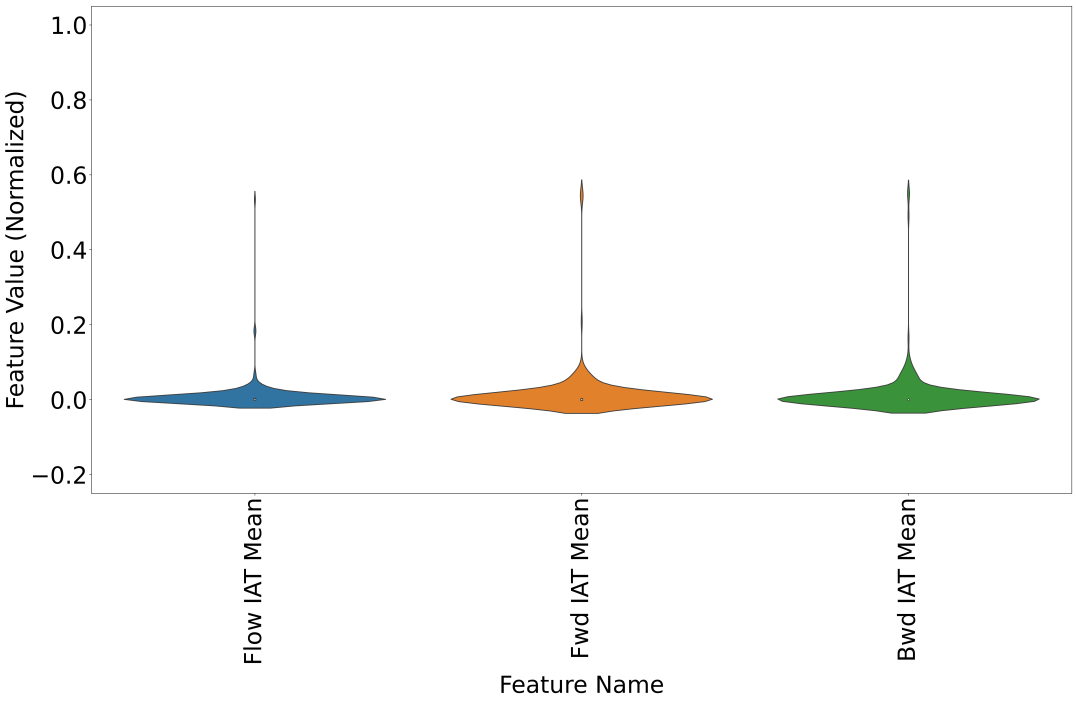


FIGURE 3.2: DDoS - This series of violin plots shows the narrower distribution of features in the DDoS class. The range is narrower and more evenly distributed than the features in the benign class. This plot illustrates that the distribution of DDoS features overlaps with the distribution of the features in the benign class. Examining the white space around the distributions of the features exposes a margin that could potentially be exploited by adversarial examples.

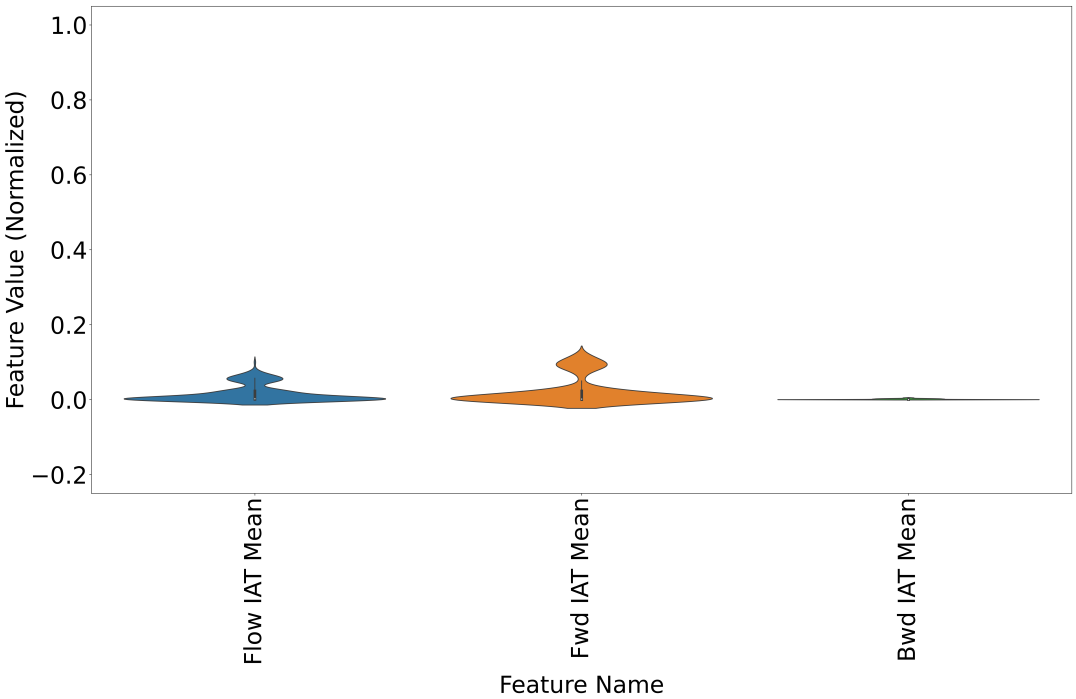
Standardization and Normalization are commonly used in machine learning to pre-process and transform by scaling the feature data before training a model. Standardization has the disadvantage that it is less sensitive to outliers. Normalization prevents bias, ensuring all features contribute proportionally to the learning process. Normalization is sensitive to outliers, which are the essence of adversarial examples. Normalization was chosen to maintain this sensitivity. Therefore, all features are “Normalized" (scaled between zero and one), and then separated by class, so that the scale factor for each feature is comparable for each of the violin plots shown.

Firstly, it can be seen that the distribution of features for the benign class in Figure 3.1 is much greater than in the malicious case in Figure 3.2. Furthermore, there is significant overlap between the two classes. The malicious class is essentially a subset within the distribution of the benign features. Plausibly, the Inter-arrival time (IAT) features are good indicators of DDoS traffic because DDoS traffic inherently has a short time between DDoS traffic. Closer inspection of the IAT features reveals that the feature distributions for the DDoS class are narrower than the corresponding distributions for the benign class. This suggests that the IAT features may be a distinguishing feature between the two classes.

Figure 3.1 and Figure 3.2 are useful in providing an overview of all the features; however, at this scale it can be unclear and difficult to understand the plots. This is in part because the plots are intricate and narrow. To clarify the story that the violin plots tell, a spotlight is thrown on three features: Flow IAT Mean, Fwd IAT Mean, and Bwd IAT Mean. The features are the statistical mean of the inter-arrival time for the flow, forward direction, and backward direction. Figure 3.3 focuses on these features. This enables easier identification of the differences in the distributions of benign and DDoS traffic. For example, where the distributions overlap or diverge. These features might help a ML model discriminate between the benign and DDoS classes.



(A) Benign IAT Mean Features



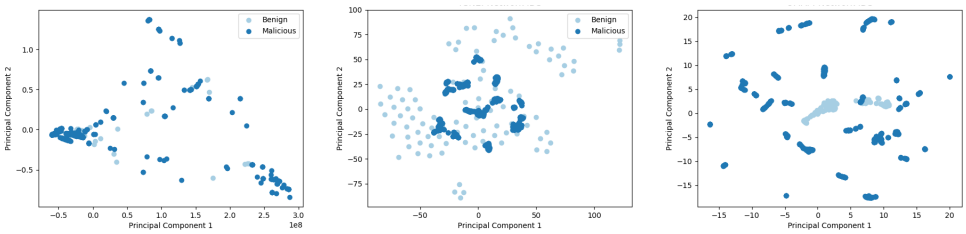
(B) DDoS IAT Mean Features

FIGURE 3.3: The violin plots in this figure show the differences between the distributions of the three features: Flow IAT Mean, Fwd IAT Mean, and Bwd IAT Mean. This plot illustrates that the distribution of DDoS features overlaps with the distribution of the features in the benign class. Examining the white space around the distributions of the features exposes a margin that could potentially be exploited by adversarial examples.

A model is trained to distinguish between benign and DDoS traffic. In order to speed the training of the model, the size of the DDoS dataset is reduced by randomly selecting 50,000 samples. Through selecting this reduced number of samples, a more balanced dataset is created with 52% of samples labelled benign and 48% of samples labelled DDoS. This is an improvement over the unmodified dataset percentages (43% benign, 57% DDoS), due to the diminished disparity between the number of class samples. The dataset is further cleaned to remove null and not applicable data.

3.4.2 Feature Analysis

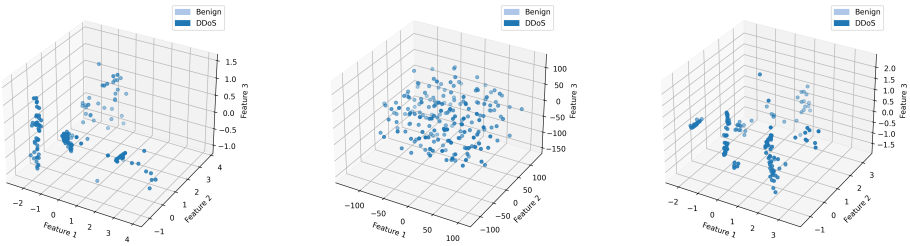
Dimensionality reduction is a common first step when analysing datasets. For convenience, one hundred random samples of each class in the CICIDS2017 dataset were extracted and separated into either benign or malicious. Subsequent analysis as shown in Figure 3.4 and Figure 3.5 suggests that this sample is sufficiently indicative because the two classes are not trivially separable; although, via dimensionality reduction and visualization the two classes can begin to be separated in higher dimensionality space. Notwithstanding, a larger random sample would have been ideal. The data are examined using dimensionality reduction methods such as Principal Component Analysis (PCA), t-Distributed Stochastic Neighbourhood Embedding (t-SNE), and Unified Manifold and Projection (UMAP), as shown in Figure 3.4. These three methods are commonly used for dimensionality reduction, allowing for visualization of the data on a 2D or 3D plot. PCA [179] is a well-known algorithm that works by identifying the hyper-plane lying closest to the data, and projecting the data onto it. Thus, largely retaining the variation in the dataset. The t-SNE algorithm [180] finds clusters in the data, reducing dimensionality whilst aiming to keep similar instances together and dissimilar instances apart [180]. UMAP [181] is an effective algorithm for visualizing clusters of data points, usually providing faster and better visualizations than t-SNE.



(A) PCA (B) t-SNE (C) UMAP

FIGURE 3.4: These plots show three common dimensionality reduction methods with improving clustering of the benign and malicious classes: PCA, t-SNE, and UMAP. The malicious samples are represented as dark-blue, and the benign samples are represented as light-blue. The complexity of the classification problem is illustrated by the benign and malicious samples occupying the same subspace.

In the PCA plot (Figure 3.4a) we see malicious traffic gathered and occupying the same subspace as benign traffic, showing the complexity of the classification problem. More sophisticated methods such as t-SNE (Figure 3.4b) and UMAP (Figure 3.4c) begin to identify the clustering of the two classes in greater detail, however even so, it is noticeable that there is no clear single cluster that can be associated with either class. This is an important observation as there is no single definition of what makes for benign or malicious traffic in respect to the features being studied within the dataset. As mentioned earlier PCA, t-SNE, and UMAP can also be used to plot 3-dimensional plots. To examine whether a 3D plot provides a different view of the relationship between benign and malicious classes. The 3D plots in Figure 3.5 show that the classes are visibly more separable; in higher dimensional space separate clusters are beginning to become clearer.



(A) PCA (B) t-SNE (C) UMAP

FIGURE 3.5: These 3D plots show three common dimensionality reduction methods: PCA, t-SNE, and UMAP. The malicious samples are represented as dark-blue, and the benign samples are represented as light-blue. The plots show improved clustering of the benign and malicious classes in the higher dimensional 3D space.

3.4.3 Parallel Co-ordinates

Therefore, the raw features are considered rather than the dimensionally reduced form. The IAT features were identified as potentially good indicators of DDoS traffic, a choice was made to plot the features as parallel coordinates as seen in Figure 3.6.

The subset of features that contain the text 'IAT' were selected, and a parallel coordinates plot was used to examine the relationship between the two classes further. In Figure 3.6 each plotline represents an individual instance from the dataset. This plot provides an alternative and simpler view that focuses on the IAT features of both classes. The distribution of the DDoS features is narrower than the corresponding distribution of the benign features, indicating that the distributions of DDoS features overlap with the distributions of benign features. On this basis successful adversarial examples might suitably perturb DDoS samples such that they could masquerade as benign samples.

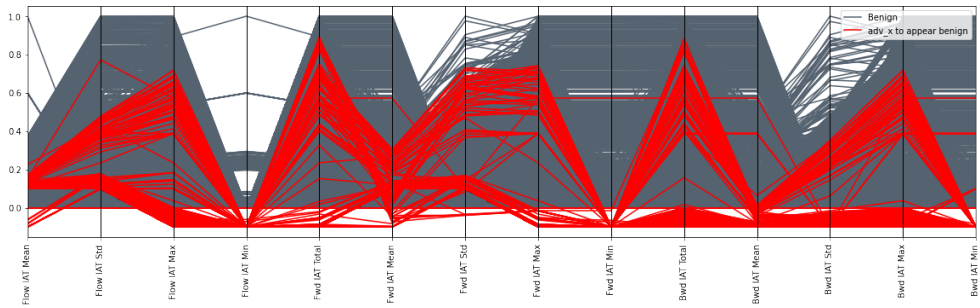


FIGURE 3.6: This parallel coordinates plot of the distributions of benign and perturbed DDoS focuses on the IAT features of both classes. The distribution of the DDoS features is narrower than the corresponding distribution of the benign features, indicating that the distributions of DDoS features overlap with the distributions of benign features. On this basis successful adversarial examples might suitably perturb DDoS samples such that they could masquerade as benign samples.

Having identified some features as better indicators of DDoS traffic, an initial study on accuracy and MSE was performed. Further, feature selection is proposed as a method to improve the classification accuracy against FGSM adversarial attacks. Recursive Feature Elimination (RFE) is considered [182], removing those

features with the largest absolute difference under FGSM attack.

3.4.4 Training the Model

The CICIDS2017 DDoS dataset was adapted and used to train a binary classifier to discriminate between benign and distributed denial-of-service (DDoS) traffic. The model was trained using shuffled stratified k-fold ($k = 5$), giving confidence of the validity of the results. The k value of 5 was selected aiming to strike a balance between long run times and reduced sample bias. Each iteration was trained with an 80/20 training/test split, yielding a training set of 180,596 samples and a validation set of 45,149 samples for each iteration. This showed encouraging results (100.00% +/- 0.00%). The initial model is trained on 76 features as shown in Table 3.2.

This well-trained model is considered a baseline; the results of the experiment were compared to this baseline. FGSM was applied to generate adversarial examples from the DDoS samples, again using k-fold ($k = 5$). Such adversarial examples significantly reduce the accuracy of the classifier, yielding an accuracy of 58.57% (+/-) 15.03%. Using the trained model with $x = 76$ features FGSM was applied and the perturbation of each feature was assessed. The feature with largest perturbation was removed and the classifier was iteratively retrained with $x - 1$ features. This iterative algorithm is described fully in Algorithm 1. The optimal solution, maximising accuracy was found. This illustrates a simple attack; however, without *masking* some features from perturbation it is not realisable in a real-world intrusion detection scenario.

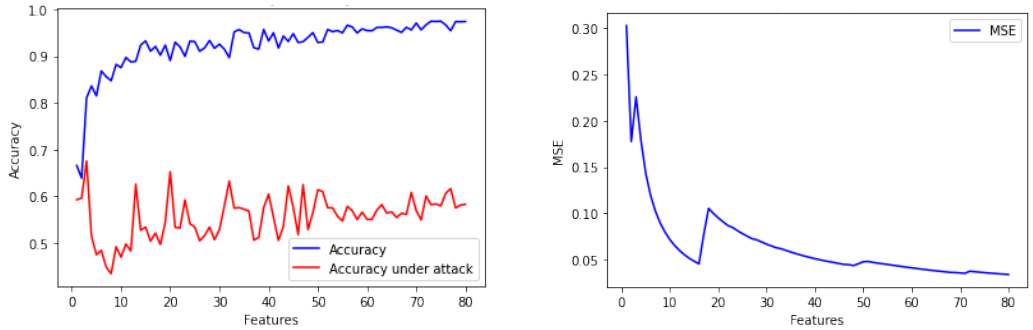
3.5 Results and Discussion

Here, the findings are described and discussed. First, the findings from the initial accuracy and Mean-Square Error study are discussed, followed by the findings from the recursive feature elimination experiments.

Feature	Name	Feature	Name
0	Destination Port	41	FIN Flag Count
1	Flow Duration	42	SYN Flag Count
2	Total Fwd Packets	43	RST Flag Count
3	Total Backward Packets	44	PSH Flag Count
4	Total Length of Fwd Packets	45	ACK Flag Count
5	Total Length of Bwd Packets	46	URG Flag Count
6	Fwd Packet Length Max	47	CWE Flag Count
7	Fwd Packet Length Min	48	ECE Flag Count
8	Fwd Packet Length Mean	49	Down/Up Ratio
9	Fwd Packet Length Std	50	Average Packet Size
10	Bwd Packet Length Max	51	Avg Fwd Segment Size
11	Bwd Packet Length Min	52	Avg Bwd Segment Size
12	Bwd Packet Length Mean	53	Fwd Header Length.1
13	Bwd Packet Length Std	54	Fwd Avg Bytes/Bulk
14	Flow IAT Mean	55	Fwd Avg Packets/Bulk
15	Flow IAT Std	56	Fwd Avg Bulk Rate
16	Flow IAT Max	57	Bwd Avg Bytes/Bulk
17	Flow IAT Min	58	Bwd Avg Packets/Bulk
18	Fwd IAT Total	59	Bwd Avg Bulk Rate
19	Fwd IAT Mean	60	Subflow Fwd Packets
20	Fwd IAT Std	61	Subflow Fwd Bytes
21	Fwd IAT Max	62	Subflow Bwd Packets
22	Fwd IAT Min	63	Subflow Bwd Bytes
23	Bwd IAT Total	64	Init_Win_bytes_forward
24	Bwd IAT Mean	65	Init_Win_bytes_backward
25	Bwd IAT Std	66	act_data_pkt_fwd
26	Bwd IAT Max	67	min_seg_size_forward
27	Bwd IAT Min	68	Active Mean
28	Fwd PSH Flags	69	Active Std
29	Bwd PSH Flags	70	Active Max
30	Fwd URG Flags	71	Active Min
31	Bwd URG Flags	72	Idle Mean
32	Fwd Header Length	73	Idle Std
33	Bwd Header Length	74	Idle Max
34	Fwd Packets/s	75	Idle Min
35	Bwd Packets/s	-	-
36	Min Packet Length	-	-
37	Max Packet Length	-	-
38	Packet Length Mean	-	-
39	Packet Length Std	-	-
40	Packet Length Variance	-	-

TABLE 3.2: The Feature-set of 76 features used to train the initial model.

Figure 3.7a shows that the model achieves acceptable accuracy (rarely falls below 90%) with fewer features. The model is most accurate when most features are used; however, accuracy under attack rarely exceeds 60%. This shows that a

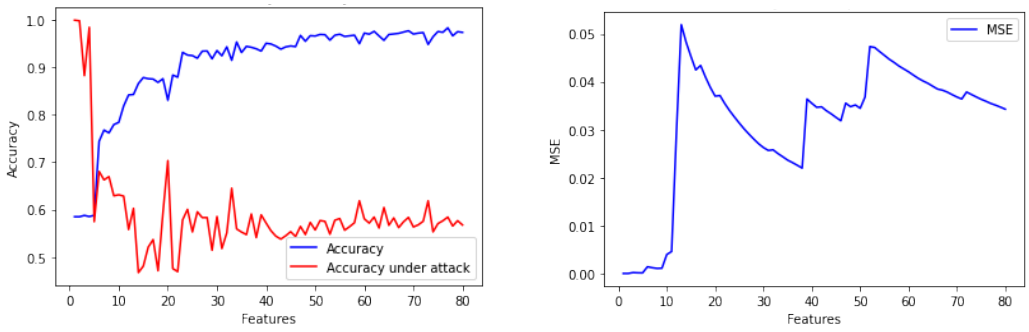


(A) Accuracy (B) MSE

FIGURE 3.7: In these figures the features are unsorted per original dataset. Plot (A) shows the relationship between features and accuracy. Plot (B) shows the relationship between features and the MSE. The more features are present, generally the size of perturbation is smaller. As fewer features are selected the perturbation size trends toward larger values. Small perturbations can be more easily overlooked and larger perturbations are more overt and therefore more easily detectable.

well-trained model is susceptible to FGSM. The graphs show an increase in accuracy under normal conditions often correlates with a decrease in accuracy under attack. Note that when Fwd Inter Arrival Time (IAT) Total (Feature 18 as listed in Table 3.2) is included, the accuracy under attack improves as shown by a spike. This accuracy boost is considered as a result of the inherent short IAT of Denial of Service (DoS) traffic, strongly indicating DoS traffic.

Figure 3.7b shows the size of perturbation required for a successful attack. The complete and unsorted feature set yields an MSE of approximately 0.05. The size of perturbation tends to reduce as more features are included. The addition of features increases the attack surface and allows more subtle adversarial perturbations. The classification results of such systems have serious consequences. Adversaries able to skew the classification accuracy of systems can leverage an advantage by making malicious conditions appear benign. As previously seen with accuracy, note an increase in perturbation size when Fwd Inter Arrival Time (IAT) Total (Feature 18 as listed in Table 3.2) is included. This feature strongly indicates DDoS traffic. It is thought that the inclusion of important features for classification may also force increases in perturbation size. This in turn means an attack must be more overt.



(A) Accuracy (B) MSE

FIGURE 3.8: In these figures the features are sorted by most importance. Plot (A) shows the relationship between features and accuracy. Plot (B) show the relationship between features and MSE. The plot resembles an imperfect saw tooth. Relatively small perturbations are necessary until a sequence of spikes which are followed by a gradual declines in perturbation size. Individual features may have an effect; however, the grouping of features may influence the perturbation size more.

Now consider the experiments with feature-set arranged in order of importance. Figure 3.8a shows a binary classifier where features have been sorted according to the extracted feature importance. Note that with fewer than five features the model predicts the class incorrectly around 40% of the time. This is a poor binary prediction model. The binary FGSM attack aims to flip the recognized class. Therefore, for poor classifiers the accuracy can curiously increase under FGSM attack. As more features are included, the accuracy wavers depending on specific properties of those features. A roughly inverse relationship between accuracy and the accuracy under FGSM attack is observed. Where accuracy falls, this coincides with an increase in accuracy under attack and vice versa. This graph is used to determine a set of features providing good accuracy under FGSM attack, whilst retaining acceptable accuracy under normal conditions. There is a promising peak at Feature 20 (_Bwd_Packet_Length_Min as shown in Table 3.3). The cumulative feature-set of 20 most important features is shown in Table 3.3.

This feature-set provides respectable accuracy under FGSM attack, whist maintaining acceptable accuracy under normal conditions. Drawing attention to either side of this peak, note drops in accuracy with the removal of _Bwd_Packet_Length_Min (19 features), or the addition of _Flow_Duration (21

Rank	Name	Meaning
1	Total_Length_of_Fwd_Packets	Length of Forward Packets
2	_CWE_Flag_Count	Congestion Window Flag
3	_Fwd_Packet_Length_Max	Max Forward Packet Length
4	_Bwd_Avg_Bytes/Bulk	Average No. Backward Bytes/Bulk
5	Subflow_Fwd_Packets	Number of Forward Packets in a Subflow
6	_Flow_IAT_Max	Maximum Inter Arrival Time for a flow
7	_Subflow_Bwd_Packets	No. Backward Packets in Subflow
8	_Total_Fwd_Packets	Total Forward Packets
9	_Flow_IAT_Mean	Mean Inter Arrival Time
10	Fwd_PSH_Flags	Forward Push Flag
11	_Down/Up_Ratio	Down/Up Ratio
12	_Source_Port	Source Port
13	_Protocol	Protocol
14	_Fwd_Packet_Length_Min	Minimum Forward Packet Length
15	_Flow_IAT_Std	Inter Arrival Time Standard Deviation for flow
16	_Flow_IAT_Min	Minimum Inter Arrival Time Standard Deviation for flow
17	_URG_Flag_Count	Urgent flag count
18	_Fwd_Avg_Bulk_Rate	Average No. Forward Bytes/Bulk
19	_Subflow_Fwd_Bytes	No. forward bytes in subflow
20	_Bwd_Packet_Length_Min	Minimum backward packet length

TABLE 3.3: Feature-set of 20 most important features.

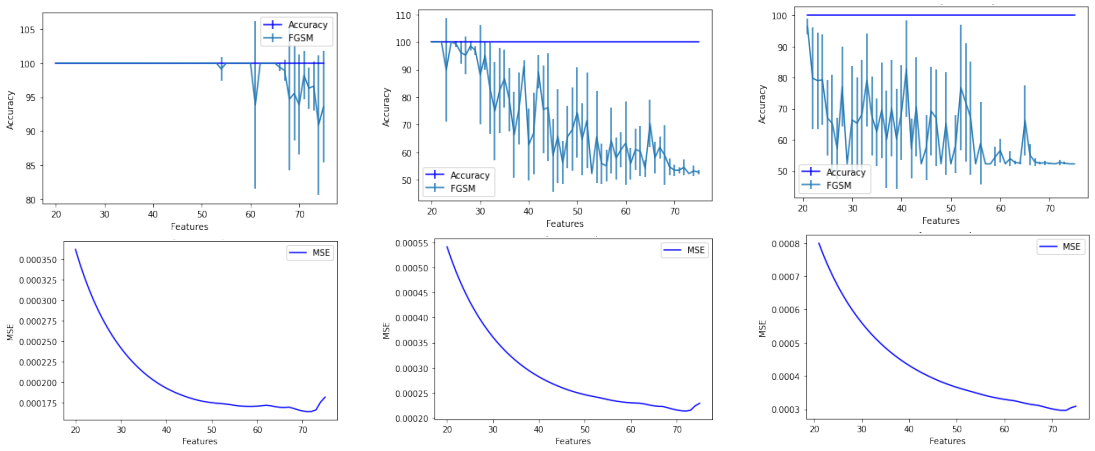
features). The inclusion of _Bwd_Packet_Length_Min (20 features) gives a local maxima for accuracy under FGSM attack. Whilst maintaining acceptable accuracy ($\approx$ 85%) under normal conditions. The _Bwd_Packet_Length_Min feature may indicate DDoS traffic through the size of returned packets. Each feature in isolation may not be an excellent indicator of DDoS traffic; however, in combination a distinct pattern may emerge. For example, a ping flood attack is performed by quickly sending a large multiple of small request packets gaining an equal number of response packets. Packet Length, Number of Packets, IAT, and Down/Up Ratio - when combined could reveal such traffic. These features are well represented in the generated feature-set (Table 3.3).

