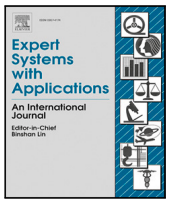
Expert Systems With Applications 266 (2025) 126044

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Detection and prevention of evasion attacks on machine learning models

Raja Muthalagu *, Jasmita Malik, Pranav M. Pawar

Department of Computer Science, BITS Pilani, Dubai, United Arab Emirates

ARTICLE INFO

Keywords:
Adversarial machine learning
Cybersecurity
Evasion attacks
Secure coding

ABSTRACT

With the increasing use of machine learning models in critical applications such as image classification, natural language processing, and cybersecurity, there is a growing concern about the vulnerability of these models to adversarial attacks. Evasion attacks, in particular, pose a significant threat by manipulating input data to mislead the model's predictions. This paper presents an overview of evasion attacks on machine learning models, its variants and conducts an adaptive white-box evasion attack to highlight how defense measures can be superseded with stronger evasion attack algorithms. It first discusses the different types of evasion attack algorithms, including Fast Gradient Sign Method (FGSM), Projected Gradient Descent (PGD), and Carlini–Wagner, highlighting their impact on model performance and security. It then reviews various detection and mitigation techniques which aim to identify adversarial examples and improve model robustness. Finally, the paper proposes effective mitigation techniques against evasion attacks and recommends a machine learning based cybersecurity architecture workflow that can be practically applied by organizations in real-world settings. Overall, this paper provides a comprehensive overview of evasion attacks on machine learning models and highlights the current state of research in defending against them.

1. Introduction

Artificial intelligence (AI) systems have rapidly transformed everyday life, with applications appearing across various domains. However, alongside their benefits, AI and machine learning (ML) systems face growing risks of cyberattacks, particularly adversarial machine learning (AML) attacks. These attacks can compromise the integrity and reliability of ML systems, leading to failures with serious consequences. A key focus of this research is evasion attacks, a prominent AML attack type where adversaries manipulate input data to cause misclassification without altering the training data or the model itself. Real-world examples of adversarial perturbations highlight these risks. Subtle modifications in images have caused autonomous vehicles to misidentify stop signs, critical objects to disappear, and high-security facial recognition systems to fail (Jing et al., 2021). In medical settings, ML models risk exposing sensitive patient data (Bak, Madai, Fritzsche, Mayrhofer, & McLennan, 2022), while in cybersecurity, tampering with ML algorithms' training data has compromised fraud detection and anti-malware systems (Price II, 2019). Addressing these vulnerabilities is essential to ensure AI systems remain secure, robust, and trustworthy.

1.1. Evasion attacks

Among AML attack types, evasion attacks are particularly concerning due to their prevalence and potential impact. These attacks occur during the post-training inference stage, where adversaries manipulate input samples to cause misclassification in trained classifiers. Unlike poisoning attacks, which alter training data, evasion attacks focus on creating adversarial perturbations in inputs to mislead the model. For instance, modifying a few pixels in an image can trick image recognition systems into making incorrect classifications (Pitropakis, Panaousis, Giannetsos, Anastasiadis, & Loukas, 2019). These attacks target applications such as autonomous vehicles, voice assistants, and cybersecurity tools like anti-malware and intrusion detection systems, where errors can lead to significant consequences. Evasion attacks occur during the inference stage in a model, where the adversary aims to introduce adversarial noise into an input and generate an adversarial sample, which when fed to a trained target model, causes predictable errors at the model's output as shown in Fig. 1. Evasion attacks can be either targeted (i.e., the noise produces a specific error at the output) or untargetted (i.e., the noise produces an error at the output, but the type of error is irrelevant to the adversary).

1.2. Motivation

Evasion attacks pose a significant threat to the reliability of AI systems, as demonstrated by real-world incidents. For instance, evasion attacks on anti-malware systems have enabled ransomware attacks

* Corresponding author.
E-mail addresses: raja.m@dubai.bits-pilani.ac.in (R. Muthalagu), p20210904@dubai.bits-pilani.ac.in (J. Malik), pranav@dubai.bits-pilani.ac.in (P.M. Pawar).

https://doi.org/10.1016/j.eswa.2024.126044
Received 29 June 2024; Received in revised form 25 November 2024; Accepted 1 December 2024
Available online 7 December 2024

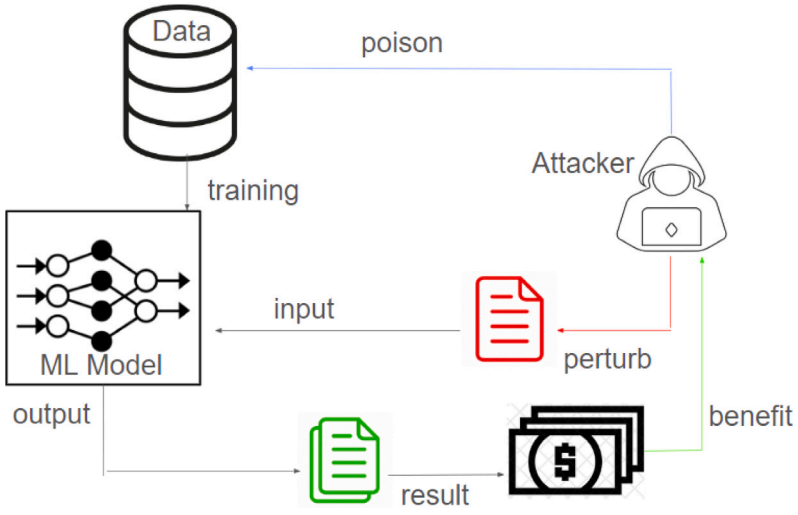


Fig. 1. Evasion attack process.

by misclassifying malware strains. Similarly, the global adoption of autonomous vehicles and ML-based applications necessitates robust protections against these threats. This research seeks to strengthen defenses and mitigate risks associated with evasion attacks, contributing to the trustworthiness of ML systems in critical domains such as medical diagnostics and cybersecurity tools.

1.3. Research gaps

Despite significant advancements, gaps persist in understanding and mitigating evasion attacks. Key challenges include limited insights into attack mechanisms, inadequate evaluation frameworks, and defense controls that lack generalizability across diverse scenarios. Existing methods often overestimate model robustness due to incomplete testing frameworks, while defenses effective in one context may fail against different attack types. Addressing these gaps requires a multidisciplinary approach combining AML and cybersecurity insights. This research aims to answer the following questions:

- **RQ1**: What are evasion attacks and its variants?
- **RQ2**: What tools are available to test ML models against evasion attacks?
- **RQ3**: What defense controls are effective against evasion attacks?
- **RQ4**: How can ML models be securely deployed?

1.4. Specific contributions

The field of adversarial machine learning (AML) has extensively explored evasion attacks and defense mechanisms, yet significant gaps remain in addressing adaptive and real-world threats. Many prior studies have focused on analyzing specific evasion attack algorithms and proposing point-based defense mechanisms. For instance, recent works such as Zhao, Li, Wang, and Liu (2023) highlighted advancements in adversarial training for improving model robustness but acknowledged challenges in generalizing across different datasets and attack types. Similarly, Wang and Chen (2023) explored image preprocessing techniques, including JPEG Compression, and demonstrated their limitations when faced with stronger adaptive attacks like PGD. These studies underscore the necessity of robust, unified frameworks for addressing adversarial vulnerabilities comprehensively. This research builds on these insights and makes the following contributions to the field:

- **Comprehensive Analysis of Adaptive Evasion Attacks and Limitations of Existing Defenses**: This study systematically evaluates the limitations of widely adopted defenses, such as JPEG Compression, against advanced attack scenarios. Unlike prior research focusing solely on simpler attack algorithms like FGSM, this work demonstrates how iterative attacks such as Projected Gradient Descent (PGD) can circumvent these defenses under adaptive conditions.
- **Innovative Integration of Defense Techniques**: By combining techniques such as Spatial Smoothing and Feature Squeezing, this study showcases the cumulative impact of layered defenses in mitigating adversarial evasion attacks. This approach moves beyond single-point defenses, providing actionable strategies for practitioners to enhance the ML model's robustness against evasion attacks effectively.
- **Empirical Validation Using Real-World Models and Datasets**: Experiments were conducted on real-world architectures, such as ResNet50V2, using large-scale datasets like ImageNet. This ensures practical applicability and addresses limitations in prior studies that relied on theoretical proofs or small synthetic datasets.
- **Proposal of a Comprehensive ML Cybersecurity Architecture Workflow**: A novel lifecycle-centric architecture is introduced, integrating pre-processing, in-training, and post-production strategies to safeguard machine learning models against evolving attacks. This workflow provides organizations with scalable and adaptable solutions to embed security measures throughout the ML lifecycle.
- **Systematic Evaluation of Defense Robustness via Adaptive Evasion Attacks Testing**: The study emphasizes adaptive evasion attacks testing, a critical but underexplored dimension in AML research. By rigorously evaluating defenses under progressively stronger evasion attack scenarios, it highlights potential vulnerabilities and ensures the proposed measures remain effective against sophisticated adversaries.

By addressing these critical gaps, this research establishes a robust foundation for advancing AML and mitigating evasion attacks in real-world applications. Table 1 compares this work with recent studies, highlighting its unique contributions.

1.5. Paper organization

This paper is structured into several sections to present a coherent flow of ideas and findings. The first introduction section introduces

Table 1
Comparison with Recent Research.

Aspect	Recent studies	This work
Attack algorithms	Most studies focused on individual attack types (e.g., FGSM, Basic Iterative Method (BIM), transfer attacks). Some works, such as PAD and IoT-specific studies, used domain-specific perturbations (Apruzzese, Andreolini, Marchetti, Venturi, & Colajanni, 2020; Wang & Chen, 2023; Xu, Li, Li, & Xu, 2024; Yang, Maina, Cheng, & Lee, 2024).	This research systematically explores multiple attack types (FGSM, PGD) while demonstrating adaptive and targeted evasion attacks, providing deeper insights into algorithm behavior.
Defense measures	Defensive strategies varied, from adversarial training, JPEG compression, Adversarial training to heuristic methods, but were often context-specific and lacked validation against adaptive attacks (Apruzzese et al., 2020; Praveena, Madhumitha, Menakadevi, & Lalith Akkash, 2023; Yang et al., 2024; Zhao et al., 2023).	This work introduces layered defense mechanisms like Spatial Smoothing, Feature Squeezing, Adversarial Training and combinations thereof, demonstrating their efficacy across scenarios and against adaptive attacks.
Evaluation scope	Prior studies were limited to specific datasets (e.g., malware samples or IoT datasets) or single model types like IDS and speech recognition (Ayub, Johnson, Talbert, & Siraj, 2020; Xu et al., 2024; Yang et al., 2024)	Experiments extend to real-world models (e.g., ResNet50V2) and diverse datasets like ImageNet, ensuring broad applicability and scalability of defenses.
Adaptive attacks	Very few studies explored adaptive evasion attacks comprehensively. Many evaluated defenses only against static attack scenarios. (Li et al., 2024; Xu et al., 2024)	This research rigorously evaluates defenses under progressively stronger adaptive scenarios, demonstrating practical resilience to advanced adversaries.
Specific domains	Some studies discussed vulnerabilities and solutions tailored for IoT applications but did not generalize to broader domains (Praveena et al., 2023; Xu et al., 2024)	While not IoT-specific, this research introduces a generalized ML cybersecurity architecture, which can be adapted to IoT or other critical systems.
Proposed architecture	No comprehensive architecture frameworks combining ML and cybersecurity were proposed; defenses were piecemeal and lacked lifecycle integration (Apruzzese et al., 2020; Li et al., 2024)	Proposes a lifecycle-centric ML cybersecurity workflow, incorporating preprocessing, output post-processing, and post-production stages for robust and adaptable defenses.
Visual evidence	Limited use of visuals to illustrate attack and defense effectiveness; some examples of data impact were shown, but few studies emphasized the interpretability of defenses (Ayub et al., 2020; Yang et al., 2024)	Provides detailed visual comparisons of perturbed and defended data, aiding in interpretability and clarity for practitioners implementing the proposed strategies.

the research problem of mitigating evasion attacks and provides an overview of the study's objectives. The second section reviews relevant literature, highlighting key evasion attacks and algorithms that contextualize the research. The third section outlines the research methodology, detailing the tools used for conducting an adaptive evasion attack and the experimental set-up used. The fourth section presents the empirical findings of the experiment, including key simulation results and model specific defense controls. The fifth section discusses the security defense controls and proposes a secure mechanism of deploying a model in real-world environments within organizations. Finally, the conclusion summarizes the main findings of the research and provides future directions for research.

2. Related work

Evasion attacks refer to attacks wherein the adversary designs input that appears normal to humans but is incorrectly classified by ML models. In an evasion attack, a deliberately perturbed input, known as an "adversarial example", is supplied to the network. This perturbed data input seems and feels precisely like its unaltered copy to a human, but it confuses the ML model classifier. The adversary solves a constrained optimization problem to find a small input perturbation that causes a significant change in the loss function, resulting in a wrong output in case of image classification models.

2.1. Fundamental concepts of evasion attacks

ML models are generally susceptible to adversarial inputs, which are deliberately selected by adversaries to alter the network's output without being apparent to a human observer. Brown, Mané, Roy, Abadi, and Gilmer (2018) states that several optimization strategies, including L-BFGS, Fast Gradient Sign Method (FGSM), DeepFool, Projected Gradient Descent (PGD) and the more recently proposed Logit-space Projected Gradient Ascent (LS-PGA), can be used to generate these adversarial examples, which typically modify each pixel by a small amount. Other attack techniques aim to alter a specific patch of the image or a particular subset of pixels in the image (in the case of Jacobian-based saliency map algorithm Papernot et al., 2016). It has been demonstrated that in real-world situations, these adversarial examples can completely paralyze a ML application. Brown et al. (2018) designed an attack that did not try to gently change one item into another. Rather, they used a method that produced a highly salient image-independent patch for a neural network. They used a patch to fully replace a portion of the image in order to execute the attack. They first trained the patch over a range of images, applying random translation, scaling, and rotation to the patch in each image, and then used gradient descent for optimization. This allowed the patch to take on any shape. The classifier then reported the desired class when this patch is positioned anywhere in its range of vision. Due to the scenario independent nature of the patch, the authors could launch an evasion attack without having to know the lighting, camera angle, classifier type being targeted or even the other objects in the scene beforehand. In Fig. 2, an attack on VGG16 model was presented by the authors, employing a patch produced by ensemble techniques. The network reported class "banana" with 97% confidence when a picture of a tabletop with a banana and a notebook (top photograph) was passed through VGG16 (top plot). Later on, after using the patch, the network reported with 99% confidence, that the image was a toaster if a sticker directed at the class "toaster" is physically placed on the table (bottom plot).

