AI SECURITY FRAMEWORK

snowflake

WHITE PAPER

TABLE OF CONTENTS

INTRO

Artificial Intelligence (AI) has revolutionized numerous domains, transforming the way we live and work. Its algorithms and models have proven their mettle by outperforming traditional methods in various applications, from natural language processing to self-driving cars. However, as AI permeates our lives, it introduces new security risks that can have catastrophic consequences. A compromised model could cause car accidents, misdiagnose illnesses, jeopardize lives, create fake content in news or manipulate stocks, impacting serious financial crises.

To harness AI's potential, while safeguarding against vulnerabilities, regular audits, adversarial testing, and transparent model development are essential. A practical framework for securing AI systems is crucial, ensuring that the future lies at the intersection of innovation and resilience. Join us as we explore the delicate balance between progress and security in the era of technological marvels.

How to read this document

- Each of the threats below has an associated category or asset, risk area, and triggers — criteria that make a given threat apply.
- You can find a summary of each threat, together with impact and examples, as well as proposed mitigations and references.

TRAINING DATA LEAKAGE

Name	Category/Asset	Main Risk	Trigger
Training data leakage	Model/Dataset	Confidentiality, Compliance, Legal	Generative models trained on shared sensitive data

Summary

In AI, training data leakage refers to the inadvertent exposure of sensitive information that may be contained within the dataset(s) used to train a model. This leakage can occur through various means, such as improper data handling practices, inadequate data anonymization techniques, or vulnerabilities in data storage and transmission systems. Training data leakage may pose significant risks to privacy, security, and intellectual property, highlighting the importance of robust data protection measures throughout an AI system's lifecycle.

Impact

The impact of training data leakage has the potential to lead to breaches of confidentiality, loss of competitive advantage and potential legal liabilities for organizations. For instance, in the context of healthcare, the exposure of patient records used to train medical diagnostic models could result in violations of privacy regulations such as the US Health Insurance Portability and Accountability Act (HIPAA) and undermine patient trust. Similarly, in the realm of financial services, the leakage of proprietary trading data used to train predictive models could enable competitors to gain insights into trading strategies, compromising market integrity and business competitiveness.

Example attack

- Attacker, via carefully crafted prompt, is able to extract information included in training data about certain individuals that may be sensitive.
- Attacker, through various techniques like vulnerabilities, improper access control, social engineering, or malicious code, is able to compromise systems that store or access datasets and gets access to training data itself, giving the attacker a wide knowledge about training sources and potentially sensitive information stored in the training dataset.

Mitigations

- **Data Anonymization:** Apply robust data anonymization techniques such as differential privacy, k-anonymity, or l-diversity to protect sensitive information in the training dataset while preserving its utility for model training.
- **Train-Test Split:** Apply data preparation techniques only to the training set, avoiding leakage. Evaluate the model on a separate test set.
- **Access Controls:** Implement strict access controls and data governance policies to restrict access to training data only to authorized personnel and ensure that data usage complies with applicable privacy regulations and organizational policies.
- **Secure Data Transmission:** Encrypt training data during transmission over untrusted networks using secure communication protocols such as transport layer security (TLS) or virtual private networks (VPNs) to prevent eavesdropping or interception by malicious actors.
- Consider using synthetic data or redacting the original datasets to remove values and properties that are sensitive, yet not important in the model training process.
- Consider proper data classification and data loss prevention (DLP) systems covering stored and processed data.

PRIVACY

Name	Category/Asset	Main Risk	Trigger
Privacy	Model/Dataset	Confidentiality, Compliance, Legal	Generative models trained on shared sensitive personal data Decision flow based on returned predictions (business context identification)

Summary

AI has become pervasive across various domains, but its reliance on large volumes of training data can give rise to privacy concerns, relating to the collection, storage, and utilization of sensitive data to train and deploy models. The challenge lies in balancing the need for accurate models with the imperative to safeguard individuals' privacy rights. Addressing privacy concerns involves implementing robust data pseudonymization and anonymization techniques, implementing access controls, and developing privacy-preserving AI systems. Privacy risks are often layered with other AI risks, such as training data leakage, bias, and lack of explainability/transparency, and can occur across the AI lifecycle, from training stages to inference.

Impact

The impact of privacy breaches in AI can be significant. For instance, unauthorized access to sensitive healthcare data used to train medical diagnostic models can compromise patient confidentiality and may result in regulatory penalties. Models often require personal and private data for accurate predictions. However, using such data poses risks of leakage and may violate privacy laws and regulations like the EU's General Data Protection Regulation (GDPR). Similarly, in the context of recommender systems, inadvertent disclosure of users' preferences and behaviors can erode user trust and lead to adverse consequences.

Example attack

An attacker attempts to determine whether a specific individual's data was used in the training dataset of a model. By exploiting the model's output probabilities or confidence scores, the attacker can infer whether a given data point was part of the training data, thus compromising individual privacy. For instance, in a medical research study, an attacker could determine whether a particular patient's data was included in the training dataset for a predictive model, revealing sensitive medical information without the patient's consent.

Mitigations

- **Differential Privacy:** Implement differential privacy mechanisms to inject noise into training data or model outputs, thereby preventing adversaries from inferring sensitive information about individual data points.
- **Data Minimization:** Adopt data minimization principles to collect and retain only the minimum amount of data necessary for model training and inference, reducing the risk of privacy breaches.
- **Secure Multiparty Computation (SMC):** Utilize SMC protocols to enable collaborative model training across multiple parties without revealing individual data points, preserving privacy while leveraging the collective knowledge of diverse datasets.
- **Federated Learning:** Employ federated learning techniques to train machine learning models directly on user devices, ensuring that sensitive data remains localized and reducing the need for centralized data storage.
- **Privacy-Preserving Evaluation:** Use privacy-preserving evaluation methods such as homomorphic encryption or secure enclaves to enable model evaluation without exposing sensitive data to unauthorized parties.

BIAS

Name	Category/Asset	Main Risk	Trigger
Bias	Model/Dataset	Integrity, Compliance, Legal	Model response used for business decisions or presented to wider audience

Summary

AI models, despite their impressive capabilities, are susceptible to bias which refers to the presence of systematic and unfair inaccuracies in predictions made by the models. These biases can arise from various sources, including biased training data, algorithmic design choices, and the inherent biases of the individuals involved in the development process. Recognizing and mitigating bias in AI models is crucial to ensure fair and equitable outcomes, especially in applications such as finance, healthcare, and criminal justice.

Impact

The impact of AI model bias can potentially lead to discriminatory outcomes that disproportionately affect certain groups. In scenarios where biased models are deployed, individuals may experience unfair treatment, denial of opportunities, or biased decision-making. This not only erodes trust in AI systems, but also perpetuates and amplifies existing societal inequalities

Example attack

An example of a bias attack is adversarial manipulation, where malicious actors deliberately inject biased data into the training dataset to influence the model's predictions. This can be achieved by subtly modifying input data to shift the model's decision boundaries in favor of a particular group or outcome. Adversarial attacks highlight the vulnerability of AI systems to intentional manipulation, emphasizing the need for robust defenses against such threats. Another example is unexpected bias in the model responses which has the potential to lead to legal implications and reputational risks. New York City, for example, introduced a new law that requires Automated Employment Decision Tools to be audited against bias before use. (**ref**)

Mitigations

- **Data collection and cleansing:** Ensuring diverse and representative datasets, carefully examining data sources for potential biases, and employing techniques like data augmentation to address imbalances.
- **Algorithm selection and design:** Utilizing algorithms known to be less susceptible to bias and carefully evaluating fairness metrics during model development.
- **Human oversight and auditing:** Implementing human review processes to catch and address biased decisions, regularly performing fairness audits to identify and mitigate potential issues.