Figure 3.8b shows the accompanying plot of the perturbation size by number of features, resembling an imperfect saw tooth. This graph is used to determine a feature-set maximising the perturbation size of successful FGSM attacks. The plot shows relatively small perturbations are necessary until a significant spike occurs

with _Source_Port (Feature 12 in Table 3.3), followed by a gradual decline until another spike at _Subflow_Bwd_Bytes (Feature 40). Further peaks and gradual declines are seen with _Packet_Length_Variance (Feature 50) and _Bwd_IAT_Min (Feature 70). Again, individual features may have an effect; however, in consideration, grouping features in a cumulative feature-set effects the perturbation size more. Note that the maximum size of perturbation is smaller when using the feature-set sorted by feature importance (0.05) compared against the maximum perturbation of the unsorted feature-set (0.30). The cumulative feature-set detailed in Table 3.3 peaks with _Source_Port (feature 12). Looking either side of this feature, note the removal of _Down/Up_Ratio or the addition of _Protocol. The inclusion of _Source_Port gives a local maxima for Mean-Squared Error ($>$0.05). Note that the cumulative feature-set of 20 features yields a relatively high Mean-Squared Error ($\approx$ 0.04); however, this value is much lower than the Mean-Squared Error yielded by the unsorted feature-set at Feature 20 ($\approx$ 0.12). The MSE is decreased in the sorted feature-set. The improved accuracy under FGSM attack is an effect of selecting and grouping features. Forced increases in perturbation size may also have a smaller effect.

3.5.1 Feature Selection

There are many types of feature importance, which highlight which features may be most/least relevant. Through identifying the relevance of features insights can be gleaned on the dataset and model. These insights are used to improve the predictive model, by discarding features more susceptible to the FGSM attack. It is known that reducing the number of features can yield benefits including: reduced time required to train a model [183], improved accuracy, and reduced execution time [93]. It is further explored whether robustness can be improved through feature selection. Three common methods for determining feature importance are: model coefficients, decision trees and permutation testing. This work focuses on the latter.



(A) Accuracy(top) and Average perturbation size per feature(bottom) $\epsilon = 0.05$

(B) Accuracy(top) and Average perturbation size per feature(bottom) $\epsilon = 0.10$

(C) Accuracy(top) and Average perturbation size per feature(bottom): $\epsilon = 0.15$

FIGURE 3.9: Accuracy and average perturbation per feature for feature sets of decreasing size, with ϵ values of: 0.05, 0.10, and 0.15.

Here, the findings from the RFE experiments are discussed. Figure 3.9 shows plots of accuracy and average perturbation per feature. In the context of adversarial examples generated with FGSM, the perturbation size refers to the magnitude of the distortion or noise applied to the input data. The purpose of the perturbation is to cause the classifier to misclassify a sample. For example, to misclassify malicious samples as benign. When using FGSM, the trade-off between the effectiveness of the adversarial example and its perceptibility is managed by the epsilon (ϵ) value which controls the maximum distortion allowed to the input. Larger epsilon values generally have a higher likelihood of fooling the model, but produce more noticeable perturbations, which in turn are less likely to preserve the functionality of a malicious sample. The plots illustrate the effect of FGSM on accuracy for different values of epsilon (ϵ). The negative effect on classification accuracy increases with the size of ϵ. For $\epsilon = 0.10$ and $\epsilon = 0.15$ accuracy for the original dataset near 50%. It should be noted that for binary classification tasks an accuracy of 50% equates to a random guess. For all values of epsilon, an incremental increase in robustness against FGSM is observed. Such adversarial examples can be successfully mitigated with feature selection. Where $\epsilon = 0.05$ a feature-set of approximately 50 features is sufficient to negate the effects of FGSM. Where $\epsilon = 0.10$ a feature-set of approximately 20 features is sufficient to negate the effects of FGSM. Whereas for $\epsilon = 0.15$ a feature set of approximately 20 features is unable

to fully negate the effect of FGSM.

All values of ϵ show a similar trend of increased average perturbation per feature; however, note that as ϵ increases the average perturbation per feature decreases. This can be explained if perturbations are unevenly distributed across features. Large perturbations of a small set of features and small or no perturbations on other features give smaller average perturbations for the sample.

It has been shown that feature selection can improve classification accuracy under adversarial conditions. A DDoS case study was employed, using the CICIDS2017 dataset. Improvement in FGSM accuracy is shown by recursively removing the features most susceptible to large perturbations from the training set. This research shows an improvement from 58.57% +/- 15.03 to 78.63% +/- 8.23 (results at the edge of the standard deviation indicate FGSM accuracy 86.86%) with no drop in accuracy for unperturbed samples.

3.5.2 Interpreting Patterns in Parallel Co-ordinates

The parallel-coordinates plot in Figure 3.6 shows DDoS features modified by FGSM fall within the distribution of benign traffic for these features. It is clear that benign and malicious traffic cannot be separated based solely on the range of their features. Instead, it is theorised that that the correlations between features can help separate benign and malicious traffic. Significantly, a pattern of peaks and troughs emerge from the FGSM distribution. It is theorised that this pattern is more easily concealed in large feature sets where such patterns may be harder to detect.

3.6 Conclusion

This chapter demonstrated a generalizable approach for assessing the vulnerability and robustness of features in a machine learning context. In particular, adversarial machine learning attacks seek to identify subtle perturbations of features that can result in misclassification. The presented approach provides researchers with a suitable methodology for assessing how susceptible features may be towards

perturbation attacks, and how vulnerable features can be systematically removed, to simultaneously maintain acceptable classifier accuracy whilst eliminating features that may introduce subtle attack vectors. To demonstrate the concept, the approach was applied to a network traffic classification task to distinguish between malicious DDoS activity and benign traffic behaviours. Feature selection was successfully used to achieve improvement in accuracy under FGSM attack.

This chapter focuses on a static classification model. The model was trained on a dataset consisting of carefully selected features. This research shows that feature selection improves robustness of models against adversarial attacks; however, one weakness in this static classification model is that classification relies on the dataset containing previous labelled examples. This should not be relied upon. Zero-day attacks, which are unknown to the security community before their first use, do exist. Moreover, as the model and the current state of the art diverge, the models suffer from hidden technical debt and need retraining to reflect current state of the art attacks and new network traffic patterns [184]. So far binary classification has been examined.

The multi-class problem is more complex. A multi-class classifier must determine to which of many classes a suspect sample belongs. Moreover, an adversary can choose the target class for an adversarial example. This could be advantageous: a network analyst would certainly treat a DDoS attack differently than a BotNet or infiltration attempt. Adversaries could gain significant advantage through camouflaging an infiltration attack as a comparatively less serious network intrusion.

Chapter 4

Defending against Adversarial Machine Learning Attacks using Hierarchical Learning

Building on work in earlier chapters, this chapter considers adversarial learning against multi-class classification, where a novel hierarchical-based approach is proposed to improve model output robustness. This chapter explores the robustness benefit gained through using multiple separately trained machine learning models, when combined into an ensemble in the form of a two layer hierarchical classifier.

4.1 Introduction

As mentioned earlier, the protection of computer and network systems is fundamental for most organizations. Machine learning is becoming widely used for the detection and classification of malicious network activity to aid the response to cyber-attacks, where a mathematical model is learned to relate input feature observations to a set of possible output classes. Whilst machine learning can help manage this wealth of information, it is not without limitation. Recent years have seen a growing interest in the domain of adversarial machine learning [6] that seeks to identify well-crafted examples that knowingly force misclassification by a model. A challenge in adversarial learning is to determine which features are most susceptible such that a minimal change can result in misclassification by the model,

whilst the overall input to the model appears unchanged or unaltered to the human observer. This is related to recall because a model with high recall has more inclusive decision boundaries, which may make it especially susceptible to adversarial examples. Care has been taken so that the model can correctly identify original unperturbed malicious samples (and also the perturbed malicious samples). It is theoretically possible that a model might correctly classify adversarial examples but not the original unperturbed samples. This concern is addressed through careful selection of benign and malicious samples that are then perturbed, generating adversarial examples. Adversarial examples and unperturbed samples are both presented to the two models to ascertain model performance on both adversarial examples and unperturbed samples. In intrusion detection, a malicious attack should exhibit the same characteristics such that the activity is still deemed malicious, whilst identifying the minimal amount of perturbation in the derived features such that the model believes the observation to be benign, hence resulting in misclassification. This characteristic is referred to as *functionality preservation*.

This chapter first examines the impact of adversarial attack generation against a well-trained network traffic classification model and shows the performance degradation. To combat this, **a novel defensive strategy is presented that uses hierarchical learning to help reduce the attack surface that an adversarial example can exploit** within the constraints of the parameter space of the intended attack. The results show that the defensive learning model can withstand crafted adversarial attacks and can achieve classification accuracy in line with the original model when not under attack.

The primary contributions of this work are:

- A **new comprehensive study of applying functionality-preserving adversarial learning attacks** against a multi-class network traffic classification model, that demonstrates successful attack misclassification within a constrained attack parameter space. Over ninety percent (90.25%) of

the attacks were able to evade detection of a well-trained classifier, while also constraining the parameter space to preserve functionality.

- A **novel defensive strategy based on hierarchical learning** is proposed to reduce the attack surface that an adversarial example can exploit within the constraints of the parameter space of the intended attack, achieving classification accuracy in line with the original model when not under attack.

The remainder of the chapter is structured as follows: Section 4.2 presents the related works on adversarial machine learning exploring both the creation of attacks, and the defence against attacks; Section 4.3 presents a study on the creation of adversarial attacks against a well-trained network traffic classification model; Section 4.4 presents a novel defensive strategy based on hierarchical learning; Section 4.5 provides discussion of the research on model robustness, attack transferability, and limitations of hierarchical classification; and Section 4.6 concludes this chapter.

4.2 Related Work

This related works section addresses relevant topics of adversarial machine learning, functionality preservation with feature perturbations, localized classification, and machine learning-based intrusion detection. It also draws upon the author's previous works on surveying functionality-preserving attacks [42], and feature vulnerability and robustness in network traffic analysis [43].

4.2.1 Adversarial Machine Learning

Unfortunately, machine learning systems are susceptible to carefully crafted attacks that aim to yield an arbitrary, or specific, misclassification. Recall that Szegedy et al. [22] first found that imperceptible perturbations of input values can result in significant differences in the output of a ML classifier. Adversarial machine learning has been a research topic for over a decade and is now an accepted topic with open problems.

Papernot *et al.* [21] note the security and privacy of ML is an active but nascent area of research. In this early work, they systematize their findings on security and privacy in machine learning. Indicating that a science for understanding the vulnerabilities of ML and countermeasures is slowly emerging. They utilize the classical confidentiality, integrity, and availability (CIA) model to analyse: training in adversarial settings; inferring adversarial settings; robust, fair, accountable, and private ML models. Their analysis points toward two related notions of sensitivity. The sensitivity of learning models to their training data is essential for privacy-preserving ML, and similarly, the sensitivity to inference data is essential for secure ML. The crux of both notions of sensitivity is the generalization error. They focus on attacks and defences for machine learning systems aiming to further understanding of the sensitivity of modern ML algorithms to input data and foster a science of privacy and security in machine learning.

A primary focus of this work is how adversarial examples might translate to the cyber security domain. A further complication in this domain is that of functionality preservation. In cyber security domains, it is critical that an intrusion detection classifier correctly identifies malicious traffic while minimizing false positive and false negative results since identifying truly malicious activity in a profusion of false positives is problematic. ML performance is often evaluated by accuracy, precision, recall, and other metrics such as F1-Score.

Zhang and Li [48] discuss opportunities and challenges arising from adversarial examples. They survey state-of-the-art adversarial example generation methods and defences before raising future research opportunities and challenges. They note three challenges for adversarial example construction:

1. The difficulty of building a generalizable method.
2. The difficulty in controlling the size of perturbation (too small will not result in adversarial examples, and too large can easily be perceived).
3. Difficulty in maintaining adversarial stability in real-world applications (some adversarial examples do not hold for transformations such as blurring).

They identify black-box attacks as a challenge for defences. This is because black-box attacks require zero-knowledge of the model architecture and therefore might not be easily resisted by modifying the model architecture or parameters. Second, defences are often specific to a single attack method and are often less suitable as a general defence. They subsequently identify an opportunity for highly transferable adversarial examples (high confidence).

4.2.2 Threat Model

A *realistic* threat model can help harden intrusion detection systems by prioritising the smaller subset of attacks that are realistic [125]. The following threat model is presented considering the strengths and weaknesses of an adversary's possible strategies.

Adversarial attacks can be designated as either poisoning attacks or inference-time attacks. Poisoning attacks effect the training phase and aim to influence classification by augmenting the training dataset with new samples or modifying existing samples. Inference attacks aim to influence classification by leveraging the sensitivity of the model to its training data. Typical strategies include gradient descent [174] and Generative Adversarial Networks (GANS) [185]. Such attacks are commonly known as evasion attacks. The goal of an attacker may be to misclassify malicious samples as benign; however, it is plausible that an attacker could gain advantage by causing misclassification between malicious classes [43].

There are three main types of adversarial attack: white-box, black-box, and gray-box. White-box attacks assume complete knowledge of the target model. Black-box attacks assume zero-knowledge of the system; although an adversary might optionally acquire knowledge through exploiting 'oracle' attack strategies using multiple queries incrementally modifying a malicious sample until it is misclassified. An alternative strategy requires the adversary to create a surrogate model. The goal here is to employ the *transferability* property of adversarial examples. Adversarial examples generated on the surrogate may *transfer* and

subsequently successfully fool the target model. An attacker with 'oracle' access might be able to reverse-engineer the target model to generate a surrogate model; however, Apruzzese [125] rightly indicates that using a NIDS as an oracle is complex and demanding, feasible only in limited scenarios. Adversaries choosing this route face two obstacles: 1) Querying the target model while stealthily avoiding detection forcing a low-and-slow approach; 2) Acquiring feedback is difficult in IDS because the output of the model may not be directly observable to the adversary.

Fortunately, 'oracle' access is unnecessary for the creation of adversarial examples on a surrogate model. Papernot *et al.* [186] states that adversarial examples affecting one model often affect another model, even where the models have different architectures or were trained on different training sets. For transferable adversarial examples it is sufficient that both models were trained to perform the *same task*. An adversary may train their own surrogate model, craft adversarial examples against the surrogate, and transfer them to the target with *very little* knowledge of the target model. Recently, Yang *et al.* [187] examined the adversarial transferability of surrogate models. Specifically, they claim the adversarial transferability of a surrogate model can be improved when *any model* for the *same task* is used to extract and leverage soft (probabilistic) labels to train a surrogate model *without* querying the target model.

Gray-box attacks assume partial knowledge of the target model. For example, an adversary might have some knowledge of the features used by the model. Indeed, this is likely. By necessity all ML IDS analyse either raw network traffic or derived metadata [125]. ML IDS systems are likely to be trained on network flows [188]. A skilful adversary with sufficient domain knowledge could estimate which features are likely to be used and how they could impact performance.

4.2.3 Functionality Preserving Adversarial Examples

Apruzzese et al. [125] examine adversarial examples to consider realistic attacks, highlighting that most literature considers adversaries with complete knowledge about the classifier who are free to interact with the target systems. They emphasize

that few works consider 'realizable' perturbations that take account of domain and/or real-world constraints. Commonly black-box attacks assume adversaries are able to repeatedly query an 'oracle' model. The attacker may iteratively and incrementally perturb samples. At each iteration, a sample is presented, and the output is examined, pursuant to determining model decision boundaries and ultimately achieving misclassification. Such 'oracle' attacks are challenging. Each query increases the risk that the strategy will be detected, and the attack foiled. Moreover, direct output from an intrusion detection system (IDS) is not easy to achieve. Apruzzese et al. [125] consider situations where such attacks gain feedback while remaining undetected. However, they acknowledge that such attacks require persistence, skill, and resources. Transferability attacks avoid these problems because the oracle access to the target model is unnecessary as the target is not queried. An adversary can build a surrogate model for the same task on which to generate adversarial examples, subsequently transferring them to the target model.

Sheatsley *et al.* [189] advise that most adversarial examples in cyber-detection domains violate one or more domain constraints. Moreover, they find that crafting adversarial examples in constrained domains requires a different process to unconstrained domains. They argue that constrained domains are inherently more robust against adversarial examples. Further, they posit that the exploitable threat surface of models in constrained domains is likely narrower than previously understood; however, it is stressed that it is important not to take succour from this statement. The attack surface may be narrower; however, carefully crafted attacks might nevertheless successfully exploit it.

Apruzzese allege that a misconception exists in the literature in that much research pursues 'minimal' perturbations. They claim that in reality adversaries are not bound by this constraint [190]. It is acknowledged that adversaries will use *any* suitable method to fool the classifier, regardless the size of perturbation; However, this should not be considered final and decisive. Adversaries exist in an arms-race environment. Defences are continually improving, and adversaries must adapt

their strategies. The stark reality is that large perturbations are more easily detected by statistical measures [143]. Smaller perturbations are less easily detected and therefore confer advantages to adversaries who wish to hold persistence and remain undetected for a period of time. Furthermore, large perturbations do not necessarily confer an additional advantage over smaller perturbations. Domain constraints might be broken by large and small perturbations alike. The author predicts that the future trend of adversarial examples in NIDS will be to constrain the adversarial example in scope, size, or both. Specifically, the author predicts perturbations to small combinations of increasingly fewer features to a lesser degree. ART [87] allows feature masks to exclude certain (constrained) features from perturbation. This work carefully limits the scope of perturbations to one feature (gamma=0.05), and the size of perturbation to 0.02 (theta=0.02)

4.2.4 Intrusion Detection

Zhang et al [188] note common classification methods for internet traffic are based on statistical properties captured as netflows. This method addresses problems of dynamic port numbers and protects user privacy. Systems can be deployed to search for patterns in the netflows. Most such systems employ machine learning to perform automated classification of traffic types, detecting and/or dropping malicious traffic.

Wu *et al.* [52] consider several types of deep learning systems for network attack detection, including supervised and unsupervised models to compare the efficiency and effectiveness of different attack detection methods using two intrusion detection datasets: "KDD Cup 99" dataset and an improved version known as NSL-KDD [53] [54]. These datasets are commonly used; however, they do not fairly represent modern network traffic analysis problems due to concept-drift. Networks have increasing numbers of connected devices, increasing communications per second, and new applications using the network. Moreover, the use of computer networks and the Internet has changed substantially in twenty years. The continued introduction of IPv6, Network address Translation, Wi-Fi, mobile 5G networks, and cloud providers has changed network infrastructure [55].

Furthermore, the internet is increasingly used for financial services. Akamai [56] report financial services see millions or tens of millions of attacks each day. These attacks were less common twenty years ago. Furthermore, social media constitutes much of today's internet traffic and most social media platforms were founded after the KDD Cup 99 and NSL-KDD datasets were introduced. For example, Facebook, YouTube, and Twitter were founded in 2004, 2005, and 2006 respectively but are now in the top five most visited sites [191].

Kok *et al.* [57] warn the dangerous trend of using outdated datasets could result in no or insufficient progress on IDS. This would ultimately lead to an untenable situation, with obsolete intrusion detection systems (IDS), while intrusion attacks continuously evolve along with user behaviour and the introduction of new technologies.

Martins *et al.* [5] note that IDS are typically signature-based and that machine learning approaches are being widely employed for intrusion detection. They describe common white-box methods to generate adversarial examples including: L-BFGS , FGSM, JSMA , Deepfool, and Carlini & Wagner attacks. They also consider black-box methods using GANS. Traditional GANS sometimes suffer problems of mode collapse. Wasserstein Generative Adversarial Networks solve some of these problems. They introduce the Zeroth-order optimization attack as a black-box method. ZOO *estimates* the gradient and optimizes an attack by iteratively adding perturbations to features. They note that most attacks have been initially tested in the image domain, but can be applied to other types of data, which poses a security threat. Furthermore, they consider there is a trade-off when choosing an adversarial attack. For example, JSMA is more computationally intensive than FGSM, but modifies fewer features. They consider JSMA to be the most realistic attack because it perturbs fewer features; however, variations of FGSM and other methods can be configured to only modify certain features [192], making other less computationally intensive attacks potentially realizable.

4.2.5 Model Training for Robust Models

The trustworthiness and quality of a model is impacted by the distribution, quality, quantity, and complexity of dataset training samples [73]. Biased models are more susceptible to adversarial examples. Therefore, models should be trained on unbiased training data; although Johnson et al. consider the *absolute* number of training samples may be more important than the ratio of class imbalance [74]. Cyber security datasets are often prone to bias, partly because of limited samples of some malicious traffic (e.g., *zero-day* attacks) and large amounts of benign traffic. Sheatsley et al. [75] state biased distributions enable successful adversarial examples with very few feature modifications. Common data-level techniques tackle biased datasets by resampling: oversampling, undersampling, and hybrid sampling by combining modest oversampling of minority classes and modest undersampling of majority classes, aiming to give better model performance than applying either technique alone. Algorithm-level techniques tackling dataset bias commonly employ cost-sensitive learning where a class penalty or weight is considered, or decision thresholds are shifted to reduce bias [74].

4.2.6 Robustness

Robustness can be defined as the performance of well-trained models facing adversarial examples [79]. Essentially, robustness considers how sensitive a model's output is to a change in the input. The robustness of a model is related to the generalization-error of the model. A recognized trade-off exists between accuracy and robustness in machine learning. That is, highly accurate models are less robust to adversarial examples. Machine learning models in adversarial domains must be both highly accurate and robust. Therefore, improving the robustness of machine learning models enables safer deployment of ML systems across a wider range of domains, including IDS. To critically evaluate and make fair comparisons of the robustness of a model under attack, robustness metrics are necessary. Common machine learning metrics can be used to provide consistency, such as Precision, Recall, and F1-Score. These are not robustness metrics. However, a simple robustness metric can be calculated using the difference between F1-Scores

for ordinary and adversarial inputs.

4.2.7 Common Defences

Some research considers how model weights can be used to disrupt the generation of adversarial examples using white-box methods [154] [141]. However, these defences can be bypassed by using black-box methods. Much research considers the detection of adversarial examples by considering whether a sample is out of distribution [143] [144]. Detection is hard because adversarial examples do not exhibit intrinsic properties. Moreover, many detection methods are susceptible to good attacker-loss functions [164]. Adversarial training is a simple method aiming to improve robustness; however, it is not scalable. Moreover, Tramér *et al.* [121] found adversarial training can be bypassed. Some research investigates hardening the architecture of the model. Perhaps, changing model parameters or employing ensembles [6]. Defensive dropout uses a dropout layer and can block black-box and transferability-based attacks [153]. Adversarial defences exist in an arms race where adversaries adapt to defences by adopting new strategies. Therefore, defences must remain secure against adversaries who understand the model defences.

4.2.8 Ensemble classification

Biggio [6] asserts that ensemble classifiers have been exploited to improve robustness; however, they must be properly constructed to avoid worsening robustness. A typical ensemble classifier often used in intrusion detection is a Random Forest (RF) [193]. Other state-of-the-art ensemble classifiers include XGBoost, Histogram-based Gradient Boosting Classifier (HBBC), and Light Gradient Based Machine (LGBM). These ensemble classifiers help with robustness because their construction generates multiple randomised estimators. This complicates the task of generating attacks capable of fooling all (or most) of the estimators. Some research focuses on how ensembles of estimators with opposing or different gradients can help robustness [194]. Hierarchical classification, also

known as multilevel classification, is a form of ensemble classifier, where multiple classifiers are usually arranged in a top-down hierarchy.

4.3 Adversarial Attack of a Network Traffic Classification Model

To examine the nature of adversarial attacks, a well-trained machine learning model for network traffic classification is first developed. The Python programming language was used in this development along with the popular machine learning libraries, Keras [195] and scikit-learn [196]. Additionally, the Adversarial Robustness Toolbox (ART) [87] was used to support the construction of the adversarial examples. All experiments were performed using a variant of the CICIDS2017 dataset as detailed below.

4.3.1 Preparing the CICIDS2017 Dataset

The Canadian Institute for Cybersecurity IDS 2017 dataset (CICIDS2017) [66] is a recent addition to modern IDS datasets that has become increasingly popular amongst researchers. The dataset consists of a packet capture trace across a multi-system infrastructure for a period of 5 days, denoting both benign traffic activity as well as 14 common attacks including Brute Force FTP, Brute Force SSH, Denial of Service (DoS), Heartbleed, Web Attack, Infiltration, Botnet, and Distributed-Denial of Service (DDoS), as shown in Table 4.1. The data are available as a series of packet capture (PCAP) files, and as a "ML-ready" set of features in CSV format, derived using their CICFlowMeter tool [197][91] providing features of the communication flow between two parties, similar to Cisco NetFlow [198]. Recently, Engelen *et al.* [38] reported on errors that occur from using the CICFlowMeter tool that are present in the "ML-ready" dataset. Engelen *et al.* resolve the issues with the CICFlowMeter tool and provide a new version of the software tool and a corrected dataset, that is, a more accurate derivation of intended features in the original PCAP. For this study the dataset made available by Engelen *et al.* [38] is used.

Typical features for each flow include: Flow Duration, Packet Statistics, Flow Bytes/s, Flow Packets/s, IAT Statistics, Flags, Header Length, Down/Up Ratio, Bulk Statistics, Subflow Statistics, Init Win bytes, Active data packets forward, Active Statistics, and Idle Statistics. A set of 25 possible classes are derived in the improved labelling of the CICIDS2017 dataset, since some attacks that had previously been labelled had not successfully executed (e.g., did not result in data transmission). These attack labels are appended with the label "Attempted".

