the-tech-trend.com

Understanding AI in Cybersecurity and AI Security: Defense Methods for Adversarial Attacks and Privacy Issues in Secure AI (UCSAISec-04)

Arash Habibi Lashkari

13–17 minutes

As AI systems become more common in the real world, powering self-driving cars, medical tools, and surveillance, they've also become exposed to new kinds of threats. One serious threat is the adversarial attack, where small changes to input data mislead a model into making incorrect predictions. These changes are usually invisible to humans but can completely alter the model's output. For example, an image classifier might mistake a stop sign for a speed limit sign after just a few pixel tweaks.

This article, as the fourth installment in the UCSAISec series from our Understanding Cybersecurity Series (UCS) program at the Behaviour-Centric Cybersecurity Center (BCCC), explores the importance of addressing failures in the AI system, particularly in environments where reliability is critical. It highlights emerging techniques designed to make models more resilient to manipulation, safeguard the data they rely on, and enhance transparency in decision-making.

1. Model Robustness

Defending against adversarial threats starts with robustness. A robust model continues to perform accurately even when inputs are slightly distorted, whether from random noise, poor lighting, or attempts to deceive it. For example, an autonomous vehicle's vision system should still recognize a pedestrian, whether it's sunny, foggy, or if someone added a sticker to a street sign.

Achieving this requires the model to learn meaningful patterns in the data rather than relying on fragile shortcuts. This is often improved through adversarial training, where models are exposed to manipulated inputs during training and learn to resist them. Regularization techniques are also used to reduce over-sensitivity to specific features.

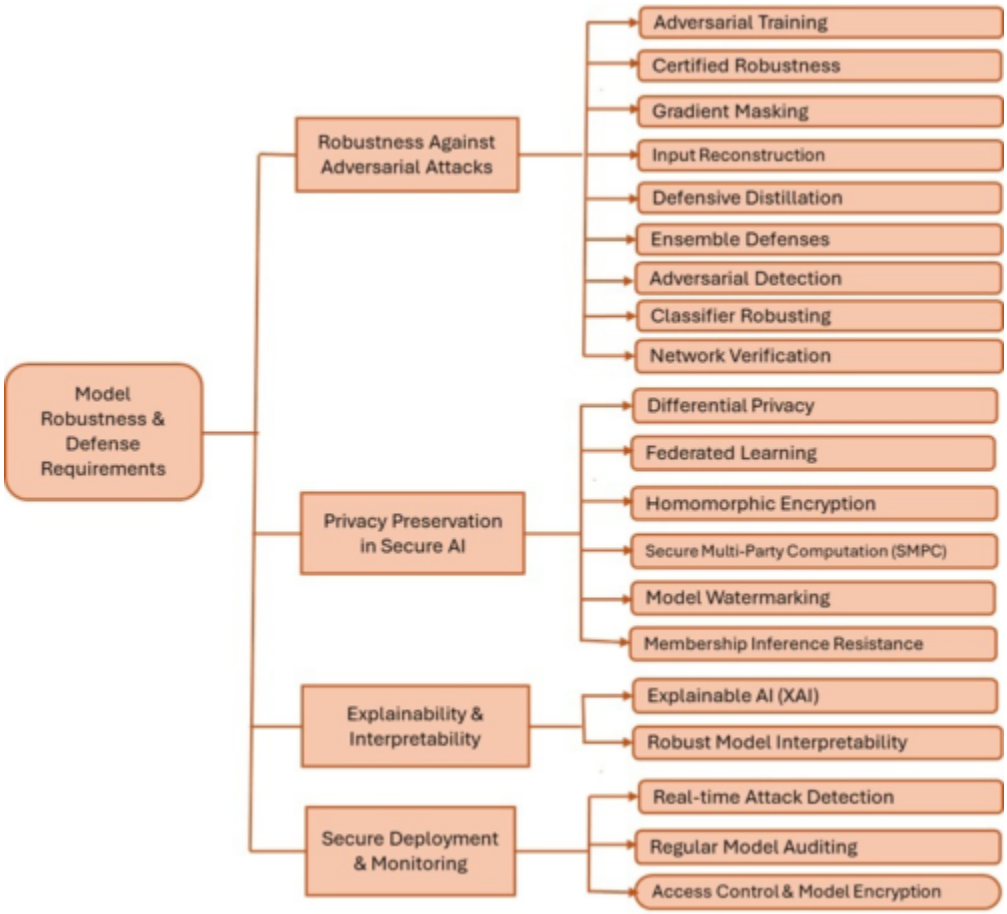

Figure 1: Model robustness and adversarial defense

requirements

2. Adversarial Samples & How They're Crafted

Adversarial samples are inputs that have been slightly altered, just enough to fool an AI model, but not enough for a human to notice anything wrong. These small, targeted changes can cause a model to make incorrect predictions. In applications like fraud detection, medical diagnosis, or autonomous control, even subtle errors can lead to costly or dangerous outcomes.

