Name: Shezonia Idrees
Rollno: BITF22M044
Subject: Information Security
Submitted to: Sir Huzaifa Nazir

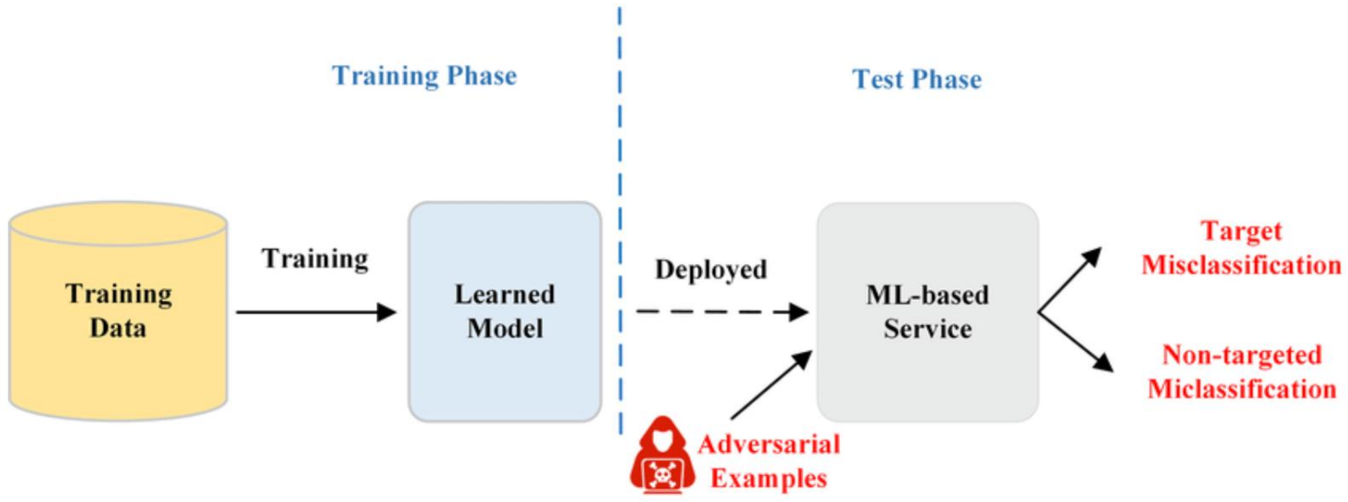
<u>Title: Adversarial ML Attacks: How Hackers Manipulate ML Models</u>

REFERENCES

1. What Is an Adversarial AI Attack- *paloaltonetworks*
2. Adversarial machine learning threat analysis and remediation in Open Radio Access Network (O-RAN)- *sciencedirect*
3. A System-Driven Taxonomy of Attacks and Defenses in Adversarial Machine Learning- *nih*
4. Adversarial Attacks in Machine Learning Based Access Control- *ceur*
5. Adversarial Attacks- *sentinelone*
6. Adversarial Machine Learning in Cybersecurity: A Review on Defending Against AI-Driven Attacks- *researchgate*
7. Adversarial Machine Learning: Techniques, Risks, and Applications-*birchwoodu*
8. Adversarial machine learning- *Wikipedia*
9. What are adversarial attacks in machine learning and how to prevent them- *labellerr*
10. The Threat of Adversarial AI- *wiz*

What is Adversarial Machine Learning?

Adversarial Machine Learning is designed in a way that it deceives the models by providing misleading inputs, which are also known as adversarial examples, to induce incorrect inferences. These attacks modify data in ways that appear harmless to humans but perplex machine learning models, undermining their accuracy. Adversarial attacks can cause significant errors in applications like image classification, speech recognition, and cybersecurity. Take, for instance, the alteration of a stop sign image can misinform an autonomous vehicle, which can lead to wrong interpretation and potentially cause accidents. The consequences of adversarial attacks are profound and far-reaching in critical systems within healthcare, finance, and autonomous transportation, as ML is deeply integrated in these sectors and is being widely used nowadays.



Origins of Adversarial Machine Learning

Adversarial Machine Learning started as early as the 2000s, when scientists realized that ML algorithms, particularly neural networks, were manipulable by minute variations in input data.

One of the first key papers arrived in 2004, when Dalvi et al. demonstrated how spam filters could be manipulated by making minor modifications to spam emails, without changing their intent. This was one of the first real-world demonstrations of adversarial attacks. However, the field only came into prominence after a landmark work by Szegedy et al. in 2013, where they showed that the addition of small, imperceptible noise to an image might lead a deep neural network to misclassify it entirely. This ignited the contemporary trend in adversarial ML.

