A Novel Testing Framework for Vision Models Using Bayesian Network

Author Name

***Abstract*—Deep learning models, particularly vision models (VMs) are critical in high-stakes domains such as autonomous driving, medical diagnostics, and security systems, where their deployment in real-world scenarios requires rigorous robustness testing due to diverse environmental conditions. Traditional metrics like neuron coverage, while essential, do not fully capture all corner cases, which can lead to unexpected model failures. To address this gap, this research introduces a comprehensive testing framework that enhances the robustness evaluation of VMs through a structured five-stage process. The first stage is specification, defines essential system properties to guide the entire testing process and ensure comprehensive coverage. The second sampling stage, gathering relevant samples for exhaustive model testing. In the test case generation stage, the defined properties are applied to create targeted test scenarios. The testing and probabilistic graph stage validates the effectiveness of these test cases and conducts robustness assessments both locally (within individual category) and globally (across multiple scenarios), employing a Bayesian network for detailed probabilistic and quantitative analysis of performance. The final stage is error summarisation, compiles and analyzes recorded errors to generate actionable graphical error reports and recommendations, thus guiding the refinement of VMs. This framework not only fills existing gaps in VM testing but also supports the development of models that are robust across varied environmental conditions.**

I. Introduction

II. Problem Statement

Deep learning models are being more widely used in a variety of applications, yet their robustness in practical applications remains a challenge.

Unlike traditional software, DNNs do not have a clear control-flow structure. They learn their decision policy through training on a large dataset, adjusting parameters gradually using several methods to achieve desired accuracy. Consequently, traditional software testing methods like functional coverage, branch coverage, etc. cannot be applied to DNNs, thereby challenging their use for safety- critical applications.Traditional software testing methods fail when applied to DNNs because the code for deep neural networks holds no information about the internal decision-making logic of a DNN.

- Input space is extremely large, **unguided simulation** are higly unlikely to find erroneous behavior
- Standards available in industry but **lack of logcial structure** and **system specification**
- Heavily depend on manual collections of test data under different conditions which become expensive as number of test condition increases
- Existing coverage criteria are not detailed enough to notice precise behaviours exhibited by DL systems.

III. Research Goal

This paper aims to develop a systematic framework for evaluating local (in a specific region of the input space) and global (across a broad range of inputs) robustness in deep learning models. The goal is to provide a comprehensive error summary to improve model design and training, ensuring their reliability for real-world applications.

IV. Contributions

This research makes the following key contributions to the field of deep learning robustness evaluation:

- We design an **end-to-end pipeline** for evaluating the robustness of system.
- We propose a **conceptual framework** that quantifies both local (individual category) and global (across diverse categories) robustness,with formalized Bayesian probabilistic approach to verify system robustness.
- A novel **error summarization** approach which allows better identification of model weaknesses related to class and property.
- We perform all our **experiments** using publicly available vision models and MNIST dataset.

V. Research Questions

This paper addresses the following research questions applicable to various vision models and datasets:

- How can we design a comprehensive framework to test system robustness?
- How can we systematically evaluated the robustness both at local (property-specific) and global (overall system) levels within framework?
- How can error summarization be employed to quantify the impacts on model robustness?

VI. Methodology

This section presents an overview of the comprehensive testing framework, which is intended to test the specified properties according to given specification. The pipeline begins by precisely describing the properties to be evaluated. To comprehensively examine the model, test cases are created and tested. The error summary step next concentrating on the progression from local to global robustness to highlight the system