File	Traffic Type	Class Samples	Benign Samples	File Ratio	Dataset Ratio
Monday	BENIGN	529918	529918	1.00000	1.0000000
Tuesday	FTP-Patator	7938	432074	0.01837	0.0034922
	SSH-Patator	5897	-	0.01365	0.0025943
Wednesday	DoS GoldenEye	10293	440031	0.02339	0.0045282
	DoS Hulk	231073	-	0.52513	0.1016556
	DoS Slowhttptest	5499	-	0.01250	0.0024192
	DoS slowloris	5796	-	0.01317	0.0025498
	Heartbleed	11	-	0.00002	0.0000048
Thursday-Morning-WebAttacks	Web Attack - Brute Force	1507	168186	0.00896	0.0006630
	Web Attack - Sql Injection	21	-	0.00012	0.0000092
	Web Attack - XSS	652	-	0.00388	0.0002868
Thursday-Afternoon-Infilteration	Infiltration	36	288566	0.00012	0.0000158
Friday-Morning	Bot	1966	189067	0.01040	0.0008649
Friday-Afternoon-DDos	DDoS	128027	97718	1.31017	0.0563227
Friday-Afternoon-PortScan	PortScan	158930	127537	1.24615	0.0699178

TABLE 4.1: The CICIDS2017 dataset. For each data file (ordered by date), the table shows the attack types covered, the number of class samples for each attack, and the number of benign samples within each data file. The dataset ratio column shows classes considered over-represented at a per file level are still under-represented in the dataset as a whole when the dataset ratio is calculated using the sum of all benign samples.

Class imbalance can skew the assessment of model performance. As common for this domain, the benign class exhibits many more samples compared to the attack classes. To overcome this, *oversampling* and *undersampling* techniques are combined in sequence to effectively balance the dataset [199]. The dataset is also cleansed to remove all instances that consist of null entries. This balanced dataset results in 7,500 samples (300 samples per class).

These experiments make use of tests employing a control group and a mitigated group. Statistical analysis confirms that this sample size is sufficient for this study. A *paired t-test* power calculation [200] was performed to ascertain a sufficient sample size for meaningful results.

Such a power calculation requires four related components:

- Effect Size. The quantified magnitude of a result present in the population.
- Sample Size. The number of observations in the sample.
- Significance. The significance level used, commonly set to 5% (0.05).
- Statistical Power. The probability of correctly accepting the alternative hypothesis.

The size of the effect sought affects the sample size. The larger the effect sought the smaller the sample size is required, as shown in Figure 4.1.

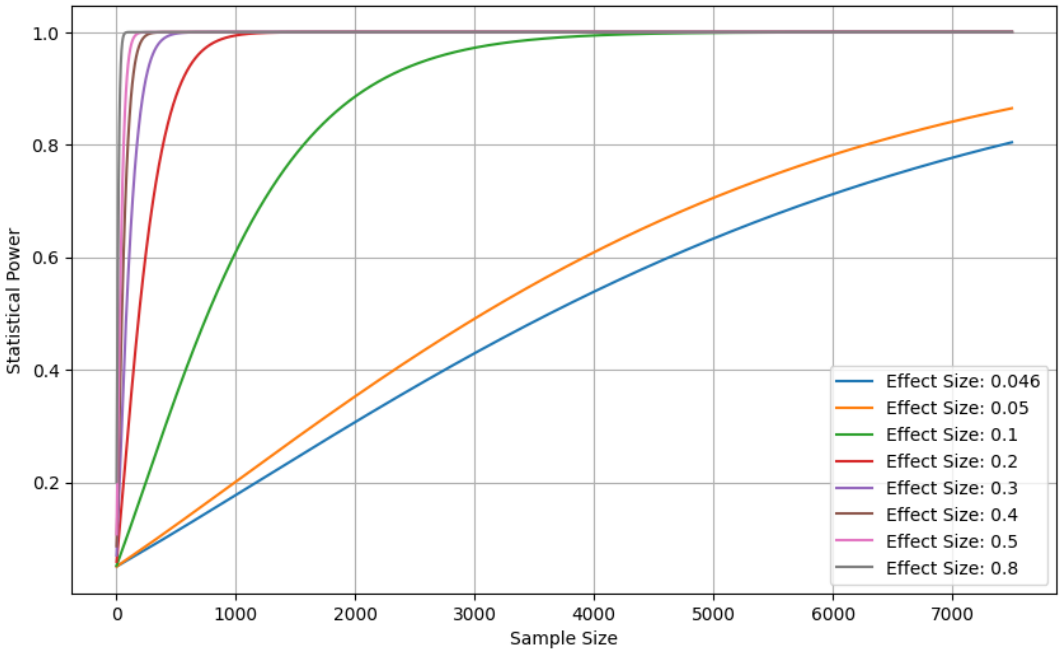


FIGURE 4.1: This plot shows paired t-test power calculation curves for effect sizes between 0.046 and 0.8 and sample sizes up to 7,500. The smallest effect of 0.046 meets a statistical-power of 0.8 with 7,500 samples. For larger effect sizes, the statistical-power threshold of 0.8 is comfortably accommodated by 7,500 samples.

A calculation adopting a *significance-value* of 0.05 and a *Statistical-Power* value of 0.8 and a sample size of 7,500 samples calculates the sample size is sufficient to find an effect of 0.05 between the control group and the mitigated group. This study hopes to mitigate the effects of JSMA adversarial examples and therefore seeks to find a large difference (effect) between the control group (JSMA) and the mitigated group. For example, a larger effect of 0.2 is a commonly used small effect. An effect

of 0.2 suggests a sample size of 394 is sufficient for a *significance-value* of 0.05 and a *Statistical-Power* value of 0.8.

4.3.2 Initial Classification Model

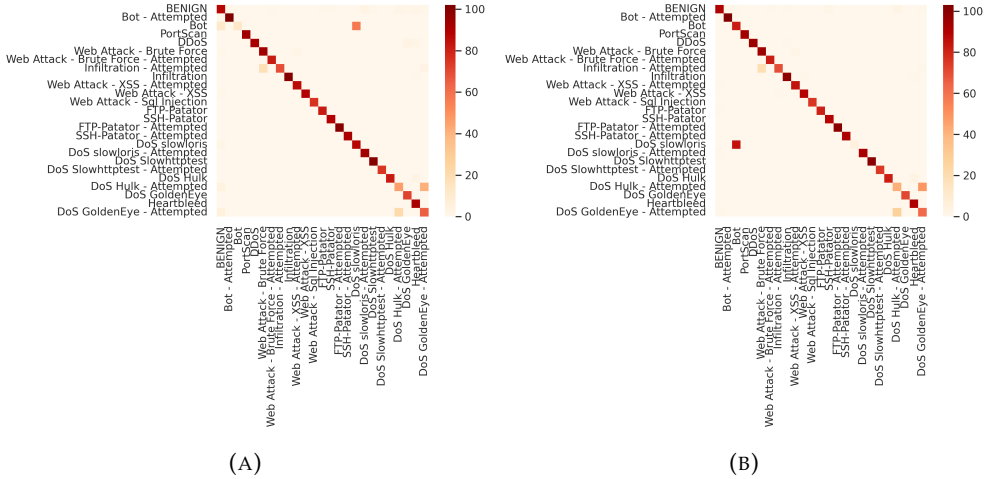
Scikit-learn offers a comprehensive selection of extensively documented machine learning models, seamlessly integrating with various Python libraries. Notably, adversarial examples were initially identified in the context of ANNs, which are particularly susceptible to such attacks owing to their inherent properties. By employing an artificial neural network, these vulnerabilities are effectively highlighted. Thus, an ANN serves as useful platform for the development and evaluation of countermeasures. Therefore, an initial *target* multi-class classifier is trained using a scikit-learn MLP model. The model consists of three dense layers and is trained for a maximum of 300 iterations. The resampled dataset is pre-processed, scaled, and split into training and validation sets with a 70/30 split. In Chapter 3 an 80/20 split was used, yielding 100% accuracy, and raising concerns of potential *overfitting*. Through reducing the proportion of the training set the risk of *overfitting* is reduced. As detailed later, the accuracy of the models presented in this chapter are slightly lower than the model presented in Chapter 3; however, the increased proportion of the validation set enhances the reliability of the evaluation. The Adam optimizer is used to train the target model, producing a well-trained model. Table 4.2 shows the performance of the classifier. The standard metrics of accuracy, precision, recall, and F1-score are used. The *support* column indicates the number of occurrences of the class in the specified sample. The target model achieves an accuracy of 91%, with precision of 91%, recall of 93%, and F1-score of 89%.

To overcome the issue of 'oracle' attacks and enable the transferability of adversarial examples [186], in addition to the target model, a surrogate neural network model is also trained for the same intrusion detection task. This sequential model consists of three dense layers and a softmax output layer. The first dense layer receives input in the form of the (67) features and has 128 units using ReLU activation function. The second dense layer has 64 units. The third dense layer has

	Target Model				Surrogate Model			
	Precision	**Recall**	**F1-Score**	**Support**	**Precision**	**Recall**	**F1-Score**	**Support**
BENIGN	0.75	0.96	0.84	79	0.88	0.98	0.93	92
Bot - Attempted	1.00	0.98	0.99	105	1.00	0.98	0.99	105
Bot	1.00	0.48	0.65	170	1.00	0.49	0.66	167
PortScan	0.98	1.00	0.99	92	1.00	1.00	1.00	94
DDoS	0.97	1.00	0.98	94	0.99	1.00	0.99	96
Web Attack - Brute Force	0.99	0.83	0.91	115	0.97	0.78	0.86	121
Web Attack - Brute Force - Attempted	1.00	0.96	0.98	85	0.99	0.99	0.99	82
Infiltration - Attempted	0.78	0.96	0.86	73	0.68	1.00	0.81	61
Infiltration	1.00	0.96	0.98	105	1.00	1.00	1.00	101
Web Attack - XSS - Attempted	1.00	0.99	0.99	87	0.99	0.96	0.97	89
Web Attack - XSS	0.97	0.99	0.98	88	1.00	1.00	1.00	90
Web Attack - Sql Injection	1.00	1.00	1.00	76	0.99	0.99	0.99	76
FTP-Patator	1.00	1.00	1.00	82	1.00	1.00	1.00	82
SSH-Patator	1.00	0.98	0.99	91	1.00	0.99	0.99	90
FTP-Patator - Attempted	1.00	1.00	1.00	100	1.00	1.00	1.00	100
SSH-Patator - Attempted	1.00	0.99	0.99	92	1.00	1.00	1.00	91
DoS slowloris	0.06	1.00	0.11	5	0.08	1.00	0.14	7
DoS slowloris - Attempted	1.00	1.00	1.00	93	0.98	0.99	0.98	92
DoS Slowhttptest	0.99	1.00	0.99	99	0.99	0.99	0.99	100
DoS Slowhttptest - Attempted	1.00	1.00	1.00	75	1.00	1.00	1.00	75
DoS Hulk	0.99	0.99	0.99	85	0.98	1.00	0.99	83
DoS Hulk - Attempted	0.55	0.61	0.58	82	0.55	0.59	0.57	85
DoS GoldenEye	1.00	0.95	0.97	76	1.00	0.97	0.99	74
Heartbleed	0.99	0.94	0.96	96	1.00	0.96	0.98	95
DoS GoldenEye - Attempted	0.68	0.59	0.63	105	0.63	0.56	0.59	102
Macro Average	0.91	0.93	0.89	2250	0.91	0.93	0.90	2250
Accuracy	0.91				0.91			

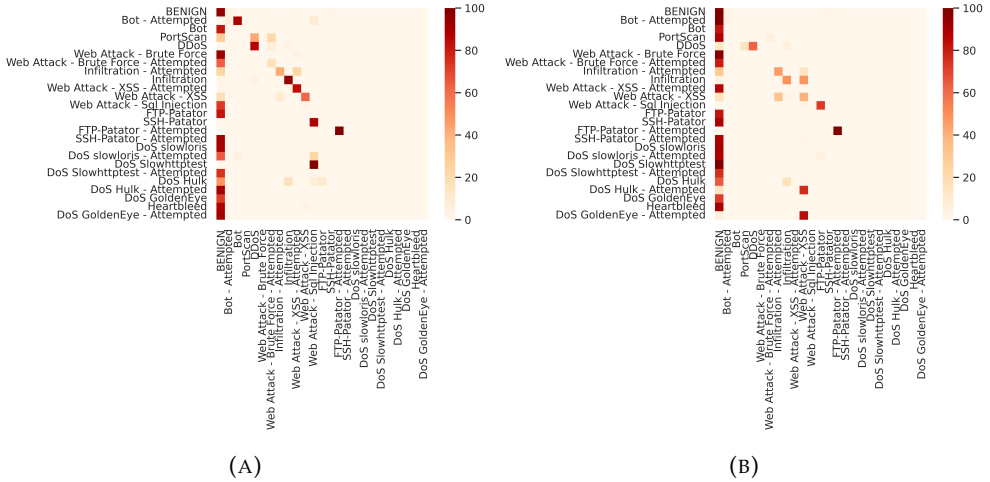
TABLE 4.2: Target Model and Surrogate Model Classification Reports.

25 units, one for each unique class. A fourth softmax layer that converts the output values into probabilities. The model uses *sparse categorical cross-entropy* loss function, together with the Adam optimizer. The model is trained for 200 epochs with an early stopping patience of 50. Whilst superficially similar to the target model, an alternative framework (Keras) is used to construct this surrogate model. Since the two models are known to be *similar* and are trained with portions of a *freely available dataset* the approach is intended to be representative of a black-box attack; however, it may well be considered to be a grey-box attack, since the author inevitably has some knowledge of the underlying models. As with the target model, the surrogate model consists of three dense layers and a softmax activation layer. Table 4.2 shows that the model achieves accuracy comparable to that of the target model (accuracy of 91%, with precision of 91%, recall of 93%, and F1-Scores of 90%). Figure 4.2 shows the confusion matrix of both the (a) target model and the (b) surrogate model to evaluate the performance of each individual class and to assess misclassification.



(A) (B)

FIGURE 4.2: Confusion matrix for (a) Target model (Scikit-learn), (b) Surrogate model (Keras).



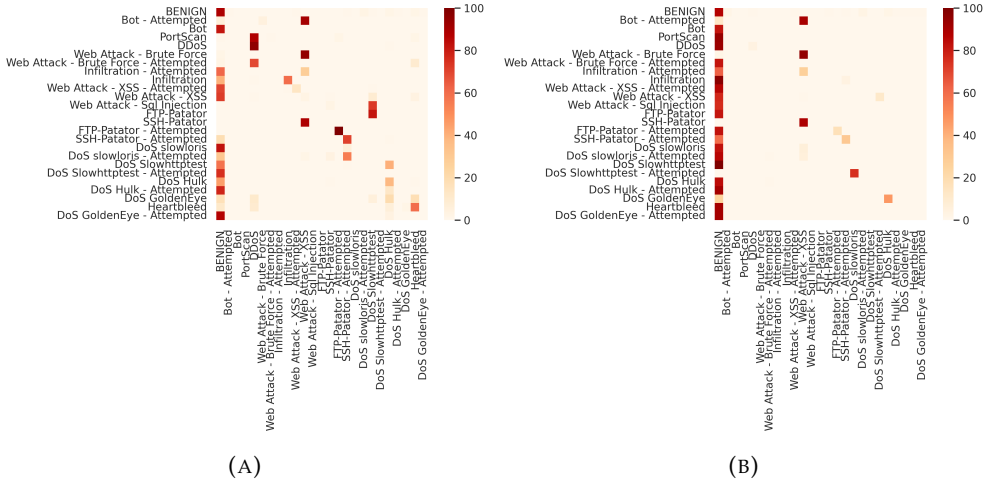
(A) (B)

FIGURE 4.3: Untargeted JSMA (θ=0.05 and γ=0.02) against (a) Target model (Scikit-learn), (b) Surrogate model (Keras).

4.3.3 Using the Surrogate Model to Attack the Target Model

From the adversarial perspective, the intention is to cause the target model to misclassify, often referred to as an *evasion attack*. Adversarial attacks can be performed on binary and multi-class classification systems, and typically fall into 3 groups: white-box; black-box; and gray-box attacks. White-box attacks provide full knowledge of the model, and access to the model. Many white-box attacks rely on the use of gradient descent search. Gray-box models assume some partial knowledge of the model. Black-box attacks offer zero knowledge of the model, and 'oracle' attacks are impractical for intrusion detection domain [125]. Therefore, some black-box attacks rely on the transferability properties of adversarial

examples using a surrogate model [201]. This is the approach taken here, whereby adversarial examples are generated on the surrogate model using the Jacobian-based Saliency Map Attack [202] [174], and then tested against the target model.



(A) (B)

FIGURE 4.4: Targeted JSMA for benign class (θ=0.05 and γ=0.02) against (a) Target model (Scikit-learn), (b) Surrogate model (Keras).

Figure 4.3 shows the confusion matrix of *untargeted* JSMA on both (a) the target model and (b) the surrogate model, where $\theta = 0.1$ and $\gamma = 0.1$. These two parameters specify the amount of perturbation to introduce to the original feature set (θ) and the maximum fraction of features to influence (γ). Both confusion matrices now exhibit a large amount of misclassification between the predicted labels and the ground-truth labels. It is important to reiterate that the adversarial examples were crafted against the surrogate model using JSMA, and then tested against the target model.

Figure 4.4 shows a similar effect for *targeted* JSMA. The untargeted attack does not specify a target class, whereas for the targeted attack the intention is to specifically misclassify each attack as benign. The surrogate model is significantly impacted, and the majority of samples are successfully misclassified as benign (Figure 4.4b). For the target model, whilst the effect of targeted misclassification to the benign class is not quite as prominent, nevertheless, the model performance is still severely degraded (Figure 4.4a).

4.3.4 Functionality-Preservation in Adversarial Example Generation

Adversarial attacks in computer vision rely on the manipulation of features (i.e., pixel intensity) that are unnoticeable to the human visual system. In this case study, whilst it is possible to manipulate features to provoke misclassification, it is important to assess whether the resulting features remain within the expected distribution of the data, such that it may be feasible to curate an attack that remains unnoticeable to a human observer, whilst exhibiting the intended underlying attack behaviour. Systematic experimentation was performed using the JSMA parameters θ and γ. To provide a clearer understanding of how the variables interact experiments were performed. To examine the size of perturbation, a range of values were chosen: large (0.8), midsize (0.5), small (0.1), and very small (0.05). The model was trained on sixty-seven features which influenced the choice of γ values. To examine the proportion of features modified, a similar range of values were chosen: large (0.8/80% of features/$\approx$ 53 features), midsize (0.5/50% of features/$\approx$ 33 features), small (0.1/10% of features/$\approx$ 6 features), very small (0.05/ 5% of features/$\approx$ 3 features), and a single feature (0.02/2% of features/$\approx$ 1 feature). An experiment was conducted using all paired combinations of $\theta = (0.8, 0.5, 0.1, 0.05)$ and $\gamma = (0.8, 0.5, 0.1, 0.05, 0.02)$, allowing the feature distribution to be studied in comparison to the statistical distribution of the original data.

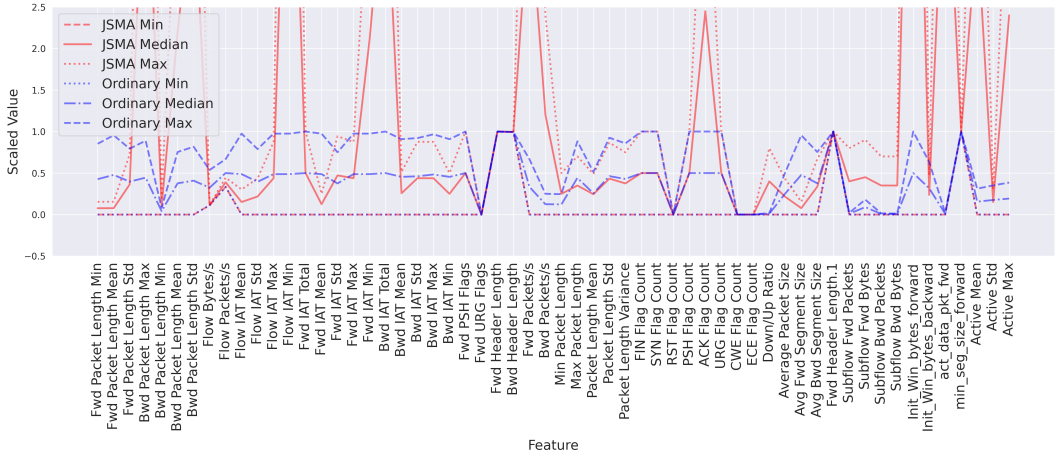


FIGURE 4.5: Parallel Coordinates to show the distribution of original features versus JSMA features ($\theta = 0.1$ and $\gamma = 1.0$). It can be observed that the perturbed JSMA features significantly exceed the expected range of the original traffic features.

Figure 4.5 shows a parallel coordinates plot for a subset of the feature domain,

to highlight the minimum, maximum and median of features for both the original dataset and the compromised adversarial examples. The discrepancies between the original data and the adversary examples are clearly visible with JSMA parameters $\theta = 0.1$ and $\gamma = 1.0$. Therefore, the crafted examples may either be clearly noticeable to a human observer investigating an attack, or the features may no longer satisfy the intended attack behaviour.

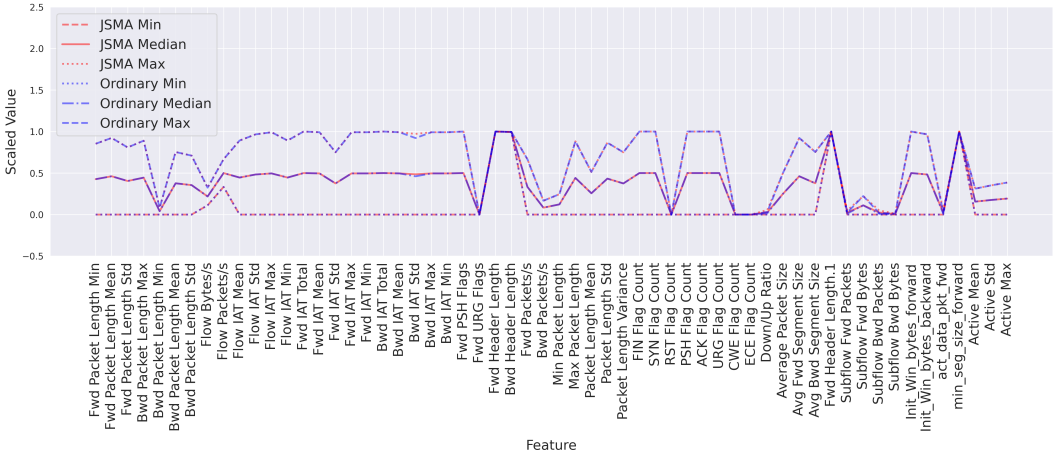
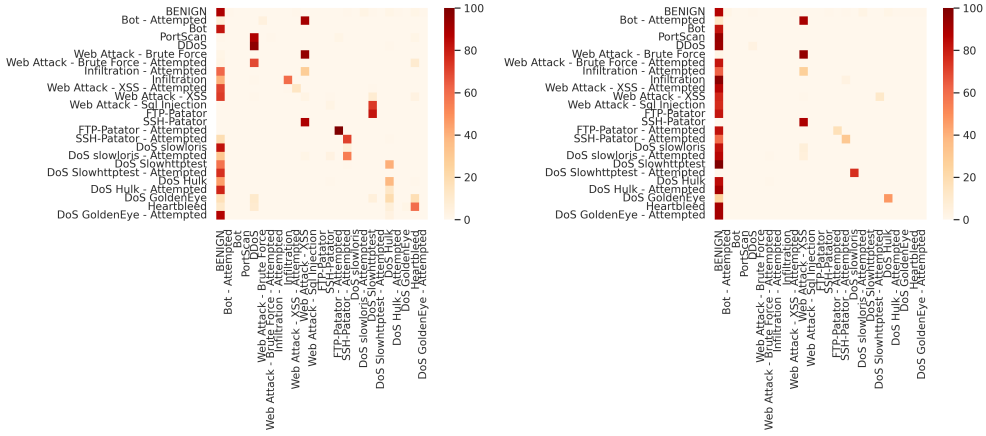


FIGURE 4.6: Parallel Coordinates to show the distribution of original features versus JSMA features ($\theta = 0.05$ and $\gamma = 0.02$). The perturbed JSMA features are within the expected range of the original traffic features.



(A) (B)

FIGURE 4.7: Targeted JSMA for benign class ($\theta = 0.05$ and $\gamma = 0.02$) against (a) Target model (Scikit-learn), (b) Surrogate model (Keras).

Figure 4.6 shows the distribution of original and adversarial examples where $\theta = 0.05$ and $\gamma = 0.02$. Importantly here, the difference between the adversarial and original distributions is unnoticeable at this scale, meaning that such adversarial

Traffic Type	Original	Correct after JSMA	Successful Attack Percentage
FTP-Patator	82	0	100.00
Web Attack - Sql Injection	76	0	100.00
Heartbleed	91	0	100.00
DoS GoldenEye	72	0	100.00
DoS Hulk - Attempted	91	0	100.00
DoS Hulk	85	0	100.00
DoS Slowhttptest - Attempted	75	0	100.00
DoS Slowhttptest	100	0	100.00
DoS slowloris - Attempted	93	0	100.00
DoS slowloris	90	0	100.00
SSH-Patator	89	0	100.00
DoS GoldenEye - Attempted	91	0	100.00
Web Attack - XSS	90	0	100.00
Infiltration - Attempted	90	0	100.00
Web Attack - Brute Force - Attempted	82	0	100.00
Web Attack - Brute Force	97	0	100.00
PortScan	94	0	100.00
Bot	82	0	100.00
Infiltration	101	1	99.01
Bot - Attempted	103	4	96.12
DDoS	97	6	93.81
SSH-Patator - Attempted	91	26	71.43
Web Attack - XSS - Attempted	86	70	18.60
FTP-Patator - Attempted	100	94	6.00
Total	2063	201	90.25

TABLE 4.3: Percentage of successful attacks, target='benign', by class ($\theta = 0.05$ & $\gamma = 0.02$).

attacks would be unlikely to be identified through statistical methods, and may well exhibit sufficient similarity to the underlying attack sequence in the packet flow communication. Therefore, this specific set of adversarial examples can be considered to be *functionality-preserving*.

Figure 4.7 shows the confusion matrix of *targeted* JSMA on both (a) the target model and (b) the surrogate model, where $\theta = 0.05$ and $\gamma = 0.02$, as determined to be a suitable set of parameters for functionality-preservation. Whilst a clear diagonal can be observed on each matrix, this is still severely degraded from the original result shown in Figure 4.2, whilst also noting that functionality of the adversarial cases would likely be preserved. Such parameter constraints would mean that not all adversarial examples will achieve misclassification, and some classes may well be more robust against the generation of adversarial example. Table 4.3 provides a detailed summary of the results, showing the number of successful targeted attacks for the constrained adversarial examples, for each of the possible attack classes. Overall, it is observed that **90.25% of the attacks were able**

to evade detection, with all DoS slowhttptest, and Heartbleed being misclassified, as well as the majority of other DoS attacks and Web Attack SQL Injection. Excluding the benign class that was not perturbed, all classes were found to exhibit some degree of vulnerability, as demonstrated by successful attacks.

