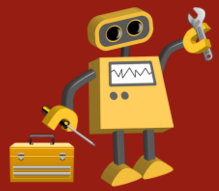


RobOT: Robustness-Oriented Testing for Deep Learning Systems

Jingyi Wang[1], Jialuo Chen[1], Youcheng Sun[2], Xingjun Ma[3], Dongxia Wang[1], Jun Sun[4], Peng Cheng[1]

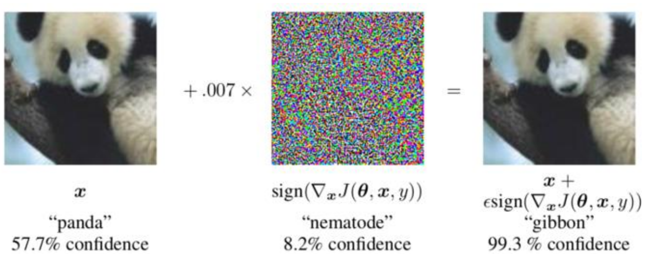
[1] Zhejiang University
[2] Queen's University, Belfast
[3] Deakin University
[4] Singapore Management University

2021/05/26

Deep Neural Network

- **Deep neural networks have made great progress.**
  - Image classification
  - Speech recognition
  - Natural language processing
  - Self-driving car

- **Are deep neural networks safe enough ? - No, LACK OF ROBUSTNESS**
  - Deployment in safety-critical applications



Adversarial Attack
[ICLR'15]

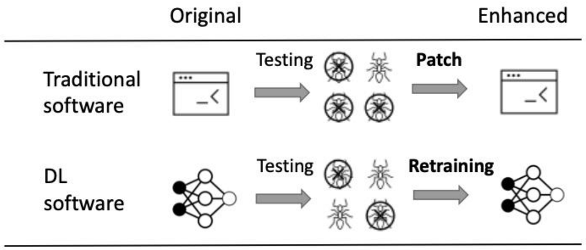
Autonomous Driving
[SOSP'17]

Medical Diagnosis
[SCIENCE'19]

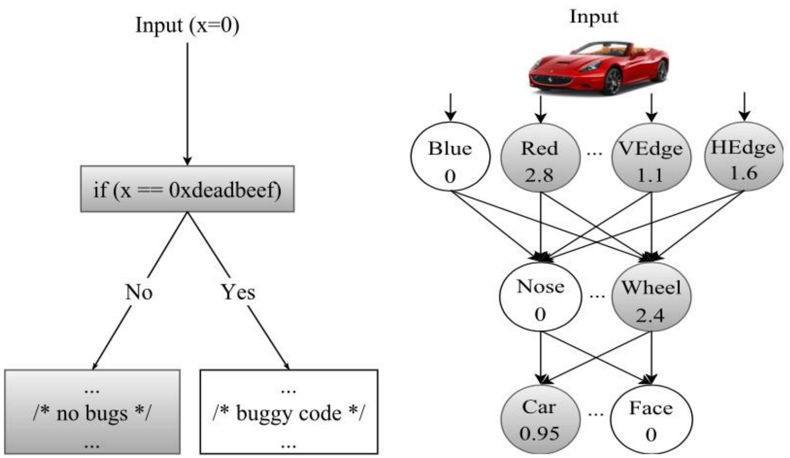
Face Recognition
[USENIX'20]

Testing Deep Neural Networks

- **Traditional coverage metrics are not sufficient.**
  - Branch coverage
  - Statement coverage
  - And more…



- **Deep neural network specialized coverage metrics.**
  - Neuron Coverage (NC) [SOSP'17]
  - TKNC, KMNC, NBC, SNAC [ASE'18]
  - Surprise Coverage (SC) [ICSE'19]
  - And more…



(a) A program with a rare branch (b) A DNN for detecting cars and faces

Testing Deep Neural Networks

- **Testing is to generate a set of inputs that are likely to**
  - **Increasing coverage metrics**
  - **Find adversarial inputs**