Major Milestones in Adversarial Research

- **2013:** Szegedy et al. propose the concept of adversarial examples in deep learning. It was an eye-opener to the AI community.

- **2014:** Goodfellow et al. introduce the Fast Gradient Sign Method (FGSM), a simple and efficient way to produce adversarial examples. This facilitated attacks to be easier to reproduce.

- **2015–2017:** Scientists delve into transferability, where adversarial examples generated for one model deceive other models as well. This brings into the limelight the deep weaknesses of most AI systems.

- **2018 and beyond:** Adversarial attacks expand to black-box models (where attackers are unaware of the model's internal details), real-world objects (such as road signs) and even physical environments, increasing the risk factor.

- **Present Time:** Adversarial ML is currently a central topic in AI security, with mounting interest in defenses such as adversarial training, robust optimization and certified defenses.

Evolution of Attack Methods

In the early days, adversarial attacks were primarily straightforward and white-box, i.e., attackers required complete model access. They used to introduce minor noise to deceive the system. Present times have seen attacks evolve significantly:

- **Black-box attacks:** These function without having knowledge about the model's specifics, making them more real-world relevant.

- **Physical-world attacks:** This comprises modifications of items such as road signs or glasses that deceive facial recognition systems.

- **Adaptive attacks:** They are engineered to circumvent certain defense measures, showcasing that even secured systems are not immune.

The transition from mere digital stunts to sophisticated real-world manipulation demonstrates that adversarial attacks are on the rise and becoming increasingly difficult to prevent

What is an Adversarial Example?

The data input that has been manipulated with the intention to deceive a machine learning model is called an Adversarial Example. These adversarial examples are almost indistinguishable from legitimate inputs to human observers; however, they are designed to influence the model's prediction accuracy. For example, a human would still recognize a slightly modified image of a dog but it may be classified by a deep learning model as a cat.

There are various techniques by which adversarial examples can be generated. The major challenge in adversarial machine learning lies in the ability to produce minimal perturbations to inputs, to make sure they go unnoticed by users but still distort machine learning outputs successfully.

Types of Adversarial Attacks on Machine Learning

Adversarial attacks can be classified by when they occur in the machine learning lifecycle and by the attacker's level of knowledge about the model. White-box attacks happen when the attacker has full access to the model's architecture and parameters. Black-box attacks are more common and involve the attacker having limited or no knowledge of the model's internal workings, instead relying on querying the model and observing its outputs.

The main types of attacks include:

a) **Poisoning Attacks:**

“Training time" attacks where malicious data is injected into the training dataset. This corrupts the model's learning process, leading to degraded accuracy or deliberate vulnerabilities. For example, an attacker could inject mislabeled spam emails into a dataset, teaching the model to ignore actual spam in the future.

b) **Evasion Attacks:**

Attacks that occur when input data is manipulated to deceive an already trained AI model. For example, adding invisible changes to an image can cause an AI system to misidentify it. Evasion attacks are categorized into two subtypes:

i. Nontargeted attacks: In nontargeted evasion attacks, the goal is to make the AI model produce any incorrect output, regardless of the production. For example, an attacker might manipulate a stop sign image so that the AI system fails to recognize it as a stop sign, potentially leading to dangerous road situations.
ii. Targeted attacks: The attacker aims to force the AI model to produce a specific, predefined, incorrect output, such as classifying a benign object as harmful.

c) **Model Extraction Attacks:** An attacker repeatedly queries a deployed model to create a replica of its functionality, thereby compromising the model's security and intellectual property without ever accessing its code.

d) **Inference-Related Attacks:** These attacks exploit a model's output to extract sensitive information or learn about its training data. This includes model inversion, which reconstructs sensitive data from the model's outputs, and membership inference, which determines if a specific data point was used in the training set.

e) **Transfer Attacks:** These attacks involve creating adversarial examples for one AI system and adapting them to attack other, different AI models.

<u>How Hackers Attacks Model?</u>

First, an adversary will try to find your ML model’s core weaknesses. They test their limits, find flaws, and enter invalid inputs to see how these systems react.

Attackers probe your models the same way they probe your network. They test against different changes and reactions the models give, based on what inputs they supply. And when they find the trigger switch or something they can flip, they change their attack strategy. How they fool ML models or break past default limits will depend on them.