This is a classic example of an evasion attack. The attack here is intriguing because when designing the attack, the adversary does not need to be aware of the image they are targeting. Once an adversarial patch has been created, it may be extensively shared online for use by more adversaries who can print and utilize it. Furthermore, the attack employs a large perturbation, thus the security strategies that are already in place which focus on protecting against smaller perturbations might not be strong enough to withstand larger perturbations like these. In fact, recent research has shown that even the most advanced adversarially trained models on MNIST are susceptible to

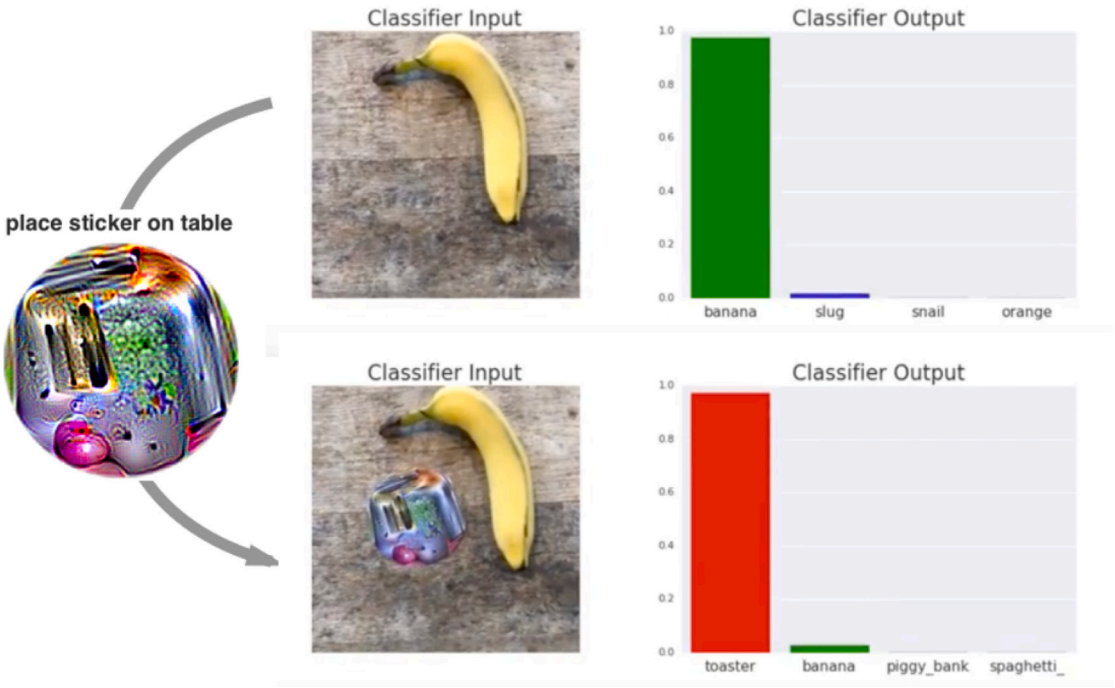


Fig. 2. Real world Evasion Attack on VGG16(Brown et al., 2018).

larger perturbations than those employed during training, either by applying large perturbations in the background or by looking for a nearby adversarial example using a different metric. Since many ML models do not require human validation for each input, adversaries do not need to be concerned if their attacks can be detected. Even if people are able to see these patches, it is possible that they will perceive them as works of art rather than comprehending the purpose behind them. This work demonstrated that protecting classifiers from minor perturbations alone is insufficient because classifiers can also be broken by large local perturbations.

While early attacks altered victim model inputs digitally, more recent developments produced images and real-world objects that caused misclassification in a variety of imaging scenarios (Athalye, Engstrom, Ilyas and Kwok, 2018; Eykholt et al., 2018; Wiyatno & Xu, 2019) described a technique for producing subtle perturbations that confuse visual object tracking systems when they are seen as posters in the real world. Their adversarial technique influences fields like autonomous vehicles, drone photography, and security monitoring and surveillance systems. The authors referred to their adversarial technique as Physical Adversarial Textures (PAT) which produced patterns to deceive object tracking systems. Emerging from a common optimization framework, these "PATterns" inspired a variety of adversarial examples with the ultimate goal of forcing the tracker to deviate from its intended target. The authors aimed to draw attention to the fact that objects with subtle colors can fool contemporary vision-based systems just by being in close proximity. The authors contend that, despite recent advancements, tracking systems that are solely reliant on vision are not resistant against adversaries. Case studies of similar evasion attacks are presented in further detail in Appendix A.

2.1.1. Evasion attack algorithms

Evasion attack algorithms are techniques used by adversaries to craft adversarial examples that exploit vulnerabilities in ML models. These adversarial examples are carefully crafted inputs designed to deceive the model into making incorrect predictions. Some common evasion attack algorithms include:

- **Fast Gradient Sign Method (FGSM):** FGSM is a fast and effective algorithm for generating adversarial examples. It works by taking the sign of the gradient of the loss function with respect to the input and using it to perturb the input data (Goodfellow et al., 2015).
- **Projected Gradient Descent (PGD):** PGD is an iterative variant of FGSM that performs multiple steps of gradient descent with small step sizes. This helps to craft more effective adversarial examples by iteratively refining the perturbation (Aleks et al., 2019).
- **Carlini–Wagner Attack:** The Carlini–Wagner attack is a powerful optimization-based attack that aims to find the minimum perturbation needed to fool the model while ensuring that the perturbed example remains close to the original input in terms of a specified distance metric Carlini and Wagner (2018).

These evasion attack algorithms demonstrated the vulnerability of ML models to adversarial examples and highlighted the importance of developing robust and secure models in the past. Table 2 further provides a list of various evasion attack algorithms identified in literature.

2.1.2. White-box evasion attacks

In addition to evasion attack algorithmic variants, evasion attacks can also be classified as white-box and black-box evasion attacks. White-box evasion attacks refer to scenarios where the adversary has full access to the details of the target model, including its architecture, parameters, and training data. This knowledge enables the adversary to craft adversarial examples more effectively compared to scenarios where only the input–output behavior of the model is known (black-box attacks). Papernot et al. (2016) demonstrated an example of a white-box evasion attack wherein the adversary gained access to the DNN's training data, functions, and algorithms, as well as knowledge of its architecture. It comprised of model layer number and type, neuron activation functions, and weight/bias matrices. The adversary was aware of the algorithm used to train the network, as well as its related loss function. This is the strongest type as the adversary can evaluate training data and simulate the deep neural network to find all its weaknesses. Carlini and Wagner (2017) as well demonstrated how in an evasion attack, the adversary has full access to the neural network, including its architecture and parameters and can use it to his or her advantage.

Table 2
List of evasion attacks.

Domain	Evasion attack algorithm	Description	Sources
Image and video classification	Adversarial texture	An attack for applying learned adversarial texture on videos.	Wiyatno and Xu (2019)
	Boundary attack/Decision-based attack	A potent black-box boundary attack that only requires the final class prediction.	Yamamura et al. (2022)
	Fast Gradient Sign Method (FGSM)	It operates by perturbing the input data in the direction that maximizes the model's loss, as determined by the gradient of the loss function with respect to the input.	Goodfellow, Shlens, and Szegedy (2015)
	Frame saliency attack	Designed to fool object identification models, notably in the context of autonomous driving.	Inkawhich, Inkawhich, Chen, and Li (2018)
	Geometric decision based attack	The Geometric decision-based approach is an adversarial approach that attacks ML models, namely classifiers, by exploiting decision boundary geometry.	Rahmati, Moosavi-Dezfooli, Frossard, and Dai (2020)
	Jacobian Saliency Map Attack (JSMA)	A collection of adversarial attack strategies for deceiving classification models, including deep neural networks for image classification tasks.	Wiyatno and Xu (2018)
	Projected Gradient Descent (PGD)	The Projected Gradient Descent attack is an iterative method in which, after each iteration, the perturbation is projected with a defined radius.	Aleks et al. (2019)
Speech recognition	Carlini and Wagner Automatic Speech Recognition (ASR) attack	Implements adversarial approach on a speech recognition model.	Carlini and Wagner (2018)
	Imperceptible ASR attack	Develops imperceptible audio adversarial instances by using the psychoacoustic principle of auditory masking.	Qin, Carlini, Cottrell, Goodfellow, and Xie (2019)
Tabular data	LowProFool attack	A method to produce imperceptible adversarial examples in the tabular domain.	Ballet et al. (2019)

2.1.3. Black-box evasion attacks

Black-box evasion attacks refer to scenarios where the adversary has limited or no access to the target model's internal details, such as its architecture, parameters, or training data. The adversary can only interact with the model by providing inputs and observing the corresponding outputs. Despite this limited knowledge, the adversary aims to craft adversarial examples that can fool the model. In Papernot et al. (2017), the authors assume the adversary has no knowledge of the deep neural network's structure or parameters and no access to the training dataset. The adversary's only capability is to monitor labels assigned by the deep neural network for specific inputs. Brendel, Rauber, and Bethge (2018) states that in real-world ML applications, attackers typically have no access to the architecture or training data, but can only examine the end outcome. This applies to security systems (e.g. face recognition), autonomous vehicles, and speech recognition systems like Alexa or Cortana. The authors demonstrate the Boundary Attack which is relevant to real-world ML algorithms as it just requires access to the model's final decision (e.g. class label or transcribed phrase) and does not rely on model information such as gradients or confidence scores as required in white-box evasion attacks.

With this the answer to **RQ1** describing evasion attacks, attack algorithms and its variants has been stated. In the next section, the researchers describe the methodology of the adaptive evasion attack experiment.

3. Methodology

This research utilized a mixed-methods approach to explore the impact of evasion attacks on ML models and available defense measures to mitigate them. As a first step, the researchers evaluated the available open-source adversarial ML testing tools that can be used to simulate an evasion attack and develop defense measures. Security tools that can be used by organizations to perform continuous assessment of vulnerabilities in their ML model environments and help in the detection and prevention of evasion attacks as shown in Table 3.

3.1. Tool review

Amongst all available testing tools, the researchers identified the IBM-owned Adversarial Robustness Toolbox (ART) to be the most comprehensive open-source tooling system that can be leveraged by organizations for testing the security posture of their ML applications. ART provides pre-packed libraries to test an ML model against vulnerabilities and provides a diverse set of modules to conduct evasion attacks and test defense countermeasures to improve the overall resilience of the model against evasion attacks.

ART is the most comprehensive open-source ML application security testing toolbox available at the time of writing this research. The main aim of ART is to provide tools to developers and researchers to enable them to evaluate, defend, certify and verify the robustness of ML models and applications against AML attacks including evasion attacks. ART aims to be general and supports all ML tasks such as classification, object detection, generation, encoding, certification, automated speech recognition and so on. ART supports commonly-used ML libraries and frameworks such as TensorFlow, Keras, PyTorch, MXNet, scikit-learn, XGBoost, LightGBM, CatBoost, GPy. ART can be customized to make it specific to a certain framework. ART supports all types of data such as images, tables, audio, video, etc. ART started with implementations of attacks and defenses and later created a combination of attack tools that a red and blue team would use in a practical real-world setting. A red team usually would play the part of an attacker and their goal would be to compromise the ML model while the blue team would work on defending it. The idea of ART is to have the state-of-the-art for a developer/security researcher to evaluate the various threat scenarios through which a real attacker would try to penetrate and compromise an ML model. ART provides evaluation tools for evasion, poisoning, extraction and inference attacks. ART also provides blue team specific defense tools. These tools help in defending the ML models from AML attacks. Some of the notable tools in ART are adversarial training examples which are one of the strongest defenses against evasion attacks. There are also other tools for evasion and poisoning attack detection. ART also has certification and validation tools for

Table 3
List of adversarial attack testing tools.

Tool name	Attacks covered	Tool limitations
AdvBox	Deep fakes; Data poisoning	Advbox generates adversarial examples to train a ML model. However, it does not contain any mitigatory measures for improving the resilience of models against evasion attacks.
Adversarial DNN playground	Jacobian Saliency Map Approach (JSMA), Fast Gradient Sign Method (FGSM)	Adversarial DNN Playground is a visualization software suite for adversarial sample generation. It only supports three attack algorithms and not Projected Gradient Descent (PGD).
AdverTorch	DeepFool, JSMA, Carlini–Wagner	AdverTorch is a Python toolbox for studying adversarial robustness. It does not support various ML libraries.
CleverHans	FGSM, PGD but not an entire group of evasion attack algorithms	CleverHans is an adversarial example library used for benchmarking, constructing attacks, and building defenses. However, it does not contain any mitigatory measures for improving the resilience of ML models as well.
Counterfit	Boundary Attack, Carlini–Wagner, DeepFool, PGD	Counterfit is a general automation layer used to evaluate the security of ML applications. However, it does not contain any mitigatory measures for improving the resilience of ML models.
DeepSec	FGSM, JSMA, PGD	No specific defenses provided against evasion attacks.
Foolbox	HopSkipJump, NewtonFool Attack	No specific defenses provided against evasion attacks.
Adversarial Robustness Toolbox (ART)	FGSM, JSMA, PGD, Carlini–Wagner, DeepFool and all other most common attack algorithms known	ART is the most comprehensive ML model security testing toolbox that provides various attack algorithms for Evasion, Poisoning and Privacy (Inference and Extraction based attacks). It also provides specific defense mechanisms against evasion attacks.
MIA	Shadow Model Attack	Membership inference attack is a type of a privacy attack. No specific defenses are provided in this tool against evasion attacks.
TextFool	Has no support for evasion attacks.	No specific defenses are provided against evasion attacks.

confirming the robustness of ML models. These tools are continuously being developed and published through ART's Github workspace. ART is hosted as a Python library by the Linux Foundation for AI in the Trusted AI workspace on Github.

ART has 6 main-modules which are referred to as attacks, defenses, estimators, evaluations, metrics and pre-processing as shown in Fig. 3. These modules will be used by the researchers in latter sections. A brief explanation of each main module is given below:

(1) **Attacks**: The attacks module has white-box and black-box attack tools for evaluating the ML model's robustness by simulating attacks. The attacks module is further divided into four sub-modules: (1) Evasion, (2) Poisoning, (3) Extraction and (4) Inference

(2) **Defenses**: The defenses module contains several defense techniques for defending against various AML attacks including evasion attacks. Some of these approaches include adversarial training, pre-processing and post-processing. Many defenses break with strong attacks therefore, it is crucial for a developer/ML engineer and a security expert in an organization to try and experiment with different types of defense approaches against AML attacks.

(3) **Estimators**: The Estimators module contains APIs for abstraction of the evaluated ML model. The estimators module has four sub-modules divided based on tasks: (1) Classification, (2) Object detection, (3) Poison mitigation and (4) Speech recognition.

(4) **Evaluations**: The Evaluations module contains high-level tools for evaluation of robustness and builds on attacks and estimators. The evaluations module has 1 sub-module i.e., the (1) Security curve, which is a tool meant for evaluating the overall attack surface of the application.

(5) **Metrics**: ART also has something referred to as the Metrics module which further has 3 sub-modules: (1) CLEVER, (2) Verification decision trees, (3) Gradient check. All these sub-modules help in calculating and quantifying the metrics of robustness within the model.

(6) **Pre-processing**: The pre-processing module connects with estimators and applies pre-processing techniques to make attacks stronger and apply larger modification to the input data. It makes the attacks more robust and helps the developer implement stronger defenses. The pre-processing module has 2 sub-modules: (1) Expectation over transformation and (2) Audio.

ART has a multitude of tools that can be used for replicating the four types of attacks: Evasion, Poisoning, Extraction and Inference. It also has attacks that are classified as white-box attacks or black-box attacks depending on the access required by the tester to the ML environment. With this the researchers confirm that ART is the most suitable tool to test ML models against evasion attacks and gain assurance of the effectiveness of the defensive controls implemented. This provides the answer to **RQ2**.

3.2. Experimental setup

In this section, the researchers have conducted a full-fledged evasion attack leveraging the open-source Adversarial Robustness Toolbox (ART) to evaluate the robustness and security of ML models against evasion attacks. For this experiment, a ResNet50V2 image classification model has been utilized. The researchers have developed algorithms for creating adversarial perturbations within the image classification model. The process followed for conducting the attack are described further. The ART toolbox has several tools available. The researchers used two specific tools within the toolbox. The first tool is of the "Evasion attack" sub-module within the "Attacks" module and the second tool is the "Classification" sub-module within the "Estimators" module. The attack being conducted in this research is an adaptive white-box evasion attack on ImageNet. Its a simplified illustrative evasion attack in which the researchers added a small imperceptible perturbation to change the classification of the image provided to the model. It is assumed that the researchers have full access to the model and can calculate the loss gradients. It is therefore considered as a white-box evasion attack. In a later step within the attack, defenses are applied and the researchers adapt the attack to a new complex attack scenario and compromise the model. Furthermore, the researchers recommend specific defenses that can be used against similar evasion attacks.