To create adversarial samples, attackers identify parts of the input that strongly influence the model's decision. This is often done by calculating gradients, which show how much each feature contributes to the output. Once the most sensitive areas are identified, minor adjustments are made to steer the model toward the incorrect result.

How Attacks Work

A basic example is the Fast Gradient Sign Method (FGSM), which calculates how to push the input in just the right direction to cause the most significant prediction error. It's fast and often effective, though more advanced methods like Projected Gradient Descent (PGD) go further, applying multiple minor tweaks while keeping the input within an acceptable range.

To automate this process, attackers often use a two-step framework:

1. **Sensitivity Estimation**: Measure how changing each part of the input shifts the output.
2. **Perturbation Selection**: Choose which features to modify, and by

how much.

Some methods use saliency maps to focus on just the most influential input areas. Others rely on distance metrics to make sure the modified input doesn't stray too far from the original.

Also read: Understanding AI in Cybersecurity and AI Security: AI in Cybersecurity (UCSAISec-01)

3. Defense Strategies Against Adversarial Attacks

Defending against adversarial attacks involves either stopping them as they occur or preparing the model to resist them ahead of time. These two approaches, known as reactive and proactive methods, form the foundation of most defense strategies.

Adversarial Training

One of the most widely used defenses is adversarial training. It involves generating adversarial samples during training and mixing them into the training data. This helps the model learn not just to classify the original input, but also to handle minor, malicious tweaks.

There are several flavors of adversarial training:

- **FGSM (Fast Gradient Sign Method)**: A simple one-shot attack used to generate quick adversarial examples. It is fast to compute but relatively weak.
- **PGD (Projected Gradient Descent)**: Builds stronger examples through multiple iterations, making the model more resistant overall.
- **VAT (Virtual Adversarial Training)**: Works even without labeled

data by using small, targeted noise to smooth decision boundaries. It's instrumental in semi-supervised learning.

Certified Robustness

Certified defenses use math to prove that a model's prediction will not change if the input is only slightly altered. Instead of relying on trial and error with known attacks, these methods define a specific range of changes that are guaranteed to leave the output unchanged. They are resource-intensive and not always practical for large models, but useful when high confidence is needed.

Gradient Masking

This method tries to make it harder for attackers to use gradients to craft adversarial samples. It adds non-differentiable steps or noise that hides the model's internal signals. While this can stop simple gradient-based attacks, it's not a complete solution, and many attackers find workarounds.

Input Reconstruction

Instead of hardening the model itself, input reconstruction methods clean the input before it reaches the classifier. They use autoencoders or generative models to remove noise and pull the input back toward the "natural" data distribution.

Defensive Distillation

Distillation works by training a second, simpler model using the probability outputs (not just the labels) of a larger model. This process smooths decision boundaries and makes the model

harder to fool.

Ensemble Defenses

Combining multiple models can make it harder for an adversarial input to fool all of them. These ensembles may use different architectures, training styles, or randomized input transformations. Even if one model is tricked, others might catch the anomaly.

Adversarial Detection

Some systems try to flag adversarial inputs at test time. For instance, **SafetyNet** adds a separate detector that monitors a classifier's internal layers for signs of tampering. It can catch inputs that look suspicious, even if they still get classified normally.

Network Verification

For critical systems, formal verification tools like Reluplex and DeepSafe are used to test whether a model holds up under small input changes. Instead of checking one input at a time, these methods scan entire regions of the input space to see where predictions might break down. This gives a clearer picture of how reliable the model is and where it might fail under pressure.

4. Privacy-Preserving Methods in AI

Modern AI models depend on large datasets that often include personal or sensitive information. This raises major privacy concerns, especially when data is shared, processed remotely, or used to train models across different organizations. Several approaches have been developed to protect privacy throughout

the AI lifecycle.