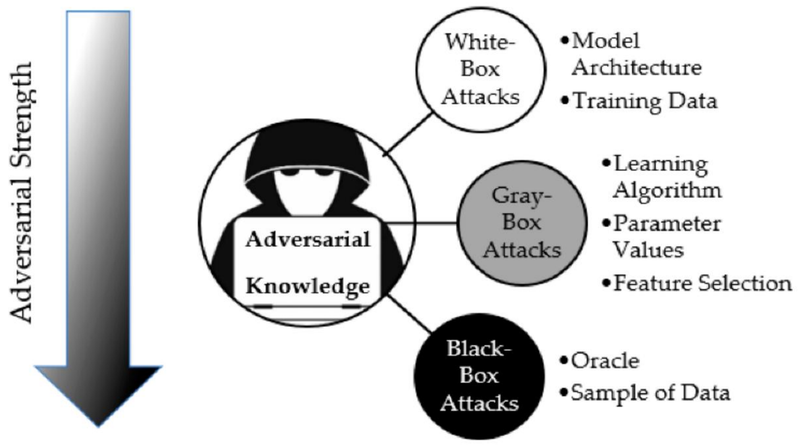
Some adversaries can even reverse engineer programs to find exploits and target them. Before they even launch an attack, they study the target victim/system and deploy various inputs to see how these systems behave against them. They basically test the sensitivity of your machine learning models.
The general attack workflow mirrors what you see daily:

- Reconnaissance maps outputs and rate limits
- Construction runs optimization to craft malicious inputs
- Exploitation sends the payload
- Adaptation refines the attack based on your response

Traditional monitoring tools miss these moves because the packets, images, or log lines look legitimate to humans.

Threat Model Taxonomy

Threat models categorize the capabilities and knowledge levels of adversaries. They specify the amount of access hackers have to the model and guide the design of defensive strategies.

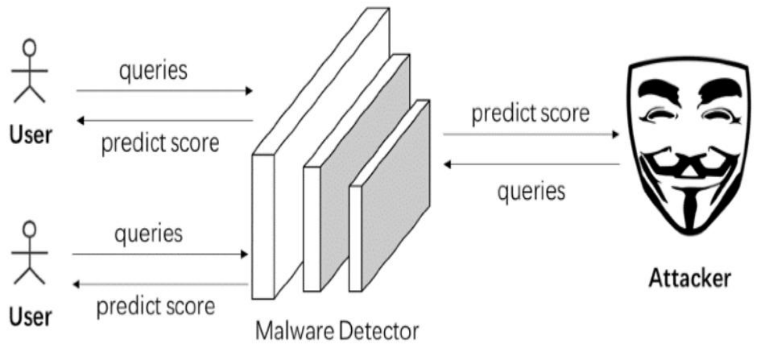


White-Box Attacks

White-box attacks need the attacker to completely grasp ML system, such as its architecture, weights, training data, as well as gradients. This enables extremely focused and optimized adversarial example creation. These attacks are powerful but assume a strong threat mode. The hacker has full usage to the model's framework, training data , as well as parameters. The owner first trains a model and inserts a backdoor using crafted samples. The attacker then uses this compromised model to perform regularization and fine-tuning, preserving the backdoor while ensuring the model performs well on clean data. This demonstrates how deep internal knowledge of the model enables sophisticated manipulation.

Black-Box Attacks

Black-box assaults presume no inside understanding of the target model. An attacker can only observe inputs and outputs, such as class labels or confidence scores. Black-box attackers often use transferability The attacker sends queries and observes outputs to craft adversarial inputs without knowing the model's internal details.



Gray-Box Attacks

Gray-box assaults presuppose a partial understanding of the system. This may include information like the model type, feature space, or access to partial training data. These assaults fall between white-box & black-box as well as more feasible in real-world circumstances.

Defense Strategies Against Adversarial Attacks

The ML and DL models are seriously threatened by adversarial attacks, particularly in the vital fields such as cybersecurity, autonomous vehicles, and healthcare [19]. In order to stop these threats, several defines strategies have been suggested. The four most important defines categories are:

Adversarial Training

Adversarial training happens to be one of the most commonly used defences against adversary attacks. It is a technique in which adversarial examples are created- inputs are artificially scrambled in order to trick the model, and then used in the training process. So, the model becomes trained to identify and reject malicious inputs, thereby strengthening its robustness. However, this strategy is computationally intensive and may not work well for all sorts of adversarial assaults.