- Employing techniques like counterfactual analysis or fairness-aware optimization to adjust model outputs and reduce bias after the fact.
- **Continuous monitoring and feedback:** Actively monitoring deployed models for signs of bias and incorporating feedback from users and stakeholders to refine them over time.
- Evaluate model fairness using metrics like demographic parity, equalized odds, and disparate impact.
- Adjust sample weights to balance underrepresented groups.
- **FairGAN:** Fairness-aware Generative Adversarial Networks may be used to generate fair synthetic data.

LACK OF EXPLAINABILITY / TRANSPARENCY

Name	Category/Asset	Main Risk	Trigger
Lack of explainability / transparency	Model/Dataset	Integrity, Compliance, Legal	Decision flow based on returned predictions (business context identification)

Summary

AI models often lack explainability (LOE), this problem is present in many powerful models, particularly complex deep learning architectures. This opacity, often referred to as the "black box" problem, presents a significant challenge across various domains. From understanding medical diagnoses to ensuring fairness in loan approvals, explainability is crucial for building trust, mitigating bias and ensuring responsible AI development. Striking the right balance can be challenging. Complex models may offer superior accuracy but often are "black boxes." Simpler models often provide greater transparency but may underperform.

Impact

- **Reduced Trust:** Users struggle to trust models they don't understand, leading to hesitancy in adopting AI solutions. This hinders the widespread adoption of beneficial technologies, particularly in high-stakes domains like healthcare and finance.
- **Bias Amplification:** If biases exist in the training data, opaque models may amplify them, leading to discriminatory outcomes. Without understanding the model's reasoning, it's difficult to identify and address these biases.
- **Security Vulnerabilities:** The complex inner workings of black box models can be exploited by attackers to manipulate their outputs, potentially leading to security and data breaches.
- **Debugging Challenges:** Troubleshooting and improving opaque models is significantly more challenging compared to interpretable ones. This hinders the development and maintenance of robust and reliable AI systems.

Example attack

Imagine an image classification system used for autonomous vehicles, an attacker could manipulate road signs or traffic signals in a way that causes the model to misinterpret them, potentially leading to hazardous driving behaviors. Because the model fails to provide explanations for its decisions, these adversarial attacks can go unnoticed until they result in real-world consequences, highlighting the importance of explainability in robust and secure AI systems.

Mitigations

- **Feature Importance:** Techniques like LIME (Local Interpretable Model-Agnostic Explanations) and SHAP (SHapley Additive exPlanations) help identify the most influential features contributing to a prediction, offering insights into the model's reasoning.
- **Model Explainability:** Utilize techniques like Explainable AI (XAI) to understand the model's decision-making and identify potential vulnerabilities.
- **Counterfactual Explanations:** Approaches like Anchors or Integrated Gradients generate counterfactual examples, i.e., "what-if" scenarios where the input is minimally changed to produce a different output. These can help users understand the boundaries and decision logic of the model.
- **Adversarial Training:** Exposing the model to adversarial examples during training can help it develop robustness against such attacks. This involves generating adversarial examples and iteratively refining the model to resist them.
- **Model Transparency and Documentation:** Document and maintain detailed records of the model's architecture, training data, and hyperparameters to facilitate transparency and reproducibility. Providing clear documentation enables users to understand and validate the model's behavior effectively.

BACKDOORING MODELS (INSIDER ATTACKS)

Name	Category/Asset	Main Risk	Trigger
Backdooring models (insider attacks)	Model	Integrity, Compliance, Legal	Creating 1P models or fine-tuning/processing 3P models

Summary

Model backdooring insider threats are a significant concern in AI, where adversaries manipulate models during training to introduce hidden vulnerabilities, allowing them to trigger malicious behavior during deployment. This insider threat poses a serious risk to the integrity and security of AI systems, potentially compromising their reliability and trustworthiness.

Impact

The impact of model backdooring or insider attacks may lead to compromised model performance, unauthorized access to sensitive information, and potential harm to individuals or organizations relying on the affected systems. Adversaries can exploit backdoors to evade detection mechanisms, manipulate model outputs, or launch targeted attacks, undermining the effectiveness and safety of machine learning applications across various domains.

Example attack

- **Trojan Injection:** An attacker with insider access subtly modifies the training data or model architecture to insert a Trojan trigger, which activates the backdoor under specific conditions. For instance, in an image classification model, the attacker may insert a hidden trigger that causes the model to misclassify images containing a certain pattern or object as benign, leading to potentially harmful consequences during deployment.
- **Data Poisoning:** Adversaries inject malicious data samples into the training dataset to bias the model towards certain behaviors or objectives. By strategically poisoning the training data, attackers can manipulate the model's decision boundaries, leading to unexpected or undesirable outcomes during inference. For example, in a spam email detection system, an adversary may inject benign emails labeled as spam to deceive the model into misclassifying legitimate messages.

Mitigations

- **Secure Development Practices:** Implementing strict access controls, code review processes, and continuous monitoring throughout the model development lifecycle.
- **Data Provenance and Auditability:** tracking data origin, modifications, and access logs to identify suspicious activity and potential tampering attempts.
- **Model Verification and Validation:** thorough verification and validation of AI models to detect anomalous behavior or deviations from expected norms. This may involve rigorous testing, model introspection, and verification of model outputs against ground truth labels or known benchmarks.
- **Access Control and Monitoring:** enforcing strict access controls and monitoring mechanisms to limit the exposure of models to potential insider threats. Monitor model activity and behavior for signs of unauthorized access, suspicious activity, or deviations from normal operation.
- **Adversary Detection and Response:** proactive measures for detecting and responding to insider threats in real-time. This includes deploying anomaly detection systems, conducting regular audits, and implementing incident response protocols to mitigate the impact of backdoor attacks.

PROMPT INJECTION

Name	Category/Asset	Main Risk	Trigger
Prompt injection	Model	Integrity, Compliance, Legal	Model response used for business decisions or inference designed to be limited in purpose

Summary

Prompt injection attacks target the way LLMs process input prompts. These prompts essentially guide the model's thinking and influence the generated output. Attackers can manipulate prompts in several ways:

- **Poisoning:** Injecting malicious keywords or phrases into the prompt to steer the model towards generating specific outputs, like hate speech, misinformation, or offensive content.
- **Adversarial Prompt Crafting:** Constructing prompts that exploit the model's internal biases or blind spots to produce biased or unfair outputs, even when the actual text used is neutral.
- **Meta-prompt Injection:** Embedding hidden instructions within the prompt that subtly influence the model's behavior beyond the intended task, potentially leading to unintended consequences.