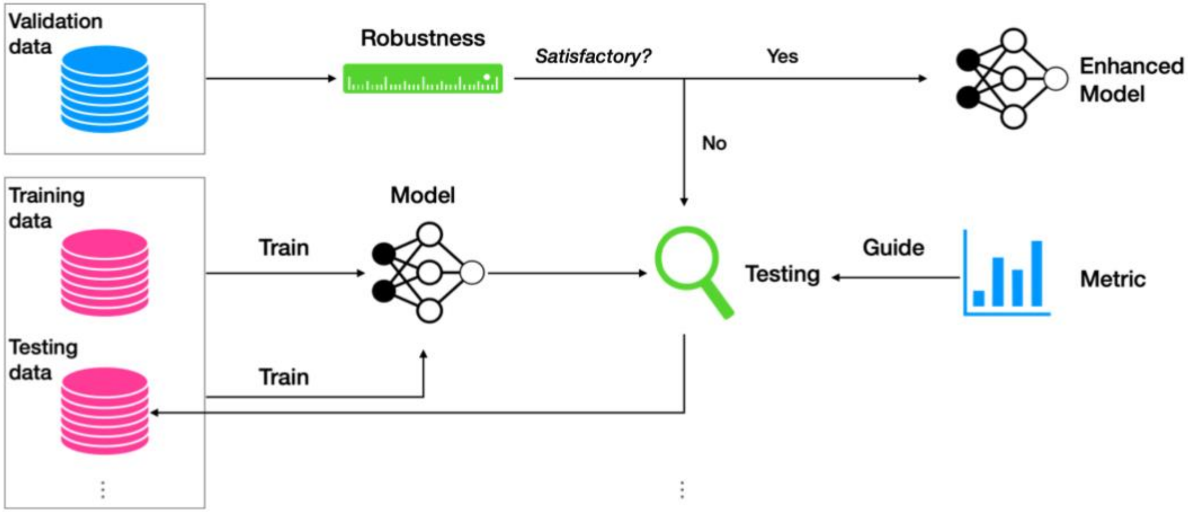
- **Existing test case generation techniques:**
  - DeepXplore [SOSP'17]
  - DLFuzz [FSE'18]
  - DeepTest [ICSE'18]
  - DeepConcolic [ICSE'19]
  - ADAPT [ISSTA'20]

How can we design metrics that are strongly correlated with model robustness?

- **Are existing NC metrics useful for improving robustness? - Not sure**
  - Recent studies found that there is little correlation [ICSE'19, FSE'20, ICECCS'20]

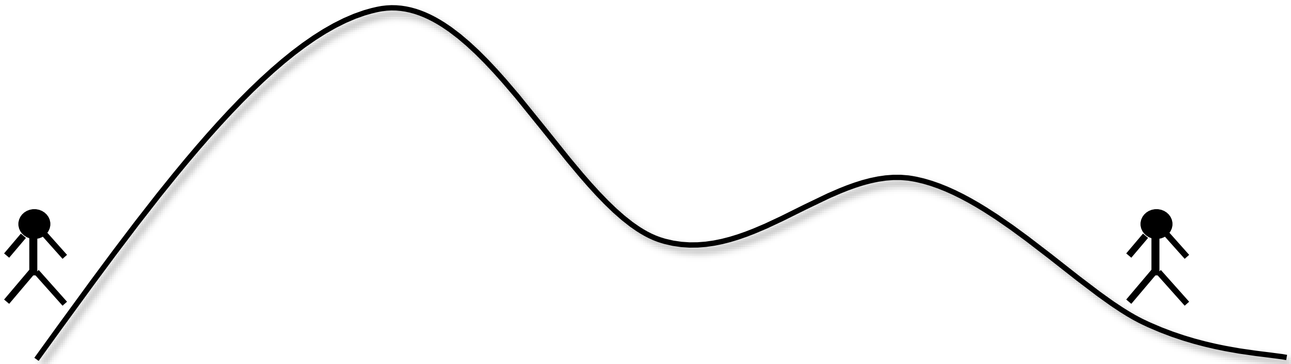
RobOT Overview

- **End-to-end Robustness-Oriented Testing framework**
  - A novel set of lightweight metrics that strongly correlated with robustness.
  - A set of fuzzing strategies to automatically generate high-quality test cases for improving robustness.
  - Aiming to bridge the gap between the DL testing and retraining