4.3.5 Summary of the Adversarial Attack Stage

This chapter provides a comprehensive study on adversarial attack generation against a well-trained machine learning model for network traffic classification. It is shown that whilst evasion attacks are significantly more challenging to conduct within the constraints of the original data distribution, attacks of this nature are still achievable. As illustrated in Table 4.3, the fact that at least one instance of each attack can evade the classification model highlights the potential vulnerabilities exhibited by such a learning system. The often-stated maxim is affirmed that whilst defenders need to be successful in detection every time, attackers only need to be successful in their attack *once* in order to achieve their goal.

This study concentrated on the use of JSMA since it can provide targeted and untargeted attacks, and because the parameters allow control of the number of features to perturb and the amount of perturbation to introduce. Recent work has explored alternative adversarial techniques including Fast Gradient Sign Method, Basic Iterative Method, and Projected Gradient Descent that can also be configured to modify fewer or specific features [192]. Libraries such as the Adversarial Robustness Toolkit also support masking parameters to modify select features [87].

It is important to note that ML-based Intrusion Detection Systems may well include a mixture of categorical, continuous, and discrete features [203]. JSMA uses random perturbations of continuous features to generate adversarial examples. Features such as destination port number and protocol are *nominal* attributes that should not be treated as numerical and should not be perturbed in this same way. For the CICIDS2017 dataset used in this study, packet flow data is used, which

consists primarily of count data and statistical-based features derived from count data. Only the continuous features were perturbed. Nevertheless, additional logical and mathematical constraints should also be considered in future studies such that statistical features are accurate. Consequently, the crafting of adversarial examples within constrained domains poses unique challenges compared to much of the prior work from the computer vision domain [189].

4.4 Hierarchical Classification for Model Robustness

Having successfully compromised the network traffic classification model, defensive strategies are explored to improve the robustness of the classifier against such adversarial attacks. Previous works have often retrained the classifier using a set of curated adversarial examples [204] [121] [23]; however, this approach is not scalable and only provides a retrospective defence after a compromise has been identified. Instead, the attack surface where misclassification can occur is restructured by using a hierarchical classification layer. By reframing the classification task such that a hierarchical label is provided, the available attack surface to the adversarial methods is effectively reduced, such that the amount of perturbation required to misclassify a label becomes greater than compared to a flat classification layer, and outside of the distribution of the original dataset. In this network traffic classification case study, this means that the number of possible output states can be reduced from 25 to approximately 2-5 states at any level of the hierarchy. It is proposed that a hierarchical classification approach is less susceptible to the influence of adversarial examples compared to the classifier used previously in this work.

The hierarchical defence is beneficial because simple classifiers with fewer outputs are more robust [194]. Hierarchical classifiers are inherently more robust because misclassification between higher level classes requires more overt perturbations [205][206]. The hierarchical nature of such models reduces the number of classes at the upper layer are reduced. Further, the top-down nature of

hierarchies means that lower levels also classify among fewer classes. The effect is a further restricted number of appropriate classes, which improves robustness. The generation of adversarial examples against hierarchical classifiers is compounded because an adversarial example may need to fool two or more separate local classifiers. Alternatively, a *specific* classifier within the hierarchy could be targeted aiming to affect its output in order to impact the overall output of the hierarchical classifier. Research on adversarial examples largely focuses on one-stage (flat) classifiers. Attacking hierarchical classifiers is a more complex task, although it is an open research topic [206].

The work presented in this thesis examines hierarchical classifiers as a defence against transferable black-box adversarial examples. This experimentation incorporates the hierarchical learning library, HiClass [207], with the previous work in Section 4.3.

4.4.1 Hierarchical Classification

Traditional classification models utilize a flat output layer where each class is associated with a probability. However, many large classification tasks exhibit some natural hierarchy. For example, within the CICIDS2017, there are multiple variants of denial-of-service attacks (DoS), as well as different web-based attacks, and patator attacks on FTP and SSH services. The grouping of similar classes may therefore mean that misclassification between classes that exist in separate groupings may become more challenging to achieve with an adversarial attack.

Koller and Sahami's [208] seminal work on local classification approach established the foundations for researchers to expand the field of hierarchical classification using local classifiers. Hierarchical models often employ multiple multi-class models (e.g., One-Vs-All/One-Vs-Rest) and therefore may be

considerably larger than flat models. There are three generally accepted forms of local classifier: Local Classifier Per Node (LCPN), Local Classifier Per Parent Node (LCPPN), and Local Classifier Per Level (LCPL).

- **Local Classifier Per Node (LCPN):** A binary classifier is trained for every node in the class hierarchy (excluding the root node). The advantage is that this becomes naturally multi-class. The disadvantage is that because each node is trained independently, it is theoretically possible to have inconsistency between levels (e.g., a coarse class prediction of 'DoS'=False and a finer prediction of 'Dos Hulk'=True). Without methods to resolve these class-membership inconsistencies, incompatible predictions between coarse and fine classes are possible. This approach is illustrated in Figure 4.8.

- **Local Classifier Per Parent Node (LCPPN):** A multi-class classifier is trained for each parent's child nodes. This method respects hierarchy constraints while avoiding class-membership inconsistencies. This approach is illustrated in Figure 4.9.

- **Local Classifier Per Level (LCPL):** A multi-class classifier is trained for each level of the hierarchy. Simplistic implementations take the output of the classifier at each level, presenting this as the final classification. This method can result in class-membership inconsistencies. Approaches avoiding class-membership inconsistencies include a top-down approach where the class prediction at coarse levels restricts classification at finer levels to only child nodes of the previous level. This approach is illustrated in Figure 4.10.

4.4.2 Hierarchical Output Class

Typically, a machine learning classifier is trained to provide a single label from a set of possible labels. This approach extends from object classification to natural language generation. In this case study, a label is predicted that defines the associated network attack type, based on the characteristics of network traffic. It is natural to think how output classes might group together, such that similar attack types are grouped together under a specific sub-group. For example, DoS slowloris, DoS Slowhttptest, DoS Hulk, and DoS GoldenEye, naturally group together as a DoS group. Similarly, all web attacks can be grouped. The definition of a suitable set of hierarchical class labels could be achieved in multiple ways: based on analyst's domain knowledge, text label similarities, feature similarities, or some higher attributes related to the data. This study did not explore hyper-parameter searching as this was not the core focus of this study; however, this remains an area of future research.

The experiments were performed using 7 different hierarchical schemes: *K*-means; Ward; Average; Complete; Single; Researcher Defined; and Dataset. Figure 4.11 shows two of the initial hierarchy schemes: Researcher Defined and Dataset. The *Dataset* scheme (a) is based on the implicit hierarchy present in the original dataset, defined by how attacks have been grouped by the original authors. The *researcher* scheme (b) is defined by the author based on his domain knowledge.

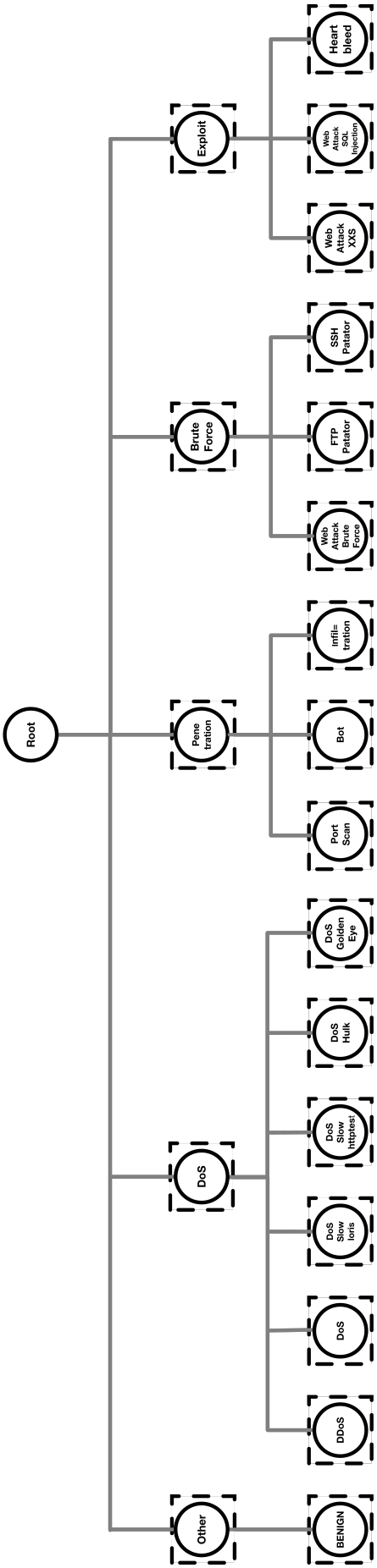


FIGURE 4.8: Local Classifier Per Node (LCPN): This diagram illustrates the local classifier per node technique, wherein binary classifiers (represented as dashed squares) are trained for each class (represented as circles) within the hierarchy, excluding the root node.

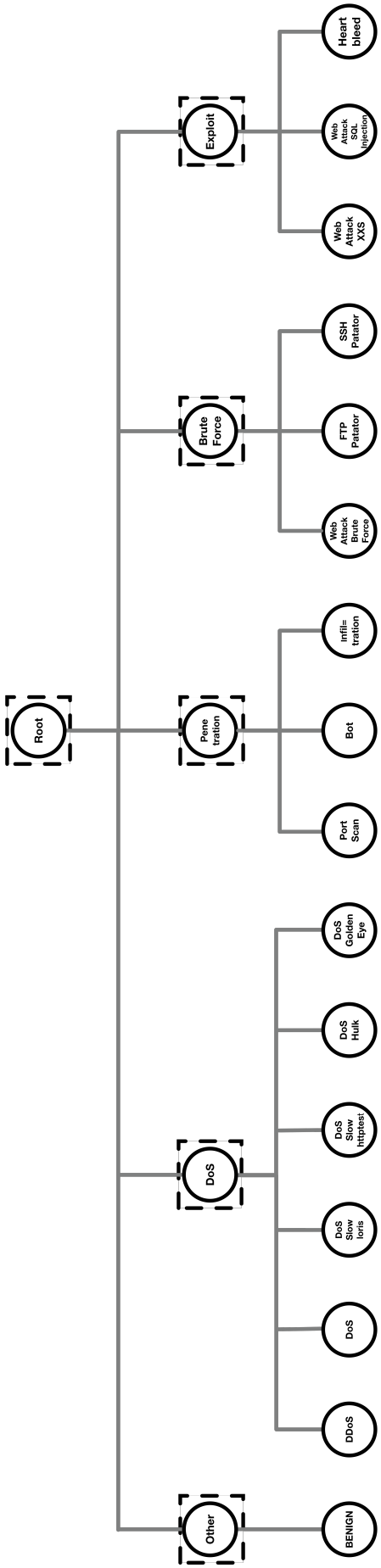


FIGURE 4.9: Local Classifier Per Parent Node (LCPPN): This diagram depicts the local classifier per parent node methodology, where multi-class classifiers (represented as dashed squares) are trained for each parent node present within the class hierarchy. Note that leaf nodes (classes without any children) are classified at the parent node.

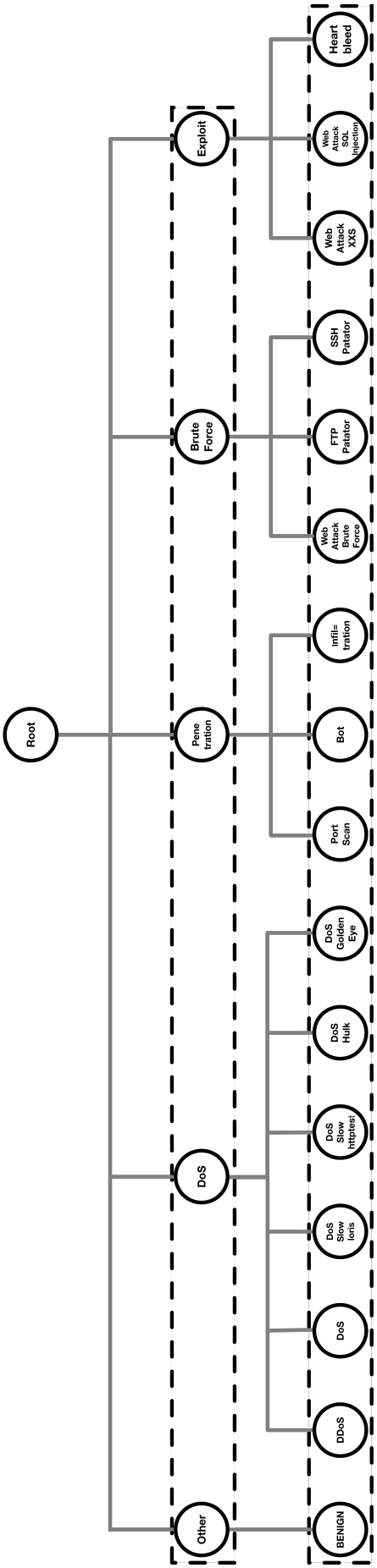
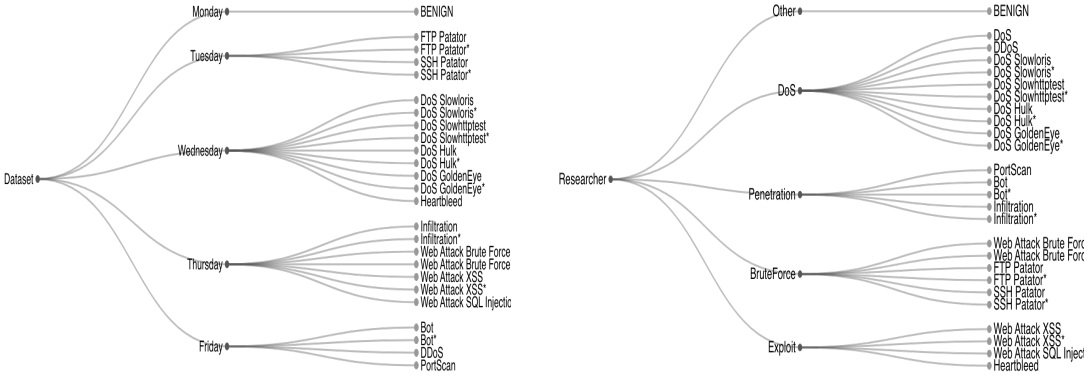


FIGURE 4.10: Local Classifier Per Level (LCPL): This diagram illustrates the local classifier per level technique, where multi-class classifiers (depicted as dashed rectangles) are trained for each level within the hierarchy.

Automated Hierarchical Clustering - K-Means

Optimistically, it is expected that hierarchies, defined by human experts, will be most robust; however, building such a hierarchy is a skilled task and introduces an overhead for analysts. To assist the process of generating class hierarchies, automated cluster techniques were considered. Firstly, *k*-means clustering was used to identify the hierarchical relationships, where k specifies the number of suitable clusters to discover. Figure 4.12 shows the hierarchy as determined by *k*-means clustering using a top-down approach. In this example, $k = 5$, for which it can be seen that the majority of classes cluster in group 0.



(A) (B)

FIGURE 4.11: Hierarchies assembled by human reasoning: (a) original data set structure, (b) researcher-defined structure. Attempted classes are denoted by $*$

.

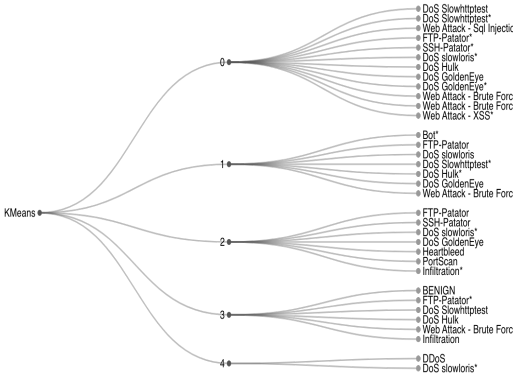


FIGURE 4.12: Hierarchy based on divisive clustering: *k*-means.

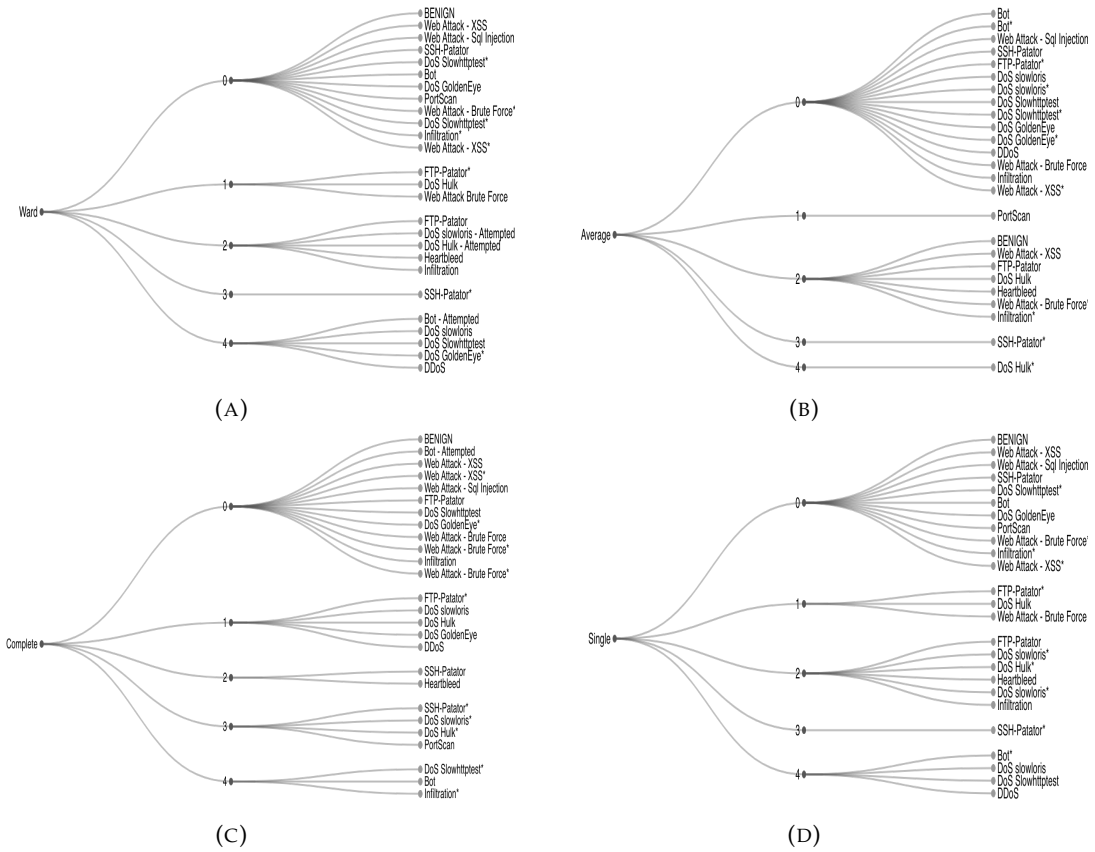


FIGURE 4.13: Hierarchies from agglomerative clustering: (a) Ward, (b) Average, (c) Complete, and (d) Single. Attempted classes are denoted by $*$

Automated Hierarchical Clustering - Agglomerative

Agglomerative clustering is a type of hierarchical clustering that differs from k-means in that it is a bottom-up approach. It begins with n clusters and sequentially combines similar clusters until all clusters belong to a single large cluster. The approach is more computationally expensive than k-means, however the scheme is especially applicable for arranging clusters into a natural hierarchy. The main parameters of agglomerative clustering are affinity and linkage, where affinity refers to the distance metrics used (Euclidean distance), and linkage refers to how the distance between clusters should be measured. Four possible linkage schemes are studied: 'Ward' [209], 'Average', 'Complete', and 'Single'. For a given pair of clusters, 'Single' will calculate the minimum distance between any pair of observations within each of the clusters, whilst 'Complete' will calculate the maximum distance between a pair of observations. 'Average' will take an average distance based on all pairs of observations within each clusters. 'Ward' is similar to average; however, it utilizes the variance of the observations within each cluster to then calculate the average. The scikit-learn library provides a function for Agglomerative Clustering that supports all four linkage schemes.

Algorithm 2: Create hierarchical labels from an unlabelled flat dataset.

```
finecluster = AgglomerativeClustering(n_clusters=15, affinity=affinity, linkage=linkage);
finecluster.fit_predict(X);
coarsecluster = AgglomerativeClustering(n_clusters=5, affinity=affinity, linkage=linkage);
coarsecluster.fit_predict(X);
top_layer_labels = coarsecluster.labels_
bottom_layer_labels = finecluster.labels_
for i in range len(X):
    hier_labels.append( [str(top_layer_labels[i]),str(bottom_layer_labels[i])])
```

Algorithm 3: Create hierarchical labels from a labelled flat dataset.

```
cluster = ClusterAlgorithm(n_clusters= 5)
cluster.fit_predict(X)
for class in range (0,max_class):
    initialize class groups
for i in range len(X):
    append cluster_label to relevant ground-truth class
for class_name in all_classes:
    find majority cluster label for ground-truth label
Initialize hier_labels = empty
for i in range(len(X)):
    append [str(majority_cluster_label),str(ground_truth_label[i])] to hier_labels
```

Algorithm 2 and Algorithm 3 show the process of constructing hierarchies for either an unlabelled or labelled dataset respectively, by generating labels for both

coarse and fine layers. The analyst can provide an integer to define the number of expected coarse and fine classes for the hierarchy, or if not provided, this will default to five.

4.4.3 Deployment of Hierarchical Classification

With the set of possible class hierarchies available, a two-layer hierarchical classifier can be built using the Python library 'HiClass' [207], each hierarchical structure for the model output layer can then be tested. This library integrates with the scikit-learn MLP classifier, allowing the target model from Section 4.3 to be easily incorporated. This also enables the comparison of the results of the hierarchical approach with that of the original flat model used previously. For hierarchical learning models, model performance can be evaluated using modified performance metrics proposed by Kritchenko *et al.* [210]: hierarchical precision (hP), hierarchical Recall (hR) and hierarchical F1-Score (hF).

4.4.4 Results of Hierarchical Classification

Figure 4.14 shows the F1-Score Macro Average for the original flat MLP model with both the normal data and the compromised adversarial example data. Similarly, it also shows the Hierarchical F1-Scores (HF) for both the coarse and fine layers of the hierarchical model, for both the normal data and the compromised adversarial example data. This is presented for each of the seven clustering strategies. It can be seen that LCPPN hierarchical classifiers can improve the robustness of classification, as measured by F1-Score. In particular, an average improvement of 84.85% in classification of presented adversarial examples: an increase of 0.28 from 0.33 to 0.61 This improvement can be seen in the difference between the orange and brown horizontal lines. Moreover, a slight improvement in robustness of classification in general is noted when only original unperturbed samples are presented.

The hierarchical defence achieves improved F1-Scores and robustness despite the presence of adversarial learning attack. Note that all the hierarchies improved

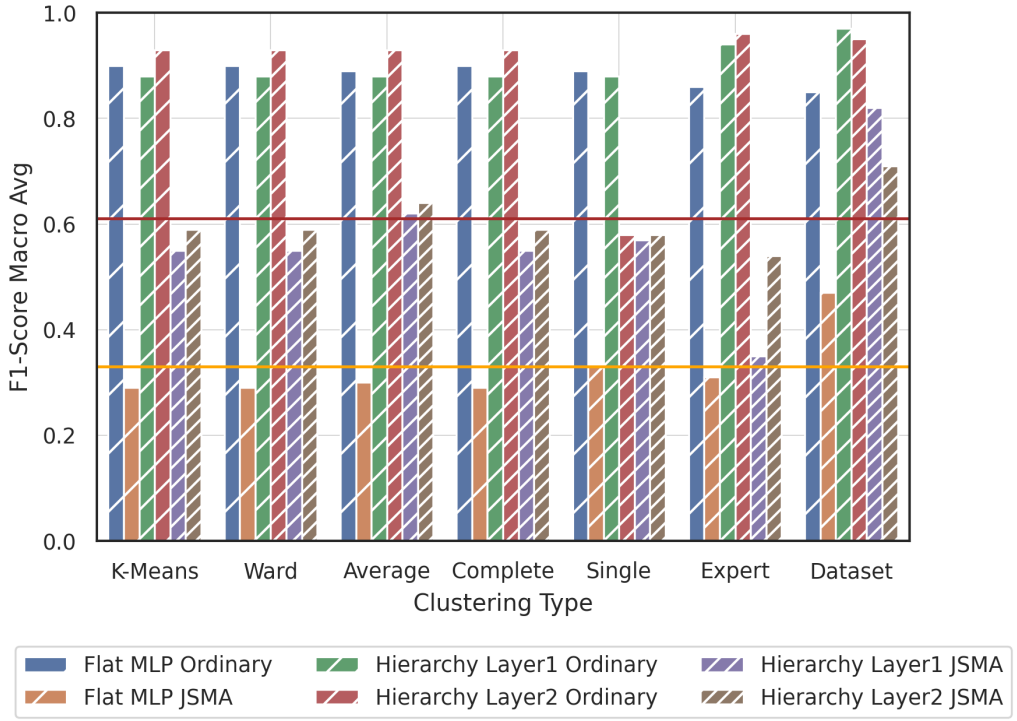
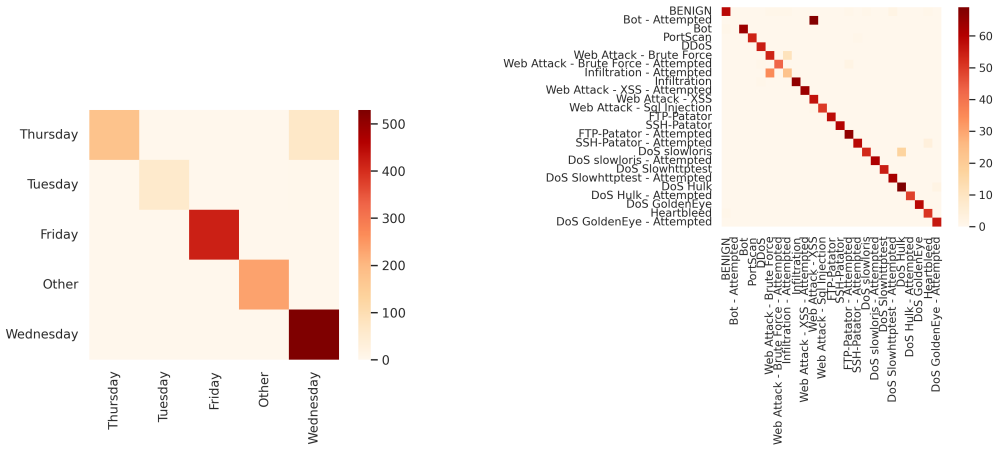


FIGURE 4.14: Bar plot to show robustness improvement by comparing appropriate F1-Score metrics across different LCPPN hierarchies. It can be seen that all hierarchies have improved F1-Scores under adversarial conditions. A decrease in the difference between F1-Scores for perturbed and unperturbed samples is visible. Two important results are highlighted: the orange horizontal line indicates the mean F1-Score for 'Flat MLP JSMA' across the hierarchies. The brown horizontal line indicates the mean 'Hierarchy Layer 2 JSMA' across the hierarchies. All hierarchies also improve the F1-Score when no adversarial traffic is present.

robustness as measured by the difference between F1-Score under perturbed samples and F1-Score under unperturbed samples. Moreover, the hierarchical approach also improves the F1-Score when no adversarial traffic is present. Figure 4.15 shows the confusion matrix for both (a) the coarse layer and (b) the fine layer, that reveals fewer misclassifications compared to the original model performance. Agreement is found with Qian *et al.* [194]: classifiers with fewer output classes are more robust. However, Hierarchical F1-Score for the 'single' (Figure 4.13d) remains steady. It is believed that because this hierarchy has a particularly large subclass, the benefit of a hierarchical structure is not fully realized, suggesting that some hierarchies will be more robust than others. Furthermore, it is found that flat models that are most susceptible to adversarial samples gain most from implementing a hierarchy.