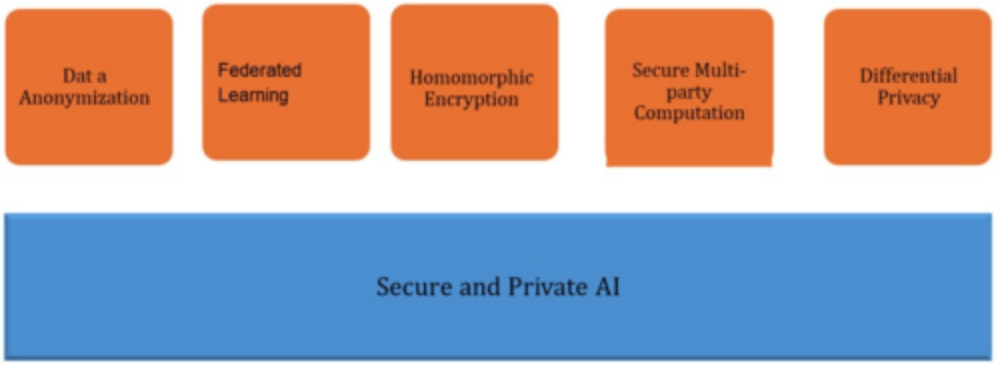

Figure 2: Privacy-Preserving Methods in AI

Data Anonymization

Data anonymization removes or masks identifiable information from datasets before they are used for training. While it helps reduce privacy risks, it can be vulnerable to re-identification attacks if done poorly or combined with auxiliary data.

Federated Learning

Federated Learning (FL) allows different parties (like hospitals, smartphones, or banks) to train a shared model without sharing raw data. Each device or institution trains a model locally, then sends back updates (not the data itself) to a central server. These updates are aggregated into a global model.

This setup keeps data decentralized and reduces exposure, but model updates can still leak information. To mitigate that:

- **Secure Aggregation** protocols make sure the server only sees the combined result, not individual updates.
- **Differential Privacy** can add noise to each client's update before it's shared.

- **Global privacy** protects data from third parties but trusts the server. Local privacy assumes the server itself could be compromised and adds stronger protection.

Homomorphic Encryption

Homomorphic encryption lets AI models operate on encrypted data. The idea is that the server never sees the raw input or the result – it performs computations blindly. Only the data owner can decrypt the output.

This is powerful but computationally expensive. Two notable implementations include CryptoNets and UniHENN.

Secure Multi-Party Computation (SMPC)

SMPC lets multiple parties jointly compute results without revealing their inputs to each other. This is useful for training models across institutions that don't want to share data.

Two common types of SMPC are garbled circuits, where one party encrypts the computation and another evaluates it blindly, and secret sharing, where data is split into shares, computed separately, and combined without revealing the original inputs.

Differential Privacy

Differential privacy is a method for protecting individual data within a dataset. It makes sure that no one can tell whether a specific person's information was used to train a model. To do this, it adds noise during key steps in the learning process. The central idea is that the model should behave the same whether your data is part of the training set or not.

Noise can be added in different ways:

- **During Training**: Random noise is applied to gradients so that updates do not reveal specific data.
- **During knowledge Transfer**: Noise is added to model outputs or labels when transferring information between models.

Also read: Understanding AI in Cybersecurity and AI Security: AI in IoT and OT Security (UCSAISec-02)

5. Explainability & Interpretability

AI systems are often black boxes. They make predictions, but it's not always clear why. In areas like cybersecurity or healthcare, that's a problem. If a model misfires, you need to understand what went wrong and whether it was a normal error, a bias, or an adversarial attack.

Explainability tools help you open up that black box. They highlight which parts of the input influenced a decision, making AI behavior more transparent and trustworthy.

- **SHAP (Shapley Additive Explanations)** attributes parts of a model's output to individual input features using principles from game theory. Helpful in isolating the exact reasons behind a prediction, such as identifying which network activity led to a security flag.
- **LIME (Local Interpretable Model-agnostic Explanations)** creates simplified local models around specific predictions by perturbing the input and observing how the output changes. It builds a simple explanation model around one prediction at a time.
- **Grad-CAM and other feature attribution methods** highlight the

input regions most responsible for a decision, often through visual maps. Initially built for image classification, they've been adapted for cybersecurity tasks like malware analysis and anomaly detection.