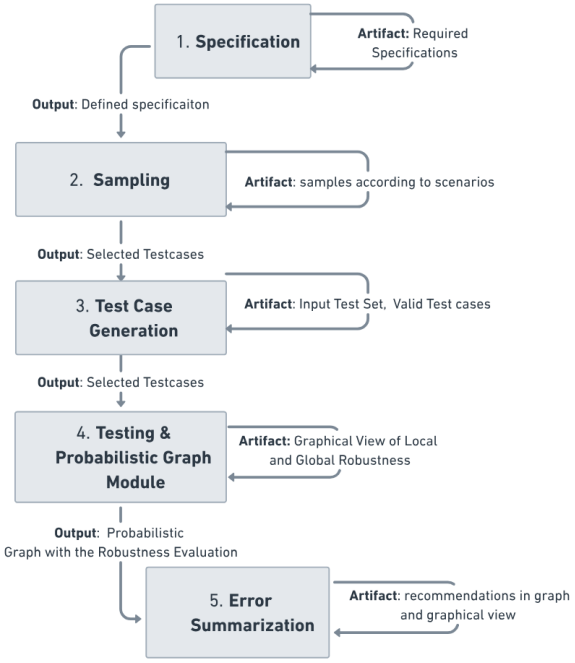

Fig. 1: Overview of Testing Framework

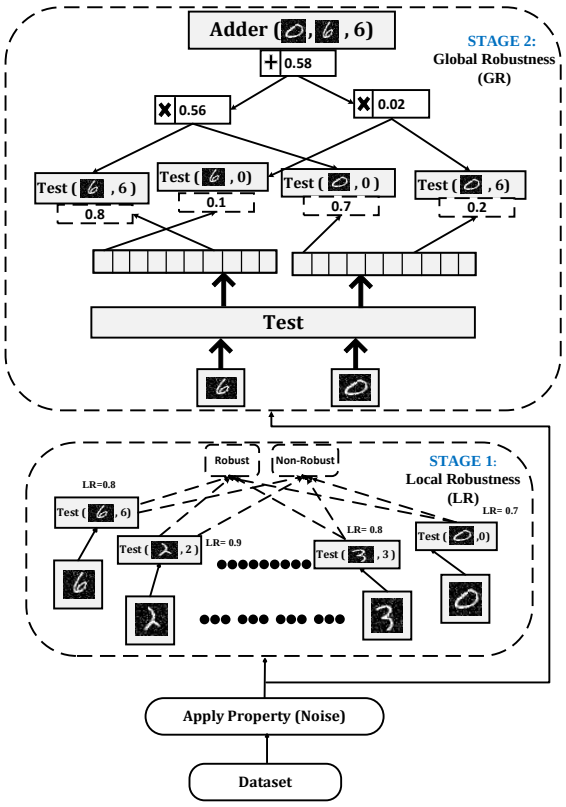

Fig. 2: Graphical View of Local and Global Robustness

systemic strengths and weaknesses. This systematic technique improves the robustness of deep learning models by tackling each essential aspect sequentially, from specification to complete error analysis.

A. Specification

B. Sampling

The sample selection process involves a random but balanced choice of samples from each class, focusing exclusively on instances that the model has correctly predicted. This method ensures a representative and fair distribution of data across all classes.

- Model Utilization:
  - A pre-trained CNN model is utilized to select samples.
  - Let $X = \{x_1, x_2, \ldots, x_N\}$ denote the set of MNIST images. The model function f predicts:

$$f(x_i) \rightarrow y_i$$

  - The filter function g identifies accurate predictions, defined as:

$$g(x_i) = \begin{cases} 1 & \text{if } f(x_i) = \text{true label of } x_i \\ 0 & \text{otherwise} \end{cases}$$

  - The subset S includes only correctly predicted images:

$$S = \{x_i \in X \mid g(x_i) = 1\}$$

- Random Selection of Samples:
  - Randomly selects 200 samples from each class in S, totaling 2000 samples.

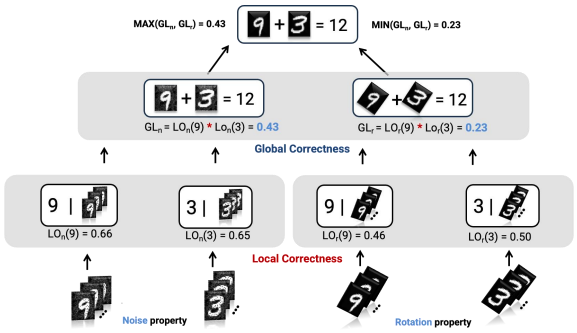

Fig. 3: Graphical View of Local and Global Robustness

  - The random selection function R is defined to ensure:

$$R(S_c, 200) \text{ for each class } c \text{ in } S$$

where S_c represents the samples of class c within S.