3.3. Adaptive white-box evasion attack against image classification model

The attack conducted in this research can be classified as an adaptive white-box evasion attack on an image classification model. The researchers add a calculated imperceptible perturbation to the original image to change the image's classification category. The original image provided to the model is of a Siamese cat. During the experiment, the researchers also calculate the loss gradients and the gradients in the image that maximize the classification. The researchers also

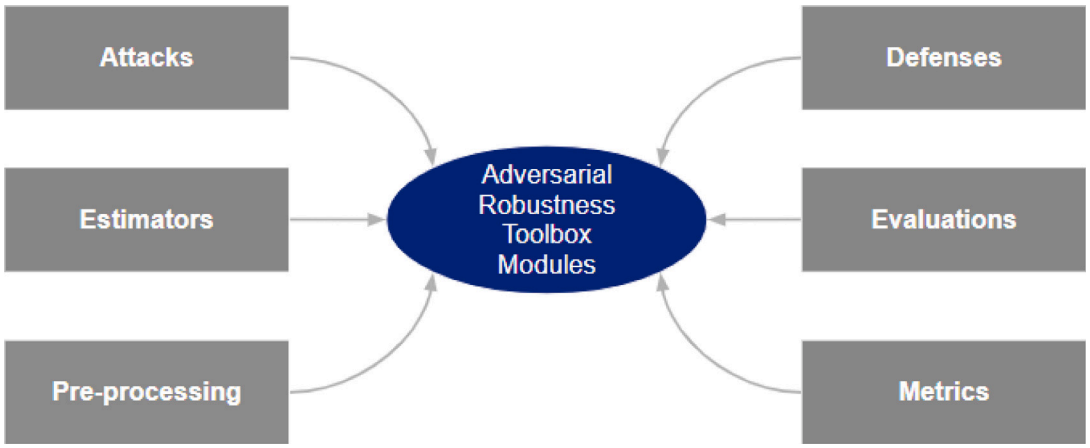


Fig. 3. ART modules.

demonstrate the adaptive part of the attack in which a defense is first implemented but later overrided by a stronger sophisticated attack. The researchers provide the algorithms used at each step of the experiment below.

Step 1: Algorithm 1: Initial Set-Up: As a first step, the researchers import from ART the relevant packages for the attack. The KerasClassifier tool is the abstraction ART tool imported for abstracting Keras ML models along with a pre-trained ResNet50V2 model. The ResNet50V2 will be attacked in this experiment. The researchers provide the model to the KerasClassifier tool within ART and provide it with clip values. Clip values are required to provide ranges of the input data that can be expected. Since, an image is being manipulated in this experiment, the pixel values i.e., 0–255 is provided as the clip values information to the KerasClassifier tool. Then an image of a 'Siamese cat' is loaded in the instance and the classifier method is used to make a prediction of the image.

Algorithm 1: Initial Set-Up

1: **Import** necessary modules
2: **from** art.estimators.classification **import** KerasClassifier
3: **from** tensorflow.keras.applications **import** ResNet50V2
4: **Load** pre-trained ResNet50V2 model
5: $model \leftarrow ResNet50V2(weights =' imagenet')$
6: **Create** an ART classifier for the model
7: $classifier \leftarrow KerasClassifier(model = model, clip\_values = (0, 255))$
8: **Load** the image
9: $image \leftarrow load\_image('siamese\_cat.jpg')$
10: $y\_prediction \leftarrow classifier.predict(original\_image)$

Output: The output of running Algorithm 1 provides the result as shown in Fig. 4. Here it is noticed that the model with an epsilon value as 2 correctly classifies the image as a Siamese cat with a confidence score of 99.9% as shown in Table 4. This is the initial set up of abstracting the ML model and verifying that the model is making the correct prediction.

Step 2: Algorithm 2: Evasion Attack using FGSM: Now, the researchers are extending this initial set up to an evasion attack. The initial algorithm remains the same with the abstraction of the model with KerasClassifier, however now the researchers import an attack method from the ART toolbox known as the Fast Gradient Sign Method. The Fast Gradient Sign Method is one of the most simple but surprisingly strong evasion attacks found in the real world. It basically calculates the loss gradient, takes the sign of it, multiples it and creates a perturbation that is given to the image and in many cases this is enough to result into a misclassification. The Fast Gradient Sign Method algorithm is implemented in Algorithm 2. One of the extensions in the below algorithm is that the researchers create an evasion attack by creating an instance of Fast Gradient Sign Method and keying in the two most important attributes to the Fast Gradient Sign Method i.e., the estimator that needs to be attacked, which in this case is the Kerasclassifier instance that has been created before as well as an epsilon (eps) value which is the argument to tell the program that the attacker will make a step of two pixel values from 0–255. An epsilon of 2 in this case means that the attacker can make a step of either +2 or −2 from the original pixel values. Then, the researchers define the attack with the method 'generate'. All evasion attacks in ART implement the method called 'generate' which accepts the argument 'x' and in this case it is the data that the researchers want to create adversarial examples of. So, the researchers provide the

Fig. 4. Original image of Siamese Cat.

Table 4
Initial baseline simulation results.

Image	Prediction class	Confidence score	Eps
Original image	Siamese cat	99.9%	2

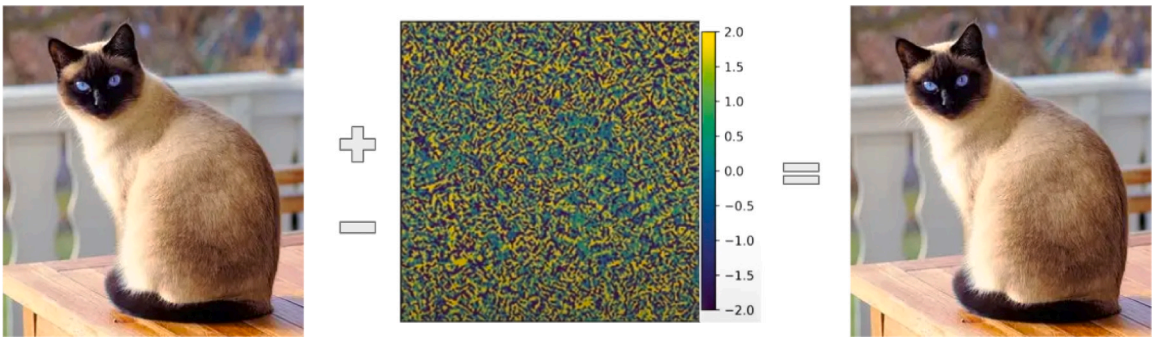


Fig. 5. Adversarial image of Siamese cat.

method 'generate' to generate adversarial examples of the Siamese cat image that has been loaded and this returns the adversarial image of the original image that was provided. At the end, the success of the attack is evaluated by providing the adversarial image to the classifier 'predict' method to see if the ML model now changes the classification of the image.

Algorithm 2: Evasion Attack using FGSM

1: **from** art.attacks.evasion **import** FastGradientMethod
2: **Create** an evasion attack of strength ϵ
3: $attack \leftarrow FastGradientMethod(estimator = classifier, \epsilon = 2)$
4: **Load** and **preprocess** the image
5: $image \leftarrow load\_image('siamese\_cat.jpg')$
6: **Create** an adversarial example
7: $adversarial\_image \leftarrow attack.generate(x = image)$
8: **Evaluate** attack
9: $y\_prediction \leftarrow classifier.predict(adversarial\_image)$

Output: The new prediction of the adversarial image is shown below. The original image had 99.9% classification of the Siamese cat class which is the expected classification for the original image but now, with the adversarial image, the ML model is classifying it as a 71% classification in the Indri (Lemur) class as shown in Table 5. So, the adversarial image that has been returned by the generate method in the model has been classified as an Indri (Lemur) instead of a Siamese cat class. Thus, the goal of creating an adversarial example and conducting an evasion attack has been achieved. The perturbation through the Fast Gradient Sign Method (FGSM) was added which is not easily perceptible by the human eye but leads the image classification model to change the classification and misclassify the image. Now, the Indri (Lemur) class can still be considered as quite close to the original classification of a Siamese cat, however, in a later stage of this research, it will be shown how the ML model changes the classification to a completely unrelated category through a stronger evasion attack. The perturbation changes from −2 to +2 which is caused by the epsilon steps that was given to the Fast Gradient Sign Method. So, this was an attack with a low attack strength on the classifier because an epsilon of 2 can be considered as quite a small change in the image's pixel values. The below image as shown in Fig. 5 shows the minimal perturbation in the image that had been generated by the Fast Gradient Sign Method which has been used in the above algorithm. The pertubation value was small, therefore it only caused a minor change to the original image.

Step 3: Algorithm 3: Mitigation using JPEGCompression: In the next algorithm, the researchers demonstrate an adaptive white-box evasion attack by adding a first level defense to the ML model to protect it from the minor perturbation already injected. The defense method being added here is referred to as a 'Pre-processing' technique. Many defense approaches against evasion attacks in literature consider pre-processing. Pre-processing is a common technique that conducts some processing on the image before it is given to the model. The pre-processing aims to remove adversarial perturbations. Here, the pre-processing approach is added as a defense and the researchers evaluate

Table 5
Fast gradient sign algorithm simulation results with low EPS.

Image	Attack method	Prediction class	Confidence score	Eps
Adversarial image	Fast Gradient Sign Method	Indri (Lemur)	71%	2

the strengths and weaknesses of this defense against evasion attacks and how far can it be relied on. In Algorithm 3 the pre-processing method 'JPEG Compression' is imported from the ART's defense module. This defense method will aim to remove adversarial perturbations by adding additional JPEG compression to the image that is provided to the model. A new instance of Joint Photographic Experts Group (JPEG) compression is added and the compression value has been set as 50% compression in the quality value. The researchers evaluate in the output whether the JPEG compression is sufficient against adaptive white-box attacks by providing the compressed adversarial image to the model.

Algorithm 3: Mitigation using JPEGCompression

1: **from** art.defences.preprocessor **import** JpegCompression
2: **Create** a defended ART classifier
3: $jpeg\_defence \leftarrow JpegCompression(quality = 50)$
4: $classifier \leftarrow KerasClassifier(model = model, clip\_values = (0, 255), preprocessor\_defences = jpeg\_defence)$
5: **Create** a defended adversarial image
6: $defended\_adv\_image \leftarrow attack.generate(x = image)$
7: **Evaluate** the defended image
8: $y\_prediction \leftarrow classifier.predict(defended\_adv\_image)$

Output: As it was noticed earlier, initially the original image had a 99% classification of the Siamese cat class and then it was misclassified as an Indri (Lemur) after adding a minor perturbation using the FGSM algorithm. Now, after adding the defense of JPEG Compression pre-processing technique to the adversarial image, the image has come back to its correct category. So, the ML model is classifying the image once again as a Siamese cat. However, the confidence score is 64% which is moderately low. The confidence score has reduced from 99% to 64% as shown in Table 6, however, the JPEG Compression pre-processing defensive technique has brought back the image to the correct classification. This shows that it was able to reduce some of the adversarial perturbations and get back to the correct class. JPEG Compression provides a decent certainty of the classification, despite being under an evasion attack, however, the researchers demonstrate next how JPEG Compression can be surpassed by a stronger attack.

3.3.1. Adaptive evasion attack approach

Up until now, the researchers showcased a simple way of attacking an image classification model with an evasion attack and defending it using a partially effective defense technique i.e., the pre-processing of input data using ART's toolbox. In Algorithm 4, the researchers demonstrate an evasion attack with increased attack strength. This

Table 6
Fast gradient sign algorithm simulation results after JPEG compression.

Image	Attack method	Prediction class	Confidence score	Eps
JPEG compressed image	Fast Gradient Sign Method	Siamese cat	64%	2

Table 7
Fast gradient sign algorithm simulation results with higher EPS.

Image	Attack method	Prediction class	Confidence score	Eps
JPEG compressed image	Fast Gradient Sign Method	Indri (Lemur)	79%	4

Table 8
Projected gradient descent algorithm simulation results.

Image	Attack method	Prediction class	Confidence score	Eps
JPEG compressed image	Projected Gradient Descent	Ambulance	27%	4

is to remind ML developers that every time they deploy a defense technique, they need to consider what an attacker could do if somehow the attacker becomes aware of the defense technique deployed and how an attacker can work towards circumventing that defense technique. That is the adaptive part of the white-box evasion attack and it is the most important part of this research. An ML developer must realize that they need to test the defense techniques implemented for any ML model. This requires developing an attacker's mindset and trying to identify ways on how can a stronger attack be executed to override the deployed defense techniques. This is demonstrated through Algorithm 4.

Step 4: Algorithm 4: Adaptive Evasion Attack using PGD: There are many advanced ways of increasing an attack's strength and making it more sophisticated. However, in this experiment, the researchers use a simple technique to increase the attack strength. Basically, to adapt to the new defense technique implemented and increase the attack strength, the researchers increase the epsilon value first. The epsilon value was 2 earlier, this has now been increased to 4 in Algorithm 4 which means every pixel value can change by plus or minus 4. Everything else will remain the same.

Algorithm 4: Adaptive Evasion Attack using PGD

1: **from** art.attacks.evasion **import** ProjectedGradientDescent
2: **from** art.defences.preprocessor **import** JpegCompression
3: **Create** a JPEG compression defense
4: $jpeg\_defence \leftarrow JpegCompression(quality = 50)$
5: **Create** an ART classifier for the attacked model
6: $classifier \leftarrow KerasClassifier(model = model, clip\_values = (0, 255), preprocessor\_defences = jpeg\_defence)$
7: **Increase** attack strength ε
8: $attack \leftarrow ProjectedGradientDescent(estimator = classifier, \varepsilon = 4)$
9: **Evaluate** the defended image
10: $y\_prediction \leftarrow classifier.predict(defended\_adv\_image)$

Output: In the output it is noticed that the new classification is back to the wrong class i.e., Indri (Lemur) with a 79% confidence score as shown in Table 7. This is even while the JPEG Compression is in place. So, just by increasing the epsilon value which increases the overall attack strength, the effect of the defense technique JPEG Compression was completely removed.

Step 5: Algorithm 5: Targeted Evasion Attack using PGD: Now, the researchers apply further complexity to the attack by changing the evasion attack algorithm in-use. Instead of the Fast Gradient Sign Method attack algorithm, the researchers use the Projected Gradient Descent attack algorithm which is an iterative application of the Fast Gradient Sign Method. There would be many more iterations of the Fast Gradient Sign Method and therefore naturally this would make the attack much more stronger.

Algorithm 5: Targeted Evasion Attack using PGD

1: **Apply** a more advanced attack
2: $attack \leftarrow ProjectedGradientDescent(classifier = classifier, targeted = True, \varepsilon = 4, \varepsilon\_step = 1, max\_iter = 1000)$
3: **Load** and **preprocess** the image
4: $image \leftarrow load\_image('siamese\_cat.jpg')$
5: **Run** targeted attack to classify as ambulance
6: $y\_target \leftarrow [407]$; `/* target class: 407 (ambulance) */`
7: $defended\_adv\_image \leftarrow attack.generate(x = image, y = y\_target)$
8: **Evaluate** the defended image
9: $y\_prediction \leftarrow classifier.predict(defended\_adv\_image)$

So in Algorithm 5, instead of the Fast Gradient Sign Method, the Projected Gradient Descent attack sub-module is imported from ART and an attack instance is created. The epsilon value remains as 4 and since this is an iterative attack, the epsilon steps is provided as 1 and the maximum iteration is provided as 1000. This will ensure it is properly converged. Another important variance in the below attack is that so far the researchers had been conducting untargeted attacks, so the model would simply classify the suitable class on its own. However, in the below case, the Projected Gradient Descent based attack is so strong against this classifier that the researchers have been able to transition easily to a targeted attack which is usually harder than the other untargeted attacks. A targeted attack means that any classification category will not be accepted, instead the ML model will be influenced towards a certain specific classification category required. In this case, the researchers require an adversarial example to be created that classifies the image to the 'ambulance' class which is target class 407 in the dataset. This has now been defined in Algorithm 5 with the target class ID as 407. Therefore, this will now execute a stronger evasion attack.