- **Robust Interpretability**: If small input changes lead to wildly different model behavior, something's wrong. Techniques like saliency maps, consistency testing, and adversarial training can reveal fragile spots in the model's logic.

6. Secure Deployment & Monitoring

Once an AI model is deployed, it remains vulnerable without proper safeguards. Security must continue beyond training into the full lifecycle.

- **Real-time monitoring tools** like AI-powered intrusion detection systems track unusual input patterns, model outputs, or system behavior. They flag threats such as adversarial probes or data exfiltration attempts.
- **Regular audits**, including vulnerability scans and red team tests, help uncover new weaknesses and monitor model drift.
- Only authorized users should be able to interact with your model or its underlying data. **Access control systems and encryption** help enforce this.
- A **secure deployment pipeline** requires ongoing surveillance, testing, and strict access governance to maintain reliability.

7. Challenges in Real-World Use

Adversarial defenses often look promising in controlled settings,

but real-world systems come with added complexity. Some of the biggest challenges include:

- **Trade-off Between Accuracy and Robustness**: Hardening a model can reduce its accuracy on clean data. In high-stakes systems like autonomous driving, this trade-off must be carefully managed to avoid introducing new risks.
- **Defending Against Unknown Attacks**: Many defenses are tailored to specific attack types. Adaptive or previously unseen methods can bypass static defenses with ease.
- **Operational and Environmental Constraints**: Real-time systems demand low-latency inference. Defenses that introduce delay or require additional computation may be impractical.
- **Transferability and Simulation Gaps**: Attacks built in one model often transfer to another. Defenses validated in simulation do not always hold up under real-world conditions.
- **Lack of Standards**: There's no single benchmark or gold standard for measuring adversarial robustness. Without standard evaluation protocols that reflect deployment realities, it isn't easy to assess the practical reliability of a defense.

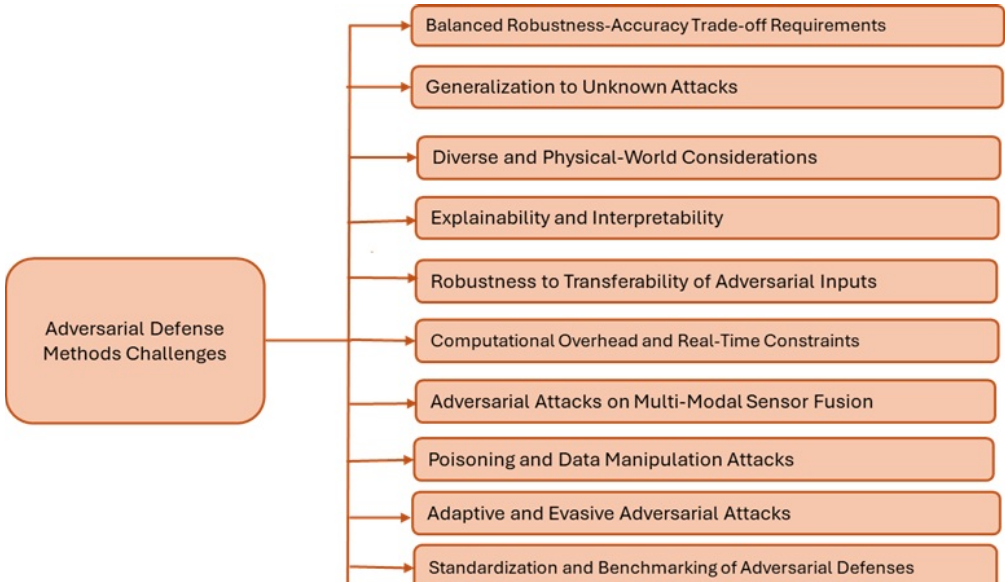

Figure 3: Challenges in applying Adversarial Methods

Making AI Defensible by Design

Adversarial attacks expose real vulnerabilities in AI systems already deployed in critical sectors. As reliance on these models grows, robustness and privacy must be treated as core design requirements.

Effective defenses go beyond patching known issues. They need to account for evolving threats, protect data throughout the pipeline, and operate reliably under real-world conditions. Techniques like adversarial training, input purification, explainability, and privacy-preserving learning all play a role, but each introduces trade-offs in performance, complexity, or accuracy.

If AI is going to be trusted, it has to be resilient by design.