- Includes adversarial cases in the training set.
- Enables the model to understand and classify perturbed inputs correctly.
- Improves robustness against known attacks.
- Increases training time and may not protect against unseen attacks.

Defensive Distillation

Defensive distillation is a strategy that trains the model in two steps to limit its sensitivity to modest input changes. To create soft labels (probability distributions), the first model gets trained applying a high-temperature SoftMax method. These soft labels are then used to train a second model, which learns to make predictions in a smoother, more generalizable way. This strategy makes it harder for attackers to create successful adversarial samples.

- Involves training a model to produce soft labels instead of hard outputs.
- Reduces model sensitivity to small perturbations.
- Helps to counter gradient-based attacks.
- May still be vulnerable to sophisticated or adaptive adversaries.

Input Preprocessing Techniques

Input pre-processing approaches try to neutralize adversarial perturbations prior reaching the model. These techniques transform the input data in ways that potentially remove or reduce the effects of adversarial noise. Common methods include image resizing, compression, denoising, and feature squeezing. While generally simple and model-agnostic, pre-processing can sometimes affect model accuracy on clean data and may be circumvented by adaptive attacks.

- Applies transformations like denoising, resizing, or compression.
- Feature squeezing reduces input precision (colour depth).
- Easy to implement and compatible with most models.
- Can degrade accuracy and be bypassed by adaptive attackers.

Model Architecture and Certification-Based Approaches

Some defines strategies focus on modifying the model architecture or using mathematical guarantees to improve robustness. These include using architectures designed for stability (like Bayesian neural networks) or certified defences that guarantee the models behaviour under Adversarial perturbations are classified into particular categories. Though highly effective in theory, these techniques may be difficult and computationally demanding, and they may affect model performance.

How to Defend Against Adversarial Machine Learning Attacks

Attackers probe your models the same way they probe your network. They find the weakest link and exploit it. Your ML models are under attack right now, and traditional security tools generally miss these threats entirely.

Defending ML systems requires the same defense-in-depth approach you use everywhere else: harden during development, detect attacks in real-time, and respond before damage spreads.
The difference? Adversarial attacks on ML target the brain of your system, not just the gates.
Your data scientists, ML engineers, and SOC analysts need to work as one team with shared threat models and response procedures. When an adversarial attack hits your fraud detection model, it's a security incident that demands the same urgency as ransomware.

1. Proactive Defense Strategies

Building robust defenses starts during model development. Adversarial training stops evasion attacks before they start by adding crafted perturbations to every training batch using multi-step PGD methods.
Your model learns to keep decisions stable when inputs get manipulated.

The trade-off is real:

- Robust accuracy goes up
- Clean accuracy can drop
- Training takes longer

Start small with perturbation budgets and increase gradually.
Data poisoning works because your training pipelines trust what they consume.

Prevent data poisoning attacks by:

- Validating every input with schema checks and outlier filters
- Recording data provenance before anything hits your optimizer
- Quarantining crowd-sourced samples until human review confirms they're clean.

Architecture choices matter for defense. Simpler networks with proper regularization drop the non-robust features attackers love to exploit. Ensemble methods force attackers to fool multiple decision boundaries simultaneously. For your highest-value models, certified robustness techniques provide formal guarantees, use them when the compute cost is justified.

Third-party model weights are attack vectors. Sign every artifact, store cryptographic hashes, and verify them in your CI/CD pipeline. If a supplier can't provide checksums, don't deploy their model. Build diversity into your defense by rotating training seeds, perturbation strengths, and data splits regularly. An attacker who succeeds against one model snapshot often fails against the next version.

2. Detection and Response Capabilities

Even hardened models face adaptive attackers, making real-time detection essential.
Monitor every request to your ML endpoints. This means you should track input distributions, embedding drift, and confidence score patterns. Sharp shifts can indicate active probing.

Inline detectors act as your first line of defense, catching attacks before they reach your model. For example, statistical tests can flag inputs that fall outside the model's expected patterns, while ensemble disagreement, when multiple models produce conflicting predictions, can signal something suspicious. Because attackers can adapt to a single defense, it's best to run several detection methods in parallel.

Once a detector triggers, your response should be automatic. That can mean throttling the suspicious client, isolating questionable requests, or switching to a more robust backup model. Capture everything, raw inputs, model outputs, and detector scores, so your team has the evidence needed for investigation.
From there, handle the incident as you would any other security breach.
Follow a runbook that includes collecting evidence, assessing impact, rolling back to a trusted model version, and retraining on clean data.