Output: In this case as well, no significant perturbations are noticeable to the human eye in the adversarial image. However, for the model, different perturbations are created within the adversarial image which leads to a misclassified result. With the new Projected Gradient Descent (PGD) algorithm in use and a higher epsilon value, the output received is in the 'Ambulance' category with a 27% confidence score as shown in Table 8. This is now completely different from how a cat image should be classified as. So, this is the final step of the adaptive white-box evasion attack using the ART tool and also investigating how a targeted version of an evasion attack looks like. This proves that image classification models are highly susceptible to evasion attacks.

Step 6: Algorithm 6: Mitigation using Spatial Smoothing: It is evident through the steps taken in the experiment so far that JPEG Compression which is a type of pre-processing technique is not an effective defensive mechanism against evasion attacks in image classification models. It can be easily overridden by stronger attack algorithms. Protecting ML models and specifically image classification models against adaptive evasion attacks does not have a straightforward answer. There are various technical ML and cybersecurity specific defense measures to be considered depending on the type of model, datasets and algorithms used which is elaborated further in a latter section of this research. Since in this particular experiment, the researchers are utilizing an image classification model with a relatively simple image of a 'Siamese cat' that was previously incorrectly classified as an 'Ambulance' by the model earlier, this can be reversed by applying a stronger mitigation technique. To improve the detection capabilities of the model and prevent the adversarial perturbation to make an effect on the image, the researchers replace the pre-processing mitigation technique 'JPEGCompression' with a better technique known as 'Spatial Smoothing'. Spatial smoothing is a mitigation technique used mainly

for image classification models (Park & Kim, 2022; Xu, Evans, & Qi, 2018). It applies a filter to an image to reduce noise and improve specific features. Spatial smoothing reduces the noise in an image and emphasizes its key features while suppressing irrelevant details. It also helps to improve the model's robustness by making it less sensitive to small perturbations in the input data. So in Algorithm 6, instead of JPEG Compression, the researchers now incorporate Spatial Smoothing. To do so, the researchers import the Spatial Smoothing sub-module from ART and the same attack instance is created. The epsilon value continues to remain as 4 and the researchers require the adversarial example to be created again which earlier classified the image to the 'ambulance' class in the dataset. However, this time the researchers utilize the Spatial Smoothing technique to help reduce the effects of the perturbation and enhance key features of the image. In Algorithm 6, the researchers have imported the Spatial Smoothing sub-module. Spatial smoothing will reduce the differences among individual pixels by using a sliding window filter. A sliding window is applied to each pixel in the image, replacing the center pixel with the median value of its neighbors. This method spreads pixel values across nearby pixels, rather than reducing the number of pixels in the image. The median filter reduces sample features by increasing similarity between adjacent pixels. In Algorithm 6, the researchers use a sliding window of 3 pixels.

Algorithm 6: Mitigation using Spatial Smoothing

1: **from** art.defences.preprocessing **import** SpatialSmoothing
2: **Create** a spatial smoothing defense
3: $ss\_defence \leftarrow SpatialSmoothing(window\_size = window\_size)$
4: **Create** an ART classifier for the attacked model with spatial smoothing
5: $classifier \leftarrow KerasClassifier(model = model, clip\_values = (0, 255), preprocessor\_defences = ss\_defence)$
6: **Evaluate** the defended image
7: $y\_prediction \leftarrow classifier.predict(defended\_adv\_image)$

Output: After applying the spatial smoothing mitigation technique, the model replaces each pixel of the image with the median of its neighboring pixels, therefore effectively eliminating the noise created by the adversarial perturbation generated by the Projected Gradient Descent (PGD) algorithm earlier. However, the image loses its high quality resolution which enhances certain features in the image. The model is now able to correctly predict the image as a 'Siamese cat' with 99% confidence as listed in Table 9. However, the confidence score can still be improved further. Therefore, the researchers will apply other mitigation techniques to examine if the confidence score can be improved.

Step 7: Algorithm 7: Mitigation using Gaussian Data Augmentation: In this step, the researchers apply another mitigation technique called as Gaussian Data Augmentation. This technique involves adding Gaussian noise to the input data to increase its robustness against adversarial perturbations. It can be effective in certain scenarios, especially when the noise level is carefully tuned to balance between robustness and preservation of useful information (Wang & Chen, 2023). Therefore, in the next step the researchers import the Gaussian Data Augmentation sub-module.

Algorithm 7: Mitigation using Gaussian Data Augmentation

1: **from** art.defences.preprocessing **import** GaussianAugmentation
2: **Create** an ART classifier with Gaussian augmentation defense
3: $classifier \leftarrow KerasClassifier(preprocessing = preprocessor, postprocessing\_defences = GaussianNoise(scale = 0.35), clip\_values = (0, 255), model = model)$
4: **Evaluate** the defended image
5: $y\_prediction \leftarrow classifier.predict(defended\_adv\_image)$

Output: After applying the gaussian data augmentation mitigation technique, the model introduces variability into the image, which can help the model learn to better generalize and become more robust to perturbations. It correctly predicts the image as a 'Siamese cat' with 72% confidence as listed in Table 10. However, the confidence score is quite low and therefore there is a need for a better mitigation technique to be applied.

Table 9
Spatial smoothing output results.

Image	Attack method	Prediction class	Confidence score	Eps
Spatial smoothened image	Projected Gradient Descent	Siamese cat	99%	4

Table 10
Gaussian data augmentation mitigation results.

Image	Attack method	Prediction class	Confidence score	Eps
Gaussian data augmented image	Projected Gradient Descent	Ambulance	72%	4

Step 8: Algorithm 8: Mitigation using Feature Squeezing: In this step, the researchers apply another mitigation technique called as Feature Squeezing (Xu et al., 2018). This technique involves reducing the precision or granularity of the features of the image. Instead of using the original high-precision pixel values (e.g., 8 bits per channel for RGB images), the pixel values are quantized to a lower precision (e.g., 4 or 2 bits per channel). By reducing the bit-depth of pixel values, each pixel value is rounded to the nearest value representable with the reduced number of bits. This reduces the number of distinct values that each feature can take on, effectively squeezing the feature space. This technique seems quite similar to Spatial Smoothing, however, these are quite distinct. While, spatial smoothing involves blurring or smoothing the input data to reduce the impact of small-scale perturbations by modifying the pixel values of the input image to create a smoother version, feature squeezing operates at a higher level than spatial smoothing and is applied to the representation of features extracted from the input data. Feature squeezing involves reducing the bit-depth of pixel values or quantizing intermediate representations in neural networks, and decreasing the number of distinct values that each feature can take on. Thus, in Algorithm 7 the researchers import the Feature Squeezing sub-module.

Algorithm 8: Mitigation using Feature Squeezing

1: **from** art.defences.preprocessing **import** FeatureSqueezing
2: **Create** an ART classifier with Feature Squeezing defense
3: $classifier \leftarrow KerasClassifier(preprocessing = preprocessor, postprocessing\_defences = FeatureSqueezing(bit\_depth = 3, clip\_values = (0, 255)), model = model)$
4: $defended\_adv\_image \leftarrow attack.generate(x = image, y = y\_target)$
5: **Evaluate** the defended image
6: $y\_prediction \leftarrow classifier.predict(defended\_adv\_image)$

Output: After applying the feature squeezing mitigation technique, the model reduces the precision of input features or feature representations and assists the model in predicting the image in the right class. The model now predicts the image as a 'Siamese cat' with 97% confidence as listed in Table 11. However, the confidence score still has potential for improvement. Therefore, the researchers continue to explore if two mitigation techniques i.e., such as Spatial Smoothing and Feature Squeezing can be combined together in the next step.

Step 9: Algorithm 9: Mitigation using Spatial Smoothing and Feature Squeezing: In this step, the researchers apply two mitigation techniques i.e., Spatial Smoothing (Park & Kim, 2022) and Feature Squeezing (Xu et al., 2018) together. It is expected that the confidence

Table 11
Feature squeezing mitigation results.

Image	Attack method	Prediction class	Confidence score	Eps
Feature squeezing	Projected Gradient Descent	Siamese cat	97%	4

score of the image will improve when two mitigation techniques are combined together. Thus, in Algorithm 9, the researchers import the Spatial Smoothing and Feature Squeezing sub-modules.

Algorithm 9: Mitigation using Spatial Smoothing and Feature Squeezing

1: **from** art.defences.preprocessing **import** SpatialSmoothing, FeatureSqueezing
2: **Create** an ART classifier with Spatial Smoothing and Feature Squeezing defenses
3: $classifier \leftarrow KerasClassifier(preprocessing = preprocessor, postprocessing\_defences = [SpatialSmoothing(clip\_values = (0, 255), window\_size = 3), FeatureSqueezing(bit\_depth = 3, clip\_values = (0, 255))], model = model)$
4: $defended\_adv\_image \leftarrow attack.generate(x = image, y = y\_target)$
5: **Evaluate** the defended image
6: $y\_prediction \leftarrow classifier.predict(defended\_adv\_image)$

Output: After applying the spatial smoothing and feature squeezing mitigation techniques together, the model is able to predict the correct class with an increased confidence score. The model now predicts the image as a 'Siamese cat' with 100% confidence as listed in Table 12. This proves that in this particular experiment and with the image that was utilized, the Spatial Smoothing and Feature Squeezing techniques were found to be most useful.

This concludes the experiment. Fig. 7 flowchart diagram explains each step carried out as part of this experiment. It is important to highlight that the effectiveness of the various mitigation techniques incorporated in this experiment for defending against evasion attacks depends on various factors including the specific characteristic of the attacks, the nature of the data, and the application context. Since, this experiment was a specific adaptive white box evasion attack and it utilized one single image of a Siamese cat. Spatial Smoothing and Feature Squeezing were the most optimal mitigation techniques. However, each technique has its own strengths and weaknesses and it is not necessary that every model will improve its defense using Spatial Smoothing and Feature Squeezing alone. There may be a need to incorporate other defense techniques as well. Below is an analysis of the merits and limitations of the techniques used in this experiment:

Gaussian Data Augmentation: This technique involves adding Gaussian noise to the input data to increase its robustness against adversarial perturbations. It can be effective in certain scenarios, especially when the noise level is carefully tuned to balance between robustness and preservation of useful information. However, it may not be optimal for all types of evasion attacks, especially if the attacker is aware of the augmentation strategy and can adapt accordingly (Wang & Chen, 2023).

Spatial Smoothing: Spatial smoothing involves blurring the input image to reduce the impact of small-scale perturbations. It can be effective against attacks that rely on fine-grained perturbations, such as pixel-level manipulations. However, it may also blur important details in the image and affect the overall performance of the system, especially if the smoothing kernel is too aggressive (Park & Kim, 2022).

Feature Squeezing: Feature squeezing reduces the precision of input features to make them less susceptible to adversarial perturbations. This can be achieved by reducing the bit-depth of pixel values or by quantizing intermediate representations in the neural network. Feature squeezing can be effective against attacks that exploit the sensitivity of neural networks to small changes in input features. However, it may also degrade the performance of the model on legitimate inputs, especially if the squeezing is too aggressive (Xu et al., 2018).

Fig. 6 demonstrates the variations of the same image when each mitigation technique is applied. In practice, a combination of these above mentioned techniques, along with other defense mechanisms is often used to enhance robustness against evasion attacks. Additionally, it is important to evaluate the effectiveness of these techniques empirically on the specific data and model architecture of interest.

4. Results and discussion

Through this experiment, the researchers showed how the robustness of an ML model can be evaluated through the ART toolbox and how certain defensive techniques can be overridden. In this experiment, the researchers initially utilized JPEG Compression which is a common pre-processing defense technique against evasion attacks but this was overriden when a stronger attack algorithm and epsilon value was leveraged. It was demonstrated how ML models are susceptible to evasion attacks and many defense techniques are not truly effective.

4.1. Summary of simulation results

In the above experiment, it was noticed that by adding slight unnoticeable perturbations to the original Siamese cat image, the ML model was initially evaded and gave an incorrect outcome.

Confidence & Prediction Class: In Step 1, the confidence of the original image of a Siamese cat was 99.9%, but in Step 2, this dropped to 71% after adding slight perturbations to the original image using the Fast Gradient Sign Method attack algorithm with an epsilon value of 2. The model predicted the wrong classification category of Indri Lemur as well. In Step 3, when a defense technique of JPEG Compression pre-processing was applied, the model predicted the right classification class of Siamese cat again, however the confidence score still remained low i.e., 64%. In Step 4, when further perturbations were inserted using a higher epsilon value of 4, the model predicted the wrong class of Indri Lemur again with a confidence of 79%. Finally, with a more advanced attack technique i.e., Projected Gradient Descent algorithm being incorporated in Step 5, the model gave an incorrect classification of the image as an Ambulance with a 27% confidence which was completely irrelevant. It classified the image of the Siamese cat as an Ambulance. This indicates how a ML model is greatly susceptible to evasion attacks and how even a small perturbation can completely dissuade the model from making the correct classification. It also indicates how defenses can be overridden by using stronger attack algorithms. Finally, in Step 6, the researchers used a different defense technique i.e., Spatial Smoothing to reduce the noise created by the adversarial perturbation and enhance its key feature. This finally led to the model accurately classifying the image as a Siamese cat, however the confidence reached only to 84%. Therefore, one can state that in image classification ML models, evasion attacks remain a great concern, especially in critical sectors such as autonomous transportation and healthcare where these models are being prominently used for self-driving cars and medical systems and any misclassification of an image can lead to serious consequences. In the next section, the experiment undertaken in this research is compared with prior work.

4.2. Comparison with prior work

In this section, the researchers survey prior work in generating adversarial examples and study the impact of evasion attacks and algorithms used to conduct similar attacks. Since, there is a lot of research on adversarial examples, we only include the most relevant papers here as listed in Table 13.

Table 12
Spatial smoothing and feature squeezing mitigation results.