Impact

- **Dissemination of Harmful Content:** LLMs can be used to generate large volumes of text, making them susceptible to being used for spreading misinformation, propaganda, or hate speech.
- **Erosion of Trust:** If LLMs are perceived as easily manipulated, it can erode public trust in these technologies and their potential benefits.
- **Privacy Violations:** Malicious prompts could be used to extract sensitive information from the model, potentially violating user privacy or revealing confidential details about the training data.

Example attack

Attackers can craft adversarial inputs into a publicly accessible system tailored to exploit vulnerabilities in LLMs and induce specific behaviors or outcomes. For instance, in a natural language processing system, an adversary may inject carefully crafted inputs containing subtle linguistic cues or triggers to manipulate the model's responses to generate outputs that leak information.

Mitigations

- **Prompt Validation and Filtering:** Implementing mechanisms to detect and filter out malicious or suspicious prompts before feeding them to the LLM.
- **Adversarial Training:** Training LLMs on diverse and adversarial prompts to make them more robust against manipulation attempts.
- **Context-Aware Processing:** Incorporating context information (e.g., user identity, task specifics) into the LLM's processing to prevent prompts from being exploited in isolation.
- **Explainable AI Techniques:** Utilizing explainable AI methods to understand how prompts contribute to the generated output, enabling better detection of potential manipulation attempts.
- **Human-in-the-Loop Systems:** Combining LLMs with human oversight and review processes can help catch malicious outputs and ensure responsible use of these models.

INDIRECT PROMPT INJECTION

Name	Category/Asset	Main Risk	Trigger
Indirect prompt injection	Model	Integrity, Compliance, Legal	Keeping model in a loop with 3rd party APIs or systems chained with model response

Summary

Indirect prompt injection is a critical security issue affecting LLMs. LLMs, widely used across various applications, combine user instructions with third-party content to create prompts for LLM processing. However, malicious actors can exploit this combination to inject harmful instructions into external content, causing LLMs to generate unexpected and potentially harmful responses. Despite the severity of this vulnerability, no comprehensive analysis or effective defense mechanisms have been proposed until recently.

Impact

Indirect prompt injection attacks pose a significant risk to users and organizations. By manipulating LLM outputs, attackers can deceive users, alter information, or even cause unintended consequences. For instance, imagine a chatbot providing medical advice based on LLM-generated responses. If an attacker injects malicious instructions, the chatbot could recommend harmful treatments, endangering lives. The impact extends beyond chatbots to any LLM-integrated application, including translation services, content generation tools, and recommendation systems.

Example attack

- **Financial Fraud:** Malicious instructions are embedded in a stock market prediction blog post. Investors rely on LLM-generated predictions for trading decisions. The LLM, influenced by the injected instructions, provides false predictions, leading to financial losses.
- **Disinformation Campaigns:** Attackers manipulate news articles by injecting biased instructions into LLMs so that, when prompted with these articles, they produce misleading summaries. Dissemination of false information affects public opinion and trust.

Mitigations

- Implement standard techniques used for detecting prompt injection.
- Training LLMs on diverse and adversarial prompts to make them more robust against manipulation attempts.
- **Sources Validation and Filtering:** Use trusted sources of external content for LLM agents.
- **Explainable AI Techniques:** Utilizing explainable AI methods to understand how prompts contribute to the generated output, enabling better detection of potential manipulation attempts.
- **Human-in-the-Loop Systems:** Combining LLMs with human oversight and review processes can help catch malicious outputs and ensure responsible use of these models.

ADVERSARIAL SAMPLES

Name	Category/Asset	Main Risk	Trigger
Adversarial samples	Model	Integrity, Compliance, Legal	Model response used for business decisions or presented to wider audience

Summary

An adversarial sample is a carefully modified input designed to mislead an AI model. The modifications, often imperceptible to humans, can be as subtle as adding minimal noise to an image or crafting a sentence with slightly altered syntax. Despite the seemingly minor changes, the model misinterprets the input, often classifying it as a completely different category with high confidence. This raises significant concerns, especially in safety-critical domains like autonomous vehicles, facial recognition, and spam filtering, where incorrect predictions can have severe consequences.

These attacks can target various types of AI models, including deep neural networks, support vector machines, and decision trees. Adversarial samples often leverage the sensitivity of AI models to small changes in input data, aiming to exploit the model's weaknesses and cause it to make incorrect predictions.

Generating adversarial samples often involves leveraging the model's internal decision boundaries. These boundaries represent the learned decision surfaces that separate different classes in the input space. By understanding how the model makes predictions, attackers can craft inputs that lie close to, but just outside, the expected decision boundary for a particular class.

Impact and examples

Misclassification leading to various scenarios depending on the model and application

- **Safety & Security:** In self-driving cars, a misclassified stop sign could lead to accidents. In facial recognition systems, manipulated images could enable unauthorized access.
- **Finance:** Fraudulent transactions might bypass detection systems. Stock prices could be manipulated through fake news fed to sentiment analysis models.
- **Healthcare:** Medical diagnosis or treatment recommendations could be compromised.

Mitigations

Addressing adversarial samples attacks requires a multifaceted approach. Some key mitigation strategies include:

- **Robust Model Training:** Incorporating adversarial training during model development by augmenting the training dataset with adversarial samples helps improve model robustness.
- **Defensive Distillation:** Applying defensive distillation, a technique involving training a secondary model on the outputs of the primary model, can enhance resistance to adversarial attacks.
- **Input Preprocessing:** Employing input preprocessing techniques, such as feature scaling and normalization, can make models less susceptible to adversarial samples.
- **Adversarial Detection Mechanisms:** Integrating detection mechanisms into AI systems can identify adversarial samples during runtime, enabling proactive responses to potential attacks.

SPONGE SAMPLES

Name	Category/Asset	Main Risk	Trigger
Sponge samples	Model	Availability, Compliance, Legal	Allowed input for inference

Summary

The "sponge samples" attack exploits vulnerabilities in machine learning models, particularly neural networks, by leveraging maliciously crafted inputs designed to significantly increase energy consumption and computational latency. These "sponge" examples, named for their ability to absorb resources, work by triggering inefficient internal computations within the model, leading to performance degradation. The attack poses a threat to diverse applications that rely on real-time or resource-constrained machine learning, such as autonomous vehicles, edge computing, and other resource-limited devices.

Impact

- **Performance Degradation:** The primary impact is a sharp rise in energy consumption and latency, potentially rendering the model inoperable or unresponsive in critical real-time scenarios.
- **Denial-of-Service (DoS) Attacks:** By targeting specific models or systems, attackers can launch DoS attacks, disrupting or disabling services that rely on machine learning.
- **Security Implications:** Sponge attacks could be combined with other adversarial techniques to evade detection, manipulate outputs, or achieve more complex goals.

Example attack

Consider a language model used in a real-time chatbot application. An attacker crafts a sponge input tailored to the model's architecture and training data. When fed to the model, this input triggers excessive internal computations, potentially causing:

- **Increased resource consumption:** The CPU/GPU demands spike, exceeding hardware/software limits, generated output is well above expected length/size.
- **Elevated latency:** The response time increases dramatically, rendering the chatbot sluggish or unresponsive to other users.
- **Degraded accuracy:** If the model's resources are depleted, its ability to process legitimate inputs accurately may be compromised.

