SECURITAS: Security Evaluation and Control Utilizing Reliable Intelligent Transparent Automated Systems

Author:

Gerald Enrique Nelson Mc Kenzie

Abstract:
This publication presents an open-source framework designed to rigorously evaluate artificial intelligence (AI) models using a suite of quantitative tests, all integrated into a Flask-based web application. The framework provides multiple endpoints for diverse evaluation needs:

- `/model/<model_name>` for testing a single model,
- `/all` for evaluating all available models, and
- `/multiple/<models_str>` for assessing a comma-separated list of models.

The evaluation suite comprises a broad range of tests that address critical aspects of model performance and robustness:

- **Adversarial Test (FGSM Attack):** Evaluates model susceptibility to adversarial perturbations by comparing predictions before and after generating adversarial examples.
- **Bias Analysis Test:** Assesses fairness by comparing performance across synthetic subgroups.
- **Security Test (Input Perturbation):** Measures the change in model output under fixed input noise.
- **Performance Test (Inference Time):** Computes average inference time to determine real-time applicability.
- **Confidence Test (Max Softmax Probability):** Gauges the model's prediction confidence.
- **Gradient Norm Test:** Quantifies sensitivity to input variations by calculating the norm of the gradient.
- **Activation Sparsity Test:** Estimates network efficiency by measuring the sparsity of activations in early convolutional layers.

- **Parameter Count Test:** Counts the total number of trainable parameters as a measure of model complexity.
- **Memory Usage Test:** Estimates the memory footprint during inference, crucial for deployment.
- **Occlusion Test:** Determines reliance on input regions by measuring confidence drop when a central patch is occluded.

Each test is accompanied by a rating and a detailed explanation, offering a comprehensive perspective on the model's robustness, efficiency, and reliability. This framework not only serves as a proof-of-concept for standardized AI model evaluation but also aims to contribute to the development of safe and transparent AI systems, informing best practices and regulatory standards.

The application is intended for researchers, practitioners, and policymakers who require a robust baseline for assessing AI models, with potential applications in national security, healthcare, finance, and beyond.

1. Introduction

The rapid evolution of AI models has created a need for systematic, transparent evaluation tools that can document their performance, robustness, and potential biases. Traditional methods of generating model cards often require significant manual effort and quickly become outdated. In response, this work introduces a comprehensive evaluation framework—implemented as an open-source Flask application—that automates the generation of detailed, verifiable reports (or “model cards”) for AI models.

This framework provides three distinct endpoints to support different evaluation scenarios:

- A single-model endpoint (`/model/<model_name>`) for focused evaluations,
- An all-models endpoint (`/all`) to assess every available model in the system, and
- A multiple-models endpoint (`/multiple/<models_str>`) that accepts a comma-separated list of models.

Each endpoint executes a suite of tests designed to capture critical metrics such as adversarial robustness, bias, security, performance, confidence, sensitivity, efficiency, and resource utilization.

2. Methodology

2.1. System Architecture

The framework is structured as a Flask web application with the following endpoints:

- **/model/<model_name>:** Loads and evaluates a specified model.
- **/all:** Iterates through all available models defined within the application.
- **/multiple/<models_str>:** Accepts a comma-separated list of model names to evaluate a selected subset.

Each endpoint triggers an evaluation process that executes a suite of quantitative tests and then displays the results in a standardized report.

2.2. Evaluation Suite

The evaluation suite includes:

- **Adversarial Test (FGSM Attack):**
  Generates adversarial examples using the Fast Gradient Sign Method (FGSM) and compares predictions before and after the attack.
  Rating: ★★★★☆

- **Bias Analysis Test:**
  Uses fixed synthetic labels and subgroup assignments to compute fairness metrics (accuracy differences, selection rate, TPR, FPR).
  Rating: ★★★☆☆

- **Security Test (Input Perturbation):**
  Adds fixed noise to inputs and measures changes in the model's output.
  Rating: ★★★☆☆
- **Performance Test (Inference Time):**
  Measures average inference time over several iterations to assess deployment feasibility.
  Rating: ★★★★☆

- **Confidence Test (Max Softmax Probability):**
  Evaluates prediction confidence by calculating the maximum softmax probability.
  Rating: ★★★☆☆

- **Gradient Norm Test:**
  Computes the norm of the gradient of the output with respect to the input to measure sensitivity.
  Rating: ★★★★☆

- **Activation Sparsity Test:**
  Assesses network efficiency by measuring the sparsity of activations in early convolutional layers.
  Rating: ★★★☆☆

- **Parameter Count Test:**
  Counts the number of trainable parameters as an indicator of model complexity.
  Rating: ★★★★★

- **Memory Usage Test:**
  Estimates the memory footprint during inference, using GPU statistics or parameter-based approximations.
  Rating: ★★★★☆

- **Occlusion Test:**
  Measures the drop in prediction confidence when a central patch of the input is occluded, indicating reliance on specific features.
  Rating: ★★★★☆

2.3. Implementation

The framework is implemented as an open-source Flask application. Each endpoint processes the input model(s), runs the complete evaluation suite, and generates a comprehensive report. The reports are rendered in a user-friendly format, providing quantitative metrics and qualitative ratings for each test.