Image	Attack method	Prediction class	Confidence score	Eps
Original image	Fast Gradient Sign Method	Siamese cat	99.9%	2
Adversarial image	Fast Gradient Sign Method	Indri (Lemur)	71%	2
JPEG compressed image	Fast Gradient Sign Method	Siamese cat	64%	2
JPEG compressed image	Fast Gradient Sign Method	Indri (Lemur)	79%	4
JPEG compressed image	Projected Gradient Descent	Ambulance	27%	4
Spatial smoothened image	Projected Gradient Descent	Siamese cat	99%	4
Gaussian data augmented image	Projected Gradient Descent	Siamese cat	72%	4
Feature squeezing	Projected Gradient Descent	Siamese cat	97%	4
Spatial smoothing and feature squeezing image	Projected Gradient Descent	Siamese cat	100%	4

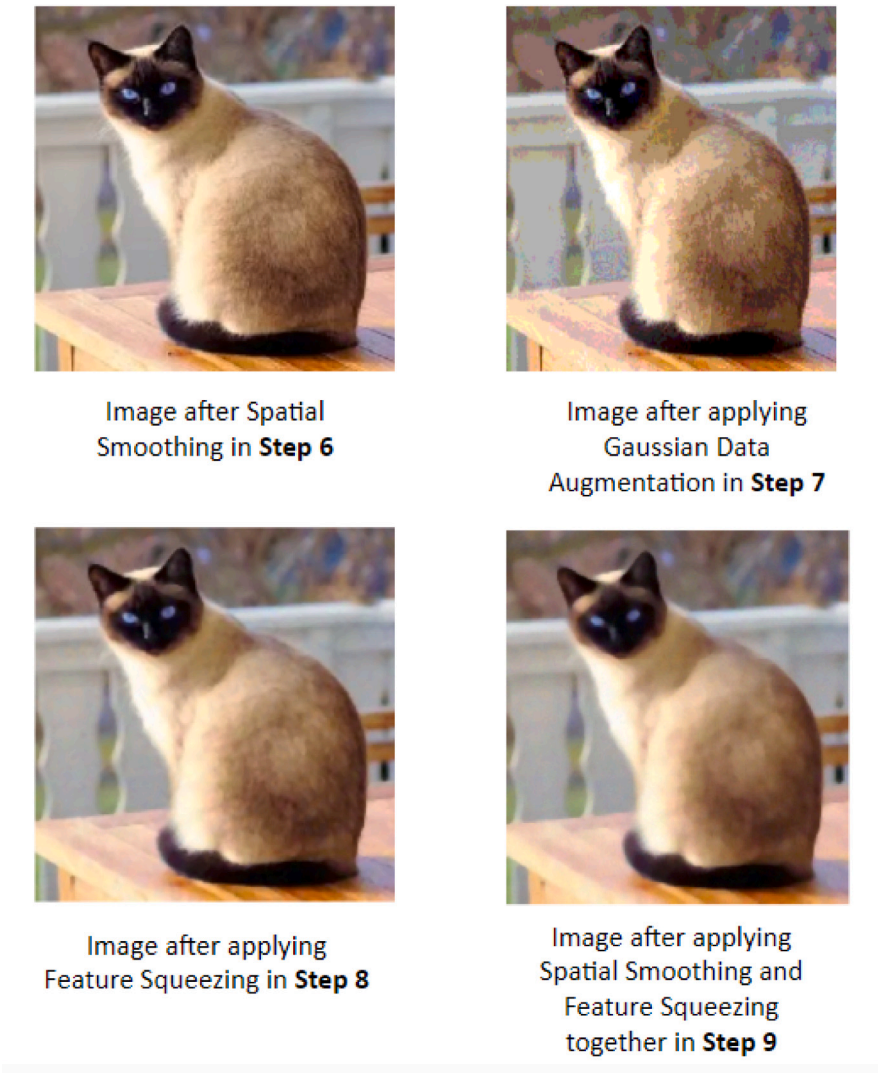


Fig. 6. Variations of Siamese cat image.

Majority of the studies conducted in the past on evading ML models and identifying potential mitigation techniques discuss specifically on the algorithms or point-based defense techniques such as JPEG Compression but barely provide a comprehensive secure architecture for mitigation of AML attacks as a whole. In contrast, our work focuses on creating a ML and cybersecurity architecture workflow to protect the ML model environment from evasion attacks. Many studies are also misleading when it comes to proposing mitigation techniques. Multiple studies have stated that JPEG Compression is an effective defense technique against evasion attacks aimed at image classification models. Das et al. (2017) proposed using JPEG compression, to eliminate adversarial noise from a dataset. They examined how different JPEG compression characteristics impact the precision of predictions and claimed that JPEG compression is an effective technique to combat evasion attacks. This is misleading as the experiment in this research demonstrated how JPEG compression is an ineffective defense technique. Also, the study only used the Fast Gradient Sign Method (FGSM) which is a low-level attack algorithm. The study did not employ stronger algorithms such as Projected Gradient Descent (PGD) that can easily evade the model in spite of leveraging JPEG Compression. Our work demonstrated an adaptive evasion attack that made use of the PGD algorithm besides FGSM to prove JPEG Compression is not an effective mitigation technique. The experiment conducted in this research proves that when a stronger attack algorithm is used to execute an attack, JPEG Compression fails in preventing the model from being evaded. Several other

Fig. 7. Adaptive evasion attack flowchart.

studies conducted similar evasion attacks with different algorithms but have major limitations in their analysis. Certain studies demonstrated how evasion attack algorithms could evade a model, however they did not provide any mitigation techniques at all. For example, Eykholt et al. (2018) introduced Robust Physical Perturbations (RP2), a general attack algorithm that produced robust adversarial visual perturbations in a variety of physical settings. They demonstrated that adversarial examples created with RP2 produces high targeted misclassification rates against standard-architecture road sign image classifiers in the actual world under a variety of environmental variables, such as views, utilizing the real-world scenario of road sign classification. Now, even though the study demonstrated how road sign classifiers in autonomous vehicles can be evaded using adversarial perturbations generated from RP2 algorithm, the authors did not provide any countermeasures to protect the model from these attacks. Similarly, Brown et al. (2018) provided insights on how large perturbations can be more effective in evading a model, however their work as well did not mention any mitigation techniques to protect the model from evasion attacks. Whereas, in this research, the authors aim to provide specific mitigation techniques against evasion attacks.

A few studies have also specified certain mitigatory techniques without providing an overall architecture that incorporates ML as well as cybersecurity aspects. For example, Aleks et al. (2019) suggested ML Networks need more capability to withstand powerful adversarial

Table 13
Comparison with prior work.

Paper title and source	Algorithms used	Impact
Towards deep learning models resistant to adversarial attacks (Aleks et al., 2019)	PGD	The study conducted white-box attacks with PGD using the Carlini–Wagner (CW) algorithm.
Feature squeezing: Detecting adversarial examples in deep neural networks (Xu et al., 2018)	No specific algorithm used	The study employed feature squeezing to compare between the prediction made by a model on the original input and the squeezed inputs to identify adversarial examples with a high degree of accuracy and few false positives. The authors investigated feature squeezing and spatial smoothing.
Deflecting adversarial attacks with pixel deflection (Prakash, Moran, Garber, DiLillo, & Storer, 2018)	Pixel deflection and JPEG compression	The study provided a defense module with certain defenses against adversarial attacks but no specific defenses provided against evasion attacks alone.
Adversarial examples in the physical world (Kurakin, Goodfellow, & Bengio, 2017)	JPEG compression	The study used multiple image transformation techniques such as blurring, adding random noise along with JPEG compression to create adversarial examples that cause misclassification in ML models.
A study of the effect of JPEG compression on adversarial images (Dziugaite, Ghahramani, & Roy, 2016)	FGSM and JPEG compression	The study first assessed the impact of adversarial perturbations on the network's classification before examining the influence of an additional JPEG compression on the adversarial image's classification. The authors assessed the variation at various levels of perturbations caused by adversarial inputs.
Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression (Das et al., 2017)	FGSM, DeepFool and JPEG compression	The study proposed using JPEG compression, to eliminate adversarial noise from a dataset. They examined in detail how different JPEG compression characteristics impact the precision of predictions.

Table 14
State-of-the-art comparison of model parameters.

A study of the effect of JPG compression on adversarial images (Dziugaite et al., 2016)				
Test cases for JPEG compression	Algorithm	Epsilon (Eps)	Confidence score	Prediction class
Original image (x)	None	None	99%	agama
Adversarial image (x′)	FGSM	1	93%	rock crab
Adversarial image (x′) after JPEG compression	FGSM	1	48%	agama
Adversarial image (y′) after JPEG compression	FGSM	5	26%	agama
Adversarial image (y′) after JPEG compression	FGSM	10	17%	agama
Our work: Detection and prevention of evasion attacks on machine learning models				
Test cases for JPEG compression	Algorithm	Epsilon (Eps)	Confidence score	Prediction class
Original image (x)	None	2	99.9%	Siamese cat
Adversarial image (x′)	FGSM	2	71%	indri (lemur)
Adversarial image (y′) after JPEG compression	FGSM	2	64%	Siamese cat
Adversarial Image (y′) after JPEG compression	FGSM	4	79%	indri (lemur)
Adversarial image (y′) after JPEG compression	PGD	4	27%	ambulance
Adversarial image (y″) after spatial smoothing	PGD	4	99%	Siamese cat
Adversarial image (y″) after Gaussian data augmentation	PGD	4	72%	Siamese cat
Adversarial image (y″) after feature squeezing	PGD	4	97%	Siamese cat
Adversarial image (y″) after spatial smoothing and feature squeezing	PGD	4	100%	Siamese cat

attacks. In spite of providing this as a recommendation, the study did not provide any further insights on improving the network's architecture and model's capacity to withstand adversarial attacks. Similarly, Ross and Doshi-Velez (2017) focused on using two specific point-based countermeasures i.e., network distillation and adversarial training. However, the study did not provide insights on how a ML model's entire network must be secured against evasion attacks. Xu et al. (2018) only focused on the effectiveness of feature squeezing as a countermeasure against evasion attacks which is known to have several limitations. The study did not provide an overall architecture to secure ML models. In this research, the authors provide an ML cybersecurity architecture workflow that can be implemented by organizations to improve the overall security of the model.

The study that closely resembled to this research work was (Dziugaite et al., 2016). This study demonstrated the effect of JPEG compression on adversarial images. In this study, Dziugaite et al. (2016) first assessed the impact of adversarial perturbations on the network's classification of a single image before examining the influence of JPEG compression on the image as well as the model's classification. The authors assessed the variation at various levels of perturbations caused by adversarial inputs and the effect noticed on the model parameters. However, this study as well only used the Fast Gradient Sign Method (FGSM) algorithm and was unable to provide any insights on JPEG compression's efficacy if a stronger algorithm such as Projected Gradient Descent (PGD) is incorporated. Table 14 compares the model parameters of this work with Dziugaite et al. (2016) and demonstrates how the confidence score and prediction class is affected before and after applying JPEG Compression to an image. It is noticed that the confidence score decreases even in Dziugaite et al. (2016) as the epsilon value increases. The prediction class remains the same. However, in our experiment, the prediction class changes as a stronger attack algorithm is utilized. Our research is the first of a kind to have demonstrated an adaptive evasion attack with more than one algorithm tested to depict the effectiveness of defense techniques on an image classification model. This research also distinguishes by using one image instead of an entire dataset. In the next section, the researchers provide specific countermeasures that can help prevent evasion attacks from being executed.

4.3. Technical ML defense measures

The truth is that there is no magical solution to protect ML models against all types of evasion attacks and other AML attacks as well such as poisoning and privacy attacks. Organizations require to perform several steps to secure their model environment as well as test out the ML specific technical defense measures applied. In this experiment,

Table 15
Technical ML defense measures.

S.No.	Stage	ML defense measures
1.	Input pre-processing	Data transformation (such as JPEG compression, spatial smoothing, Feature squeezing and Gaussian data augmentation) (Guo, Rana, Cisse, & van der Maaten, 2018; Kunwar, 2021; Nayak & Friedland, 2023; Rochac, Liang, Zhang, & Oladunni, 2019; Xu et al., 2018)
2.	Output post-processing	Adversarial training (Bai, Luo, Zhao, Wen, & Wang, 2021; Cao & Xue, 2023; McClintick, Harer, Flowers, Headley, & Wyglinski, 2022) Regularization (Tian & Zhang, 2022) Certifiable training (Calzavara, Ferrara, & Lucchese, 2023; Kumar et al., 2024) Gradient masking (Bountakas, Zarras, Lekidis, & Xenakis, 2023; Shaukat, Luo, & Varadharajan, 2022; Wang et al., 2023) Distillation network (Papernot & McDaniel, 2017)
3.	Post-production	Adversarial testing (Papers, 2024) Adversarial example detection (Aldahdooh, Hamidouche, Fezza, & Déforges, 2022; Zhao et al., 2023)

even though Spatial Smoothing was found to be more effective in defending against evasion attacks compared to JPEG Compression, it has been found that Spatial Smoothing as well has its own limitations (Sharma & Chen, 2018). Every ML specific defense measure has its own merits and limitations. In this section, the researchers simplify the various types of defense techniques that can be applied to a ML model.

In Table 15, the technical code-level ML defenses against evasion attacks are listed. Each of these defense techniques are further described below:

Input Pre-Processing Measures: There are several defensive measures that can be combined to improve the resilience of a model to evasion attacks. One strategy is to focus on the training dataset through data transformation. Data transformation increases the difficulty of creating adversarial examples while maintaining model accuracy but at the expense of computational overhead. Data transformation pre-processing techniques such as JPEG Compression, Spatial Smoothing and several others can be used to enhance the robustness of image classification models against evasion attacks. These pre-processing techniques are also referred to as data transformation techniques. The most commonly data transformation techniques used are spatial smoothing, feature squeezing, normalization, gaussian smoothing, randomization and quantization. These have been useful to a certain extent but have many limitations. JPEG Compression is also one of them which was used in this experiment. However, it was noticed that JPEG Compression is a weak defensive technique when faced against stronger evasion attack algorithms such as PGD. Other data transformation techniques like spatial smoothing have also been identified as ineffective in certain studies but some pre-processing techniques such as feature squeezing and quantization are helpful as well and can prevent evasion attacks that involve minor perturbations. However, these techniques tend to fail in attacks with larger perturbations and also compromise on the image quality. Not all pre-processing techniques should be classified as ineffective. Many of them are effective when used in combination with other post-processing and post-production defense techniques as illustrated in Fig. 8.

Output Post-Processing Measures:

Output post-processing is another strategy that improves model robustness while reducing training overhead and maintaining accuracy. The development of output post-processing model hardening techniques for specific attacks is a competitive effort. Over time, adaptive attackers can beat some of them, limiting their usefulness. Below are some model hardening strategies, including their strengths and limitations:

Adversarial training: Adversarial training is a technique used in ML to improve the robustness of a model against adversarial attacks. The basic idea behind adversarial training is to augment the training data with adversarial examples, which are carefully crafted to deceive the model. By training the model on both clean and adversarial examples, the model learns to be more resilient to adversarial perturbations. Adversarial training is one of the most widely researched approaches for countering evasion attacks. Adversarial training involves using a training data set with correctly labeled adversarial examples to prevent the model from being deceived by them. Adversarial examples are generated using attack algorithms used by attackers such as FGSM, PGD, or Carlini–Wagner. To train models with adversarial examples, the Fast Gradient Sign method (FGSM) (Goodfellow et al., 2015) has often been used. Past studies such as Aleks et al. (2019) also indicate that the Projected Gradient Descent (PGD) was used to produce resilient models against adaptive evasion attacks. PGD can identify the most likely adversarial cases for a threat model by generalizing other attack strategies. According to Aleks et al. (2019), adversarial training using PGD is more effective than other algorithms. However, adversarial training provides limited protection against evasion attacks, making it insufficient for certain applications. Adversarial training increases the training duration as well which is another limitation and cannot be solely relied upon as a defense technique McClintick et al. (2022).

Regularization: Regularization prevents small input perturbations from affecting model output by providing protection against adversarial samples. It is used to prevent over-fitting and improve the generalization of a model. Over-fitting occurs when a model learns the noise and details in the training data to the extent that it performs poorly on new, unseen data. Regularization introduces a penalty term to the model's loss function, discouraging the model from fitting the training data too closely and instead encourages it to learn simpler patterns that generalize better. Existing techniques include constructing specialized regularizations, such as Lipschitz regularization and randomized smoothing. Randomized smoothing (Cohen, Rosenfeld, & Kolter, 2019) is based on probabilistic robustness verification. Adapting the training technique to regularization can limit the gradient of the neural network's implemented function (Pitropakis et al., 2019). There is a strong correlation between these two approaches, as any differentiable function with a bounded gradient is continuous. However, the assumed constant can be enormous, and the model function in question is not always differentiable. Classifiers are learned using a training dataset enriched with Gaussian noises, resulting in a probability distribution across all classes. The final model is trained using the expanded dataset, with labels indicating the most probable class. The resulting model is provable due to randomized smoothing procedures. However, in spite of regularization significantly enhancing robustness of model, it has certain drawbacks as well. Gradient boundaries are often promoted by penalty terms rather than rigidly enforced and enforcing these boundaries can impair the network's generalization performance and accuracy on complicated tasks.