Mitigations

- **Input Validation and Normalization:** Implement pre-processing steps to detect and reject suspicious inputs based on statistical or domain-specific criteria.
- **Resource Monitoring and Throttling:** Monitor resource usage (e.g., CPU, memory, latency) and dynamically adjust processing parameters to prevent overload and multi-tenant "noisy neighbor" issues.
- **Adversarial Training:** Train models with diverse adversarial examples to improve their robustness to manipulation and increase their attack detection capabilities.
- **Hardware-Level Optimizations:** Explore specialized hardware that can efficiently handle computationally expensive operations or detect and bypass anomalous patterns.

MODEL STEALING

Name	Category/Asset	Main Risk	Trigger
Model stealing	Model	Confidentiality, Legal	Allowed input for inference and visibility to model response or model weights and code

Summary

Model stealing is an attack aimed at extracting and replicating the functionality of a target model. In this breach, adversaries seek to gain unauthorized access to a pre-trained model, allowing them to recreate a replica that mirrors its behavior. This poses a significant threat to the intellectual property and proprietary algorithms of organizations that invest in developing advanced machine learning models.

Impact

- **Intellectual Property Theft:** Stealing a unique model may constitute intellectual property theft, depriving the owner of its potential benefits.
- **Economic Loss:** Stolen models can be used to compete directly with the owner, leading to lost revenue and market share.
- **Reputational Damage:** Breaches involving sensitive data or model theft can harm an organization's reputation and erode customer trust.
- **Security Risks:** Stolen models might be used for malicious purposes, such as generating deepfakes or manipulating financial markets.

Example attack

Consider a scenario where a financial institution has developed a highly accurate model to predict stock market trends. An adversary with malicious intent could deploy model stealing techniques to extract the underlying architecture, weights, and parameters of the model. With this information, the attacker can then recreate a duplicate model, giving them access to the same predictive capabilities without the need for extensive training data. This stolen model could be used for making profitable trades, putting the financial institution at a significant disadvantage.

Mitigations

- **Secure Model Deployment:** Employ secure deployment practices to safeguard the model architecture and parameters. Use encryption and secure channels to transmit model-related data.
- **Watermarking and Fingerprinting:** Embed watermarks or unique fingerprints into the model to identify its origin. This helps in tracing the stolen model back to its source.

- **Differential Privacy Techniques:** Integrate differential privacy techniques into the training process to introduce noise, making it harder for attackers to accurately reconstruct the model.
- **Access Control Measures:** Implement strict access controls and permissions to limit who can interact with the model and its components. Regularly audit and monitor access logs for suspicious activities.

FUZZING

Name	Category/Asset	Main Risk	Trigger
Fuzzing	Model	Integrity, Availability, Compliance	Allowed input for inference and visibility to model response

Summary

Machine learning model attack fuzzing (MLAF) leverages fuzz testing principles to discover vulnerabilities in machine learning models. It works by feeding the model with a deluge of diverse, often malformed, inputs, aiming to elicit unexpected behaviors or trigger crashes. This technique complements traditional adversarial example crafting methods, expanding the threat landscape beyond targeted perturbations.

In general computer science fuzzing, also known as fuzz testing or fuzzing analysis, is a powerful technique used to uncover vulnerabilities in software systems. In machine learning context it involves feeding in the context of model input that aims to discover vulnerabilities in ML models, algorithms, and their implementations.

Impact

The impact of successful fuzzing attacks on ML systems can be severe:

- **Incorrect Predictions:** The attacker manipulates the model's input to produce inaccurate predictions, potentially impacting safety-critical systems (e.g., self-driving cars).
- **Privacy Violations:** Fuzzing can reveal sensitive information encoded in the model, leading to privacy breaches.
- **Adversarial Attacks:** Fuzzing can reveal attack vectors that adversaries can exploit to manipulate ML models - an effective way to identify the right adversarial sample.
- **Resource Exhaustion:** Fuzzing can overload ML systems, causing resource exhaustion and denial-of-service (DoS) scenarios.

Example attack

Consider an image classification model used in facial recognition. A fuzzing attack might identify tiny pixel-level perturbations that need to be inserted into an image to:

- **Evade Detection:** The model misclassifies the image as a different person, allowing unauthorized access.
- **Privacy Breach:** The perturbations reveal hidden information about the person in the image.

Mitigations

- **Input Validation:** Implement robust input validation mechanisms to reject malformed data before it reaches the model.
- **Adversarial Training:** Train the model on adversarial examples to improve its robustness against such attacks.
- **Model Explainability:** Utilize techniques like Explainable AI (XAI) to understand the model's decision-making and identify potential vulnerabilities.
- **Fuzzing as a Defense:** Use fuzzing as a defensive tool to identify and fix vulnerabilities in models before attackers exploit them.

MODEL INVERSION

Name	Category/Asset	Main Risk	Trigger
Model inversion	Model	Confidentiality, Legal	Allowed input for inference and visibility to model response

Summary

Model inversion is a class of privacy threats in AI where an adversary aims to reverse-engineer a trained model to extract sensitive information about its training data. This type of attack exploits the vulnerabilities present in the model's output and can have serious implications for privacy and security. In a model inversion attack, an adversary leverages the model's predictions to infer details about individual data points, potentially compromising confidential information.

Impact

- **Intellectual property theft:** Reversing engineering models might reveal proprietary information or trade secrets embedded in their training data.
- **Privacy Breach:** Attackers can uncover personal or confidential information from the model's outputs.
- **Adversarial Exploitation:** Model inversion can aid adversaries in crafting targeted attacks or manipulating the original AI system.

Example attack

Consider a model designed for a mortgage loan scoring system. An attacker, through repeated interactions with the system, submits carefully crafted queries and observes the model's responses. By analyzing the output probabilities or decision boundaries, the adversary can gradually reconstruct certain features of individuals in the training dataset, ultimately revealing the identities of those individuals.

Mitigations

- Limit access to the model and its predictions.
- Require authentication, encryption, or other security measures.
- Validate input data to prevent malicious inputs.
- Check format, range, and consistency before processing.
- Log all inputs and outputs for auditing.
- Compare predictions to ground truth data.
- Monitor performance over time.
- Regularly update the model with new data.
- Correct inaccuracies and prevent outdated information leaks.

DISTRIBUTED DENIAL OF SERVICE ON ML MODEL

Name	Category/Asset	Main Risk	Trigger
Distributed denial of service on ML model	Model	Availability, Compliance, Legal	Unrestricted or unaccounted access to inference

Summary

Distributed Denial of Service (DDoS) attacks work in a similar way that sponge samples, aiming to overwhelm ML systems, exhaust their computational resources, and render them unusable. Traditional DDoS attacks target web servers or networks. In the ML context, the target becomes the model itself or the infrastructure it runs on. DDoS attacks against ML systems can take various forms:

- **Resource Exhaustion:** Attackers overload the system with requests, consuming critical resources like CPU, memory, or network bandwidth. This can cripple the model's ability to process legitimate queries.
- **Complex Inference Requests:** Attackers deliberately craft computationally expensive inference requests, forcing the model to expend significant resources and time on each query, eventually slowing it to a crawl.
- **False Data Injection:** Attackers flood the system with erroneous or misleading data during training or retraining phases. This can corrupt the model and lead to degraded performance or biased predictions.
- **Multi-Vector Attacks:** Attackers may combine these methods to overwhelm the system's defenses from multiple angles, amplifying their disruption potential.