C. Test Case Generation

This section outlines the generation of test cases to assess model robustness through properties such as noise, rotation, and brightness adjustments.

- **Noise Addition:** Noise is added to an image x_i using a Gaussian noise model. The noise function p_n is defined as:

$$p_n(x_i, \sigma) = x_i + \epsilon$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ denotes the Gaussian noise with mean zero and standard deviation σ.

- **Rotation:** The rotation of an image x_i by an angle θ is modeled by the rotation function p_r:

$$p_r(x_i, \theta) = \text{rotate}(x_i, \theta)$$

where $\text{rotate}(\cdot, \theta)$ represents the rotation operation.

- **Brightness Adjustment:** Brightness adjustment of an image x_i is controlled by a multiplicative factor β, which scales the intensity of all pixels. The brightness function p_b can be defined as:

$$p_b(x_i, \beta) = \beta \cdot x_i$$

where $\beta > 1$ increases brightness, and $\beta < 1$ decreases it. This approach ensures that the image's contrast is preserved while adjusting its brightness.

D. Testing and Probabilistic Graph

The Testing section evaluates how accurately and confidently the model predicts under various properties (noise, rotation, brightness) applied to images from each class. This phase focuses on directly measuring and quantifying the robustness of the model.

- Confidence Level Assessment:
  - After generating test cases, measure the model's confidence for each class under each type of property.
  - Aggregate these measurements to assess the overall robustness of individual properties (noise, rotation, brightness).

E. Introduction to Local Robustness

Local robustness assesses how consistent a model's predictions are when faced with alterations such as noise, rotation, and brightness adjustments.

1) Definition: Local robustness for a class c subjected to a property p is formally defined as the conditional probability of the model correctly classifying a perturbed input:

$$LR_{c,p} = P(\hat{Y}_{c,p} = Y_c \mid X_c = x'_{c,p}), \tag{1}$$

where $X_c = x'_{c,p}$ denotes the input from class c modified by property p, such as adding Gaussian noise or altering brightness.

2) Empirical Estimation: To measure local robustness in practice, the model is subjected to a series of tests involving perturbed inputs. The robustness is then quantified by averaging the outcomes of these tests:

$$\hat{LR}_{c,p} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}(f(p(x_{c,i}, \theta_p)) = y_{c,i}), \tag{2}$$

where N represents the number of test cases, and $\mathbf{1}$ is the indicator function, which is 1 if the prediction is correct and 0 otherwise.

3) Aggregation of Robustness Measures: While individual measures of local robustness are insightful, aggregating these metrics across different properties and classes provides a comprehensive view of the model's overall stability and reliability. For instance, averaging the robustness scores across all classes for a specific property can highlight how well the model handles that property in general:

$$LR_p = \frac{1}{C} \sum_{c=1}^{C} LR_{c,p}, \tag{3}$$

where C is the total number of classes. This aggregated metric helps in identifying properties that might require further optimization to enhance the model's overall performance.

F. Global Robustness Evaluation

Global robustness in this context evaluates the model's ability to process and synthesize outputs from multiple inputs, each subjected to the same type of perturbation, directly within an integrated adder function. This adder not only generates predictions but also combines them, assessing the collective output against a predefined combined result.

1) Definition and Integrated Functionality: The global robustness is defined with respect to an adder function that both predicts and combines outputs from two perturbed inputs:

$$GR_{(c1,c2),p} = P(\text{Adder}(x'_{c1,p}, x'_{c2,p}) = Y_{\text{combined}} \mid X_{c1} = x'_{c1,p}, X_{c2} = x'_{c2,p}) \tag{4}$$

where: - $x'_{c1,p}$ and $x'_{c2,p}$ are the inputs from classes $c1$ and $c2$, respectively, after the application of property p. - The Adder function now encapsulates the prediction process for each input and the subsequent combination of these predictions. - Y_{combined} is the expected result of this combination, reflecting the correct combined output for the inputs processed together.