(A) (B)

FIGURE 4.15: Confusion matrices for (a) coarse layer and (b) fine layer that shows fewer misclassifications for the original dataset when utilising a hierarchical classification model.

4.5 Discussion

This section discusses topics proceeding from this work, including how hierarchical learning can help improve the robustness of ML models and assist in blocking the transferability of adversarial examples. The identified benefits of hierarchical classifiers and hierarchies are discussed, along with the benefits of clustering techniques. The importance of blocking transferability is discussed, highlighting the reduction in transferability seen when using the presented hierarchical defence. Next, the classifiers and clustering techniques used in hierarchical learning, are examined. Subsequently, the feasibility of attacking a hierarchical classifier is discussed.

4.5.1 Benefits of Hierarchical Classifiers

Hierarchical classification is a simple ensemble technique that offers promise in protecting machine learning systems from adversarial examples, as this work has explored for intrusion detection systems.

It is posited that for hierarchical models with a few parent nodes and more child nodes, the higher layers are more robust than the lower layers. This offers advantages for network defence because misclassification among subclasses (e.g.,

DoS Hulk and DoS slowhttp) is a less serious prospect than misclassification among superclasses (e.g., DoS and Benign) [211]. Indeed, Jeanneret *et al.* [205] note that hierarchical attacks aiming for severe outcomes are less successful in evading detection. Hierarchical classifiers offer improvements in F1-Score and generalization error. Moreover, they may also reduce memory consumption, disk usage, and training time [207].

4.5.2 Hierarchies

Simple classifiers with fewer output classes are more robust [194]. It follows that in hierarchical models with a few parent nodes and more child nodes, the higher layers are more robust than the lower layers; however, if the classification task is truly hierarchical, by the nature of the class and subclasses, Layer 1 and Layer 2 will intuitively have aligned gradients. For example, the classifier for the parent node 'Denial of Service' will likely have gradients aligned with classifiers detecting specific subclasses of Denial-of-Service attacks such as 'Dos Hulk'. Intuitively, the construction of a traditional hierarchical classifier means that Layer 1 and Layer 2 ought to have aligned gradients. Using strong ensemble classifiers with misaligned gradients in a hierarchy may help improve robustness for hierarchical classifiers, combining the advantages of misaligned gradients and the robustness improvement gained through hierarchical classification.

Top-down methods restrict classification at finer levels to only child nodes of the previous level, meaning that lower levels also have fewer classes, further increasing robustness. A disadvantage is recognized that when descending a hierarchy there is no way to retrace one's steps. Therefore, misclassification at a coarse level might forbid correct classification at the fine level; however, improved robustness may be considered sufficient for this trade-off.

4.5.3 Clustering Techniques

Divisive and agglomerative clustering techniques could be employed to find other groupings of classes. Hierarchical labels simplify the generation of hierarchical

classifiers. The semi-automated techniques described in this work could be used to build hierarchical labels for unlabelled or labelled flat datasets.

It is noted that that any clustering algorithm is unlikely to generate clusters that correspond exactly to the known classes. Indeed, the experiments found that a particular known class could be dispersed among multiple clusters. The objective here is to explore the generation of hierarchies rather than to label datasets. Clustering techniques are unlikely to perform as well as supervised or semi-supervised learning techniques [212]. Instead, clustering is only used as a guide to which ground-truth labels could be grouped.

It is noted that some hierarchies have diverse fine-classes covered by one large coarse-class. For example, Single has a particularly large coarse-class. The prospect of misclassification at the coarse level is likely proportionate to the number of classes in that branch of the hierarchy.

4.5.4 Blocking Transferability

The transferability property of adversarial examples can effectively be used to perform gray-box or black-box attacks. Breaking transferability is an important goal. In the experiments a reduction in transferability to the hierarchical model is observed. Other work considers other ways to block the transferability of adversarial examples. For example, Hosseini *et al.* [213] propose NULL-labeling as a form of adversarial training.

4.5.5 Effectively Attacking Hierarchies

The hierarchical defence presented in this work improves robustness against adversarial learning attacks. Attacks causing misclassification between subclasses are less severe than attacks causing misclassification at class level. Successful adversarial learning attacks on intrusion detection systems must preserve the functionality of malicious network traffic. Research in other domains might apply in cyber security. For example, Jeanneret *et al.* [205] consider hierarchy aware

attacks that generate adversarial perturbations considering the hierarchical distance between labels. Moreover, they consider the severity of Hierarchical Attacks and apply Curriculum Learning to enhance the performance of models through learning concepts from coarse to fine.

4.6 Conclusion

This chapter proposes hierarchical learning as a defensive strategy to mitigate against adversarial machine learning attacks. The presented defence is independent of the attack algorithm and based on a robust hierarchical learning scheme. When under attack, the presented defence achieves accuracy scores close to the accuracy of the original flat model with no adversarial machine learning attacks. A decrease in the generalization error is seen. The presented approach is intended to be representative of a functionality-preserving black-box attack; however, this approach may be considered a gray-box attack, because inevitably the author has some knowledge of the underlying models. This work identifies the vulnerability of machine learning models and explicitly with artificial neural networks. This work examines how the vulnerabilities might be mitigated, using two novel methods to improve robustness. If the designers and developers of machine learning systems are not easily able to identify the vulnerabilities in their models they will unknowingly or unwittingly deploy vulnerable models in unsafe environments. The following chapter considers how the research findings can be generalized to other domains.

Chapter 5

Further Exploration of Adversarial Machine Learning

5.1 Introduction

This chapter evaluates the extent to which the findings can be generalized to other conditions, settings, and situations. For example, how well does this research generalize across other domains and particularly cyber security domains, such as cyber physical systems and the internet of things (IoT)? Consideration is given to the main factors that could affect the generalizability of the results and the possible limitations of this research are outlined. Two miniature case studies are used to explore the generalizability of this work to other models and datasets.

5.2 Case Study 1 - Consequences of Model and Dataset

This case study examines how this work can be generalized to other models and datasets.

5.2.1 Model Choice

This section goes beyond Neural Networks, to explore how adversarial learning can impact other forms of machine learning classification models. There are a range of machine learning models that are each vulnerable to adversarial examples to some extent. Through the experiments it was found that ensemble classifiers are more robust than stand-alone classifiers.

Model Name	Original F1-Score Macro Avg	Adversarial F1-Score Macro Avg	Generalisation Error	Percentage Change
Support Vector Classifier	0.08	0.08	0.00	00.00
Light Gradient Boosting Machine	0.78	0.73	0.05	-06.41
Histogram Based Boosting Classifier	0.78	0.72	0.06	-07.69
AdaBoost	0.10	0.09	0.01	-10.00
Logistic Regression	0.45	0.39	0.06	-13.33
k Nearest Neighbour	0.60	0.51	0.09	-15.00
Random Forest	0.79	0.66	0.13	-16.46
XGBoost	0.72	0.57	0.15	-20.83
Naive Bayes	0.39	0.29	0.10	-25.64
Decision Tree	0.79	0.26	0.53	-67.09
Quadratic Discrimination Analysis	0.53	0.05	0.48	-90.57

TABLE 5.1: Model Types sorted by Adversarial Generalisation Error (F1-Score).

Table 5.1 shows how the macro average F1-score of models can deteriorate in the presence of adversarial examples. The worst performing models are Quadratic Discrimination Analysis and Decision Tree with performance dips of approximately 90 % and 67 % respectively. Superficially the best performing model is the Support Vector Classifier (SVC) because its performance does not deteriorate in the presence of adversarial examples; however, this SVC model performs very poorly under ordinary conditions, and should be considered an anomaly. The best performing models under adversarial conditions are the ensemble classifiers Light Gradient Boosting Classifier and Histogram Based Boosting with low percentage drops of approximately 6% and 8%. Interestingly the Random Forest model is an ensemble version of the decision tree and is more robust than the decision tree with a performance dip of approximately 16%, compared with 67.09% for the Decision Tree.

5.2.2 Defensive Hierarchical Approach

Carefully applying a hierarchical approach could further improve robustness of all models to adversarial examples. The robustness of models is explored with a selection of *untuned* classifiers, that are trained on the CICIDS2017 dataset. The experiments essentially replicate the experimental conditions described in Chapter 4 using different base classifiers. JSMA adversarial examples are generated using a *surrogate* Keras model trained on the same dataset. The following common models were selected: Quadratic Discrimination Analysis (QDA), Decision Tree (DT),

Naive Bayes (NB), Multi Layer Perceptron (MLP), Logistic Regression (LR), Support Vector Classifier (SVC), K Nearest Neighbour (KNN), AdaBoost, XGBoost (eXtreme Gradient Boosting), Histogram Based Boosting Classifier (HBBC), Light Gradient Boosting Machine (LGBM), and Random Forest (RF). Each model is used as a base classifier in a hierarchical classifier and its performance against adversarial examples is evaluated for different hierarchical paradigms: LCPN, LCPPN. The untuned classifiers are attacked by presenting the generated JSMA adversarial examples, utilising the transferability property. The hierarchical F1-Score Macro Avg is noted as a robustness metric as shown in Figure 5.1. To find the most robust classifiers they are sorted by F1-Score.

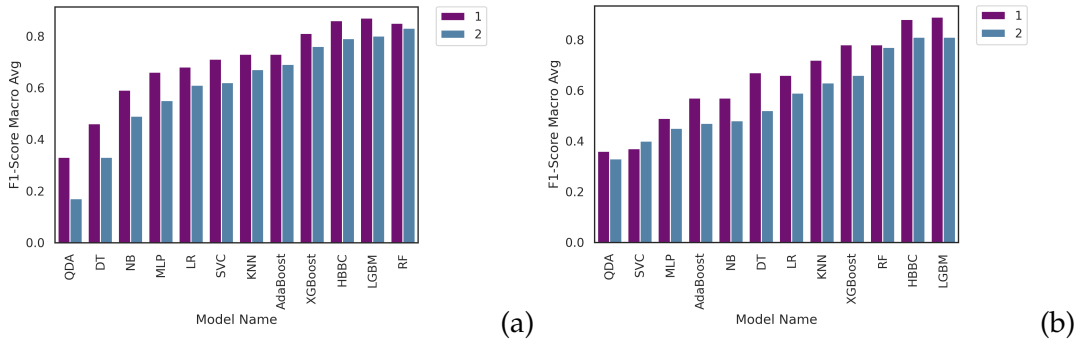


FIGURE 5.1: LCPN F1-Score (a) and LCPPN F1-Score (b) by Model.

It is noted that Layer 1 commonly achieves higher F1-Scores than Layer 2 (with the exception of SVC for LCPPN). It is posited that this phenomenon is because Layer 1 in the hierarchy has fewer classes than Layer 2. Further, it is noted that ensemble classifiers are typically more robust than single classifiers, with the exception of AdaBoost, which is composed of weak learners. In general, there is a visible trend that ensemble classifiers perform well in hierarchical classification.

5.2.3 Dataset

This research is generalizable to different problem domains. The proposed methods are found to work well on classification problems for intrusion detection scenarios. In the CIFAR-10 and CIFAR-100 datasets [214] each image comes with a fine label and a coarse label. The robustness benefits of hierarchical learning could easily be gained in the visual domain by using similar hierarchical datasets for classification. To explore the generalizability of this work, the research is applied to a different

dataset: MQTT-IoT-IDS2020 [71]. Dataset bias might limit the generalizability because common hierarchical classifiers are affected by class imbalance [215]. Steps were taken to resample the datasets, using undersampling to adjust for bias.

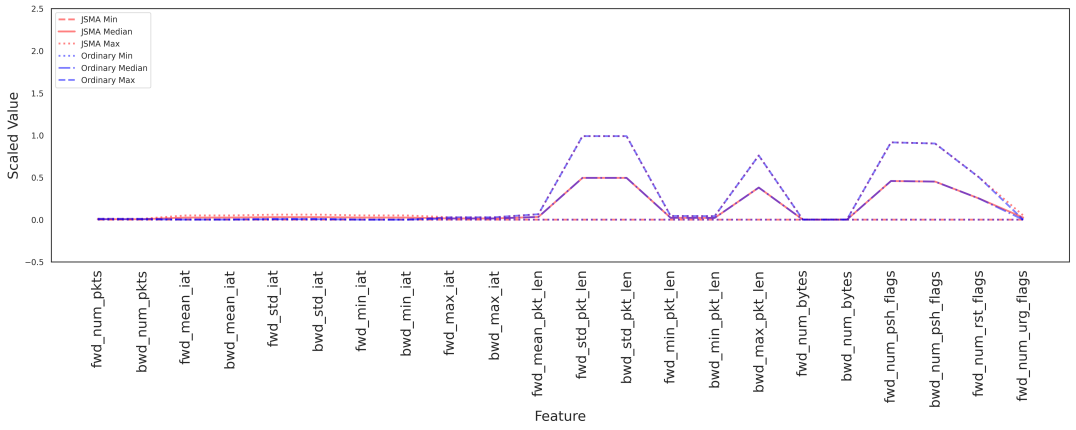
5.2.4 Features

In Chapter 2 a generalizable method for feature vulnerability and robustness assessment was presented. The method used recursive feature elimination, eliminating the least robust features as identified by the feature with the largest perturbation. It was found that reducing the number of vulnerable features forced more overt adversarial examples with larger perturbations. When applying JSMA adversarial examples to the MQTT dataset which already has fewer features than the CICIDS2017 dataset, it was observed that adversaries wishing to generate effective JSMA adversarial examples on feature sets with fewer samples are faced with a choice:

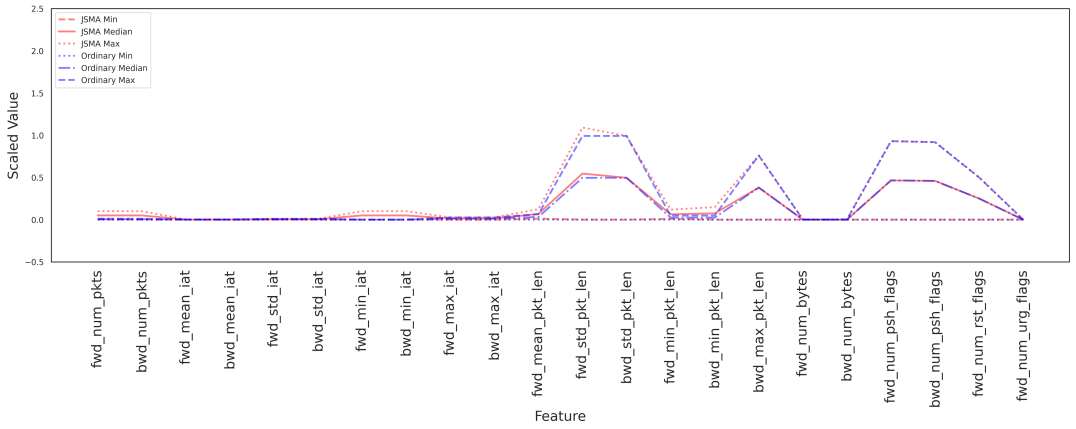
- Increase the perturbation size to a feature
- Increase the number of features to perturb
- Increase both the size of the perturbation and the number of affected features.

Figure 5.2a, Figure 5.2b, and Figure 5.2c show parallel coordinate plots indicating the effect on the distribution of adversarial examples. Looking carefully at the parallel coordinate plots, they show that as the perturbation size increases the variance from the ordinary traffic profile also increases. It can be seen that JSMA selects a feature to perturb. This feature is not the same for each sample. It can be seen that 'IAT' statistics are commonly perturbed. The more overt adversarial examples are, the more likely they are to be detected. Moreover, larger perturbations imply that samples are less likely to function as intended. Table 5.3 and Table 5.4 show the improvement in successful attacks with larger theta and gamma values.

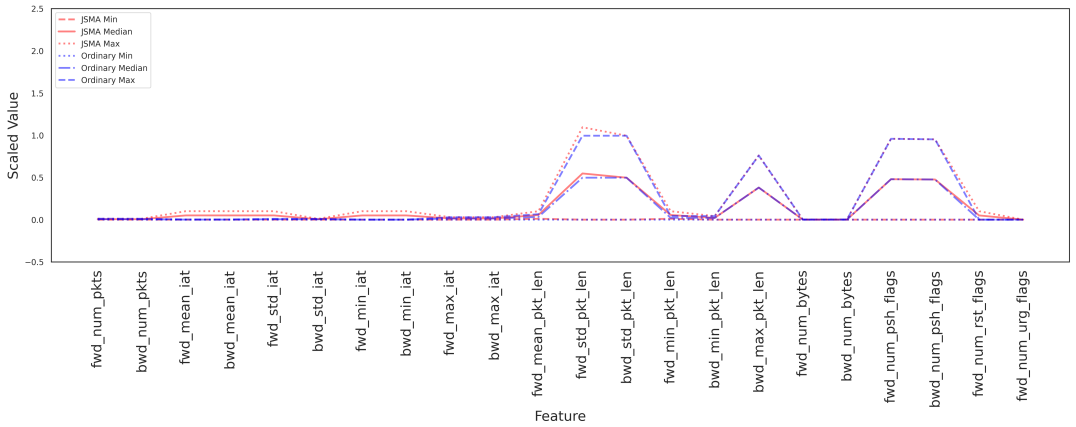
Certain features may be more important to the output of a model. Whereas less important or unused features provide attackers with a larger attack surface while



(A)



(B)



(C)

FIGURE 5.2: Parallel Coordinate Plots for (A) Small Perturbation to approximately three features - $\theta = 0.05\gamma = 0.07$, (B) Larger Perturbation to approximately one feature - $\theta = 0.1\gamma = 0.04$, (C) Larger Perturbation to approximately three features - $\theta = 0.1\gamma = 0.07$.

offering little or nothing of consequence to the model output. Previous chapters considered ways to select the most robust features using feature engineering techniques such as Recursive Feature Elimination (RFE). In the experiments

presented in Chapter 2 it was found that feature sets of twenty features were sufficient to negate the effects of FGSM while retaining acceptable classification accuracy. For datasets with very small numbers of features, eliminating the most vulnerable features could start to impact on correct classification. Similarly, datasets where all features contribute equally (indicating high sensitivity to each feature) may suffer unacceptable classification accuracy with the reduction of vulnerable features.

5.2.5 Classes

In the context of intrusion detection and network traffic classification, adversaries employ adversarial examples mainly to subvert a model by causing unexpected outputs or classifications. Therefore, the proportion of incorrectly classified classes can be used as a proxy measure for robustness. In other words, the fewer samples that are misclassified, the more robust a model is. To this aim the generalisation error between F1-Scores is also a suitable metric.

Chapter 4 presented a defensive hierarchical-based approach to improving robustness of multi-class classification against adversarial examples. It is argued that hierarchical classification could be a simple way of hardening machine learning models. The number of identified classes could limit generalizability. For example, in binary classification problems, there are only two classes. It is difficult to envisage a useful hierarchy with two classes. As the number of classes increases it is possible that natural hierarchies will form, and useful hierarchies could be built; although natural hierarchies present in the dataset may not form the most robust hierarchy possible.

5.3 Applying the Proposed Methods to the MQTT Dataset

This section applies the proposed methods to a different dataset: MQTT [71]. There are five classes in this dataset: Benign, Sparta Brute Force, MQTT Brute Force, Scan_A and Scan_sU. The experimental conditions in Chapter 4 are replicated. The target and surrogate models are constructed. Recall that the target model is

implemented using a scikit-learn MLP model, and the surrogate model is implemented using Keras. Subsequently the prepared MQTT dataset is pre-processed, scaled, and split into train and test samples (0.7/0.3 split). The Adam optimizer is used to train the target model. The well-trained model achieves accuracy of 99% as show in Table 5.2.

Class	precision	recall	F1-score	support
Benign	1.00	0.97	0.98	86
Sparta-BF	0.97	1.00	0.98	87
MQTT-BF	1.00	1.00	1.00	79
Scan_A	1.00	1.00	1.00	109
Scan_sU	1.00	1.00	1.00	89
accuracy	0.99	0.99	0.99	0.99

TABLE 5.2: MQTT: Target Model Classification Report.

Untargeted JSMA adversarial examples are generated on the surrogate model and transfer them to the target model. Only two of the classes were sufficiently perturbed to be considered a successful attack. Table 5.3 shows that JSMA perturbations with theta=0.05 gamma=0.02 are effective in only a small percentage of cases.

Class	Original	Correct after JSMA	Successful Attack Percentage
Scan_A	109	107	1.83
Sparta-BF	90	89	1.11
Benign	83	83	0.00
MQTT-BF	79	79	0.00
Scan_sU	89	89	0.00

TABLE 5.3: Flat Classifier: MQTT Successful Attack Percentage with JSMA (theta= 0.05, gamma= 0.02).

In large part this is due to a smaller number of features in the MQTT dataset. Table 5.4 shows that as the JSMA theta and gamma values are increased (theta=0.10 gamma=0.07), the successful attack percentage improves. Over half (58.23%) of the MQTT-BF samples and nearly one fifth (19.27%) of Scan_A samples and 6% of Sparta-BF samples were considered successful attacks. There is a potential impediment of larger perturbations to more features in that larger perturbations over a wider number of features will be more easily detected and are less likely to retain functionality. The successful attack percentage is reported in Table 5.4.

Class	Original	Correct after JSMA	Successful Attack Percentage
MQTT-BF	79	33	58.23
Scan_A	109	88	19.27
Sparta-BF	90	84	6.67
Benign	83	83	0.00
Scan_sU	89	89	0.00

TABLE 5.4: Flat Classifier: MQTT Successful Attack Percentage with JSMA (theta=0.10 gamma=0.07).

The hierarchical defence is now applied, reducing the percentage of successful attacks as shown in Table 5.5 (coarse) and Table 5.6 (fine). The hierarchy is generated with the 'Ward' linkage. When considering the three coarse classes class 0 and class 2 are most robust with 0% successful attacks. The coarse class 1 is least robust with approximately 3% of the attacks deemed successful. The two most vulnerable classes: MQTT-BF and Scan_A are grouped as coarse class 0. The least vulnerable class: Scan_uS is paired with the Benign class which is not perturbed. The Sparta-BF attack are in a class of their own.

Coarse Class	Original	Correct after JSMA	Successful Attack Percentage
(Sparta) 1	90	87	3.33
(MQTT & Scan_A) 0	188	188	0.00
(Benign & Scan_uS) 2	172	172	0.00

TABLE 5.5: Ward Hierarchy Classifier Layer 1 (Coarse): MQTT Successful Attack Percentage with JSMA (theta=0.10 gamma=0.07).

Fine Class	Original	Correct after JSMA	Successful Attack Percentage
Sparta-BF	90	87	3.33
Benign	83	83	0.00
MQTT-BF	79	79	0.00
Scan_A	109	109	0.00
Scan_sU	89	89	0.00

TABLE 5.6: Ward Hierarchy Classifier Layer 2 (Fine): MQTT Successful Attack Percentage with JSMA (theta=0.10 gamma=0.07).

Recall from Chapter 4 that building an appropriate hierarchy is a skilled task and introduces an overhead for analysts. In assisting the process of generating class hierarchies, automated cluster techniques were considered. Specifically, *k*-Means, and the agglomerative clustering linkages: Ward, Average, Complete, and Single.

Figure 5.3 shows the hierarchies built from the MQTT dataset.

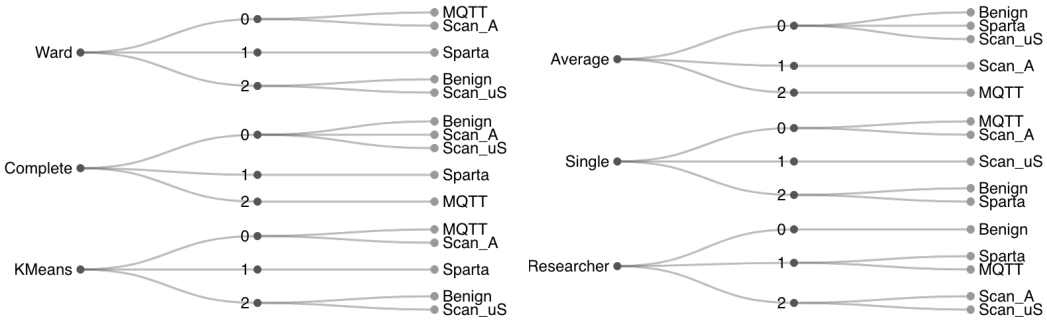


FIGURE 5.3: Hierarchies Built from MQTT Dataset.

Note that none of the automatically built hierarchies separated the benign class from other classes. Only the Researcher defined hierarchy places benign samples in a class of their own. Further, note that Ward and *k*-Means produce identical hierarchies. There does not appear to be a shared pattern among the other hierarchies. The experiments conducted in Chapter 4 are repeated for the MQTT dataset, and the results presented in Figure 5.4.