Speed is critical: the longer a compromised model runs, the more damage it can cause. Treat your detection-to-containment time the same way you would for ransomware, because a poisoned or manipulated model can create cascading business failures.

3. Enterprise ML Security Architecture

Protecting machine learning at the enterprise level means treating it like any other critical system, integrating defenses into your existing security stack, closing blind spots, and making attacks visible before they cause real business damage.

Start by validating data at every entry point in the pipeline. Enforce strict format checks, verify where the data came from, and use signed datasets before anything reaches long-term storage.

Protect your model registry the same way you protect code: require signed model files, track their history, and only allow deployment after passing robustness tests. At runtime, monitor model servers alongside your other workloads.

Collect process, network, and system activity, and feed those metrics into your central security console so analysts see ML anomalies alongside endpoint and network alerts. Keep an up-to-date inventory of all models with clear owners, risk ratings, and robustness scores, and review these during change-control meetings just as you would patch levels. Make adversarial testing a hard requirement before anything goes live.

Clear role separation keeps the system manageable. For example, CISOs can own the risk and set policy, SOC managers are in charge of integrating detection into daily workflows, and analysts tune alerts and investigate incidents.

Challenges in Detecting Adversarial Attacks

You may experience some challenges in detecting adversarial attacks, such as minimal distortions. These are subtle and unnoticeable signs of attacks incoming. These kinds of attacks make minimal changes to the original inputs, which makes them difficult to detect using simple filters and anomaly detection. From the outside, they look very normal.

Then you have the second problem of exploiting non-linearities. Deep neural networks can have high dimensional and very complex decision boundaries. Adversaries can exploit sharp regions in these boundaries, where small inputs and manipulating them can cause drastic changes in larger outputs, which can lead to misclassification.

Adversarial attacks that are used to target one model can be transferred over and used against other different models, even if they use a different architecture or training data. Black box attacks are becoming very common. And then we have the issue of circumventing defenses.

No universal defense will work for all models, since models can change and adapt. We also have adaptive attacks, which means adversaries can bypass specific defenses. They can neutralize common defensive techniques, like input sanitization and defensive distillation.

Targeted attacks can be more specific and may also cause random misclassification sometimes. You may also deal with high false positive rates depending on the detection methods and techniques you use. Some boundaries between naturally occurring attacks versus ones launched by adversaries can be blurred depending on the data you are dealing with. You also have to deal with degrading clean inputs, which can trigger incorrect detection and decision making, thus reducing the reliability of your security solutions.

Real-World Examples of Adversarial Attacks

Documented incidents demonstrate how adversarial attacks move from academic research to active exploitation in enterprise environments.

- **Tesla Autopilot Manipulation (2019):** Security researchers demonstrated that small stickers placed on road signs could cause Tesla's autopilot system to misread speed limits, potentially causing the vehicle to accelerate inappropriately. The attack exploited the computer vision system's reliance on specific visual patterns, showing how physical adversarial examples can impact safety-critical systems.

- **Microsoft's Tay Chatbot (2016):** Within 24 hours of launch, coordinated users manipulated Microsoft's AI chatbot through carefully crafted conversational inputs that gradually shifted its responses toward inappropriate content. This demonstrated how continuous learning systems can be corrupted through coordinated adversarial feedback.

- **ProofPoint Email Security Bypass (2020):** Attackers discovered they could evade enterprise email security by making minimal modifications to malicious attachments. By changing file headers and embedding patterns, they created variants that looked identical to security analysts but bypassed ML-based threat detection systems.

- **Chinese Traffic Camera Evasion (2021):** Researchers showed that strategically placed infrared LEDs could fool facial recognition systems used in traffic enforcement. The technique made license plates unreadable to automated systems while remaining clearly visible to human traffic officers.

- **Credit Card Fraud Detection Failures (2022):** Financial institutions reported sophisticated attacks where criminals gradually trained fraud detection systems to accept increasingly risky transaction patterns. By starting with borderline-legitimate transactions and slowly escalating, attackers established new baseline behavior that allowed larger fraudulent transactions to pass undetected.

These examples highlight a critical pattern: successful adversarial attacks often exploit the gap between human perception and machine learning model decision-making, allowing malicious activity to hide in plain sight.

Conclusion

Adversarial Machine learning brings significant threats to the model's integrity while offering opportunities for strengthening robustness, security, and fairness. Along with a continuous growth of deceptive attacks, researchers and organizations must advance in line with defense approaches, such as adversarial training and resilient optimization.