Certifiable training: Certifiable training in ML refers to a training process that guarantees certain properties or bounds on the model's behavior. This approach is particularly important in safety-critical applications where it is crucial to have formal guarantees on the model's performance. Certifiable training can be considered as a robustness verification technique that aims to improve neural networks' lower bounds. It requires using specialized algorithms or techniques to solve the optimization problem while ensuring that the specified properties are satisfied. This may involve using advanced optimization methods or incorporating domain-specific knowledge (Calzavara et al., 2023; Kumar et al., 2024).

Gradient masking: Gradient masking is another way to fight against gradient-based attacks. Shattered gradients create non-differentiable

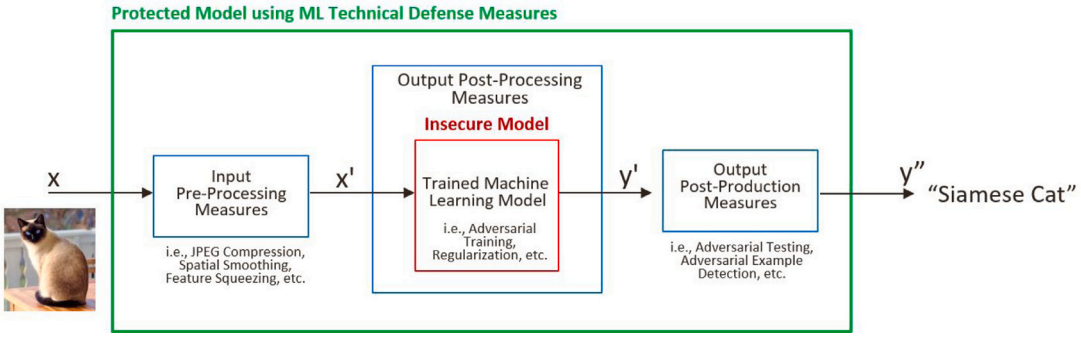


Fig. 8. ML technical defense measures.

actions, making it difficult for attackers to create adversarial examples with proper gradients. Another example is stochastic gradients, which adds randomization to gradients. However, Athalye, Carlini, Wagner (2018) demonstrated that the gradient masking strategy can be successfully overcome by advanced attacks. Backward pass differentiable approximation algorithms are provided for recovering gradient signals from shattered gradients (Athalye et al., 2018). Estimating stochastic gradients using expectation over transformation strategies can help reduce uncertainty. As a result, the gradient masking strategy is significantly less effective against targeted gradient-based evasion attacks.

Distillation network: Distillation refers to a technique used to transfer knowledge from a large, complex model (teacher) to a smaller, simpler model (student). The goal of distillation is to train the student model to mimic the behavior of the teacher model, typically achieving similar performance but with lower computational resources and memory requirements. A distillation network is offered as a reliable training strategy for classifiers against malicious samples. To train the distillation network, a teacher model is initially created on a training dataset. The distilled network is trained using the identical inputs and output probabilities as the instructor network. The training dataset's initial labels are considered hard labels, whereas the output probabilities are referred to as soft labels. However, distillation networks have been demonstrated to be less effective against adaptive evasion attacks and should not be solely relied upon as a defensive technique Pang, Cheng, Hu, and Liu (2021). It would require to be combined with other techniques to ensure defense-in-depth.

Post-Production Measures: Monitoring after the model is put in production is crucial. Combining post-production techniques such as adversarial testing and adversarial example detection along with pre-processing and post-processing techniques creates a feedback loop for improving a model's robustness. Here are some ways for assessing a model robustness after it has been put into production:

Adversarial testing: Adversarial testing is strongly related to adversarial training, which was discussed earlier. Adversarial testing process evaluates the robustness of a trained model against malicious samples. These attacks can be constructed by several ways including the usage of projected gradient descent algorithm. Adversarial testing evaluates a model's robustness against various attacks. If testing results indicate low model robustness, adversarial training might be used to further harden it Papers (2024).

Adversarial example detection: Adversarial example detection evaluates input samples to determine whether they have been changed. To recognize adversarial examples, several studies look for appropriate statistics. For convolution neural networks, binary classifiers are created using the statistics of convolutional layer results from a regular training dataset and adversarial instances. Experimental results demonstrate the usefulness of ImageNet-based datasets with AlexNet and VGGNet network architectures against L-BFGS assaults at the cost of building a binary classifier. This method requires knowledge of the training dataset. Another statistical method for adversarial example detection used predicted perturbed log-odds across random perturbations to determine if a given input x classed as class y is manipulated and the true class is z (Roth, Kilcher, & Hofmann, 2019). This approach required access to the results before the final layer during runtime. Experimental results demonstrated that CIFAR-10 and ImageNet are successful against PGD attacks, but need several inferences for a single input. Even though, there are various ML-based defenses available, one must note that these are not sufficient to secure ML models from strong adaptive evasion attacks. Adversaries can always try finding loopholes in them and circumvent ML based defenses to gain access to the organization's ML environment. Therefore, there is a need to implement additional cybersecurity defense measures to provide stronger protection against evasion attacks. To ensure basic cybersecurity hygiene, the researchers recommend the technical cybersecurity defense measures as part of the next section.

4.4. Technical cybersecurity defense measures

In the below section, the researchers describe each cybersecurity defense technique and propose the Cyber Resilient ML Model Architecture Framework for creating a secure ML application environment within an organization.

5. Cyber resilient ML model architecture framework

Each phase of the Cyber Resilient ML Model Architecture Framework plays a vital role in creating a secure and fully functional machine learning model. Incorporating cybersecurity defense techniques at every stage of the development lifecycle is essential to minimize the overall security risks within the model's environment before it reaches production. By embedding these defenses early, the data and ML components are designed and developed with foundational cybersecurity hygiene principles in mind, ensuring a secure base. In contrast, organizations that test models only at the end of their development lifecycle are more likely to encounter critical cybersecurity vulnerabilities and risks. To mitigate this, it is recommended that organizations integrate cybersecurity principles directly into the development pipeline of models. The following sections detail each phase of the model lifecycle and the corresponding cybersecurity measures that should be implemented, as illustrated in Fig. 9.

(A) Business and Data Understanding (Initial Ideation Phase): Within the model lifecycle, the first phase of Business and Data Understanding is the initial ideation phase and this phase is crucial for ensuring that the ML project addresses the right problem and has access to the appropriate data. The objective of this phase is to understand the business problem or opportunity that the ML model aims to solve or leverage. This involves collaborating closely with stakeholders to clarify goals, success criteria, and constraints. It also involves identifying and determining the data required to address the business objectives. This includes understanding the type, quality, and quantity of data needed,

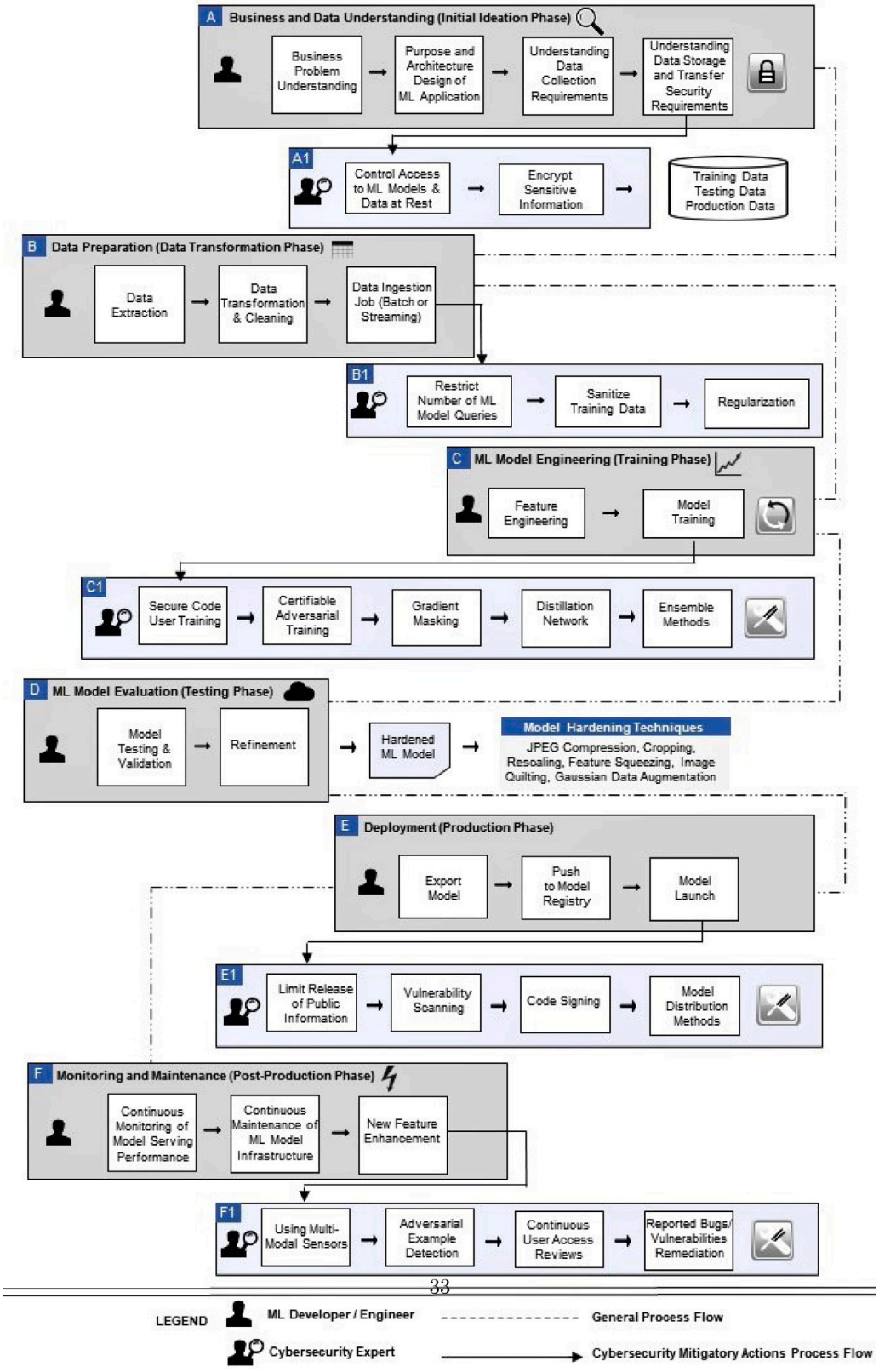


Fig. 9. Cyber Resilient ML Model Architecture Framework.

as well as any data sources that may be available. Organizations must pre-plan the data storage and transfer security requirements at this stage as well.

(A1) Technical ML and Cybersecurity Defense Measures: In terms of security, this phase requires the organization to consider the access control and storage of any sensitive data that is planned to be used within the ML model. Organizations must protect the data through encryption methods and access must be restricted to authorized personnel only. This may involve all datasets related to training, testing and production if split beforehand. To prevent adversaries from trying to access sensitive data, it is crucial for organizations to encrypt sensitive data, so that even if an adversary gains access to it, it cannot be comprehended. Organizations must also limit internal access to production data and model as well as implement access controls on internal model registries. They must restrict training data access to authorized users only. They may use an identity and access management system to better regulate this process and avoid manual errors.

(B) Data Preparation (Data Transformation Phase): This phase involves collecting and integrating the necessary data from various sources. This may involve data cleaning, pre-processing, and transformation to prepare the data for analysis. Organizations would require that the data is of high quality and meets the business requirements of the ML model. This includes checking for missing values, outliers, and inconsistencies. The data may be ingested in batches or might flow in as streams at a real-time basis.

(B1) Technical ML and Cybersecurity Defense Measures: During this phase, organizations must ensure they limit the amount of queries users can submit within a certain time frame to prevent attackers from sending large, computationally demanding inputs. They may limit the total quantity and rate at which a user is able to ask questions. As a result, this would reduce the amount of knowledge an attacker can obtain about the ontology of an ML model by using API queries. In order to control the quantity and quality of potentially sensitive information that an attacker can obtain, organizations may also limit the number of API queries that are sent in a given amount of time. Additionally, they must incorporate training data sanitization procedures for removing contaminated training data. For an active learning model, training data should be cleaned periodically before the model is trained. To reduce the amount of training data ingested, implementing a sanitizing data filter will certainly help. Additionally, creating a content policy that would prohibit the use of undesirable content, such as explicit or offensive language will prevent an adversary from evading certain text analysis based ML models such as chat bot applications. It would also help in obfuscating an ML application's query output, to limit the quantity of results displayed. Limiting the specificity of the ontology for output classes, by making use of randomized smoothing and regularization techniques and decreasing the numerical outputs' precision will help prevent adversaries to gather information about the model through its outputs and craft attacks specifically for it to be hindered.

(C) ML Model Engineering (Training Phase): The ML model engineering training phase is a crucial step in the ML model lifecycle where features are selected or constructed from raw data, and the model is trained on the prepared data to learn patterns and make predictions. It involves selecting the most relevant features from the raw data to use as inputs for the model. This helps reduce complexity and improve model performance. It may also include choosing the appropriate ML algorithm based on the nature of the problem and the characteristics of the data and iteratively adjusting the model's parameters to minimize the difference between predicted and actual values (i.e., the loss function). The goal is to develop a model that can accurately generalize to new, unseen data and make reliable predictions.

(C1) Technical ML and Cybersecurity Defense Measures: This is a crucial stage where most amount of code development takes place to create the model and it is associated algorithms. It is imperative that the developers are trained for secure code development so that they do not use insecure coding methods that may result in the execution of malicious code. It would certainly help to teach developers to spot manipulation attempts so they will not run dangerous code that, if run, could produce dangerous artifacts. The system may suffer as a result of these artifacts. Additionally, organizations must conduct certifiable adversarial training to validate that ML models function as intended and do not have adversarial bias. This will ensure an organization's ML models do not react to adversarial bias or possible backdoor triggers. Organizations must keep an eye out for concept drift and training data drift in the model, as these could be signs of evasion attacks. Another important defense measure is clipping the gradients, also referred to as gradient masking. Gradient masking is typically achieved by modifying the architecture or the training process of the model. For example, one common approach is to clip the gradients to a certain range during training, which prevents attackers from using the gradients to compute adversarial perturbations. Another approach is to add noise to the gradients, making them harder to use for crafting adversarial examples. Gradient masking aims to hide or obfuscate the gradients of the model with respect to the input, making it harder for attackers to compute the perturbations needed for adversarial examples. It is not a foolproof defense and can be circumvented by more sophisticated attackers. As such, it is often used in conjunction with other defense mechanisms to enhance the overall security of the model. Furthermore, organizations can opt for a distillation network to transfer knowledge from a larger, more complex model (the teacher) to a smaller, simpler model (the student). The goal of knowledge distillation is to distill the knowledge learned by the teacher model into a more compact form that can be more easily deployed in resource-constrained environments.This can make the model more resilient to adversarial attacks, as the distilled knowledge may include information about how to generalize better and avoid being misled by adversarial examples. Finally, organizations must use ensemble methods to prevent evasion attacks. Certain evasion attacks might successfully avoid one model or model family while failing miserably against others. Robustness against attacks is increased by using multiple different models through ensemble methods. Additionally, if a security flaw in the tool for one model or family is discovered, it would guarantee the least amount of performance loss. It might deceive adversaries about the kind of model being used and its application. Therefore, using an ensemble of models will make an organization's model more resilient to malicious inputs.