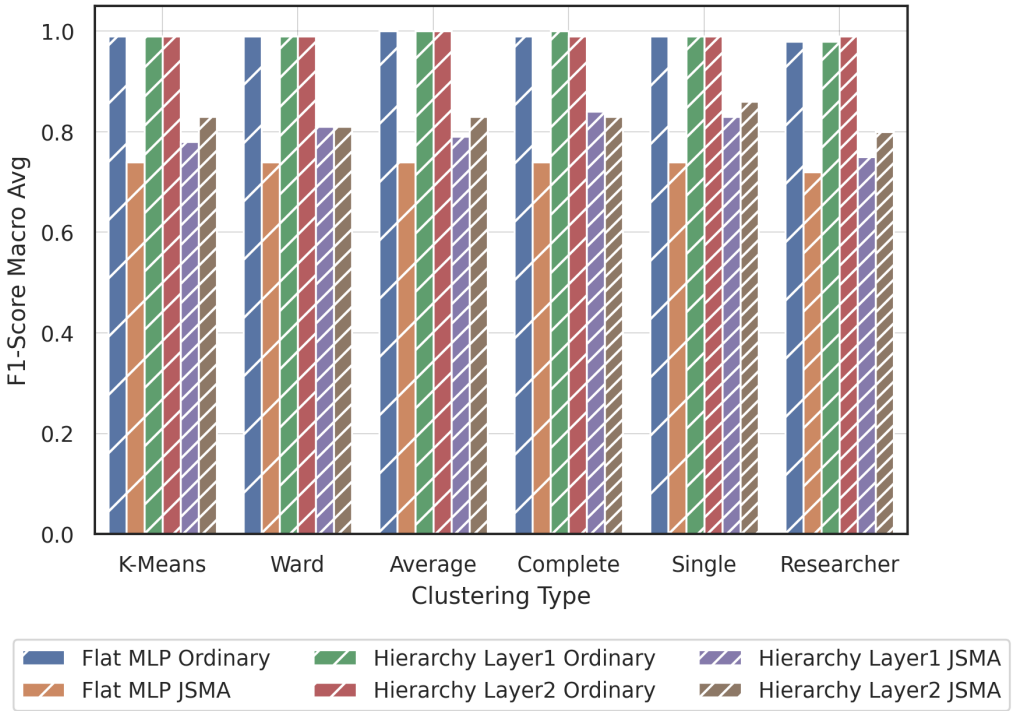


FIGURE 5.4: For all the hierarchies the hierarchical defence improves F1-Score when compared with the flat F1-Score.

Similar results are observed in that all the hierarchies improve robustness under adversarial examples. It is posited that the use of hierarchies could be a simple method of hardening models against adversarial examples. In Chapter 4 it was noted that the Hierarchical F1-Score for the 'single' remained steady. It is argued that because this hierarchy has a particularly large subclass, that the benefit of a hierarchical structure was not fully realized, suggesting that some hierarchies will be more robust than others. Poorly constructed hierarchies may allow adversarial examples to more easily cross a decision boundary than carefully constructed hierarchies. Particular hierarchies might facilitate adversarial examples, rather than defend against them. Further research is necessary to determine how to construct the most effective and robust hierarchies.

5.4 Case Study 2 - Discrete Datatypes

The second miniature case study examines how this work generalizes to discrete datatypes.

5.4.1 Challenges of Discrete Features

To date, most research has focused on inputs comprised solely of continuous features. In these scenarios, the most common adversarial example generation methods are gradient-based perturbations to the original input. Other problem domains often have discrete types of data. One popular use of AI is in Natural Language Processing (NLP) for the automation of document analysis. Logically discrete values are constrained to be one of a range of permitted values. Therefore, 'small perturbations' are not applicable to many cyber security features. For example, a dynamic analysis malware classifier might use API calls. A small perturbation of an API call *WriteFile()* might result in an invalid API call *WriteFilo()* that likely causes the program to crash. The discrete *WriteFile()* API call would need to be replaced with another valid API call. For example, *ReadFile()* [216]. This clearly also affects functionality of the malware, a topic discussed in Chapter 2.

5.4.2 Generating Adversarial Examples for Discrete Features

From an adversarial viewpoint the work on feature vulnerability and robustness assessment could be used to identify the key features to perturb, that is, the features most susceptible to adversarial examples and thus most likely to result in successful adversarial examples. Gradient descent methods for generating adversarial examples are not directly applicable to discrete features; however, variations on gradient descent can generate effective adversarial examples. For example, Yang *et al.* [217] propose a two stage 'Greedy Attack' capable of attacking text classification models. First the key features to be perturbed are identified and then perturbed in the second stage using values chosen from a dictionary. Huang *et al.* [114] take an alternative approach and propose a two-stage attack 'PWPSA' to generate DDoS adversarial examples by estimating the saliency of each position, selecting the best replacement or insertion packet for each position which is then used to perturb the original input.

Importantly ML-based intrusion detection systems can include a blend of categorical, continuous, and discrete features. Features such as port number refer to a process or service. They should generally not be perturbed. Adversaries might seek to avoid some constraints by only perturbing continuous features; however, logical, and mathematical constraints still exist. Crafting adversarial examples in constrained domains is challenging.

5.4.3 Lemmatizing and Hierarchies

In natural language processing it is common for text to be pre-processed so that words with similar semantics are grouped together. This process is called stemming or lemmatizing. This research indicates that such hierarchical groupings could help improve robustness.

5.5 Scalability of Adversarial Machine Learning

Chapter 2 assessed feature vulnerability and robustness using Recursive Feature Elimination. The features measured as least robust (those with the largest

perturbations) are eliminated. This is an iterative process and therefore computationally expensive for large numbers of features. A less precise but simpler method might use feature importance as a way to eliminate features unused for classification, or those least important to the classification. However, it is important to note that feature importance and feature robustness are different measures and must not be conflated or used interchangeably.

The generation of adversarial examples can be time consuming and computationally expensive; however, universal adversarial perturbations [126] allow the generation of a single adversarial example that causes most samples to be effectively misclassified. It is unclear if a UAP could also preserve functionality of the perturbed sample.

Chapter 4 proposed a novel hierarchical defence against adversarial examples. In addition to robustness benefits, hierarchical classifiers offer improvements in memory consumption, disk usage, and training time [207]. Therefore, hierarchical approaches may be even more appropriate for large number of classes than traditional flat classifiers.

5.6 Potential Limitations

This research focused on the robustness of models without considering privacy. Song *et al.* [218] argue that the areas of security and privacy should ideally be considered together. This is because defences in security and robustness can have unintended consequences for privacy and vice-versa. Specifically, models defended against adversarial examples might increase the risk of successful membership inference attacks. It has not been examined whether the proposed defence methods are also helpful for privacy.

The proposed hierarchical defence is limited by the number of classes: a hierarchy makes little sense for binary classifiers. Similarly, hierarchies are not useful for anomaly detection models which only focus on changes from a known

baseline. Hierarchical classifiers do not provide a mechanism for backtracking, poor decisions made at the top layer will unfortunately taint decisions at lower levels and ultimately the final classification. Moreover, hierarchical classifiers do not easily cater for probabilistic outputs and usually only provide a single output for each input. Therefore, hierarchical classifiers are less suitable for reflecting any uncertainty of a model's predictions.

No single defence can be perfect. It is suggested that the proposed novel defences could be used in combination with each other, and/or other defences such as adversarial training [79], and dropout [153].

This research considers supervised learning in which data items are labelled. From this research, it is unclear to what extent the results could apply to unsupervised, semi-supervised or active learning approaches. Moreover, adversarial examples are applicable in reinforcement learning [101], and regression models [219]. Both of these topics were out of the scope of this research.

5.7 Conclusion

In summary, this research considered current machine learning models and how their vulnerabilities could be defended. It has been shown that adversarial attacks and the proposed defences can credibly generalize to other domains and data types. Technological advances have seen machine learning systems deployed in autonomous vehicles, cyber physical systems, and defence. With such technological improvements and the increased scope of machine learning models comes additional risks. The generalizable approaches could help improve the robustness and security of models. Hence, allowing safer deployments in sensitive domains.

Chapter 6

Conclusion

6.1 Introduction

The aim of this thesis is to investigate to what extent machine learning systems can be compromised or degraded, and what methods can improve robustness in machine learning systems. The vulnerabilities of machine learning systems and how they might be mitigated were demonstrated. The trade-off between accuracy and robustness of neural networks was examined. Moreover, novel ways were discovered to improve the robustness of machine learning models whilst retaining, or improving accuracy, or F1-score. This concluding chapter revisits the research objectives, summarises the findings, and offers specific conclusions and recommendations for each objective. Additionally, the wider contributions of this work are clarified.

6.2 Knowledge Gained

This thesis generates new knowledge about the vulnerability of ML systems to adversarial examples. Further, it generates knowledge of the possibility and effectiveness of mitigations and countermeasures. Constrained methods are demonstrated for the generation of plausible functionality-preserving adversarial examples, using JSMA, such that the generated adversarial examples are within the range of 'normal' traffic. Mitigations and defences to adversarial examples are provided: a novel generalized approach for assessing the vulnerability and robustness of features, systematically removing the most vulnerable features

implementing Recursive Feature Elimination; a novel defence using hierarchical learning to decrease the attack surface of models, making the models more robust. This defence obstructs transferable adversarial examples, and additionally helps to reduce the severity of attacks employing adversarial examples.

In Chapter 2 the expectation was to achieve good coverage of the relevant literature. Approaching the literature study using the PRISMA framework as a base was a useful and effective way of systematically surveying the literature.

In Chapter 3 the expectation was that feature selection could be used to improve the robustness of models against adversarial examples by maximising the feature perturbation required to successfully manipulate a classifier. The research with a DDoS case study met these expectations. It was shown that feature selection can improve classification accuracy. This research shows a feature selection method to be effective with an increased accuracy under attack with no drop in accuracy for unperturbed samples.

In Chapter 4 it was expected that the use of hierarchical classification would improve robustness as measured by F1-Score. These expectations were met with all hierarchies improving F1-Scores above their corresponding flat classifier. This research clearly illustrates that hierarchical classification *can* help improve robustness but also raises the question of whether some hierarchies could be detrimental to robustness. Moreover, how might hierarchies be effectively attacked by an adversary?

Chapter 5 examined the generalizability of this research. It was expected that the research would be applicable to other classification models. Indeed, it was found that hierarchical classification can improve robustness on a range of classification models. Specifically, it was found that hierarchical classification can be effective when combined with other ensemble methods such as LGBM, and HBBC. This research illustrates that hierarchical classification might be a simple method of hardening models to adversarial examples. It was expected that this

research would not be less applicable to domains with discrete features; however, additional synergies were found with other domains that could be further explored. For example, might lemmatising text in natural language processing help improve robustness to adversarial examples?

6.3 Evaluation

6.3.1 Answers to Research Questions

This evaluation considers each research question in turn and summarizes how far each research question has been answered.

RQ1 - To what extent can adversarial examples influence the output of machine learning systems for intrusion detection

Chapter 2 Contributions As discussed previously most prior research in adversarial examples has been done in the visual domain to fool human perception. This work focuses on adversarial examples in cyber security domains. Contributions from Chapter 2 include:

1. In cyber security domains adversarial examples must comply with stringent constraints in order to be effective and undetected. This is advantageous for defenders because it complicates the generation of effective adversarial examples; however, it has been shown that *functionality-preserving* methods that comply with stringent constraints are feasible. Furthermore, traditional gradient descent methods if constrained to modify limited or specific features might be sufficient to generate functionality-preserving adversarial examples.

2. Adversarial examples in cyber security domains are not restricted to 'oracle' based attacks and may use the transferability property of adversarial examples in effective black-box attacks. Knowledge of the underlying machine learning model is not a prerequisite for successful attacks.

3. Human perception is not necessarily the best criterion to judge a successful attack in cyber security domains. The success of an adversarial example

depends on the specific task and context of the machine learning system. In intrusion detection the goal of an adversary is to fool the detection routine while conforming to the expected inputs of the system. It is argued that in cyber security domains a successful attack must also retain its intended function regardless of whether a human can detect the perturbation to the input.

Chapter 2 Implications Implications from Chapter 2 include:

1. Adversarial examples are part of a larger adversarial arms race [6]. Security problems are most readily thought of as a reactive process. Both defender and attacker adapt their tools and tactics in response to their opponents. Reactive approaches are unable to prevent *zero-day* attacks. Instead, through following a proactive approach defenders are able to anticipate attacks and develop appropriate countermeasures. It is proposed that researchers should follow this proactive approach. If adversarial examples are not addressed and countermeasures are not developed there is potential for novel and sophisticated attack strategies to overcome ML systems. Adversaries will adapt to defences; however, defenders must aim to be a few steps ahead.

2. Some developers may argue that their machine learning systems are safe because access to the system is safeguarded, and white-box access is not granted to the system. Therefore, they may claim that they would easily detect any oracle-based attack. This argument is invalid. The *transferability* property of adversarial examples means that successful black-box attacks are possible. Hard to attack models may be susceptible to transferable adversarial examples generated against easy to attack models. Researchers must urgently focus on countermeasures aiming to disrupt the transferability of adversarial examples.

3. Adversarial examples that do not respect domain constraints risk marking themselves as obvious anomalies. This is to the defender's advantage; however, adversarial examples that comply with these constraints may be possible.

4. Strategic attacks triggered at crucial moments might cause unwanted effects before any human could reasonably act. This establishes the need for automatic resilience to be built into systems.

Chapter 3 Contributions Contributions from Chapter 3 are:

1. The cyber security domain has strict constraints: certain features must remain unchanged; however, adversarial examples offer attackers a way to exploit weaknesses in IDS classifier, essentially masquerading a network attack as benign activity.

2. Some simple countermeasures are achievable, raising the bar for successful adversarial examples. For example, with Recursive Feature Elimination it is possible to assess the vulnerability of features before discarding the most vulnerable features. An inverse relationship between number of features and the robustness against adversarial examples is demonstrated, finding that feature selection at the training stage can improve a model's robustness to adversarial examples. Each feature in isolation may not be an excellent indicator; however, combinations might be better indicators. Feature selection can improve robustness of machine learning systems through improving the classification accuracy under attack.

3. Robustness need not come at the expense of degrading a model's performance. It can be considered that there is a trade-off between accuracy and accuracy under attack; however, accuracy will waiver depending on the properties of the selected features. Often a robust model is not as accurate as an unprotected model. Some judgement is required to consider whether the trade-off is beneficial; however, it does not follow that all countermeasures will degrade the performance of a model. Protected models might yield other benefits. For example, with Recursive Feature Elimination (Chapter 3) protected models can benefit from reduced training time, reduced execution time, and improved accuracy. In Chapter 4 The presented hierarchical defence improved F1-Scores regardless of the presence of adversarial examples.

4. A generalizable approach for assessing the vulnerability and robustness of features is provided: removing the most vulnerable features to simultaneously maintain acceptable classification accuracy whilst eliminating features that may introduce subtle attack vectors.

Chapter 3 Implications Implications from Chapter 3 are:

1. Constraints must be applied to the generation of adversarial examples, otherwise they mark themselves as aberrations and can be easily discounted as out of distribution.

2. Adversaries able to skew the classification accuracy of systems can leverage an advantage by making malicious conditions appear benign. Adversaries could gain significant advantage through camouflaging an infiltration attack as a comparatively less serious network intrusion.

3. Defences need not degrade the performance of a model. Reducing the features can yield other benefits including reduced training time, improved accuracy, and reduced execution time. Including important features for classification may force increases in perturbation size. This in turn makes the attack more overt and potentially less likely to preserve functionality.

Chapter 4 Contributions Contributions from Chapter 4 are:

1. Feasible adversarial examples attacks are demonstrated on a multiclass intrusion detection system whilst preserving functionality. Importantly, it is demonstrated that the scope of the perturbation can be constrained to one feature, remain in distribution, and yet still feasibly preserve functionality. A generalizable method for deploying black-box functionality preserving adversarial examples is provided. Subsequently a demonstration of a hierarchical defensive strategy able to mitigate against adversarial examples is presented. This defence is effective in obstructing transferability attacks.

Chapter 4 Implications Implications from Chapter 4 are:

1. In common with all other domains, machine learning models in cyber security are susceptible to adversarial examples; however, machine learning models in cyber security are a valuable target for adversaries. They are perhaps more likely to be targeted, and the consequences are likely more severe. A classifier in the cyber security domain, if fooled by an adversarial example, risks allowing malware onto a network. This is concerning because adversarial attacks against intrusion detection systems are feasible and can be both successful and undetected. Therefore, the existence of adversarial examples limits the domains in which machine learning systems can be deployed. Machine learning could possibly make great contributions in the domains of cyber security, cyber-physical systems, and defence. Improving the robustness of machine learning models to adversarial examples enables safer deployment of ML across a wider range of domains.

2. Attackers can avoid problems of 'oracle' attacks by using black-box attacks based on the transferability property. Black-box attacks need zero knowledge.

3. Hierarchical defences force attacks to be more overt, and thus adversarial examples have a greater likelihood of being detected.

RQ2 - To what extent can countermeasures and defensive approaches mitigate the effects of adversarial examples for Intrusion Detection Systems?

Chapter 2 Contributions As discussed previously, most prior research in adversarial examples has been done in the visual domain to fool human perception. This work focuses on adversarial examples in cyber security domains. Contributions from Chapter 2 are:

1. A suite of complimentary defences is likely the best approach. A single defence is unlikely to defend against all adversarial examples. For example, adversarial training is a common and simple defence. It presents a useful but minor hurdle to adversaries; however, successful evasion attacks can be found for machine learning systems that have been explicitly trained on

adversarial examples. Adversarial training is not scalable and needs to be bolstered by other defences. Data-level resampling techniques provide some robustness benefit; however, experiments in Chapter 4 show they were insufficient to prevent adversarial examples even in combination with hierarchical learning strategies. It is unlikely that any one defence is a panacea against adversarial examples. Instead, a Swiss-cheese approach [220] is proposed, in which a suite of imperfect defences are imperfect in different ways; however, they present a much stronger defence as a whole.

2. Traditional gradient descent methods, if constrained to modify limited or specific features, might be sufficient to generate functionality-preserving adversarial examples.

Chapter 2 Implications Implications from Chapter 2 are:

1. Adversarial examples exist as part of an adversarial arms race. Adversaries will adapt and adopt new strategies, and defenders must also adapt to thwart their adversaries. Game Theory approaches allowing Nash equilibrium could potentially end the evolutionary arms race.

2. No single defence is likely to block all adversarial examples; therefore, a Swiss-cheese defence could incorporate multiple defence methods. For example, architectural, feature selection, adversarial training, ensembles, randomisation, non-determinism.

Chapter 3 Contributions Contributions from Chapter 3 are:

1. A demonstration of a useful countermeasure is given, including a generalizable approach for assessing the vulnerability and robustness of features through removing the most vulnerable features to simultaneously maintain acceptable classification accuracy whilst eliminating features that may introduce subtle attack vectors. Applying systematic feature selection for model training can improve the robustness of the model against adversarial examples. Each feature in isolation may not be an excellent indicator; however, combinations might be better indicators. Feature selection can

improve robustness of machine learning systems through improving the classification accuracy under attack.

Chapter 3 Implications Implications from Chapter 3 include:

1. Inclusion of important features for classification may force increases in perturbation size. This in turn makes the attack more overt and potentially less likely to preserve functionality.

2. Reducing the features can yield other benefits including reduced training time, improved accuracy, and reduced execution time. Model performance need not be degraded by defences.

Chapter 4 Contributions Contributions from Chapter 4 include:

1. A demonstration that the scope of perturbation can be constrained to one feature, yet still feasibly preserve functionality, whilst remaining in distribution.

2. A hierarchical learning defensive strategy to mitigate against adversarial examples. It was shown that model performance need not be degraded by defences.

3. Algorithms were defined for creating hierarchical labels from labelled and unlabelled datasets. This could help the adoption of hierarchical defences for other machine learning scenarios and datasets.

Chapter 4 Implications Implications from Chapter 4 include:

1. Hierarchical defences force adversarial examples to be more overt, and thus have a greater likelihood of being detected. Improving the robustness of ML models enables safer deployment of ML across a wider range of domains. Hierarchical defences could be used as part of a suite of robustness enhancing methods and defences.

RQ3 - To what extent is this work generalizable to other scenarios, datasets, and data types?

Chapter 5 Contributions Contributions from Chapter 5 are:

1. An evaluation of the vulnerability of common ML models to adversarial examples.
2. A focus on the generalizability of this work to other scenarios, models, and data types. This work could inform design choices of model designers.
3. Recommendations that could inform deployment policy for ML systems.

Chapter 5 Implications Implications from Chapter 5 include:

1. The evaluation of common models helps inform model designers and raise awareness of the security risks and trade-offs of model design. This helps developers and researchers design robust models. More robust models help extend the domains where machine learning models can be deployed, in particular extending their useful deployment to adversarial environments.
2. Automatic analysis tools allow greater understanding of the risks of model deployment and help prevent the deployment of vulnerable models.

6.4 Recommendations and Future Challenges

It was found that adversarial examples can easily influence the output of machine learning systems. Therefore, developers and users of these systems must recognize, understand, and mitigate these risks. In this section, recommendations are highlighted.

- REC01 System designers recognize that keeping their model architecture and parameters secret is insufficient to prevent adversarial attacks. It was found that transferability property of adversarial examples is powerful. Knowledge of the underlying machine learning model and parameters is not necessary to effect attacks.

- REC02 All machine learning systems in uncontrolled environments should have at least some basic protection against adversarial examples. No single defence is likely to block all attacks. Therefore, it is recommended that that a suite of complementary defences is used, such as those presented in this thesis.

- REC03 System designers should use up-to-date datasets. The use of modern datasets is recommended in order to represent the current situation. In Chapter 2 it was found that many academic researchers use old datasets, given the difficulty in collecting or obtaining new datasets. It should, however, be recognised where concept-drift may occur, and if older datasets are no longer indicative of modern networks and their usage.

- REC04 System designers should carefully consider the most appropriate features to use in model training and test. Designers should consider feature importance as part of their decision process as to whether to include certain features. This requires an understanding of the problem domain. Designers should have a rationale as to why that feature may influence the output class. Combinations of important features are likely to perform better than a single feature. System designers should carefully consider the most appropriate features in order to achieve acceptable classification accuracy while eliminating features that could introduce subtle attack vectors. It is recommended that system designers employ combinations of robust features to improve robustness.

- REC05 System designers should sanitize inputs by checking any constraints. Although more complex adversarial example algorithms may meet constraints, simple adversarial examples often break constraints. Simple constraints checks could offer basic protection against some simple adversarial example algorithms.

- REC06 System designers should perform analysis of the risks of adversarial examples. The tolerance to false positives and false negatives should be defined. It is recommended that risk analysis is performed before

deployment. Valuable targets are perhaps more likely to be targeted and consequences may be more severe.

- REC07 Particular attention is required when deploying machine learning in environments where the risks are high. Attention is especially required if the environment is not fully controlled.
- REC08 A suite of complementary defences is recommended over a single powerful defence.
- REC09 Test the model for robustness using an appropriate robustness metric, such as the generalization error between F1-Scores. Models unable to meet a minimum robustness threshold should not be deployed. It is recommended that defences against adversarial examples be incorporated in design and deployment policy. Requirements on a minimum level of robustness should be incorporated into system requirements.
- REC10 Developers should anticipate attacks and develop appropriate countermeasures. It is recommended that that developers follow a *proactive* approach to the adversarial arms race.

6.4.1 Future Challenges

Adversarial ML is a critical area of research. If not addressed, there is increasing potential for novel attack strategies that seek to exploit the inherent weaknesses that exist within machine learning models. There are many challenges. Here a broad range of open research topics are outlined. Researchers must address the robustness of ML models against adversarial examples allowing safer deployment of ML models across cyber security domains. Standardized robustness metrics should be used and developed. Few works consider '*realisable*' perturbations that take account of domain and/or real-world constraints. Successful adversarial examples must be crafted to comply with domain and real-world constraints. This may be challenging since even small modifications may corrupt network packets that are likely to be dropped by firewalls. It was found that the traditional benchmark of human perception may be less relevant in functionality-preservation.

Randomisation of decision boundaries can make evasion more difficult. Moreover, research into multi-classifier systems could help thwart evasion attacks, making it harder to evade classification. Dropout is currently a promising defence against adversarial examples; although multiple defences may be required, and a combination of defences will likely offer better defence capability. Game theory approaches could potentially end the adversarial arms race by achieving a Nash equilibrium. Concept-drift requires further research. Many researchers are using outdated datasets. Simply using newer datasets could postpone problems of concept-drift and is a good first step. Unsupervised/semi-supervised and active learning could potentially offer longer term solutions to concept-drift, aiming for models to learn and detect novel attack methods. The transferability of adversarial examples remains an open issue, more research here has the potential to disrupt many attack strategies. Additionally, more research is required in the area of functionality-preserving adversarial attacks, recognising the limits and trade-offs between functionality-preserving adversarial examples and their ability to evade classification; moreover, research into adversarial attacks in other constrained domains could improve robustness against complex attacks.

In a broader cyber security context, risks arising from adversarial examples are not yet fully understood. Additionally, algorithms and models from other domains may not readily apply, because of distributed sensors and inherent real-world constraints. It is uncertain whether current defences are sufficient. Furthermore, adversarial example detectors must function efficiently in a real-time monitoring environment while maintaining low false alarm rates.

The future research areas are prioritised, setting the agenda for research in this area. Critical areas of research include: breaking the transferability of adversarial examples that would hopefully be applicable across domains. Non-visual domains including cyber security and cyber-physical systems have been under-explored and this oversight should be rectified urgently. Further research on transformations in nonvisual domains could provide useful knowledge. Detection of adversarial examples and pushing the fields of cyber security, intrusion detection, and

cyber-physical systems will yield benefits beyond cyber security and may be applicable in other nonvisual domains. Moreover, research is required in areas beyond instance classifiers. Areas of RNNs and reinforcement learning have been under-explored. More research is required to understand the use of domain constraints and functionality-preserving adversarial examples. Further research is needed towards effective countermeasures.

Few researchers address the problem of transferability, which remains a key area of concern because hard-to-attack models are nevertheless susceptible to *transferable* adversarial examples generated against easy-to-attack models. Breaking the transferability of adversarial examples is a key challenge for the research community. Currently, defensive dropout [153] is a promising defence. Defences like dropout exchange a relatively small decrease in accuracy for significant reduction of successful attacks, even successfully blocking black-box and transferability-based attacks. Hardening techniques force successful attacks to use larger perturbations, which in turn may be more readily recognized as adversarial.