Impact

- **Disrupted Operations:** Downtime for ML-powered systems in healthcare, finance, or autonomous systems can lead to delayed diagnoses, financial losses, or even safety hazards.
- **Loss of Revenue:** If a business relies on ML models to generate revenue (e.g., product recommendations), DDoS attacks translate into direct losses.
- **Reputational Damage:** Unreliable ML systems can erode user trust, hindering model adoption and impacting a company's reputation.
- **Increased Costs:** Mitigating and recovering from DDoS attacks often involves additional infrastructure investment and operational expenses.

Example attack

- **Targeting ML-based Spam Filters:** Attackers can overwhelm email spam filters with complex, resource-intensive queries, allowing malicious emails to slip through and potentially causing further breaches.
- **Manipulating AI-powered Trading Systems:** DDoS attacks on stock trading algorithms can disrupt market operations, triggering delayed or erroneous trades leading to financial losses.
- **Sabotaging Medical AI:** Attackers may target ML-based medical diagnostic systems, delaying critical treatment decisions and potentially impacting patient care.

Mitigations

- **Capacity Planning and Scaling:** Ensuring sufficient computational resources and implementing auto-scaling mechanisms to absorb initial surges of requests.
- **Resilient Infrastructure:** Distributing ML workloads across cloud-based or hybrid infrastructures can prevent single points of failure and improve resilience against attacks.
- **Filtering and Traffic Anomaly Detection:** Employing intrusion detection systems (IDS) and traffic analysis tools that can identify and filter out suspicious or malicious requests.
- **Rate Limiting and Prioritization:** Implementing rate-limiting controls and prioritizing legitimate traffic in real-time to maintain critical operations under attack.
- **Adaptive Defense:** Utilizing ML-based anomaly detection and real-time threat assessment to adapt defenses dynamically in response to evolving attack patterns.

MODEL POISONING

Name	Category/Asset	Main Risk	Trigger
Model poisoning	Model	Integrity, Availability, Legal	Ability to tamper datasets or model codebase

Summary

Poisoning attacks infiltrate malicious data into a model's training process, manipulating its learning and causing significant performance degradation or biased outputs. Attackers can target: data integrity: injecting noise, outliers, or manipulated samples to confuse the model or label integrity: corrupting or flipping labels to alter the ground truth the model learns from.

Impact

- **Degraded accuracy:** The model's ability to make correct predictions suffers, leading to unreliable results. In critical applications such as autonomous vehicles, healthcare diagnostics, or financial fraud detection, the consequences can be life-threatening or result in substantial financial losses.
- **Biased outputs:** The model learns skewed patterns that favor the attacker's objectives, potentially leading to discrimination or unfairness.
- **Reputational damage:** Compromised models can erode trust in AI systems and their applications. This can undermine confidence in AI systems, hindering the widespread adoption of these technologies.

Example attack

Consider a spam email filtering system that relies on an AI model to identify and filter out spam emails. In a model poisoning attack, the attacker strategically injects malicious examples into the training data, making the model more likely to misclassify legitimate emails as spam or vice versa. This can lead to a degraded filtering performance, allowing more malicious emails to reach users' inboxes, thereby compromising the system's effectiveness and potentially causing harm.

Mitigations

- Consider the supply chain of the model training infrastructure and process - integrity, invariants and access control should be assessed for each component and data flow.
- Regularly monitor the quality and integrity of training data to detect any anomalies or malicious injections. Implementing robust data validation checks can help identify unusual patterns indicative of model poisoning.
- Plant innocuous-looking but distinct data points to detect potential data injections.

- Train machine learning models with adversarial examples to improve their robustness against malicious attacks. Adversarial training involves exposing the model to crafted malicious inputs during training, enabling it to learn and adapt to potential poisoning attempts.
- Incorporate explainability features into machine learning models to understand their decision-making processes. This can help identify unexpected biases or deviations from expected behavior, signaling a potential poisoning attack.
- Implement dynamic model updating mechanisms that allow models to be retrained with fresh and verified data regularly. This reduces the impact of any poisoning attempt by continuously adapting the model to the evolving data landscape.

TRAINING DATA POISONING

Name	Category/Asset	Main Risk	Trigger
Training data poisoning	Dataset	Integrity, Compliance, Legal	Training models with datasets that are untrusted or could have been tampered with

Summary

Training data poisoning in AI refers to the malicious manipulation of training data to compromise the integrity and performance of AI models. By injecting malicious examples or subtly altering existing data, attackers aim to induce specific behaviors in the model, to induce biases, skew model predictions, or cause targeted misclassifications. Poisoned data can be introduced via an AI supply chain compromise or the data may be poisoned after the adversary gains initial access to the system. Training data poisoning poses significant challenges to the security and reliability of AI systems, highlighting the importance of robust data validation and anomaly detection mechanisms to mitigate these threats.

Impact

- **Model Performance Degradation:** Poisoned data can significantly reduce model accuracy, reliability, and generalizability, leading to erroneous predictions and decisions.
- **Biased Outcomes:** Attackers can exploit data imbalances or inject discriminatory information to introduce unwanted biases into the model, perpetuating unfairness and discrimination.
- **Security Vulnerabilities:** Poisoning can be used to create backdoors or bypass security measures, enabling attackers to gain unauthorized access or manipulate system behavior.
- **Reputational Damage:** Models compromised by poisoning can lead to public mistrust, negative publicity, and financial losses for organizations.

Example attack

Training model that generates new blog posts with gathered interactions from users. They make coordinated efforts to poison data by sending inappropriate manipulated content which is registered as part of a training dataset. As a result the model begins to replicate these behaviors creating inappropriate posts in undermining the system's effectiveness and trustworthiness.

Mitigations

- **Data Sanitization:** Implement rigorous data preprocessing and cleaning procedures to detect and remove anomalous or suspicious instances from the training dataset before model training begins.
- **Robust Model Training:** Employ adversarial training techniques, such as incorporating adversarial examples into the training process or using robust optimization algorithms, to enhance the resilience of machine learning models against data poisoning attacks.
- **Input Validation:** Validate input data at runtime to detect and mitigate potential attacks during model inference, ensuring that models can accurately classify legitimate inputs while robustly handling adversarial inputs.

- **Diverse Ensemble Learning:** Train multiple diverse models using disjoint subsets of the training data and aggregate their predictions to improve robustness against data poisoning attacks. Ensemble methods can help mitigate the impact of poisoned data on individual models and enhance overall system resilience.
- **Continuous Monitoring and Retraining:** Continuously monitor model performance and behavior in production environments and retrain models periodically using updated and validated datasets to adapt to evolving threats and mitigate the impact of data poisoning over time.