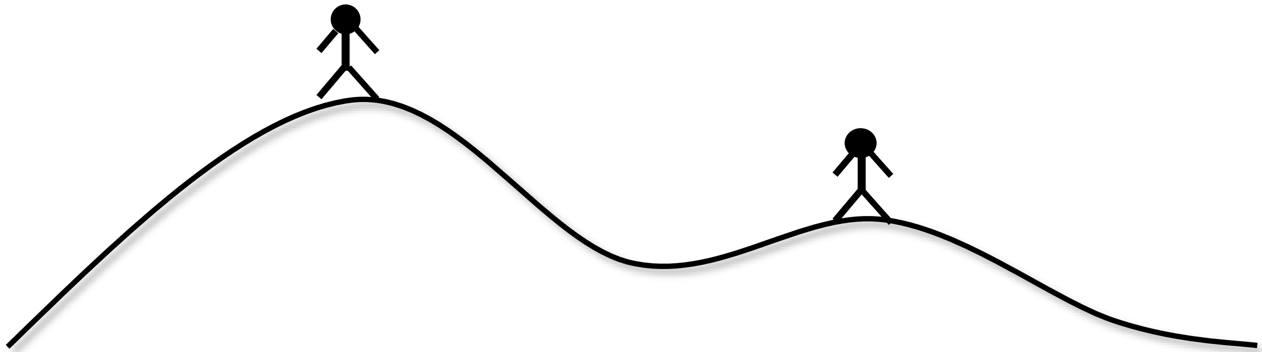
Robustness-Oriented Testing metrics

- **Intuition**
  - A test case which induces a higher loss is a stronger adversarial example, consequently more helpful in training robust models.



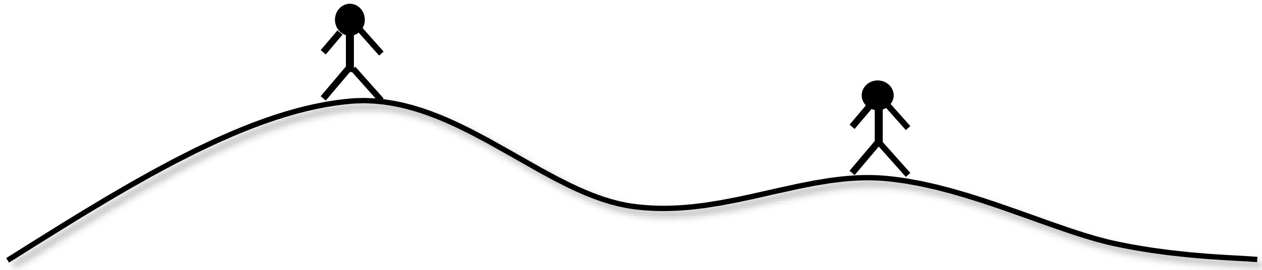
Robustness-Oriented Testing metrics

- **Intuition**
  - A test case which induces a higher loss is a stronger adversarial example, consequently more helpful in training robust models.



Robustness-Oriented Testing metrics

- **Intuition**
  - A test case which induces a higher loss is a stronger adversarial example, consequently more helpful in training robust models.