3. Experimental Setup

Experiments were conducted under controlled conditions to ensure consistency across evaluations. Models are evaluated using standardized input data, and each test runs for a

predetermined duration (e.g., 5 seconds) to produce comparable results. This setup allows for rapid, reproducible assessments of diverse AI models.

4. Results

The framework generates detailed reports that include:

- **Adversarial Test:** Displays the change in predicted class due to adversarial perturbation.
- **Bias Analysis:** Summarizes fairness metrics across synthetic subgroups.
- **Security Test:** Reports the variation in model output due to fixed noise.
- **Performance Test:** Provides average inference time.
- **Confidence Test:** Indicates maximum softmax confidence.
- **Additional Tests (Gradient Norm, Activation Sparsity, Parameter Count, Memory Usage, Occlusion):** Offer insights into model sensitivity, efficiency, complexity, and resource requirements.

Each model receives an overall rating based on a penalty system derived from key tests, offering a comprehensive overview of its robustness, efficiency, and reliability.

5. Discussion

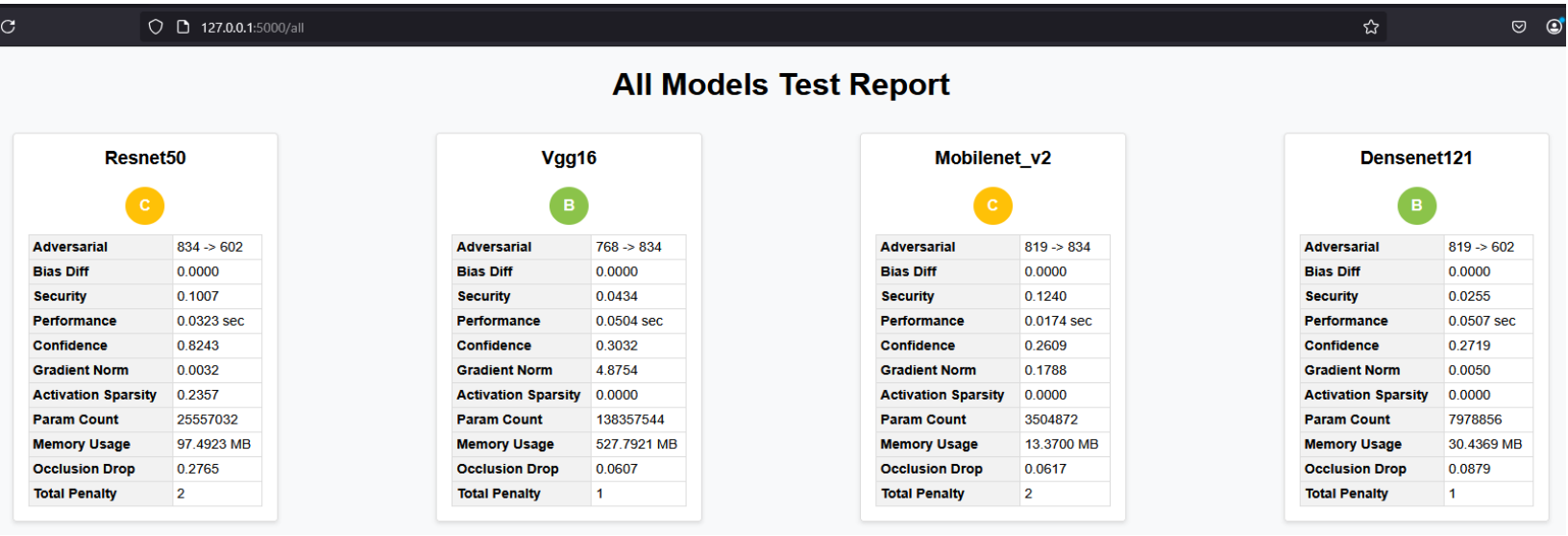
SECURITAS demonstrates the feasibility and value of an automated, multi-endpoint evaluation framework for AI models. Its modular design enables the addition of new tests and adaptation to evolving industry standards. The framework's comprehensive test suite provides critical insights, especially in light of recent industry developments. For example, while the ambitious Stargate Project under President Donald Trump underscores the need for rigorously tested AI systems, the market disruptions that can be caused by an inadequately tested AI model highlights the risks of insufficient evaluation. SECURITAS addresses these challenges by ensuring that only thoroughly vetted AI models are deployed.