MULTITENANCY IN ML ENVIRONMENTS

Name	Category/Asset	Main Risk	Trigger
Distributed denial of service on ML model	Infrastructure	Confidentiality, Compliance, Legal	Access to ML services running in multi tenant environment

Summary

Providing MLaaS - Accessible AI solutions have propelled the need for multi-tenant AI environments. These environments allow multiple clients (tenants) to share the same underlying infrastructure and resources while maintaining isolation and security. However, achieving effective tenant isolation presents the significant challenge of ensuring that tenants cannot access or influence the data and models of others.

We consider three common approaches for handling AI models in multi-tenant environments:

1. **Tenant-Specific Models:** Each tenant trains and utilizes a dedicated model specifically on their data. This approach is ideal for highly sensitive data or situations where limited transfer learning potential exists between datasets.
2. **Shared Models:** All tenants utilize the same pre-trained model, potentially sourced from third parties or trained on aggregated, anonymized data from consenting tenants. This approach can be cost-effective but raises concerns about fairness and potential data leakage.
3. **Tuned Shared Models:** A shared model is initially trained on aggregated data and then further fine-tuned on each tenant's specific data. This approach offers a balance between efficiency and security but requires careful design and management of the fine-tuning process.

Impact

- **Impact on Data Privacy and Security:** Sharing infrastructure raises concerns about data privacy and security. Tenants might inadvertently or maliciously access sensitive information belonging to other tenants, potentially compromising confidentiality and violating trust.
- **Model Fairness:** Shared models trained on data from diverse tenants can inherit biases present in individual datasets. This can lead to unfair or discriminatory outputs when applied across all tenants, undermining the model's fairness and reliability.
- **Performance and Resource Allocation:** Sharing resources across multiple tenants can impact model performance and system responsiveness. Inefficient resource allocation can lead to resource exhaustion for specific tenants, impacting their model performance and overall user experience.
- **Intellectual Property:** Models created or fine tuned by company research teams may be used by clients but are a unique intellectual property which requires protection. Although they contain client data there should not be direct access to them or the possibility to extract them outside the hosting environment to protect IP.

Example attack

- **Data Exfiltration:** Attackers might exploit weaknesses in isolation mechanisms to access or steal data from other tenants, potentially enabling identity theft, fraud, or competitive advantage.
- **Model Poisoning:** Malicious actors could inject poisoned data into shared datasets, manipulating the training process and influencing the behavior of the model for specific tenants, leading to biased or erroneous outputs.
- **Model Inference Attacks:** Attackers might analyze the model's behavior for one tenant to infer information about the data or models of other tenants, potentially breaching privacy or revealing sensitive information.

Mitigations

- **Data Encryption:** Encrypting data at rest and in transit can prevent unauthorized access, even if an attacker gains access to the underlying infrastructure.
- **Federated Learning:** This approach allows training models on distributed datasets without directly sharing the data, enhancing privacy and security.
- **Differential Privacy:** Introducing statistical noise into the training process can mask individual contributions, protecting sensitive information and mitigating privacy concerns.
- **Secure Multi-Party Computation (SMPC):** This cryptographic technique allows multiple parties to jointly compute a function on their data without revealing the data itself, enabling secure collaboration and model training.
- **Resource Management and Monitoring:** Implementing robust resource management policies and monitoring systems can ensure fair allocation of resources and prevent performance degradation for individual tenants.

EXPOSURE OF SENSITIVE INFERENTIAL INPUTS

Name	Category/Asset	Main Risk	Trigger
Exposure of sensitive inferential inputs	Infrastructure	Confidentiality, Integrity, Legal	Communication with model via an API

Summary

Model inference is often offered as a wrapped up application service which, if not properly secured, might be a target to eavesdropping. This is not exactly an attack on the model rather on information in transit where sensitive information is inadvertently leaked or exposed during the inference process, posing risks to entities sending the information. Therefore, implementing robust security measures is essential to protect sensitive information from unauthorized access and ensure the confidentiality of the data.

Impact

- **Privacy Violations:** Leaked information can be used for identity theft, discrimination, or targeted attacks, jeopardizing individual privacy and autonomy.
- **Reputational Damage:** Breaches of trust due to leaked information can damage the reputation of institutions using AI systems and erode public trust in AI technology.
- **Legal and Regulatory Violations:** Depending on the nature of the exposed information and applicable regulations, organizations deploying vulnerable service could face legal repercussions.

Example attack

- **Inference Side-Channel Attacks:** Adversaries can exploit side-channel information leakage during the inference process to infer sensitive attributes or inputs. For example, in a speech recognition system, an attacker may analyze the timing or duration of model responses to infer the presence of certain keywords or phrases, compromising user privacy and confidentiality.
- **Direct access to sent data:** Attacker can directly access to data sent for inference that contain sensitive information as a result of lack of proper security in transit of use of deprecated protocols like SSL.

Mitigations

- **Proper configuration of encryption in transit:** This includes enforcing the latest cryptographic standards, such as TLS 1.3 Additionally, strong cipher suites should be prioritized to bolster encryption strength while minimizing the risk of cryptographic attacks.
- **Implementing authentication mechanisms:** OAuth 2.0 or JSON Web Tokens (JWT), ensures that only authorized users can access sensitive resources, thereby mitigating the risk of unauthorized data access.
- **Secure communication channels:** For example,WebSocket over TLS, enables real-time bidirectional communication between clients and servers while maintaining the confidentiality and integrity of transmitted data.

ATTACKS ON THE INFRASTRUCTURE HOSTING AI SERVICES

Name	Category/Asset	Main Risk	Trigger
Distributed denial of service on AI model	Infrastructure	Confidentiality, Integrity, Availability, Compliance, Legal	1P infrastructure hosting ML services

Summary

While the focus often lies on the algorithms and models themselves, the security of the underlying AI hosting infrastructure is equally critical. This infrastructure typically comprises various non-AI components like Apache Kafka, Hadoop, and Flink, forming the backbone for data storage, processing, and model deployment. However, these components are susceptible to a range of technical attack vectors, potentially jeopardizing the entire ML system and exposing sensitive information. Understanding these vulnerabilities, their potential impact, and effective mitigation strategies is crucial for building secure and reliable AI deployments.

Impact

Attackers can exploit vulnerabilities in various non-AI components of the AI hosting infrastructure:

- Remote Code Execution (RCE) on Servers
- SQL Injection
- Cloud Account Compromise
- CI/CD Pipeline Attacks
- Data Pipeline Attacks

This can have can have far-reaching consequences leading to:

- **Data Breaches:** Exposed sensitive data like training datasets, user input, or model outputs can lead to privacy violations, identity theft, or financial losses.
- **Model Tampering:** Manipulated models can generate erroneous predictions, leading to incorrect decisions, biased outcomes, or even safety hazards in critical applications.

- **Disrupted Operations:** Attacks can disrupt data processing, model training, or inference pipelines, impacting service availability and hindering successful AI deployments.
- **Reputational Damage:** Security breaches can damage the reputation of organizations relying on AI, eroding user trust and potentially hindering broader adoption of the technology.