Robustness-Oriented Testing metrics

- **Two-levels testing metrics**
  - Zero-Order Loss (ZOL) measures the loss of a test case

$$ZOL(x^t, f) = L(f(\theta, x^t), y)$$

  - First-Order Loss (FOL) measures how well the loss converges

$$FOL(x^t, f) = \max_{x \in \mathcal{X}} \langle x - x^t, \nabla_x f(\theta, x^t) \rangle$$

$$= \begin{cases} \epsilon ||\nabla_x f(\theta, x^t)||_1 - \langle x^t - x_0, \nabla_x f(\theta, x^t) \rangle & L_{inf} \\ \epsilon ||\nabla_x f(\theta, x^t)||_2 & L_2 \end{cases}$$

Robustness-Oriented Testing metrics

- **Comparison with Neuron Coverage metrics**
  - Both ZOL and FOL are strongly correlated to the adversarial strength of the test cases and the model robustness.
  - Can help us select valuable test cases from a large amount of test cases to reduce the retraining cost.

ZOL

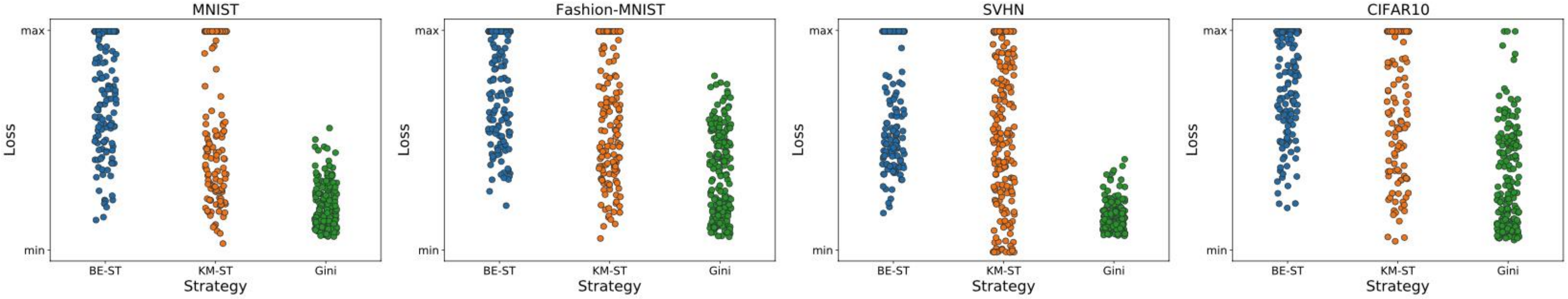
NC

FOL

FOL-Guided Test Case Selection

- **K-Multisection Strategy (KM-ST)**
  - Uniformly sample the FOL space
  - We equally divide the range of FOL into K sections
  - Randomly select N/K samples from each section

- **Bi-End Strategy (BE-ST)**
  - Combine test cases with small and large FOL values
  - Mix test cases with strong and weak adversarial strength



Loss of selected test cases for different datasets using different strategies.

FOL-Guided Fuzzing

How to use the proposed strategies for generating test cases?

BE-ST: greedily search for test cases in two directions, i.e., with both small or large FOL values.

How can we search in the FOL directions?

Through joint optimization.

How do we perturb the test cases?

Along the gradient.

FOL-Guided Fuzzing

- **A simple yet efficient fuzzing framework based on FOL**
  - Joint optimization question
    - Find adversarial examples
    - Search in FOL directions
  - Joint optimization objective

$$obj = \left(\sum_{i=2}^{k} P(c_i) - P(c_1)\right) + \lambda \cdot FOL(x')$$

Adversarial input + **Metric**

Algorithm 3 FOL-Fuzz($f, seeds\_list, \epsilon, \xi, k, \lambda, iters$)

```
Let fuzz_result = ∅
for seed ∈ seeds_list do
    Maintain a list s_list = [seed]
    while s_list is not empty do
        Obtain a seed x = s_list.pop()
        Obtain the label of the seed c1 = f(x)
        Let x' = x
        for iter = 0 to iters do
            Set optimization objective obj using Eq. 8
            Obtain grads = ∇obj / ∇x'
            Obtain perb = processing(grads)
            Let x' = x' + perb
            Let c' = f(x')
            Let dis = Dist(x', x)
            if FOL(x') ≥ FOL_m and dis ≤ ϵ then
                FOL_m = FOL(x')
                s_list.append(x')
                if c' != c1 then
                    fuzz_result.append(x')
                end if
            end if
            if FOL(x') < ξ and dis ≤ ϵ then
                s_list.append(x')
                if c' != c1 then
                    fuzz_result.append(x')
                end if
            end if
        end for
    end while
end for
return fuzz_result
```