6. Conclusion

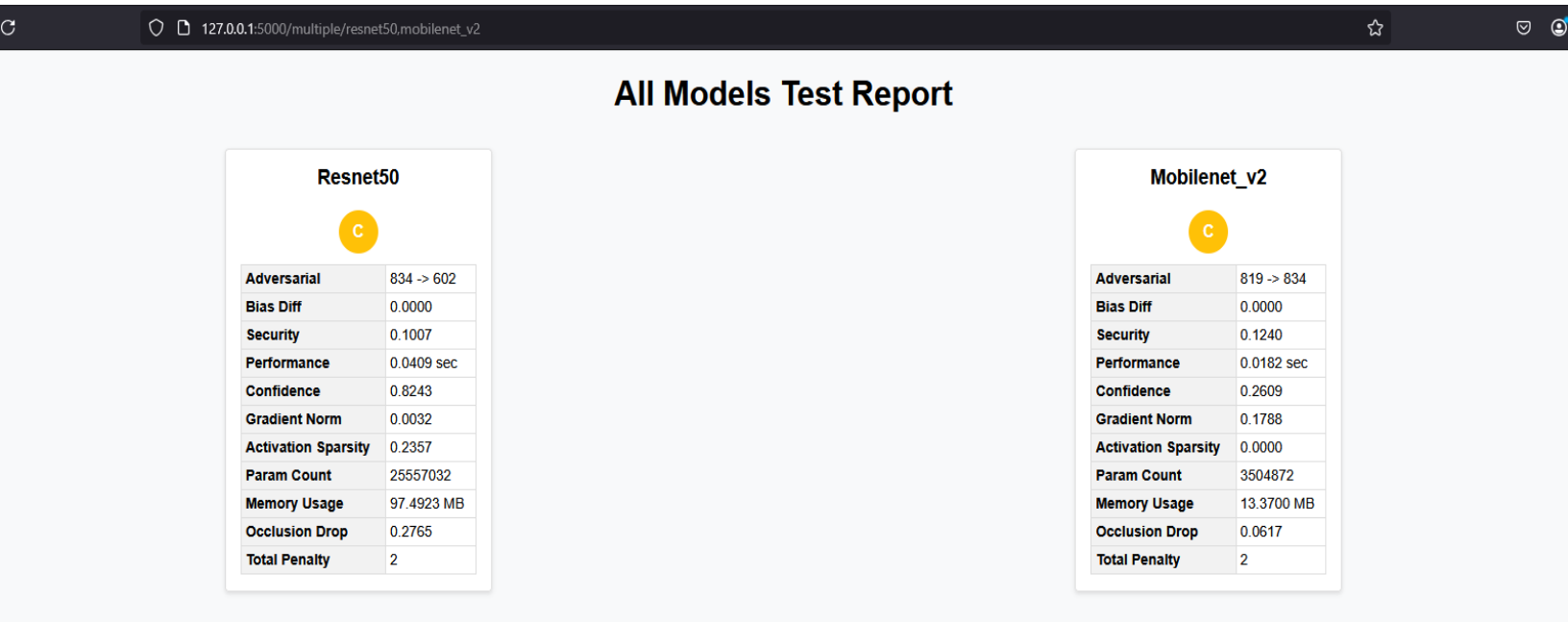
This work presents a comprehensive, open-source framework for AI model evaluation, enabling the generation of detailed, verifiable model cards through a Flask-based application. By integrating diverse tests that assess adversarial robustness, bias, security, performance, and more, the framework provides a robust baseline for safe and transparent AI deployment. Its alignment with national security initiatives and its potential to inform regulatory standards make it an essential tool for researchers, practitioners, and policymakers alike.