Example attack

- **Analytic or MLops platform compromise:** A cyber attack on a healthcare organization exploited vulnerabilities in an cluster used for data analysis, resulting in the exfiltration of patient information.
- **SQL Injection in Training Data Pipeline:** A cloud-based AI platform suffered a data breach due to a SQL injection vulnerability in their data ingestion pipeline, leading to the exposure of sensitive training data.
- **Supply Chain Attack on CI/CD Pipeline:** Attackers gained access to a third-party CI/CD platform and injected malicious code into multiple ML models deployed by various organizations, impacting their predictions and causing financial losses.

Mitigations

- **Patch Management and Vulnerability Scanning:** Regularly applying security patches and conducting vulnerability scans on servers, databases, and other components can identify and address potential weaknesses before attackers exploit them.
- **Least Privilege and Access Control:** Implementing strict access control measures, including role-based access control (RBAC), can limit user privileges and prevent unauthorized access to sensitive data or systems.
- **Data Encryption:** Encrypting data at rest and in transit can protect it from unauthorized access even if attackers breach the infrastructure.
- **Security Testing and Penetration Testing:** Regularly conducting security testing and penetration testing can identify and address vulnerabilities in the infrastructure before attackers discover them.
- **Continuous Monitoring and Logging:** Implementing continuous monitoring and logging of system activity can help detect suspicious behavior and identify potential security incidents promptly.
- **Security Awareness Training:** Raising awareness among personnel involved in managing the AI infrastructure about cybersecurity best practices can help prevent human errors that could lead to security breaches.

SELF-HOSTED OSS LLMS SECURITY

Name	Category/Asset	Main Risk	Trigger
Distributed denial of service on ML model	General	Confidentiality, Integrity, Availability, Compliance, Legal	Hosting 3rd party LLMs processing sensitive or internal use data

Summary

The computational complexity of the training process has created a market in pretrained models which provides efficient training and enables rapid prototyping. However, hosting and utilizing third-party, pretrained models poses security challenges. The very essence of these models incorporating code opens avenues for malicious actors to exploit vulnerabilities and inject malicious functionality. Understanding the nature of these threats, their potential impact, and mitigation strategies is crucial for ensuring the safe and secure adoption of third-party models.

Impact

- **Traditional Malware:** Malicious actors can embed trojans, crypto-lockers, botnet code, or cryptominers within the model structure. These can execute upon installation, compromising the host system and potentially causing financial losses or disrupting operations.
- **Data Exfiltration:** Models can be designed to leak sensitive information embedded in the input data (e.g., personal details, medical records) or the generated outputs, breaching user privacy and potentially violating regulations like GDPR.
- **Reflective Attacks:** Malicious models might exploit reflection vulnerabilities in serving systems, allowing attackers to upload their own code through crafted inputs, leading to arbitrary code execution on the hosting platform.
- **Deliberate Misclassification:** Attackers can inject code that manipulates the model's decision-making process, causing it to misclassify specific inputs or produce erroneous results, impacting decision-making and potentially causing harm.
- **Trigger-Based Actions:** Malicious code within the model can be programmed to perform specific actions upon encountering specific inputs, enabling targeted attacks or causing disruptions based on predefined conditions.

Example attack

Attackers could embed malicious code within a pre-trained image classifier. When deployed on a vulnerable system, the code could exfiltrate sensitive information from the user's device, highlighting the potential for data breaches through compromised models.

Mitigations

- **Supply Chain Security:** Implementing rigorous verification processes for third-party models, including assessing the developer's reputation, reviewing code provenance, and employing secure download channels.
- **Cryptographic Signing:** Utilizing digital signatures on models to ensure authenticity and tamper detection. Any modifications during download or transfer will invalidate the signature, alerting users to potential tampering.
- **Model Testing:** Conducting thorough testing frameworks to detect potential biases, vulnerabilities, and malicious functionality within the model before deployment. This may involve static analysis, dynamic analysis, and adversarial testing techniques.
- **Malware Scanning:** While technically challenging due to the complexity of models, exploring techniques like model compression and obfuscation may enable scanning for traditional malware signatures in specific scenarios.
- **Secure Deployment and Monitoring:** Deploying models in secure environments with robust access controls and network segmentation to minimize potential damage in case of compromise. Additionally, continuous monitoring for anomalous behavior is crucial for timely detection and mitigation of threats.

SOURCES

Explaining and Harnessing Adversarial Examples Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy https://arxiv.org/abs/1412.6572

Adversarial Attacks and Defenses in Deep Learning Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li https://arxiv.org/abs/1809.00886

Towards Deep Learning Models Resistant to Adversarial Attacks Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu https://arxiv.org/abs/1706.06083

Privacy-preserving Machine Learning in Cloud Ehsan Hesamifard, Hassan Takabi,Mehdi Ghasemi, Catherine Jones https://dl.acm.org/doi/10.1145/3140649.3140655

Encrypted statistical machine learning: new privacy preserving methods. Louis J. M. Aslett, Pedro M. Esperança, and Chris C. Holmes https://arxiv.org/abs/1508.06845

IEEE EuroS&P 2021 - Sponge Examples: Energy-Latency Attacks on Neural Networks

The SpongeNet Attack: Sponge Weight Poisoning of Deep Neural Networks Jona te Lintelo, Stefanos Koffas, Stjepan Picek https://arxiv.org/pdf/2402.06357.pdf

A systematic review of fuzzing based on machine learning techniques Yan Wanga, Peng Jiaa, Luping Liub, Jiayong Liua https://arxiv.org/pdf/1908.01262.pdf

Privacy Leakage on DNNs: A Survey of Model Inversion Attacks and Defenses Hao Fang, Yixiang Qiu, Hongyao Yu, Wenbo Yu, Jiawei Kong, Baoli Chong, Bin Chen, Xuan Wang, Shu-Tao Xia https://arxiv.org/pdf/2402.04013.pdf

Model Inversion Attack with Least Information and an In-depth Analysis of its Disparate Vulnerability Sayanton V. Dibbo; Dae Lim Chung; Shagufta Mehnaz https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10136179

Boosting Model Inversion Attacks with Adversarial Examples Shuai Zhou, Tianqing Zhu, Dayong Ye, Xin Yu, Wanlei Zhou https://arxiv.org/pdf/2306.13965.pdf

Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, Bo Li https://arxiv.org/pdf/1804.00308.pdf

MPAF: Model Poisoning Attacks to Federated Learning based on Fake Clients Xiaoyu Cao, Neil Zhenqiang Gong https://arxiv.org/pdf/2203.08669.pdf

Preference Poisoning Attacks on Reward Model Learning Junlin Wu, Jiongxiao Wang, Chaowei Xiao, Chenguang Wang, Ning Zhang, Yevgeniy Vorobeychik https://arxiv.org/pdf/2402.01920.pdf

Efficient Defense Against Model Stealing Attacks on Convolutional Neural Networks Kacem Khaled, Mouna Dhaouadi, Felipe Gohring de Magalhães, Gabriela Nicolescu https://arxiv.org/pdf/2309.01838.pdf

Isolation and Induction: Training Robust Deep Neural Networks against Model Stealing Attacks Jun Guo, Aishan Liu, Xingyu Zheng, Siyuan Liang, Yisong Xiao, Yichao Wu, Xianglong Liu https://arxiv.org/pdf/2308.00958.pdf