The area of functionality-preserving adversarial examples is under-explored. The generation of adversarial examples that respect constraints may be critical to generating functionality-preserving adversarial examples. Research into improving robustness against such adversarial examples is an area requiring urgent research. Adapting defences used in the visual domain and CNN models to other model types such as RNNs could offer potential solutions; however, caution should be exercised when adapting defences in the visual domain to other domains. For example, denoising defences may not apply directly to discrete or noncontinuous data. Constraints on adversarial examples are not limited to preserving the functionality of malware or IDS attacks and have wider implications for cyber physical systems and industrial control systems. CPSs model the real world where linear and other physical constraints must be respected. Adversarial examples that do not respect domain constraints risk marking themselves as obvious anomalies. This research uses the statistical information available in netflows that is collated from the raw packet data on networks. Netflows often form the basis of intrusion

detection systems. The next logical step for this work is to construct raw data packets, such that the resulting collated statistics embody an adversarial example, enabling misclassification network traffic.

More research attention could be given to dataset resampling strategies as a defence against adversarial examples. There is a need for standardized robustness metrics. Some researchers simply state improvement in accuracy, others might state the better F1-Score; however, F1-Score is biased by unbalanced datasets which are widespread in intrusion detection partly due to large numbers of benign samples. Using F1-Score on unbalanced datasets could lead to a false sense of security. Researchers should adopt stronger metrics such as CLEVER [80] or Empirical Robustness [83]. It was identified that data-level techniques, such as resampling, balancing datasets, and Cross Validation could have effects on robustness against adversarial examples. Further research is required to explore how the bias-variance trade-off can affect robustness.

Traditionally, adversarial examples are thought of as having imperceptible noise. That is, that humans cannot perceive the difference between the original and perturbed inputs. It is posited that human perception may not be the best criterion for analysing adversarial examples. Indeed, human perception in some domains might be immaterial. For example, strategic attacks triggered at crucial moments might cause damage to CPS before any human could reasonably act. In cyber security domains it is proposed that adversarial examples must preserve functionality.

Concept-drift is a real concern for cyber security [8], as new attacks and techniques are discovered daily. As the model and the current state of the art diverge, the model suffers from hidden technical debt. Therefore, the model must be retrained to reflect the current state-of-the-art attacks and new network traffic patterns [184]. Researchers might develop and use more up-to-date datasets. Problems of labelling data and retraining systems provide an impetus to explore the further avenues for research of semi-supervised/unsupervised ML, and active

learning methods that continuously update the underlying model, and do not rely on labelled datasets. Unfortunately, adversarial attacks are possible on active learning systems [221]. Lin *et al.* [101] describe an enchanting attack to lure a machine learning system to a target state through crafting a series of adversarial examples. It is conceivable that similar attacks could lure anomaly detection systems towards normalizing and accepting malicious traffic. It would be constructive to examine whether hierarchical classifiers are less prone to problems of concept-drift than flat classifiers.

In cyber security domains traditional gradient descent algorithms may be insufficient; although JSMA may be a reasonable choice because it perturbs few features. Stringent constraints exist in the cyber security domain and extreme care must be taken to create valid adversarial examples. Some guidelines for generating functionality-preserving adversarial examples are offered. Functionality-preserving adversarial examples should: only perform legitimate transformations; respect mathematical dependencies, real-world, and domain constraints; minimize the number of perturbed features; restrict modification to non-critical features; and where possible, retain the original payload and/or packet order.

Defences against adversarial examples must consider that adversaries are likely to adapt by adopting new strategies. Many researchers propose adversarial training to improve robustness. Adversarial training is a simple method aiming to improve robustness; however, it is potentially a cosmetic solution: the problem of adversarial examples cannot be solved only through ever greater numbers of adversarial examples in the training data. Adversarial training, if used, must be bolstered by other defences. Interesting defence strategies include randomisation, including dropout: randomising decision boundaries makes evasion more difficult because attackers have less information on the exact position of a decision boundary. They must therefore make too conservative or too risky choices when generating adversarial examples. It should be considered how hierarchical classifiers could be attacked and indeed whether they can be made more robustly. It is hoped that the proposed defences prove to be robust; however, other researchers

are encouraged to research offensive strategies to find and exploit vulnerabilities in hierarchical classifiers.

Game theoretic models could be used to address more complex situations with many adversaries of different types, as found in intrusion detection. Equilibrium strategies acceptable to both defender and adversary mean neither has an incentive to change. Therefore, assuming rational opponents, game theory-based approaches allowing a Nash equilibrium could potentially end the evolutionary arms race. Although, it is difficult to conceive a world where no advantage is possible.

This research could be further expanded through the use of: multiple and larger datasets; a greater selection of machine learning models; The generation of adversarial examples using a greater range of algorithms. The appropriate robustness metrics of generalization error based on F1-Score are used; however, other robustness metrics such as CLEVER [80] might provide a more detailed analysis of the robustness of the presented models and defences. A noticeable omission is the comparison of the presented defences against existing defences. It is difficult to rank the proposed defences among other existing defences; however, the proposed defences could be considered as part of a Swiss-cheese defence [220]: Reason describes defences as like slices of Swiss cheese, having many holes. The presence of holes in any *one* 'slice' does not normally cause bad outcomes. Bad outcomes usually only occur when the holes in many layers momentarily line up to permit a 'trajectory of opportunity'. Multiple layers of defence are unlikely to have vulnerabilities in exactly the same place. Therefore, multiple layers of defence decrease the likelihood of any exploitable gap in defence. In cyber security this is related to concepts of defence in depth and layered security. Importantly, it may be that model designers and developers are unwittingly developing and deploying models that are vulnerable to adversarial examples. Research into methods to warn, influence, and educate model designers of the risks is a worthy pursuit.

6.5 Overall Evaluation

This thesis is successful in its aims. Two novel scalable defences are offered that successfully improve the robustness of models to adversarial examples. Additionally, consideration was given to: the most common models; common adversarial example algorithms; and modern datasets. The nascent area of functionality-preserving adversarial examples received much consideration. For convenience, the main contributions of this work are restated:

- In Chapter 2 a systematic literature survey of adversarial machine learning was presented, with a focus on the relatively new research topic of functionality-preserving adversarial examples in the cyber security and intrusion detection domains. Attacks and current defences were examined [42].

- In Chapter 3 a novel defence against adversarial examples was presented that employs feature selection and Recursive Feature Elimination [43].

- Chapter 4 presents the **first comprehensive study of applying functionality-preserving adversarial learning attacks** against a multi-class network traffic classification model and demonstrates successful attack misclassification within a constrained attack parameter space. Over ninety percent (90.25%) of the attacks were able to evade detection of a well-trained classifier, while also constraining the parameter space to feasibly preserve functionality [44].

- **Generalizability:** All domains (to date) are vulnerable to adversarial examples. Common models and modern datasets were used. There is confidence that this research is applicable to cyber security and intrusion detection domains. For example, it has been shown to generalize to other datasets in Chapter 5; however, it is less certain how this research might relate to unsupervised, semi-supervised and active learning methods.

- **Validity:** Valid research requires an appropriate research strategy, data collection, and analysis. Appropriate experiments were designed, controlling for extraneous variables by cleansing the dataset of null or missing data.

Steps were taken to balance the classes and scale the features. The experiments partitioned the data into training and validation data to accurately determine how the models performed on unseen data. Standardized versions of adversarial attack algorithms were used to allow fair comparisons. Regular supervision has also helped strengthen the validity of this research.

- **Reliability:** Reliable research is trustworthy, unbiased, and evidenced. The experiments were regularly and repeatedly rerun with similar results observed each time. The source code used in the experiments can be found in the author's repository, aiding reproducibility and reliability of this research. Peer review of the associated published papers bolsters the reliability of this research. Regular supervision has also helped bolster the reliability of this research. Further, the author declares no competing interests.

This thesis provides model designers and researchers an insight into the risks of adversarial examples and effective mitigations. This thesis is applicable to other security-sensitive domains such as cyber-physical systems and industrial control systems. Two novel defences are offered, hoping to influence the design and deployment policies related to security-sensitive machine learning models. This thesis has three related publications, sharing the knowledge gained through this work with model designers, researchers, and the wider community. It is hoped that this thesis offers other researchers a base for exploring the areas of robustness in adversarial machine learning and functionality-preserving adversarial examples. Two novel defences are offered for improving robustness, and it is hoped that these will enable the design and deployment of more robust machine learning models.

A more precise interpretation of the results acknowledges that the two defences presented in this thesis have been demonstrated using the modern CICIDS2017 [66] dataset and its derivative [38]. Additionally, the research was limited in the types of adversarial example that were employed: JSMA and FGSM. The author is optimistic that the defences are likely to be appropriate in a wider context. In particular, the hierarchical defence was shown to generalize to the

MQTT-IOT-IDS2020 dataset [71]. For the MQTT study (Chapter 5, the hierarchical defences improved F1-Scores for all the presented hierarchies. It is posited that the hierarchical defence drives the generated adversarial examples to be larger and perhaps more readily identified as adversarial. However, the experiment was limited to JSMA adversarial examples. The performance of the defended models with other common algorithms is untested. The presented attack is designed to be representative of a black-box attack. The generation of white-box adversarial examples against the hierarchical model is compounded because an adversarial example may need to fool two or more separate local classifiers. Alternatively, an attack could target a specific classifier within the hierarchy. Therefore, the generation of adversarial examples against a hierarchical model may be possible. The robustness of the hierarchical defence against architecture aware adversaries employing white-box attacks is not examined in this thesis, although it is now becoming recognised as an open research topic [205][206].

Appendix A

Code Repository Links

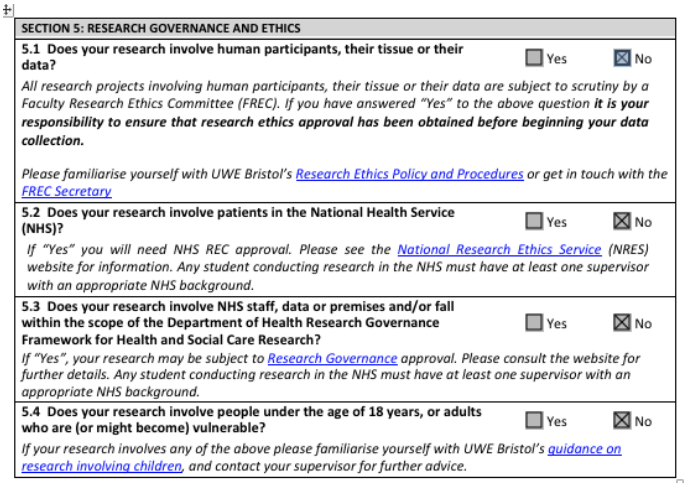
The code for experiments has been developed in Python notebooks often using Google Colaboratory. The Python code for the experiments is available in the author's repository, here:

- https://github.com/mccarthyajb/HL-NTAC
- https://github.com/mccarthyajb/ML-Feature-Robustness

Appendix B

Ethics

This research received ethical approval from the University of the West of England, after completing the ethics governance and research documentation as shown in Figure B.1. All work was carried out under the University of the West of England Code of Good Research Conduct and a Policy on Good Research Conduct: https://www.uwe.ac.uk/research/policies-and-standards/code-of-good-research-conduct.



SECTION 5: RESEARCH GOVERNANCE AND ETHICS

5.1 Does your research involve human participants, their tissue or their data? ☐ Yes ☒ No

All research projects involving human participants, their tissue or their data are subject to scrutiny by a Faculty Research Ethics Committee (FREC). If you have answered "Yes" to the above question ***it is your responsibility to ensure that research ethics approval has been obtained before beginning your data collection.***

Please familiarise yourself with UWE Bristol's Research Ethics Policy and Procedures or get in touch with the FREC Secretary

5.2 Does your research involve patients in the National Health Service (NHS)? ☐ Yes ☒ No

If "Yes" you will need NHS REC approval. Please see the National Research Ethics Service (NRES) website for information. Any student conducting research in the NHS must have at least one supervisor with an appropriate NHS background.

5.3 Does your research involve NHS staff, data or premises and/or fall within the scope of the Department of Health Research Governance Framework for Health and Social Care Research? ☐ Yes ☒ No

If "Yes", your research may be subject to Research Governance approval. Please consult the website for further details. Any student conducting research in the NHS must have at least one supervisor with an appropriate NHS background.

5.4 Does your research involve people under the age of 18 years, or adults who are (or might become) vulnerable? ☐ Yes ☒ No

If your research involves any of the above please familiarise yourself with UWE Bristol's guidance on research involving children, and contact your supervisor for further advice.

FIGURE B.1: Completed Research Ethics Form.

Bibliography

[1] Maddy Ell and Robbie Gallucci. “Cyber security breaches survey 2022: Statistical release”. In: (2022).

[2] Nour Alqudah and Qussai Yaseen. “Machine learning for traffic analysis: a review”. In: *Procedia Computer Science* 170 (2020), pp. 911–916.

[3] Gueltoum Bendiab et al. “IoT malware network traffic classification using visual representation and deep learning”. In: *2020 6th IEEE Conference on Network Softwarization (NetSoft)*. IEEE. 2020, pp. 444–449.

[4] Amirah Alshammari and Abdulaziz Aldribi. “Apply machine learning techniques to detect malicious network traffic in cloud computing”. In: *Journal of Big Data* 8.1 (2021), pp. 1–24.

[5] Nuno Martins et al. “Adversarial machine learning applied to intrusion and malware scenarios: a systematic review”. In: *IEEE Access* 8 (2020), pp. 35403–35419.

[6] Battista Biggio and Fabio Roli. “Wild patterns: Ten years after the rise of adversarial machine learning”. In: *Pattern Recognition* 84 (2018), pp. 317–331.

[7] Tom M Mitchell and Tom M Mitchell. *Machine learning*. Vol. 1. 9. McGraw-hill New York, 1997.

[8] Giuseppina Andresini et al. “INSOMNIA: Towards Concept-Drift Robustness in Network Intrusion Detection”. In: *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security (AISec)*. ACM. 2021, p. 0.

[9] Chandni Raghuraman et al. “Static and dynamic malware analysis using machine learning”. In: *First International Conference on Sustainable Technologies for Computational Intelligence*. Springer. 2020, pp. 793–806.

[10] Harel Berger, Chen Hajaj, and Amit Dvir. "Evasion Is Not Enough: A Case Study of Android Malware". In: *International Symposium on Cyber Security Cryptography and Machine Learning*. Springer. 2020, pp. 167–174.

[11] Ruitao Hou et al. "Universal Adversarial Perturbations of Malware". In: *International Symposium on Cyberspace Safety and Security*. Springer. 2020, pp. 9–19.

[12] Sergei Parshutin et al. "Classification with LSTM Networks in User Behaviour Analytics with Unbalanced Environment". In: *Automatic Control and Computer Sciences* 55.1 (2021), pp. 85–91.

[13] Duc C Le and Nur Zincir-Heywood. "Exploring anomalous behaviour detection and classification for insider threat identification". In: *International Journal of Network Management* 31.4 (2021), e2109.

[14] Sumitra Biswal. "Real-Time Intelligent Vishing Prediction and Awareness Model (RIVPAM)". In: *2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*. IEEE. 2021, pp. 1–2.

[15] Nikhil Kumar, Sanket Sonowal, et al. "Email Spam Detection Using Machine Learning Algorithms". In: *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*. IEEE. 2020, pp. 108–113.

[16] Douwe Kiela et al. "The hateful memes challenge: Detecting hate speech in multimodal memes". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 2611–2624.

[17] Salman Bin Naeem and Maged N Kamel Boulos. "COVID-19 misinformation online and health literacy: A brief overview". In: *International Journal of Environmental Research and Public Health* 18.15 (2021), p. 8091.

[18] Travis Coan et al. "Computer-assisted detection and classification of misinformation about climate change". In: *SocArXiv* (2021).

[19] Z Khanam et al. "Fake News Detection Using Machine Learning Approaches". In: *IOP Conference Series: Materials Science and Engineering*. Vol. 1099. IOP Publishing. 2021, p. 012040.

[20] Sauptik Dhar et al. "A survey of on-device machine learning: An algorithms and learning theory perspective". In: *ACM Transactions on Internet of Things* 2.3 (2021), pp. 1–49.

[21] Nicolas Papernot et al. "Sok: Security and privacy in machine learning". In: *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2018, pp. 399–414.

[22] Christian Szegedy et al. "Intriguing properties of neural networks". English (US). In: *International Conference on Learning Representations, ICLR 2014*. 2nd International Conference on Learning Representations, ICLR 2014 ; Conference date: 14-04-2014 Through 16-04-2014. Jan. 2014, p. 0.

[23] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *arXiv* (2014).

[24] Susan G Wardle et al. "Rapid and dynamic processing of face pareidolia in the human brain". In: *Nature communications* 11.1 (2020), pp. 1–14.

[25] Christopher Summerfield et al. "Mistaking a house for a face: neural correlates of misperception in healthy humans". In: *Cerebral cortex* 16.4 (2006), pp. 500–508.

[26] Yonghong Huang et al. "Malware Evasion Attack and Defense". In: *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. 2019, pp. 34–38. DOI: 10.1109/DSN-W.2019.00014.

[27] Md. Ahsan Ayub et al. "Model Evasion Attack on Intrusion Detection Systems using Adversarial Machine Learning". In: *2020 54th Annual Conference on Information Sciences and Systems (CISS)*. 2020, pp. 1–6. DOI: 10.1109/CISS48834.2020.1570617116.

[28] Chris Hankin and Martín Barrère. "Trustworthy Inter-connected Cyber-Physical Systems". In: *International Conference on Critical Information Infrastructures Security*. Springer. 2020, pp. 3–13.

[29] Jin-Hee Cho et al. "Stram: Measuring the trustworthiness of computer-based systems". In: *ACM Computing Surveys (CSUR)* 51.6 (2019), pp. 1–47.

[30] Algirdas Avizienis et al. "Basic concepts and taxonomy of dependable and secure computing". In: *IEEE transactions on dependable and secure computing* 1.1 (2004), pp. 11–33.

[31] E Barker et al. "A Framework for Designing Cryptographic Key Management Systems, Draft Special Publication 800–130". In: *National Institute of Standards and Technology* (2010).

[32] Charles P. Pfleeger, Shari Lawrence Pfleeger, and Jonathan Margulies. *Security in Computing (5th Edition)*. 5th. USA: Prentice Hall Press, 2015. ISBN: 0134085043.

[33] Yacov Y. Haimes. "On the Definition of Resilience in Systems". In: *Risk Analysis* 29.4 (2009), pp. 498–501. DOI: https://doi.org/10.1111/j.1539-6924.2009.01216.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1539-6924.2009.01216.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1539-6924.2009.01216.x.

[34] Tao Bai, Jinqi Luo, and Jun Zhao. "Recent Advances in Understanding Adversarial Robustness of Deep Neural Networks". In: *arXiv preprint arXiv:2011.01539* (2020).

[35] Jin-Hee Cho, Ananthram Swami, and Ray Chen. "A survey on trust management for mobile ad hoc networks". In: *IEEE communications surveys & tutorials* 13.4 (2010), pp. 562–583.

[36] Jin-Hee Cho, Hasan Cam, and Alessandro Oltramari. "Effect of personality traits on trust and risk to phishing vulnerability: Modeling and analysis". In: *2016 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*. IEEE. 2016, pp. 7–13.

[37] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. "A Detailed Analysis of the CICIDS2017 Data Set". In: Cham: Springer International Publishing, 2019, pp. 172–188.

[38] Gints Engelen, Vera Rimmer, and Wouter Joosen. "Troubleshooting an intrusion detection dataset: the CICIDS2017 case study". In: *2021 IEEE Security and Privacy Workshops (SPW)*. IEEE. 2021, pp. 7–12.

[39] Mahendran Botlagunta et al. "Classification and diagnostic prediction of breast cancer metastasis on clinical data using machine learning algorithms". In: *Scientific Reports* 13.1 (2023), pp. 1–17.

[40] Raphael Satter. *Experts who wrestled with SolarWinds hackers say cleanup could take months - or longer*. en. -12-24 2020. URL: https://www.reuters.com/article/us-global-cyber-usa-solarwinds-idUSKBN28Y1K3.

[41] Sara Sirota. "Air Force response to SolarWinds hack: Preserve commercial partnerships, improve transparency into security efforts". English. In: *Inside Cybersecurity* (Jan. 2021). Name - Department of Defense; Copyright - Copyright Inside Washington Publishers Jan 12, 2021; Last updated - 2021-01-13. URL: https://search-proquest-com.ezproxy.uwe.ac.uk/trade-journals/air-force-response-solarwinds-hack-preserve/docview/2477182241/se-2?accountid=14785.

[42] Andrew McCarthy et al. "Functionality-Preserving Adversarial Machine Learning for Robust Classification in Cybersecurity and Intrusion Detection Domains: A Survey". In: *Journal of Cybersecurity and Privacy* 2.1 (2022), pp. 154–190.

[43] Andrew McCarthy et al. "Feature Vulnerability and Robustness Assessment against Adversarial Machine Learning Attacks". In: *2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*. 2021, pp. 1–8. DOI: 10.1109/CyberSA52016.2021.9478199.

[44] Andrew McCarthy et al. "Defending against adversarial machine learning attacks using hierarchical learning: A case study on network traffic attack classification". In: *Journal of Information Security and Applications* 72 (2023), p. 103398.

[45] Xiaozhe Gu and Arvind Easwaran. "Towards Safe Machine Learning for CPS: Infer Uncertainty from Training Data". In: *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*. ICCPS '19.

Montreal, Quebec, Canada: Association for Computing Machinery, 2019, pp. 249–258. ISBN: 9781450362856. DOI: 10.1145/3302509.3311038. URL: https://doi.org/10.1145/3302509.3311038.

[46] Amin Ghafouri, Yevgeniy Vorobeychik, and Xenofon Koutsoukos. "Adversarial regression for detecting attacks in cyber-physical systems". In: *International Joint Conference on Artificial Intelligence*. 2018.

[47] Igino Corona, Giorgio Giacinto, and Fabio Roli. "Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues". In: *Information Sciences* 239 (2013), pp. 201–225.

[48] Jiliang Zhang and Chen Li. "Adversarial examples: Opportunities and challenges". In: *IEEE transactions on neural networks and learning systems* (2019).

[49] Giovanni Apruzzese et al. "Modeling Realistic Adversarial Attacks against Network Intrusion Detection Systems". In: *Digital Threats: Research and Practice* 0.ja (2021). ISSN: 2692-1626. DOI: 10.1145/3469659. URL: https://doi.org/10.1145/3469659.

[50] C. E. Shannon. "Communication theory of secrecy systems". In: *The Bell System Technical Journal* 28.4 (1949), pp. 656–715. DOI: 10.1002/j.1538-7305.1949.tb00928.x.

[51] Olga Taran, Shideh Rezaeifar, and Slava Voloshynovskiy. "Bridging machine learning and cryptography in defence against adversarial attacks". In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 2018, pp. 0–0.

[52] Yirui Wu, Dabao Wei, and Jun Feng. "Network attacks detection methods based on deep learning techniques: a survey". In: *Security and Communication Networks* 2020 (2020).

[53] Mahbod Tavallaee et al. "A detailed analysis of the KDD CUP 99 data set". In: *2009 IEEE symposium on computational intelligence for security and defense applications*. IEEE. 2009, pp. 1–6.

[54] John McHugh. "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory". In: *ACM Transactions on Information and System Security (TISSEC)* 3.4 (2000), pp. 262–294.

[55] Vinton G Cerf. "2021 Internet Perspectives". In: *IEEE Network* 35.1 (2021), pp. 3–3.

[56] Martin McKeay. *Akamai state of the Internet / security: A Year in Review*. http://akamai.com/soti. 2020.

[57] SH Kok et al. "A review of intrusion detection system using machine learning approach". In: *International Journal of Engineering Research and Technology* 12.1 (2019), pp. 8–15.

[58] Huda Ali Alatwi and Charles Morisset. "Adversarial Machine Learning In Network Intrusion Detection Domain: A Systematic Review". In: *arXiv e-prints* (2021), arXiv–2112.

[59] S Revathi and A Malathi. "A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection". In: *International Journal of Engineering Research & Technology (IJERT)* 2.12 (2013), pp. 1848–1853.

[60] M Gharaibeh and C Papadopoulos. "DARPA 2009 intrusion detection dataset". In: *Colorado State Univ., Tech. Rep* (2014).

[61] Sebastian Garcia et al. "An empirical comparison of botnet detection methods". In: *computers & security* 45 (2014), pp. 100–123.

[62] Jungsuk Song et al. "Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation". In: *Proceedings of the first workshop on building analysis datasets and gathering experience returns for security*. 2011, pp. 29–36.

[63] Nour Moustafa and Jill Slay. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)". In: *2015 military communications and information systems conference (MilCIS)*. IEEE. 2015, pp. 1–6.

[64] Iman Almomani, Bassam Al-Kasasbeh, and Mousa Al-Akhras. "WSN-DS: A dataset for intrusion detection systems in wireless sensor networks". In: *Journal of Sensors* 2016 (2016).

[65] Quamar Niyaz, Weiqing Sun, and Ahmad Y Javaid. "A Deep Learning Based DDoS Detection System in Software-Defined Networking (SDN)". In: *EAI Endorsed Transactions on Security and Safety* 4.12 (2017), e2–e2.

[66] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. "Toward generating a new intrusion detection dataset and intrusion traffic characterization." In: *ICISSp* 1 (2018), pp. 108–116.

[67] Manos Antonakakis et al. "Understanding the mirai botnet". In: *26th {USENIX} security symposium ({USENIX} Security 17)*. 2017, pp. 1093–1110.

[68] Nickolaos Koroniotis et al. "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset". In: *Future Generation Computer Systems* 100 (2019), pp. 779–796.

[69] Yisroel Mirsky et al. "Kitsune: an ensemble of autoencoders for online network intrusion detection". In: *arXiv preprint arXiv:1802.09089* (2018).

[70] Andrzej Janusz et al. "IEEE BigData 2019 cup: suspicious network event recognition". In: *2019 IEEE International Conference on Big Data (Big Data)*. IEEE. 2019, pp. 5881–5887.

[71] Hanan Hindy et al. "MQTT-IoT-IDS2020: MQTT Internet of Things Intrusion Detection Dataset". In: (2020). DOI: 10.21227/bhxy-ep04. URL: https://dx.doi.org/10.21227/bhxy-ep04.

[72] Andrey Ferriyan et al. "Generating Network Intrusion Detection Dataset Based on Real and Encrypted Synthetic Attack Traffic". In: *Applied Sciences* 11.17 (2021). ISSN: 2076-3417. DOI: 10.3390/app11177868. URL: https://www.mdpi.com/2076-3417/11/17/7868.

[73] David Gonzalez-Cuautle et al. "Synthetic minority oversampling technique for optimizing classification tasks in botnet and intrusion-detection-system datasets". In: *Applied Sciences* 10.3 (2020), p. 794.