(D) ML Model Evaluation (Testing Phase): The main goal of the ML Model evaluation testing phase is to evaluate how well the model generalizes to new, unseen data and to assess its performance according to predefined metrics. The trained model is used to make predictions on the test set, and the predictions are compared to the actual labels in the test set. Various performance metrics are calculated to evaluate the model, depending on the nature of the problem. These metrics are further tuned based on the performance on the test set to improve the model's performance. The final model is selected based on its performance on the test set, and it is ready for deployment in a production environment.

(D1) Technical ML and Cybersecurity Defense Measures: This phase requires a slightly different approach than other phases depending on the type of data and model in use. It is necessary to pre-process all tested inference data and eliminate or reversing any possible adversarial perturbations. This may require using specific model hardening techniques such as JPEG Compression, Cropping, Rescaling, Feature Squeezing, Image Quilting, Gaussian Data Augmentation, etc. However, these techniques must be tested thoroughly using testing tools like ART as demonstrated in the experiment conducted earlier. Many model hardening techniques are not strong enough to prevent advanced evasion attacks. Therefore, these techniques must be combined with others to create defense-in-depth across the ML model environment. There are many techniques available for different types of ML models such as Image classification, speech recognition, text classification and so on. Plenty of these techniques are available in the ART toolbox for

experimentation. The ML developers/engineers would be more capable to understand which model hardening techniques suit their model requirements and they can validate and confirm these by using the ART toolbox's testing tools. After model hardening has been successfully implemented, an adversary will not be able to analyze the input–output relationship as an additional layer of uncertainty and randomness is assumed to be added to the output data. Therefore, the ML model can be shielded from malicious data by processing model inputs through various model hardening techniques.

(E) Deployment (Production Phase): In this phase, the trained model is deployed into a production environment to make predictions on new, unseen data. The trained model, along with any necessary pre-processing or post-processing code, is packaged into a deployable unit. This unit may include scripts, configuration files, and dependencies needed to run the model. The model is optimized for scalability and performance in the production environment. This may involve optimizing the model's architecture, tuning hyper parameters, and implementing caching or parallel processing techniques.

(E1) Technical ML and Cybersecurity Defense Measures: During the deployment stage, it is extremely important that the organization restricts the public dissemination of technical details regarding model such as libraries, algorithms, datasets, APIs, etc., that is employed in the organization's products or services. An adversary's technical understanding of ML can be used to target and customize attacks for the ML application. Therefore, this must be highly controlled. Furthermore, the organization must consider limiting the dissemination of organizational details as well that could be used to deduce technical information about ML methods, model architectures, or datasets, such as department names, physical locations, and researcher names. The organization must restrict the relationship between approaches that are made public and the models, data, and algorithms that are utilized in actual production. Restricting the amount of technical project information that is made public, such as data, algorithms, model architectures, and model checkpoints that are representative of or used in production must be considered. Additionally, usage of publicly available datasets must be avoided and if not an option, must be completely hidden from public knowledge. Reducing the amount of datasets, model architectures and checkpoints that are released can make it harder for adversaries to target production models that have been trained on the same or comparable data. It is also recommended that before deployment, organizations conduct a vulnerability scan to identify software vulnerabilities that may be exploited and fix them proactively. Commonly used file formats for storing ML models, like pickle files, can include exploits that lead to arbitrary code execution. Organizations must check both downstream products generated by models and model artifacts for any vulnerabilities. Organizations can also deploy a scanner to continuously check for vulnerabilities in ML artifacts. Furthermore, to stop untrusted code from running, organizations can enforce binary and application integrity using digital signature verification. Malicious code can be embedded by adversaries in ML models or software. Enforcing code signatures will stop malicious code from running and the ML supply chain from being compromised. Lastly, deploying ML models to edge devices increase the system's attack surface. Organizations must consider providing models in the cloud to limit the adversary's access to the model. Not distributing the ML model software to edge devices will limit an adversary's ability to acquire complete access to the model.

(F) Monitoring and Maintenance (Post-Production Phase): Monitoring and Maintenance in the post-production phase of the ML model life cycle are essential activities for ensuring that the deployed model remains effective, efficient, and secure over time. By proactively monitoring and maintaining the model, organizations can ensure that it continues to deliver value and meets the needs of its users. This phase is crucial for detecting and addressing security issues that may arise over time, such as adversarial inputs causing degradation of the quality of the data or new vulnerabilities that pose risk to the model environment.

(F1) Technical ML and Cybersecurity Defense Measures: Organizations must consistently look for indications of compromise even after the model has been put to production. Organizations may opt for using a variety of sensors to combine different viewpoints and modalities in order to prevent the creation of a single point of failure that could be attacked physically. It would increase the difficulty of an attacker gaining access and causing damage. They may also send logs from the sensors to the centralized security operations center for real-time monitoring. Additionally, they must keep a tight control on access management and conduct timely user access reviews to revoke access of resigned employees and ensure it follows the least privilege mechanism. Lastly, organizations must continuously keep an eye for adversarial data inputs and new vulnerabilities on the horizon. With this, the researchers provide answers to **RQ3** and **RQ4** respectively. By incorporating the above mentioned technical ML and cybersecurity defense measures and in the sequence as described in the workflow, organizations can ensure ML models are securely deployed.

6. Conclusion and future scope of research

This research provided a detailed exploration of evasion attacks on machine learning (ML) models and presented a structured approach to understanding and mitigating these threats. The findings demonstrated how adaptive evasion attacks can bypass widely-used defenses and highlighted the effectiveness of a comprehensive lifecycle-based defense strategy.

6.1. Key implications derived from the research

- **Identification of Critical Vulnerabilities** The research demonstrated how small, imperceptible perturbations can mislead ML models, leading to incorrect predictions. Widely-used defenses such as JPEG Compression were found inadequate against adaptive white-box attacks, especially those employing stronger algorithms like Projected Gradient Descent (PGD).
- **ML Model Lifecycle-Centric Defense Framework** A novel ML cybersecurity architecture was developed to embed defenses across the entire model lifecycle:
  Input Pre-Processing Stage: Techniques like Spatial Smoothing and Feature Squeezing showed effectiveness in mitigating adversarial perturbations.
  Output Post-Processing Stage: Adversarial training was used to improve model resilience by exposing the model to adversarial samples during training.
  Post-Production Stage: Continuous adversarial testing was conducted to evaluate and improve the deployed model's robustness.
- **Comprehensive Testing of Defensive Measures** The study conducted rigorous evaluations using adaptive white-box attack scenarios on a ResNet50V2 model trained with the ImageNet dataset. Techniques like JPEG Compression were systematically tested and found insufficient against stronger attacks. Combining methods, such as Spatial Smoothing and Feature Squeezing, was shown to improve robustness, achieving 100% classification accuracy in some cases under attack conditions.
- **Use of Advanced Testing Tools** The IBM Adversarial Robustness Toolbox (ART) was employed to simulate high-fidelity adversarial scenarios, offering a practical demonstration of vulnerabilities and defenses. This open-source tool proved effective in systematically evaluating attack surfaces and defensive strategies.
- **Practical Relevance for Real-World Applications** The findings emphasize the need for robust ML systems in high-stakes domains. For instance, the demonstrated adaptive attack scenarios highlight potential risks in critical image and facial recognition systems and emphasize the importance of integrating multi-layered defenses.

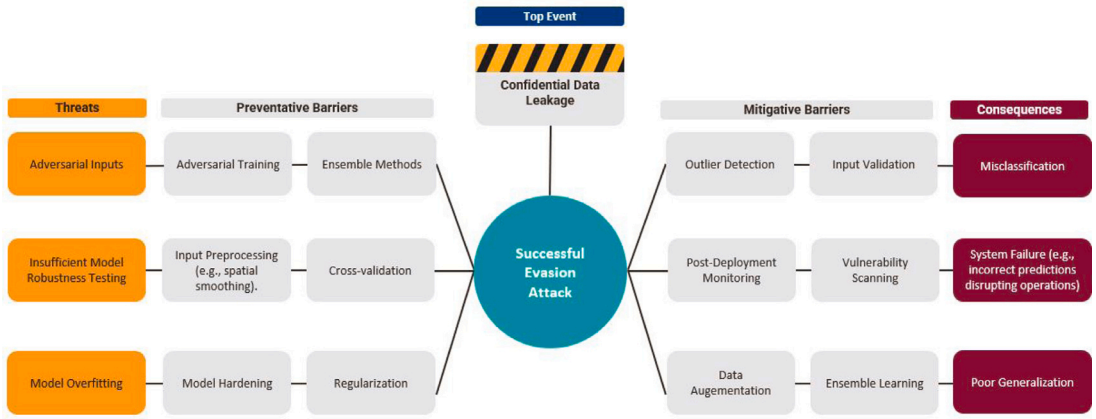


Fig. 10. Evasion attacks risk bowtie.

Table 16
Evaluation for higher perturbation strength (Eps (ϵ)).

Image	Attack algorithm	Prediction class	Confidence score	Eps (ϵ)
Image	Attack method	Prediction class	Confidence score	Eps (ϵ)
Original image	No attack algorithm	Siamese cat	99.9%	–
Adversarial image	Fast Gradient Sign Method	Indri (Lemur)	65%	6
JPEG compressed image	Fast Gradient Sign Method	Indri (Lemur)	40%	6
Spatial smoothing + Feature squeezing	Fast Gradient Sign Method	Siamese cat	92%	6
Adversarial image	Projected Gradient Descent	Ambulance	30%	8
JPEG compressed image	Projected Gradient Descent	Ambulance	18%	8
Spatial smoothing + Feature squeezing	Projected Gradient Descent	Siamese cat	88%	8

6.2. Specific future research directions

- **Cross-Domain Applicability** Extending the proposed Cyber Resilient ML Model Architecture Framework to domains such as natural language processing (NLP) and speech recognition can uncover unique challenges and enhance its generalizability.
- **Optimization of Defense Trade-Offs** Balancing robustness with computational efficiency remains critical, particularly for real-world applications requiring scalability.

By addressing these focused areas, other researchers can contribute to building secure and resilient ML systems, particularly in safety-critical applications like autonomous vehicles and healthcare.

CRediT authorship contribution statement

Raja Muthalagu: Validation, Software, Writing – review & editing. **Jasmita Malik:** Methodology, Investigation, Conceptualization, Writing – original draft. **Pranav M. Pawar:** Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Case studies of similar evasion attacks

Evasion attacks on image recognition systems have been extensively studied, with recent case studies highlighting their severe consequences and effective mitigation strategies. For example, Kotenko, Saenko, Lauta, Vasiliev, and Sadovnikov (2024) demonstrated how adversarial inputs, specifically crafted perturbations, can manipulate classifiers in critical applications such as facial recognition and autonomous vehicles. It was found that techniques like JPEG compression can significantly reduce the impact of these adversarial manipulations without compromising image quality, while adversarial training enhances robustness by incorporating adversarial examples during the training process (Chen, Zhang, Li, & Kuang, 2022). Dong et al. (2020) emphasized the vulnerability of image classifiers that undergo insufficient robustness testing, particularly under diverse adversarial conditions. Systematic testing approaches were proposed, improving classifier resilience in high-stakes domains like autonomous navigation. Model overfitting also poses a significant threat to image recognition systems, as demonstrated by Tian et al. (2024). Overfitting can lead to poor generalization on unseen datasets, which is especially problematic in real-world applications such as edge-device AI. The evasion attack bowtie diagram as shown in 10 contextualizes these findings by linking threats like adversarial inputs, insufficient testing, and overfitting to their consequences such as misclassification, system failures, and poor generalization. Preventive measures, including adversarial training, regularization, and robustness testing, combined with mitigative strategies like input validation and post-deployment monitoring, underscore the importance of layered defenses. Together, these insights from other case studies further aligns with our research by emphasizing that proactive strategies are essential to safeguarding the reliability of image recognition systems against evolving evasion attacks. By integrating preventive measures, such as adversarial training and systematic robustness testing, and mitigative steps, including input validation and ensemble learning, these approaches reinforce the same conclusions presented in our findings. They highlight how layered defense mechanisms provide a critical shield against adversarial threats, poor generalization, and operational disruptions, ensuring the integrity and resilience of models in real-world applications.

Appendix B. Additional scenarios and results

To strengthen the evaluation of the proposed defense mechanisms, experiments were conducted under additional scenarios. These scenarios included evaluations on higher perturbation strength, different

Table 17
Evaluation on different model architectures.

Model architecture	Attack algorithm	Defense	Prediction class	Confidence score
Model architecture	Attack method	Defense	Prediction class	Confidence score
ResNet50V2	Projected Gradient Descent	Spatial smoothing + Feature squeezing	Siamese cat	97%
DenseNet	Projected Gradient Descent	Spatial smoothing + Feature squeezing	Siamese cat	92%
MobileNet	Projected Gradient Descent	Spatial smoothing + Feature squeezing	Siamese cat	89%
VGG16	Projected Gradient Descent	Spatial smoothing + Feature squeezing	Siamese cat	85%

Table 18
Results on Different Datasets and Algorithms.

Dataset	Attack algorithm	Defense	Prediction class	Confidence score (%)	Eps
Dataset/Algorithm	Attack algorithm	Defense	Prediction class	Confidence score (%)	Eps
CIFAR-10	No attack algorithm	No defense	Truck (Original image)	99%	4
CIFAR-10	Fast Gradient Sign Method	JPEG compression	Truck	75%	4
CIFAR-10	Fast Gradient Sign Method	Spatial smoothing + Feature squeezing	Truck	92%	4
MNIST	No attack algorithm	No defense	Digit 7 (Original image)	99%	4
MNIST	Projected Gradient Descent	JPEG compression	Digit 9	68%	4
MNIST	Projected Gradient Descent	Spatial smoothing + Feature squeezing	Digit 7	94%	4
Tiny ImageNet	No attack algorithm	No defense	Golden retriever (Original image)	99%	4
Tiny ImageNet	Carlini–Wagner	JPEG compression	Dog	80%	4
Tiny ImageNet	Carlini–Wagner	Spatial smoothing + Feature squeezing	Golden retriever	89%	4

model architectures, datasets and algorithms. The results are summarized in Tables 16–18 and further validate the robustness of the combined defenses.

Key Observation:

As perturbation strength increased, the effectiveness of standalone defenses like JPEG Compression diminishes. However, the combined defense approach retains high accuracy and confidence even at $(\varepsilon) = 6$ and $(\varepsilon) = 8$.

Key Observation:

The combined defense as proposed in this research performs well across multiple architectures, with the highest confidence achieved on ResNet50V2 and DenseNet models.

Key Observation:

The defense Spatial Smoothing + Feature Squeezing, shows strong effectiveness across various attack algorithms and datasets, including CIFAR-10, MNIST, and Tiny ImageNet.

Data availability

Data will be made available on request.

References

Aldahdooh, A., Hamidouche, W., Fezza, S. A., & Déforges, O. (2022). Adversarial example detection for DNN models: a review and experimental comparison. *Artificial Intelligence Review*, *55*(6), 4403–4462. http://dx.doi.org/10.1007/s10462-021-10125-w.

Apruzzese, G., Andreolini, M., Marchetti, M., Venturi, A., & Colajanni, M. (2020). Deep reinforcement adversarial learning against botnet evasion attacks. *IEEE Transactions on Network and Service Management*, *17*(4), 1975–1987. http://dx.doi.org/10.1109/TNSM.2020.3031843.

Athalye, A., Carlini, N., & Wagner, D. A. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. URL https://api.semanticscholar.org/CorpusID:3310672.

Athalye, A., Engstrom, L., Ilyas, A., & Kwok, K. (2018). Synthesizing robust adversarial examples. In *Proceedings of the 35th international conference on machine learning*. Stockholm, Sweden.