Are You Stealing My Model? Sample Correlation for Fingerprinting Deep Neural Networks Jiyang Guan, Jian Liang, Ran He https://arxiv.org/pdf/2210.15427.pdf

Bias and unfairness in machine learning models: a systematic literature review Tiago Palma Pagano, Rafael Bessa Loureiro, Fernanda Vitória Nascimento Lisboa, Gustavo Oliveira Ramos Cruz, Rodrigo Matos Peixoto, Guilherme Aragão de Sousa Guimarães, Lucas Lisboa dos Santos, Maira Matos Araujo, Marco Cruz, Ewerton Lopes Silva de Oliveira, Ingrid Winkler, Erick Giovani Sperandio Nascimento https://arxiv.org/abs/2202.08176

Evaluating Proposed Fairness Models for Face Recognition Algorithms John J. Howard, Eli J. Laird, Yevgeniy B. Sirotin, Rebecca E. Rubin, Jerry L. Tipton, Arun R. Vemury https://arxiv.org/abs/2203.05051

Fairness in Deep Learning A Computational Perspective Mengnan Du, Fan Yang, Na Zou, Xia Hu https://arxiv.org/pdf/1908.08843.pdf

LIME: Local Interpretable Model-Agnostic Explanations Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin https://arxiv.org/abs/2210.04533

SHAP: SHapley Additive exPlanations Scott Lundberg and Sudeep Agarwala (2017) https://arxiv.org/abs/1705.07874

Anchors: Anchors: High-Precision Model-Agnostic Explanations Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin https://ojs.aaai.org/index.php/AAAI/article/view/11491

Privacy-Preserving Machine Learning: Methods, Challenges and Directions Runhua Xu, Nathalie Baracaldo, James Joshi https://arxiv.org/abs/2108.04417

SoK: Training Machine Learning Models over Multiple Sources with Privacy Preservation Lushan Song, Guopeng Lin, Jiaxuan Wang, Haoqi Wu, Wenqiang Ruan, Weili Han https://arxiv.org/pdf/2012.03386.pdf

Privacy-Preserving Federated Recurrent Neural Networks Sinem Sav, Abdulrahman Diaa, Apostolos Pyrgelis, Jean-Philippe Bossuat, Jean-Pierre Hubaux https://arxiv.org/pdf/2207.13947.pdf

A Survey of Trustworthy Federated Learning with Perspectives on Security, Robustness and Privacy Yifei Zhang, Dun Zeng, Jinglong Luo, Zenglin Xu, Irwin King https://arxiv.org/pdf/2302.10637.pdf

Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists 1st Edition Alice Zheng, Amanda Casari

Training Data Leakage Analysis in Language Models Huseyin A. Inan, Osman Ramadan, Lukas Wutschitz, Daniel Jones, Victor Rühle, James Withers, Robert Sim https://arxiv.org/abs/2101.05405

Shadowcast: Stealthy Data Poisoning Attacks Against Vision-Language Models Yuancheng Xu, Jiarui Yao, Manli Shu, Yanchao Sun, Zichu Wu, Ning Yu, Tom Goldstein, Furong Huang https://arxiv.org/abs/2402.06659

Hidden in Plain Sight: Undetectable Adversarial Bias Attacks on Vulnerable Patient Populations Pranav Kulkarni, Andrew Chan, Nithya Navarathna, Skylar Chan, Paul H. Yi, Vishwa S. Parekh https://arxiv.org/abs/2402.05713

Data Poisoning for In-context Learning Pengfei He, Han Xu, Yue Xing, Hui Liu, Makoto Yamada, Jiliang Tang https://arxiv.org/abs/2402.02160

Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, Bo Li https://arxiv.org/pdf/1804.00308.pdf

MPAF: Model Poisoning Attacks to Federated Learning based on Fake Clients Xiaoyu Cao, Neil Zhenqiang Gong https://arxiv.org/pdf/2203.08669.pdf

Preference Poisoning Attacks on Reward Model Learning Junlin Wu, Jiongxiao Wang, Chaowei Xiao, Chenguang Wang, Ning Zhang, Yevgeniy Vorobeychik https://arxiv.org/pdf/2402.01920.pdf

Machine Learning based Framework for User-Centered Insider Threat Detection https://dalspace.library.dal.ca/bitstream/handle/10222/80731/DucLe2021.pdf

OWASP Top 10 Machine Learning Security Risks https://owasp.org/www-project-machine-learning-security-top-10/

An Early Categorization of Prompt Injection Attacks on Large Language Models Sippo Rossi, Alisia Marianne Michel, Raghava Rao Mukkamala, Jason Bennett Thatcher https://arxiv.org/abs/2402.00898

In-Context Learning Can Re-learn Forbidden Tasks Sophie Xhonneux, David Dobre, Jian Tang, Gauthier Gidel, Dhanya Sridhar https://arxiv.org/abs/2402.05723

StruQ: Defending Against Prompt Injection with Structured Queries Sizhe Chen, Julien Piet, Chawin Sitawarin, David Wagner https://arxiv.org/abs/2402.06363

Benchmarking and Defending Against Indirect Prompt Injection Attacks on Large Language Models https://arxiv.org/pdf/2312.14197.pdf

Indirect prompt injection https://greshake.github.io/

LLM Prompt Injection: Indirect https://atlas.mitre.org/techniques/AML.T0051.001/

Detecting Distributed Denial of Service Attacks using Machine Learning Models Ebtihal Sameer Alghoson, Onytra Abbass https://thesai.org/Downloads/Volume12No12/Paper_77-Detecting_Distributed_Denial_of_Service_Attacks.pdf

Trustworthy confidential virtual machines for the masses Anna Galanou, Khushboo Bindlish, Luca Preibsch, Yvonne-Anne Pignolet, Christof Fetzer, Rüdiger Kapitza https://arxiv.org/pdf/2402.15277.pdf

Why open source software matters to your enterprise The TODO Group (European Chapter) | September 2020 https://project.linuxfoundation.org/hubfs/Reports/Why-open-source-software-matters-to-your-enterprise_090820.pdf?hsLang=en

OWASP Top 10 Web Application Security Risks https://owasp.org/www-project-top-ten/

An Experience Report on Producing Verifiable Builds for Large-Scale Commercial Systems Yong Shi; Mingzhi Wen; Filipe R. Cogo; Boyuan Chen; Zhen Ming Jiang https://ieeexplore.ieee.org/document/9465650

Pickle Scanning https://huggingface.co/docs/hub/security-pickle

Never a dill moment: Exploiting machine learning pickle files Evan Sultanik https://blog.trailofbits.com/2021/03/15/never-a-dill-moment-exploiting-machine-learning-pickle-files/

ABOUT SNOWFLAKE

Snowflake enables every organization to mobilize their data with Snowflake's Data Cloud. Customers use the Data Cloud to unite siloed data, discover and securely share data, power data applications and execute diverse AI/ML and analytic workloads. Wherever data or users live, Snowflake delivers a single data experience that spans multiple clouds and geographies. Thousands of customers across many industries, including 691 of the 2023 Forbes Global 2000 (G2K) as of January 31, 2024, use the Snowflake Data.

Learn more at **snowflake.com**