SECURING AI systems against data, model, and tool poisoning attacks

Dr. Meryem Abouali
John Jay College of Criminal Justice , New York
mabouali@jjay.cuny.edu

Danish Merchant
John Jay College of Criminal Justice , New York
mohammeddanish.merchant63@login.cuny.ed

Mauricio Embus Perez
John Jay College of Criminal Justice , New York
Mauricio.embusperez24@login.cuny.edu

Perla Dahan
John Jay College of Criminal Justice , New Yor
perla.dahan07@login.cuny.edu

Background and Motivation

Large Language Models (LLMs) are increasingly deployed in applications such as:
•Healthcare decision support
•Financial systems
•Customer service automation
•AI agents interacting with external tools
However, these systems introduce **new security risks**, including:
•**Data poisoning attacks**
•**Prompt injection attacks**
•**Jailbreak techniques**
•**Tool poisoning in AI agents**
Understanding these threats is essential for **secure AI deployment**.

Research Objective

Research Objective
This study investigates:
•How adversarial attacks can manipulate LLM behavior
•The effectiveness of different attack strategies
•Whether practical defenses can mitigate these threats The research evaluates both:
Training-time attacks and Runtime attacks on a locally deployed language model.

Key Insights

- No single defense method is sufficient.
  Layered defenses significantly improve AI security.
- Pattern detection catches known attacks. Anomaly detection finds unusual behavior. Output validation prevents malicious responses

Together they provide **stronger protection.**

Conclusion

Adversarial attacks against large language models are **practical and effective**. However, this research demonstrates that **layered defensive strategies can significantly reduce these risks**. Security must be considered **throughout the entire AI lifecycle**, from training data to deployed applications.

Attack Types Evaluation

Training-Time Attacks
Backdoor Trigger Injection
•Hidden token triggers malicious output
Bias Injection
•Introduces harmful stereotypes into data
Label Flipping
•Reverses training labels to corrupt learning
Toxic Content Injection
•Inserts harmful language into training data

Experimental Setup

Model: LLaMA 3.2 (1B parameters)
Environment: Local deployment using Ollama
Hardware: MacBook Pro with Apple M-series chip
Tools Used: Python, NumPy , Scikit-learn, Isolation Forest (anomaly detection)

Dataset:

•100 sentiment analysis samples
•Balanced positive / negative labels

Key Results

Training Data Poisoning Detection

Attack Type	Detection Rate
Backdoor	100%
Toxic Injection	100%
Bias Injection	87%
Label Flipping	75%

Overall detection rate: **87%**

Runtime Attacks
Without defense: **67% attack success rate**
With layered defense: **0% successful attacks**

Future Work

Future research should explore:
Larger datasets, Multiple models, and Adaptive defense mechanisms

Methodology

Experiments used a locally deployed LLaMA 3.2 model. Training-time attacks included backdoor triggers, bias injection, label flipping, and toxic content insertion.
Runtime attacks included prompt injection, context poisoning, and jailbreak techniques. A layered defense architecture combining pattern
filtering, Isolation Forest
anomaly detection, and runtime validation was implemented.

Defense Architecture

A **three-layer defense framework** was developed.
Training-Time Defense

1.Pattern-based scanning
2.Anomaly detection (Isolation Forest)
3.Statistical consistency checks

Runtime Defense

Runtime Defense
1.Input filtering
2.Prompt sanitization
3.Output validation
This layered approach provides **defense-in-depth**

Security Implications

AI systems deployed in real-world applications may be vulnerable to:
•Training data manipulation
•Prompt-based adversarial attacks
•Tool poisoning in AI agents
Organizations should implement:
•Dataset validation
•Runtime prompt filtering
•Output verification
•Defense-in-depth architectures