Empirical estimation of global robustness now involves:

$$\hat{GR} = \frac{1}{M} \sum_{j=1}^{M} \mathbf{1}(\text{Adder}(x'_{j1,p}, x'_{j2,p}) = Y_{j,\text{combined}}) \tag{5}$$

where: - M is the number of test pairs. - $\mathbf{1}$ is the indicator function, returning 1 if the combined output from the adder matches the expected result, and 0 otherwise. - The Adder function within each test pair handles the prediction from perturbed inputs and their aggregation, effectively simulating a real-world application where input data are integrated to form a single decision output.

G. Error Summarization

This section details the error summarization process following the global robustness testing, where we identify and analyze discrepancies in the model's predictions. By tracing errors from the combined outputs back

to individual properties and classes, we systematically pinpoint and address underlying weaknesses.

1) Example Setup: We consider two classes from the MNIST dataset:

- Class A: Digit '5'
- Class B: Digit '0'

Both classes are tested under the properties of noise and rotation, with the following local robustness confidence levels:

- $LR_{A,p_1} = 85\%$ (Noise)
- $LR_{A,p_2} = 78\%$ (Rotation)
- $LR_{B,p_1} = 90\%$ (Noise)
- $LR_{B,p_2} = 88\%$ (Rotation)

2) Global Robustness Testing Scenario: The model is tasked with processing two images, one from each class, both subjected to noise. The ideal prediction should reflect the sum '5' (Class A) + '0' (Class B) = '5'. An incorrect sum indicates a prediction error.

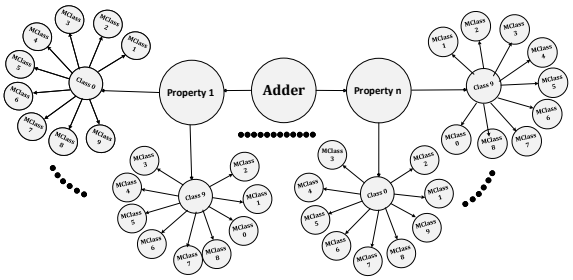

Fig. 4: Diagram of Error Summarization Highlighting Class-Property Impact

3) Error Summarization Process:

1) **Initial Error Detection:** The combined prediction incorrectly sums to '6' instead of '5'.
2) **Identify the Inaccurate Predictions:** Examination reveals the model predicts '6' for Class A and correctly predicts '0' for Class B.
3) **Drill Down to Property Level:** Both images were tested under noise. The noise robustness for each class is assessed:
   - $LR_{A,p_1} = 85\%$ — suggesting a 15% failure rate.
   - $LR_{B,p_1} = 90\%$
4) **Assess Individual Local Robustness:** Focus is placed on Class A's noise robustness, identifying potential misclassification trends under noisy conditions.
5) **Further Analysis:** Investigate if certain noise patterns consistently mislead the model concerning digit '5', potentially confusing it with a similar appearance to '6'.
6) **Systematic Error Identification:** Patterns of error involving Class A under noise are sought to determine if specific adjustments in model training or data handling can mitigate these issues.
7) **Propose Adjustments:** Modifications to the training process or data augmentation strategies are suggested to enhance noise robustness, especially for digits with appearances similar to '5'.

VII. Experiments

VIII. Threats to Validity

This section outlines significant limitations and assumptions in our study that may affect the validity and reliability of our findings.

- **Random Sampling:** Our current approach assumes a uniform distribution of samples across all classes, which may not represent the true complexity and variability within real-world data. This uniform sampling can lead to biased evaluations if the class distribution in practical applications is skewed or non-uniform. We plan to enhance our sampling techniques to better capture the diversity and distribution of data in realistic scenarios. Improved sampling strategies will help in developing more robust and generalizable error summarization methods.

IX. Related Work

X. Conclusion

References

[1] Reference details