Ayub, M. A., Johnson, W. A., Talbert, D. A., & Siraj, A. (2020). Model evasion attack on intrusion detection systems using adversarial machine learning. In *2020 54th annual conference on information sciences and systems* (pp. 1–6). http://dx.doi.org/10.1109/CISS48834.2020.1570617116.

Bai, T., Luo, J., Zhao, J., Wen, B., & Wang, Q. (2021). Recent advances in adversarial training for adversarial robustness. URL https://api.semanticscholar.org/CorpusID:231749855.

Bak, M., Madai, V. I., Fritzsche, M.-C., Mayrhofer, M. T., & McLennan, S. (2022). You can't have AI both ways: Balancing health data privacy and access fairly. *Frontiers in Genetics*, *13*, URL https://www.frontiersin.org/articles/10.3389/fgene.2022.929453.

Ballet, V., Renard, X., Aigrain, J., Laugel, T., Frossard, P., & Detyniecki, M. (2019). Imperceptible adversarial attacks on tabular data. arXiv:1911.03274.

Bountakas, P., Zarras, A., Lekidis, A., & Xenakis, C. (2023). Defense strategies for adversarial machine learning: A survey. *Computer Science Review*, *49*, Article 100573. http://dx.doi.org/10.1016/j.cosrev.2023.100573, URL https://www.sciencedirect.com/science/article/pii/S1574013723000400.

Brendel, W., Rauber, J., & Bethge, M. (2018). Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. arXiv:1712.04248.

Brown, T. B., Mané, D., Roy, A., Abadi, M., & Gilmer, J. (2018). Adversarial patch. arXiv:1712.09665.

Calzavara, S., Ferrara, P., & Lucchese, C. (2023). Certifying machine learning models against evasion attacks by program analysis. http://dx.doi.org/10.3233/JCS-210133.

Cao, H., & Xue, M. (2023). Adversarial training for better robustness. In S. I. Lopes, P. Fraga-Lamas, T. M. Fernández-Caramés, B. R. Dawadi, D. B. Rawat, & S. Shakya (Eds.), *Smart technologies for sustainable and resilient ecosystems* (pp. 75–84). Cham: Springer Nature Switzerland.

Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. arXiv:1608.04644.

Carlini, N., & Wagner, D. (2018). Audio adversarial examples: Targeted attacks on speech-to-text. arXiv:1801.01944.

Chen, Y., Zhang, M., Li, J., & Kuang, X. (2022). Adversarial attacks and defenses in image classification: A practical perspective. In *2022 7th international conference on image, vision and computing* (pp. 424–430). http://dx.doi.org/10.1109/ICIVC55077.2022.9886997.

Cohen, J. M., Rosenfeld, E., & Kolter, J. Z. (2019). Certified adversarial robustness via randomized smoothing. arXiv:1902.02918.

Das, N., Shanbhogue, M., Chen, S.-T., Hohman, F., Chen, L., Kounavis, M. E., et al. (2017). Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression. arXiv:1705.02900.

Dong, Y., Fu, Q.-A., Yang, X., Pang, T., Su, H., Xiao, Z., et al. (2020). Benchmarking adversarial robustness on image classification. In *2020 IEEE/CVF conference on computer vision and pattern recognition* (pp. 318–328). http://dx.doi.org/10.1109/CVPR42600.2020.00040.

Dziugaite, G. K., Ghahramani, Z., & Roy, D. M. (2016). A study of the effect of JPG compression on adversarial images. arXiv:1608.00853.

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., et al. (2018). Robust physical-world attacks on deep learning visual classification. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 1625–1634). IEEE.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. CoRR abs/1412.6572. URL https://api.semanticscholar.org/CorpusID:6706414.

Guo, C., Rana, M., Cisse, M., & van der Maaten, L. (2018). Countering adversarial images using input transformations. arXiv:1711.00117.

Inkawhich, N., Inkawhich, M., Chen, Y., & Li, H. (2018). Adversarial attacks for optical flow-based action recognition classifiers. arXiv:1811.11875.

Jing, P., Tang, Q., Du, Y., Xue, L., Luo, X., Wang, T., et al. (2021). Too good to be safe: Tricking lane detection in autonomous driving with crafted perturbations. In *30th USeNIX security symposium (USeNIX security 21)* (pp. 3237–3254). USENIX Association.

Kotenko, I., Saenko, I., Lauta, O., Vasiliev, N., & Sadovnikov, V. (2024). An approach to countering adversarial attacks on image recognition based on JPEG-compression and neural-cleanse. In *2024 IEEE ural-siberian conference on biomedical engineering, radioelectronics and information technology* (pp. 076–079). http://dx.doi.org/10.1109/USBEREIT61901.2024.10584049.

Kumar, A., Agarwal, C., Srinivas, S., Li, A. J., Feizi, S., & Lakkaraju, H. (2024). Certifying LLM safety against adversarial prompting. arXiv:2309.02705.

Kunwar, S. (2021). Strategies in JPEG compression using convolutional neural network (CNN). arXiv:2112.04500.

Kurakin, A., Goodfellow, I., & Bengio, S. (2017). Adversarial examples in the physical world. arXiv:1607.02533.

Li, D., Cui, S., Li, Y., Xu, J., Xiao, F., & Xu, S. (2024). PAD: Towards principled adversarial malware detection against evasion attacks. *IEEE Transactions on Dependable and Secure Computing*, *21*(2), 920–936. http://dx.doi.org/10.1109/TDSC.2023.3265665.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2019). Towards deep learning models resistant to adversarial attacks. arXiv:1706.06083.

McClintick, K. W., Harer, J., Flowers, B., Headley, W. C., & Wyglinski, A. M. (2022). Countering physical eavesdropper evasion with adversarial training. *IEEE Open Journal of the Communications Society*, *3*, 1820–1833. http://dx.doi.org/10.1109/OJCOMS.2022.3213371.

Nayak, G., & Friedland, G. (2023). Deep layers beware: Unraveling the surprising benefits of jpeg compression for image classification pre-processing. In *2023 IEEE international symposium on multimedia* (pp. 182–185). http://dx.doi.org/10.1109/ISM59092.2023.00033.

Pang, Y., Cheng, S., Hu, J., & Liu, Y. (2021). Evaluating the robustness of Bayesian neural networks against different types of attacks. arXiv:2106.09223.

Papernot, N., & McDaniel, P. (2017). Extending defensive distillation. arXiv:1705.05264.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2017). Practical black-box attacks against machine learning. arXiv:1602.02697.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., & Swami, A. (2016). The limitations of deep learning in adversarial settings. In *Security and privacy (euroS&p), 2016 IEEE European symposium on* (pp. 372–387). IEEE.

Papers with code - RNN-test: Towards adversarial testing for recurrent neural network systems — paperswithcode.com. (2024). https://paperswithcode.com/paper/rnn-test-adversarial-testing-framework-for. [Accessed 10 May 2024].

Park, N., & Kim, S. (2022). Blurs behave like ensembles: Spatial smoothings to improve accuracy, uncertainty, and robustness. arXiv:2105.12639.

Pitropakis, N., Panaousis, E., Giannetsos, T., Anastasiadis, E., & Loukas, G. (2019). A taxonomy and survey of attacks against machine learning. *Computer Science Review*, *34*, Article 100199. http://dx.doi.org/10.1016/j.cosrev.2019.100199.

Prakash, A., Moran, N., Garber, S., DiLillo, A., & Storer, J. (2018). Deflecting adversarial attacks with pixel deflection. arXiv:1801.08926.

Praveena, M., Madhumitha, S., Menakadevi, J., & Lalith Akkash, V. (2023). A comprehensive taxonomy of adversarial attacks on machine learning in IoT application. In *2023 7th international conference on electronics, communication and aerospace technology* (pp. 1398–1403). http://dx.doi.org/10.1109/ICECA58529.2023.10394928.

Price II, W. N. (2019). Risks and remedies for artificial intelligence in health care. Brookings Report https://www.brookings.edu/research/risks-and-remedies-for-artificial-intelligence-in-health-care/.

Qin, Y., Carlini, N., Cottrell, G. W., Goodfellow, I., & Xie, C. (2019). Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. arXiv preprint arXiv:1903.10346. URL https://arxiv.org/abs/1903.10346.

Rahmati, A., Moosavi-Dezfooli, S.-M., Frossard, P., & Dai, H. (2020). GeoDA: a geometric framework for black-box adversarial attacks. arXiv:2003.06468.

Rochac, J. F. R., Liang, L., Zhang, N., & Oladunni, T. (2019). A Gaussian data augmentation technique on highly dimensional, limited labeled data for multiclass classification using deep learning. In *2019 tenth international conference on intelligent control and information processing* (pp. 145–151). http://dx.doi.org/10.1109/ICICIP47338.2019.9012197.

Ross, A. S., & Doshi-Velez, F. (2017). Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. arXiv:1711.09404.

Roth, K., Kilcher, Y., & Hofmann, T. (2019). The odds are odd: A statistical test for detecting adversarial examples. arXiv:1902.04818.

Sharma, Y., & Chen, P.-Y. (2018). Bypassing feature squeezing by increasing adversary strength. arXiv:1803.09868.

Shaukat, K., Luo, S., & Varadharajan, V. (2022). A novel method for improving the robustness of deep learning-based malware detectors against adversarial attacks. *Engineering Applications of Artificial Intelligence*, *116*, Article 105461. http://dx.doi.org/10.1016/j.engappai.2022.105461, URL https://www.sciencedirect.com/science/article/pii/S0952197622004511.

Tian, P., Poreddy, S., Danda, C., Gowrineni, C., Wu, Y., & Liao, W. (2024). Evaluating impact of image transformations on adversarial examples. *IEEE Access*, 1. http://dx.doi.org/10.1109/ACCESS.2024.3487479.

Tian, Y., & Zhang, Y. (2022). A comprehensive survey on regularization strategies in machine learning. *Information Fusion*, *80*, 146–166. http://dx.doi.org/10.1016/j.inffus.2021.11.005, URL https://www.sciencedirect.com/science/article/pii/S156625352100230X.

Wang, Z., & Chen (2023). Robust recommendation with adversarial Gaussian data augmentation. In *Proceedings of the ACM web conference 2023* (pp. 897–905). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3543507.3583273.

Wang, Y., Sun, T., Li, S., Yuan, X., Ni, W., Hossain, E., et al. (2023). Adversarial attacks and defenses in machine learning-powered networks: A contemporary survey. arXiv:2303.06302.

Wiyatno, R., & Xu, A. (2018). Maximal Jacobian-based saliency map attack. arXiv:1808.07945.

Wiyatno, R. R., & Xu, A. (2019). Physical adversarial textures that fool visual object tracking. arXiv:1904.11042.

Xu, W., Evans, D., & Qi, Y. (2018). Feature squeezing: Detecting adversarial examples in deep neural networks. In *NDSS 2018*, *Proceedings 2018 network and distributed system security symposium*. Internet Society, http://dx.doi.org/10.14722/ndss.2018.23198.

Xu, Y., Li, D., Li, Q., & Xu, S. (2024). Malware evasion attacks against IoT and other devices: An empirical study. *Tsinghua Science and Technology*, *29*(1), 127–142. http://dx.doi.org/10.26599/TST.2023.9010005.

Yamamura, K., Sato, H., Tateiwa, N., Hata, N., Mitsutake, T., Oe, I., et al. (2022). Diversified adversarial attacks based on conjugate gradient method. arXiv:2206.09628.

Yang, C.-H., Maina, B. M., Cheng, S.-M., & Lee, H.-M. (2024). An adversarial attack on artificial intelligence malware detection in consumer internet of things. *IEEE Consumer Electronics Magazine*, 1–12. http://dx.doi.org/10.1109/MCE.2024.3482700.

Zhao, C., Li, H., Wang, D., & Liu, R. (2023). Adversarial example detection for deep neural networks: A review. In *2023 8th international conference on data science in cyberspace* (pp. 468–475). http://dx.doi.org/10.1109/DSC59305.2023.00074.

Raja Muthalagu received the B.E. and M.E. degrees in electronics and communication engineering from Anna University, Chennai, in 2005 and 2007, respectively, and the Ph.D. degree in wireless communication from the National Institute of Technology (NIT), Tiruchirappalli, India, in 2014. He was a Postdoctoral Research Fellow with the Air Traffic Management Research Institute, Nanyang Technological University, Singapore, from 2014 to 2015. He is currently an Associate Professor with the Birla Institute of Technology and Science, Pilani, Dubai Campus, Dubai, United Arab Emirates. He has published more than 55 research articles in reputed journals and conferences. His current research interests include wireless communications, signal processing, aeronautical communications, cybersecurity, applying intelligent techniques for detecting and mitigating a security attack in the IoT, SDN, and other computer networks. He was a recipient of the Canadian Commonwealth Scholarship Award 2010 for Graduate Student Exchange Program from the Department of Electrical and Computer Engineering, University of Saskatchewan, Saskatoon, SK, Canada.

Jasmita Malik is an Information Security and Data Privacy practitioner with over 8+ years of experience delivering and advising executive leadership on large scale Business Continuity and ICT Recovery, Cybersecurity, Data Protection and Resilience programs. She has worked across industries such as fintech, banking, retail, real-estate and hospitality. She is currently serving as the Information Security Manager at Majid Al Futtaim (MAF), a leading multi-national lifestyle conglomerate in the EMEA region. In MAF, Jasmita is in-charge of the enterprise information security and IT risk management program. She is passionate about enabling organizations in taking a holistic approach to security by embedding risk management practices with ongoing business processes and working towards overall technological resilience. Over the years, Jasmita has received several accolades for accelerating the IT and cybersecurity posture of different organizations. She was awarded the Cyber Strategist award by CXOInsights Middle-east and was shortlisted for Women in IT Awards in Asia. She is an active researcher on new-age AI and ML-driven cybersecurity and incident response solutions. On the academic front, Jasmita is currently pursuing Ph.D. in Computer Science and Information Systems in the area of AI Assurance and Cybersecurity from the Birla Institute of Technology and Science, Dubai Campus, UAE. She received the B.E. degree in computer systems engineering from The Middlesex University, Dubai, UAE and completed the M.Sc. degree in network security with a focus in cybersecurity from the Heriot Watt University, Dubai, UAE. Jasmita holds several industry-based certifications in cybersecurity, risk management and business continuity.

Pranav M. Pawar received the degree in computer engineering from Dr. Babasaheb Ambedkar Technological University, Maharashtra, India, in 2005, the master's degree in computer engineering from Pune University, in 2007, and the Ph.D. degree in wireless communication from Aalborg University, Denmark, in 2016. He is an IBM DB2 and an IBM RAD certified professional and completed NPTEL certification in different subjects. From 2006 to 2007, he was a System Executive with POS-IPC, Pune, India. He was an Associate Professor with the Department of Information Technology, STES's Smt. Kashibai Navale College of Engineering, Pune, from 2008 to 2018; and MIT ADT University, Pune, from 2018 to 2019. He is currently an Assistant Professor with the Department of Computer Science, Birla Institute of Technology and Science (BITS), Dubai. Before joining BITS, he was a Postdoctoral Fellow with Bar-Ilan University, Israel, from March 2019 to October 2020, in the areas of wireless communication and deep leaning. He received the Recognition from Infosys Technologies Ltd., for contribution in Campus Connect Program and different funding for research and attending conferences at international level. He has published more than 40 papers at national and international levels. His research interests include energy-efficient MAC for WSN, QoS in WSN, wireless security, green technology, computer architecture, database management systems, and bioinformatics. His Ph.D. thesis received nomination for the Best Thesis Award from Aalborg University. He was a recipient of the Outstanding Postdoctoral Fellowship from the Israel Planning and Budgeting Committee.