[74] Justin M Johnson and Taghi M Khoshgoftaar. "Survey on deep learning with class imbalance". In: *Journal of Big Data* 6.1 (2019), pp. 1–54.

[75] Ryan Sheatsley et al. "Adversarial Examples in Constrained Domains". In: *arXiv preprint arXiv:2011.01183* (2020).

[76] Payam Refaeilzadeh, Lei Tang, and Huan Liu. "Cross-validation." In: *Encyclopedia of database systems* 5 (2009), pp. 532–538.

[77] Muhammad Shafique et al. "Robust machine learning systems: Challenges, current trends, perspectives, and the road ahead". In: *IEEE Design & Test* 37.2 (2020), pp. 30–57.

[78] Raphael Labaca-Castro, Battista Biggio, and Gabi Dreo Rodosek. "Poster: Attacking malware classifiers by crafting gradient-attacks that preserve functionality". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2019, pp. 2565–2567.

[79] Tao Bai et al. "Recent Advances in Adversarial Training for Adversarial Robustness". In: *arXiv e-prints* (2021), arXiv–2102.

[80] Tsui-Wei Weng et al. "Evaluating the Robustness of Neural Networks: An Extreme Value Theory Approach". In: *International Conference on Learning Representations*. 2018.

[81] Ian Goodfellow. "Gradient masking causes clever to overestimate adversarial perturbation size". In: *arXiv preprint arXiv:1804.07870* (2018).

[82] Tsui-Wei Weng et al. "On extensions of clever: A neural network robustness evaluation algorithm". In: *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE. 2018, pp. 1159–1163.

[83] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016, p. 0.

[84] Nicholas Carlini and David Wagner. "Towards evaluating the robustness of neural networks". In: *2017 ieee symposium on security and privacy (sp)*. IEEE. 2017, pp. 39–57.

[85] Nicolas Papernot et al. "Technical report on the cleverhans v2. 1.0 adversarial examples library". In: *arXiv preprint arXiv:1610.00768* (2016).

[86] Jonas Rauber, Wieland Brendel, and Matthias Bethge. "Foolbox: A python toolbox to benchmark the robustness of machine learning models". In: *arXiv preprint arXiv:1707.04131* (2017).

[87] Maria-Irina Nicolae et al. "Adversarial Robustness Toolbox v1. 0.0". In: *arXiv preprint arXiv:1807.01069* (2018).

[88] Gavin Weiguang Ding, Luyu Wang, and Xiaomeng Jin. "AdverTorch v0. 1: An adversarial robustness toolbox based on pytorch". In: *arXiv preprint arXiv:1902.07623* (2019).

[89] Arash Habibi Lashkari et al. *CICFlowMeter*. https://www.unb.ca/cic/research/applications.html. 2017.

[90] Arash Habibi Lashkari. et al. "Characterization of Tor Traffic using Time based Features". In: *Proceedings of the 3rd International Conference on Information Systems Security and Privacy - ICISSP,* INSTICC. SciTePress, 2017, pp. 253–262. ISBN: 978-989-758-209-7. DOI: 10.5220/0006105602530262.

[91] Gerard Draper-Gil. et al. "Characterization of Encrypted and VPN Traffic using Time-related Features". In: *Proceedings of the 2nd International Conference on Information Systems Security and Privacy - ICISSP,* INSTICC. SciTePress, 2016, pp. 407–414. ISBN: 978-989-758-167-0. DOI: 10.5220/0005740704070414.

[92] Iqbal H Sarker et al. "Intrudtree: a machine learning based cyber security intrusion detection model". In: *Symmetry* 12.5 (2020), p. 754.

[93] Omar Almomani. "A feature selection model for network intrusion detection system based on PSO, GWO, FFA and GA algorithms". In: *Symmetry* 12.6 (2020), p. 1046.

[94] Nicolas Papernot et al. "Distillation as a defense to adversarial perturbations against deep neural networks". In: *2016 IEEE symposium on security and privacy (SP)*. IEEE. 2016, pp. 582–597.

[95] Matthew J Page et al. "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews". In: *BMJ* 372 (2021). DOI: 10.1136/bmj.n71. eprint: https://www.bmj.com/content/372/bmj.n71.full.pdf. URL: https://www.bmj.com/content/372/bmj.n71.

[96] Battista Biggio et al. "Evasion attacks against machine learning at test time". In: *Joint European conference on machine learning and knowledge discovery in databases*. Springer. 2013, pp. 387–402.

[97] Nicolas Papernot et al. "Crafting adversarial input sequences for recurrent neural networks". In: *MILCOM 2016-2016 IEEE Military Communications Conference*. IEEE. 2016, pp. 49–54.

[98] N. Papernot et al. "The Limitations of Deep Learning in Adversarial Settings". In: *2016 IEEE European Symposium on Security and Privacy (EuroS P)*. 2016, pp. 372–387.

[99] Robin Jia and Percy Liang. "Adversarial Examples for Evaluating Reading Comprehension Systems". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2017, pp. 2021–2031.

[100] Zhengli Zhao, Dheeru Dua, and Sameer Singh. "Generating Natural Adversarial Examples". In: *International Conference on Learning Representations*. 2018, p. 0.

[101] Yen-Chen Lin et al. "Tactics of adversarial attack on deep reinforcement learning agents". In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2017, pp. 3756–3762.

[102] Maria Rigaki. *Adversarial deep learning against intrusion detection classifiers*. 2017.

[103] Weiwei Hu and Ying Tan. "Black-box attacks against RNN based malware detection algorithms". In: *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*. 2018, p. 0.

[104] Ivan Homoliak et al. "Improving Network Intrusion Detection Classifiers by Non-payload-Based Exploit-Independent Obfuscations: An Adversarial Approach". In: *EAI Endorsed Transactions on Security and Safety* 5.17 (2018).

[105] Ishai Rosenberg et al. "Generic black-box end-to-end attack against state of the art API call based malware classifiers". In: *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer. 2018, pp. 490–510.

[106] Zheng Wang. "Deep learning-based intrusion detection with adversaries". In: *IEEE Access* 6 (2018), pp. 38367–38384.

[107] Arkadiusz Warzyński and Grzegorz Kołaczek. "Intrusion detection systems vulnerability on adversarial examples". In: *2018 Innovations in Intelligent Systems and Applications (INISTA)*. IEEE. 2018, pp. 1–4.

[108] Zilong Lin, Yong Shi, and Zhi Xue. "Idsgan: Generative adversarial networks for attack generation against intrusion detection". In: *arXiv preprint arXiv:1809.02077* (2018).

[109] Kaichen Yang et al. "Adversarial examples against the deep learning based network intrusion detection systems". In: *MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM)*. IEEE. 2018, pp. 559–564.

[110] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. "One pixel attack for fooling deep neural networks". In: *IEEE Transactions on Evolutionary Computation* 23.5 (2019), pp. 828–841.

[111] Aditya Kuppa et al. "Black box attacks on deep anomaly detectors". In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*. 2019, pp. 1–10.

[112] Olakunle Ibitoye, Omair Shafiq, and Ashraf Matrawy. "Analyzing adversarial attacks against deep learning for intrusion detection in IoT networks". In: *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE. 2019, pp. 1–6.

[113] Muhammad Usama et al. "Generative adversarial networks for launching and thwarting adversarial attacks on network intrusion detection systems". In: *2019 15th international wireless communications & mobile computing conference (IWCMC)*. IEEE. 2019, pp. 78–83.

[114] Weiqing Huang et al. "Adversarial Attack against LSTM-based DDoS Intrusion Detection System". In: *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE. 2020, pp. 686–693.

[115] Yuji Ogawa, Tomotaka Kimura, and Jun Cheng. "Vulnerability Assessment for Machine Learning Based Network Anomaly Detection System". In: *2020 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan)*. IEEE. 2020, pp. 1–2.

[116] Jiming Chen et al. "Generating Adversarial Examples against Machine Learning based Intrusion Detector in Industrial Control Systems". In: *IEEE Transactions on Dependable and Secure Computing* (2020).

[117] Luca Demetrio et al. "Functionality-preserving black-box optimization of adversarial windows malware". In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 3469–3478.

[118] Raphael Labaca-Castro et al. "Universal Adversarial Perturbations for Malware". In: *arXiv preprint arXiv:2102.06747* (2021).

[119] Feiyang Cai, Jiani Li, and Xenofon Koutsoukos. "Detecting adversarial examples in learning-enabled cyber-physical systems using variational autoencoder for regression". In: *2020 IEEE Security and Privacy Workshops (SPW)*. IEEE. 2020, pp. 208–214.

[120] Jiangnan Li et al. "Conaml: Constrained adversarial machine learning for cyber-physical systems". In: *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*. 2021, pp. 52–66.

[121] Florian Tramèr et al. "Ensemble adversarial training: Attacks and defenses". In: *6th International Conference on Learning Representations, ICLR 2018*. 2018, p. 0.

[122] Alexey Kurakin et al. "Adversarial attacks and defences competition". In: *The NIPS'17 Competition: Building Intelligent Systems*. Springer, 2018, pp. 195–231.

[123] Mahmood Sharif et al. "A General Framework for Adversarial Examples with Objectives". In: *ACM Trans. Priv. Secur.* 22.3 (June 2019). ISSN:

2471-2566. DOI: 10.1145/3317611. URL: https://doi.org/10.1145/3317611.

[124] Justin Gilmer et al. "Motivating the rules of the game for adversarial example research". In: *arXiv preprint arXiv:1807.06732* (2018).

[125] Giovanni Apruzzese et al. "Modeling Realistic Adversarial Attacks against Network Intrusion Detection Systems". In: *Digital Threats: Research and Practice* (2021).

[126] Seyed-Mohsen Moosavi-Dezfooli et al. "Universal adversarial perturbations". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1765–1773.

[127] Alesia Chernikova and Alina Oprea. "Fence: Feasible evasion attacks on neural networks in constrained environments". In: *ACM Transactions on Privacy and Security* 25.4 (2022), pp. 1–34.

[128] Giovanni Apruzzese, Michele Colajanni, and Mirco Marchetti. "Evaluating the effectiveness of adversarial attacks against botnet detectors". In: *2019 IEEE 18th International Symposium on Network Computing and Applications (NCA)*. IEEE. 2019, pp. 1–8.

[129] Mohammad J Hashemi, Greg Cusack, and Eric Keller. "Towards evaluation of nidss in adversarial setting". In: *Proceedings of the 3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks*. 2019, pp. 14–21.

[130] Pavlos Papadopoulos et al. "Launching Adversarial Attacks against Network Intrusion Detection Systems for IoT". In: *Journal of Cybersecurity and Privacy* 1.2 (2021), pp. 252–273.

[131] Eirini Anthi et al. "Adversarial attacks on machine learning cybersecurity defences in industrial control systems". In: *Journal of Information Security and Applications* 58 (2021), p. 102717.

[132] Dongqi Han et al. "Evaluating and Improving Adversarial Robustness of Machine Learning-Based Network Intrusion Detectors". In: *IEEE Journal on Selected Areas in Communications* (2021).

[133] Phan The Duy et al. "DIGFuPAS: Deceive IDS with GAN and Function-Preserving on Adversarial Samples in SDN-enabled networks". In: *Computers & Security* (2021), p. 102367.

[134] Elie Alhajjar, Paul Maxwell, and Nathaniel Bastian. "Adversarial machine learning in network intrusion detection systems". In: *Expert Systems with Applications* 186 (2021), p. 115782.

[135] Dongqi Han et al. "Practical traffic-space adversarial attacks on learning-based nidss". In: ().

[136] Jianyu Wang et al. "Def-IDS: An Ensemble Defense Mechanism Against Adversarial Attacks for Deep Learning-based Network Intrusion Detection". In: *2021 International Conference on Computer Communications and Networks (ICCCN)*. IEEE. 2021, pp. 1–9.

[137] Hassan Ali et al. "Analyzing the Robustness of Fake-news Detectors under Black-box Adversarial Attacks". In: *IEEE Access* (2021).

[138] Jules Chenou, George Hsieh, and Tonya Fields. "Radial Basis Function Network: Its Robustness and Ability to Mitigate Adversarial Examples". In: *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE. 2019, pp. 102–106.

[139] Wenqi Wei et al. "Adversarial examples in deep learning: Characterization and divergence". In: *arXiv preprint arXiv:1807.00051* (2018).

[140] Florian Tramèr et al. "Stealing machine learning models via prediction apis". In: *25th {USENIX} Security Symposium ({USENIX} Security 16)*. 2016, pp. 601–618.

[141] Timothy P Lillicrap et al. "Random synaptic feedback weights support error backpropagation for deep learning". In: *Nature communications* 7.1 (2016), pp. 1–10.

[142] Abigail Graese, Andras Rozsa, and Terrance E. Boult. "Assessing Threat of Adversarial Examples on Deep Neural Networks". In: *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2016, pp. 69–74. DOI: 10.1109/ICMLA.2016.0020.

[143] Kathrin Grosse et al. “On the (statistical) detection of adversarial examples”. In: *arXiv preprint arXiv:1702.06280* (2017).

[144] Jan Hendrik Metzen et al. “On detecting adversarial perturbations”. In: *arXiv preprint arXiv:1702.04267* (2017).

[145] Reuben Feinman et al. “Detecting adversarial samples from artifacts”. In: *arXiv preprint arXiv:1703.00410* (2017).

[146] Dongyu Meng and Hao Chen. “Magnet: a two-pronged defense against adversarial examples”. In: *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*. 2017, pp. 135–147.

[147] Weilin Xu, David Evans, and Yanjun Qi. “Feature squeezing: Detecting adversarial examples in deep neural networks”. In: *arXiv preprint arXiv:1704.01155* (2017).

[148] Hyeungill Lee, Sungyeob Han, and Jungwoo Lee. “Generative adversarial trainer: Defense to adversarial perturbations with gan”. In: *arXiv preprint arXiv:1705.03387* (2017).

[149] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. “Defense-gan: Protecting classifiers against adversarial attacks using generative models”. In: *arXiv preprint arXiv:1805.06605* (2018).

[150] Guneet S Dhillon et al. “Stochastic activation pruning for robust adversarial defense”. In: *arXiv preprint arXiv:1803.01442* (2018).

[151] Yuchi Tian et al. “Deeptest: Automated testing of deep-neural-network-driven autonomous cars”. In: *Proceedings of the 40th international conference on software engineering*. 2018, pp. 303–314.

[152] Mengshi Zhang et al. “DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems”. In: *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE. 2018, pp. 132–142.

[153] Siyue Wang et al. “Defensive dropout for hardening deep neural networks under adversarial attacks”. In: *Proceedings of the International Conference on Computer-Aided Design*. 2018, pp. 1–8.

[154] Mohammed Amer and Tomás Maul. "Weight Map Layer for Noise and Adversarial Attack Robustness". In: *arXiv preprint arXiv:1905.00568* (2019).

[155] Ishai Rosenberg et al. "Defense methods against adversarial examples for recurrent neural networks". In: *arXiv preprint arXiv:1901.09963* (2019).

[156] Giovanni Apruzzese et al. "Deep reinforcement adversarial learning against botnet evasion attacks". In: *IEEE Transactions on Network and Service Management* 17.4 (2020), pp. 1975–1987.

[157] Giovanni Apruzzese et al. "Addressing adversarial attacks against security systems based on machine learning". In: *2019 11th International Conference on Cyber Conflict (CyCon)*. Vol. 900. IEEE. 2019, pp. 1–18.

[158] Yan Zhou, Murat Kantarcioglu, and Bowei Xi. "A survey of game theoretic approach for adversarial machine learning". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 9.3 (2019), e1259.

[159] Giovanni Apruzzese et al. "Hardening random forest cyber detectors against adversarial attacks". In: *IEEE Transactions on Emerging Topics in Computational Intelligence* 4.4 (2020), pp. 427–439.

[160] Chen Zhang et al. "A robust generative classifier against transfer attacks based on variational auto-encoders". In: *Information Sciences* 550 (2021), pp. 57–70.

[161] Ning Wang et al. "MANDA: On Adversarial Example Detection for Network Intrusion Detection System". In: *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE. 2021, pp. 1–10.

[162] Dawn Song et al. "Physical adversarial examples for object detectors". In: *12th {USENIX} Workshop on Offensive Technologies ({WOOT} 18)*. 2018, p. 0.

[163] Yarin Gal and Zoubin Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning". In: *international conference on machine learning*. PMLR. 2016, pp. 1050–1059.

[164] Nicholas Carlini and David Wagner. "Adversarial examples are not easily detected: Bypassing ten detection methods". In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. 2017, pp. 3–14.

[165] Kexin Pei et al. "Deepxplore: Automated whitebox testing of deep learning systems". In: *proceedings of the 26th Symposium on Operating Systems Principles*. 2017, pp. 1–18.

[166] Battista Biggio, Giorgio Fumera, and Fabio Roli. "Adversarial pattern classification using multiple classifiers and randomisation". In: *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer. 2008, pp. 500–509.

[167] Nicholas Carlini. *A complete list of all (arxiv) adversarial example papers*. https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html. 2019.

[168] Andreas Sfakianakis et al. *ENISA Threat Landscape Report 2018: 15 Top Cyberthreats and Trends*. https://www.enisa.europa.eu/publications/enisa-threat-landscape-report-2018. 2019.

[169] Anish Athalye, Nicholas Carlini, and David Wagner. "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples". In: *International conference on machine learning*. PMLR. 2018, pp. 274–283.

[170] Tsung-Yi Lin et al. "Microsoft COCO: Common Objects in Context". In: *Computer Vision – ECCV 2014*. Ed. by David Fleet et al. Cham: Springer International Publishing, 2014, pp. 740–755. ISBN: 978-3-319-10602-1.

[171] Cameron Buckner. "Understanding adversarial examples requires a theory of artefacts for deep learning". In: *Nature Machine Intelligence* 2.12 (2020), pp. 731–736. ISSN: 2522-5839. DOI: 10.1038/s42256-020-00266-y. URL: https://doi.org/10.1038/s42256-020-00266-y.

[172] Florian Tramèr et al. *The Space of Transferable Adversarial Examples*. 2017. arXiv: 1704.03453 [stat.ML].

[173] Eduardo K Viegas, Altair O Santin, and Luiz S Oliveira. "Toward a reliable anomaly-based intrusion detection in real-world environments". In: *Computer Networks* 127 (2017), pp. 200–216.

[174] Ayyaz Ul Haq Qureshi et al. "An Adversarial Approach for Intrusion Detection Systems Using Jacobian Saliency Map Attacks (JSMA) Algorithm". In: *Computers* 9.3 (2020), p. 58.

[175] Tarfa Hamed, Rozita Dara, and Stefan C Kremer. "Network intrusion detection system based on recursive feature addition and bigram technique". In: *computers & security* 73 (2018), pp. 137–155.

[176] Gholamreza Farahani. "Feature selection based on cross-correlation for the intrusion detection system". In: *Security and Communication Networks* 2020 (2020).

[177] Phil Legg, Jim Smith, and Alexander Downing. "Visual analytics for collaborative human-machine confidence in human-centric active learning tasks". In: *Human-centric Computing and Information Sciences* 9.1 (Feb. 2019), p. 5.

[178] Seunghoon Yoo et al. "Hyperion: A Visual Analytics Tool for an Intrusion Detection and Prevention System". In: *IEEE Access* 8 (2020), pp. 133865–133881.

[179] Hervé Abdi and Lynne J. Williams. "Principal component analysis". In: *Wiley interdisciplinary reviews. Computational statistics* 2.4 (2010), pp. 433–459.

[180] Laurens van der Maaten et al. "Visualizing non-metric similarities in multiple maps". In: *Machine learning* 87.1 (2012), pp. 33–55.

[181] Leland McInnes et al. "UMAP: Uniform Manifold Approximation and Projection". English. In: *Journal of open source software* 3.29 (2018), p. 861.

[182] Serpil Ustebay, Zeynep Turgut, and Muhammed A. Aydin. "Intrusion Detection System with Recursive Feature Elimination by Using Random Forest and Deep Learning Classifier". In: IEEE, 2018, pp. 71–76.

[183] Sydney M Kasongo and Yanxia Sun. "Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset". In: *Journal of Big Data* 7.1 (2020), pp. 1–20.

[184] David Sculley et al. "Hidden technical debt in machine learning systems". In: *Advances in neural information processing systems*. 2015, pp. 2503–2511.

[185] Zilong Lin, Yong Shi, and Zhi Xue. "Idsgan: Generative adversarial networks for attack generation against intrusion detection". In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2022, pp. 79–91.

[186] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples". In: *arXiv preprint arXiv:1605.07277* (2016).

[187] Dingcheng Yang, Zihao Xiao, and Wenjian Yu. "Boosting the Adversarial Transferability of Surrogate Model with Dark Knowledge". In: *arXiv preprint arXiv:2206.08316* (2022).

[188] Jun Zhang et al. "Internet traffic classification by aggregating correlated naive bayes predictions". In: *IEEE transactions on information forensics and security* 8.1 (2012), pp. 5–15.

[189] Ryan Sheatsley et al. "On the robustness of domain constraints". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2021, pp. 495–515.

[190] Giovanni Apruzzese et al. "The Role of Machine Learning in Cybersecurity". In: *Digital Threats: Research and Practice* (2022).

[191] Report SimilarWeb. *Top Websites Ranking"*. https://www.similarweb.com/top-websites/. 2022.

[192] Ángel Luis Perales Gómez et al. "Crafting adversarial samples for anomaly detectors in industrial control systems". In: *Procedia Computer Science* 184 (2021), pp. 573–580.

[193] Paulo Angelo Alves Resende and André Costa Drummond. "A survey of random forest based methods for intrusion detection systems". In: *ACM Computing Surveys (CSUR)* 51.3 (2018), pp. 1–36.

[194] Sharon Qian et al. "Robustness from Simple Classifiers". In: *arXiv preprint arXiv:2002.09422* (2020).

[195] François Chollet et al. *Keras*. https://github.com/fchollet/keras. 2015.

[196] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[197] Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, Ali A Ghorbani, et al. "Characterization of tor traffic using time based features." In: 2017.

[198] R.J. Hofstede et al. "Ethernet Flow Monitoring with IPFIX". English. In: *TERENA Networking Conference 2011*. TERENA Networking Conference 2011 ; Conference date: 16-05-2011 Through 19-05-2011. Trans-European Research and Education Networking Association, May 2011, p. 23.

[199] Rodolfo M Pereira, Yandre MG Costa, and Carlos N Silla Jr. "Toward hierarchical classification of imbalanced data using random resampling algorithms". In: *Information Sciences* 578 (2021), pp. 344–363.

[200] Benedikt Langenberg et al. "A tutorial on using the paired t test for power calculations in repeated measures ANOVA with interactions". In: *Behavior Research Methods* (2022), pp. 1–18.

[201] Nicolas Papernot et al. "Practical black-box attacks against machine learning". In: *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. 2017, pp. 506–519.

[202] Nicolas Papernot et al. "The limitations of deep learning in adversarial settings". In: *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE. 2016, pp. 372–387.

[203] Ryan Sheatsley et al. "Adversarial examples for network intrusion detection systems". In: *Journal of Computer Security* Preprint (2022), pp. 1–26.

[204] Aleksander Madry et al. "Towards Deep Learning Models Resistant to Adversarial Attacks". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=rJzIBfZAb.

[205] Guillaume Jeanneret, Juan C Pérez, and Pablo Arbelaez. "A Hierarchical Assessment of Adversarial Severity". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 61–70.

[206] Ismail Alkhouri, George Atia, and Wasfy Mikhael. "Fooling the Big Picture in Classification Tasks". In: *Circuits, Systems, and Signal Processing* 42.4 (2023), pp. 2385–2415.

[207] Fábio M Miranda, Niklas Köhnecke, and Bernhard Y Renard. "Hiclass: a python library for local hierarchical classification compatible with scikit-learn". In: *Journal of Machine Learning Research* 24.29 (2023), pp. 1–17.

[208] Daphne Koller and Mehran Sahami. "Hierarchically Classifying Documents Using Very Few Words". In: *Proceedings of the Fourteenth International Conference on Machine Learning*. 1997, pp. 170–178.

[209] Fionn Murtagh and Pierre Legendre. "Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion?" In: *Journal of classification* 31.3 (2014), pp. 274–295.

[210] Svetlana Kiritchenko et al. "Learning and evaluation in the presence of class hierarchies: Application to text categorization". In: *Conference of the Canadian Society for Computational Studies of Intelligence*. Springer. 2006, pp. 395–406.

[211] Avery Ma et al. "Improving Hierarchical Adversarial Robustness of Deep Neural Networks". In: *arXiv preprint arXiv:2102.09012* (2021).

[212] Erxue Min et al. "Su-ids: A semi-supervised and unsupervised framework for network intrusion detection". In: *International conference on cloud computing and security*. Springer. 2018, pp. 322–334.

[213] Hossein Hosseini et al. "Blocking transferability of adversarial examples in black-box learning systems". In: *arXiv preprint arXiv:1703.04318* (2017).

[214] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning multiple layers of features from tiny images". In: (2009).

[215] Rodolfo M. Pereira, Yandre M.G. Costa, and Carlos N. Silla Jr. "Toward hierarchical classification of imbalanced data using random resampling algorithms". In: *Information Sciences* 578 (2021), pp. 344–363. ISSN: 0020-0255.

DOI: https://doi.org/10.1016/j.ins.2021.07.033. URL: https://www.sciencedirect.com/science/article/pii/S0020025521007234.

[216] Ishai Rosenberg et al. "Adversarial machine learning attacks and defense methods in the cyber security domain". In: *ACM Computing Surveys (CSUR)* 54.5 (2021), pp. 1–36.

[217] Puyudi Yang et al. "Greedy attack and gumbel attack: Generating adversarial examples for discrete data". In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 1613–1648.

[218] Liwei Song, Reza Shokri, and Prateek Mittal. "Privacy Risks of Securing Machine Learning Models against Adversarial Examples". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. CCS '19. London, United Kingdom: Association for Computing Machinery, 2019, pp. 241–257. ISBN: 9781450367479. DOI: 10.1145/3319535.3354211. URL: https://doi.org/10.1145/3319535.3354211.

[219] Emilio Rafael Balda, Arash Behboodi, and Rudolf Mathar. "Perturbation analysis of learning algorithms: Generation of adversarial examples from classification to regression". In: *IEEE Transactions on Signal Processing* 67.23 (2019), pp. 6078–6091.

[220] James Reason. "Human error: models and management". In: *Bmj* 320.7237 (2000), pp. 768–770.

[221] Dule Shu et al. "Generative adversarial attacks against intrusion detection systems using active learning". In: *Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning*. 2020, pp. 1–6.