6. Sample Screenshots

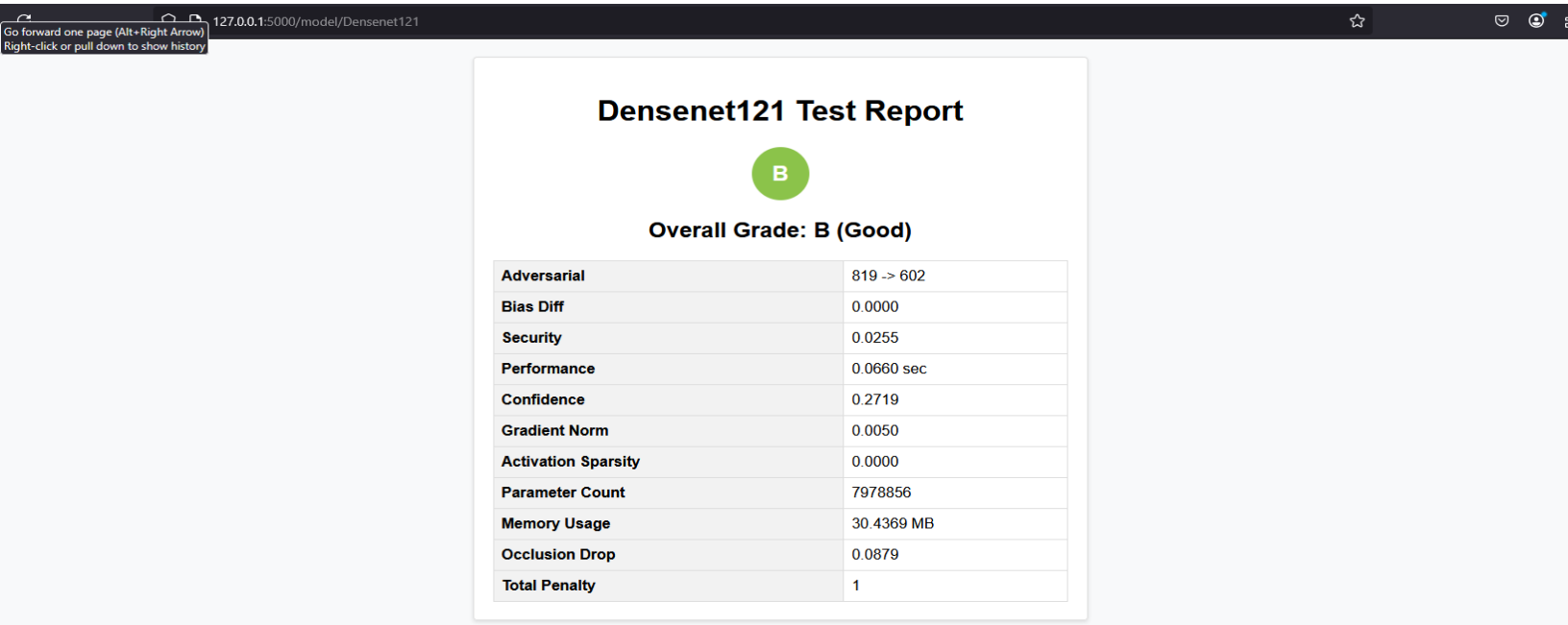
- **/ALL Endpoint**



- **/MULTIPLE/ <MODEL_STR> Endpoint**



- **/MODEL/ <MODEL_NAME> Endpoint**



References

- **Doshi-Velez, F. & Kim, B. (2017).**
  "Towards A Rigorous Science of Interpretable Machine Learning."

arXiv preprint arXiv:1702.08608.
This work outlines a systematic approach to interpretability in machine learning, setting a benchmark for evaluation metrics and methodologies.

- **Ilyas, A., Engstrom, L., Athalye, A., & Lin, J. (2019).**
  "Adversarial Examples Are Not Bugs, They Are Features."
  Advances in Neural Information Processing Systems, 32.
  This paper discusses the inherent vulnerability of models to adversarial perturbations, providing context for adversarial testing.
- **Rudin, C. (2019).**
  "Stop Explaining Black Box Models for High Stakes Decisions."
  Harvard Data Science Review, 1(1).
  Rudin argues for the use of interpretable models in critical applications, reinforcing the need for transparency in AI systems.
- **Selbst, A. D., Boyd, D., Friedler, S. A., Venkatasubramanian, S., & Vertesi, J. (2019).**
  "Fairness and Abstraction in Sociotechnical Systems."
  Proceedings of the Conference on Fairness, Accountability, and Transparency (FAccT).
  This reference provides insight into fairness in complex AI systems and underscores the importance of comprehensive bias auditing.
- **Molnar, C. (2020).**
  "Interpretable Machine Learning."
  Book (available online).
  Molnar's work offers an extensive overview of interpretability methods and techniques, complementing the explainability components of the evaluation framework.

Appendix: Code Availability

The complete implementation of the framework, including all endpoints and detailed test modules, is available in the GitHub repository:
https://github.com/lordxmen2k/SECURITAS