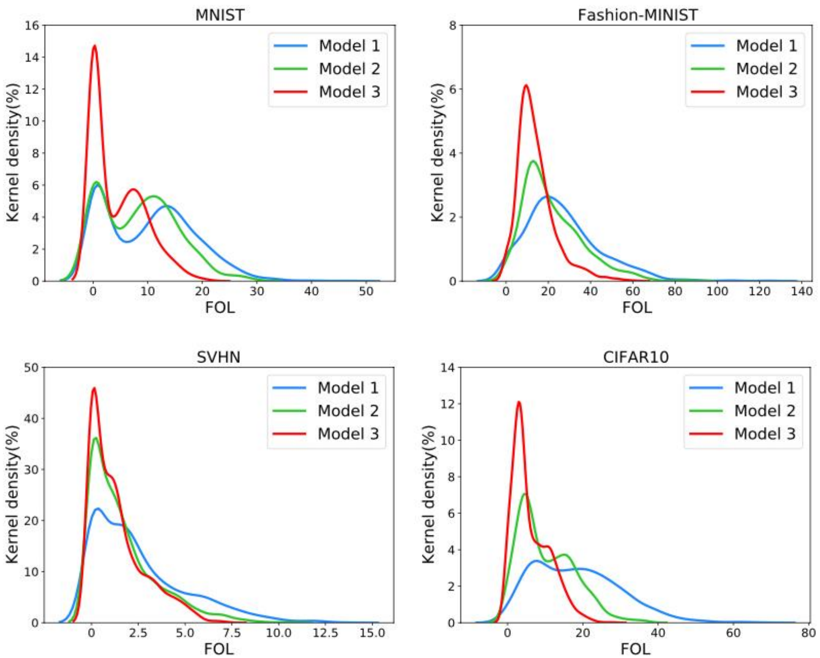
Evaluation

- **Benchmark / Model**
  - MNIST / Lenet-5
  - Fashion-MNIST / Lenet-5
  - SVHN / Lenet-5
  - CIFAR10 / ResNet-20

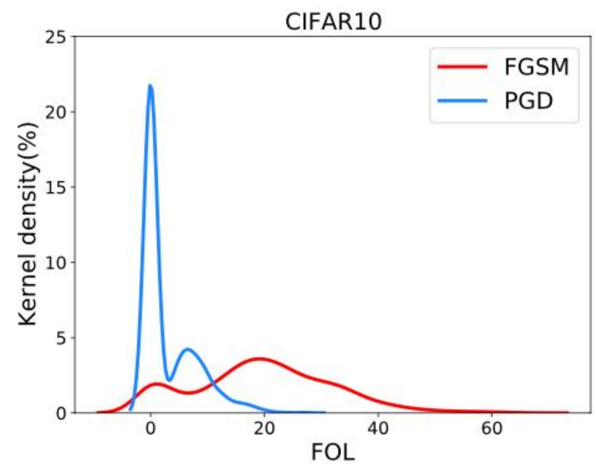
- **Test case generation**
  - FGSM & PGD
  - DeepXplore
  - DLFuzz
  - ADAPT

- **Research Questions**
  - RQ1: What is the correlation between our FOL metric and model robustness?
  - RQ2: How effective is our FOL metric for test case selection?
  - RQ3: How effective and efficient is our FOL guided fuzzing algorithm?

Evaluation



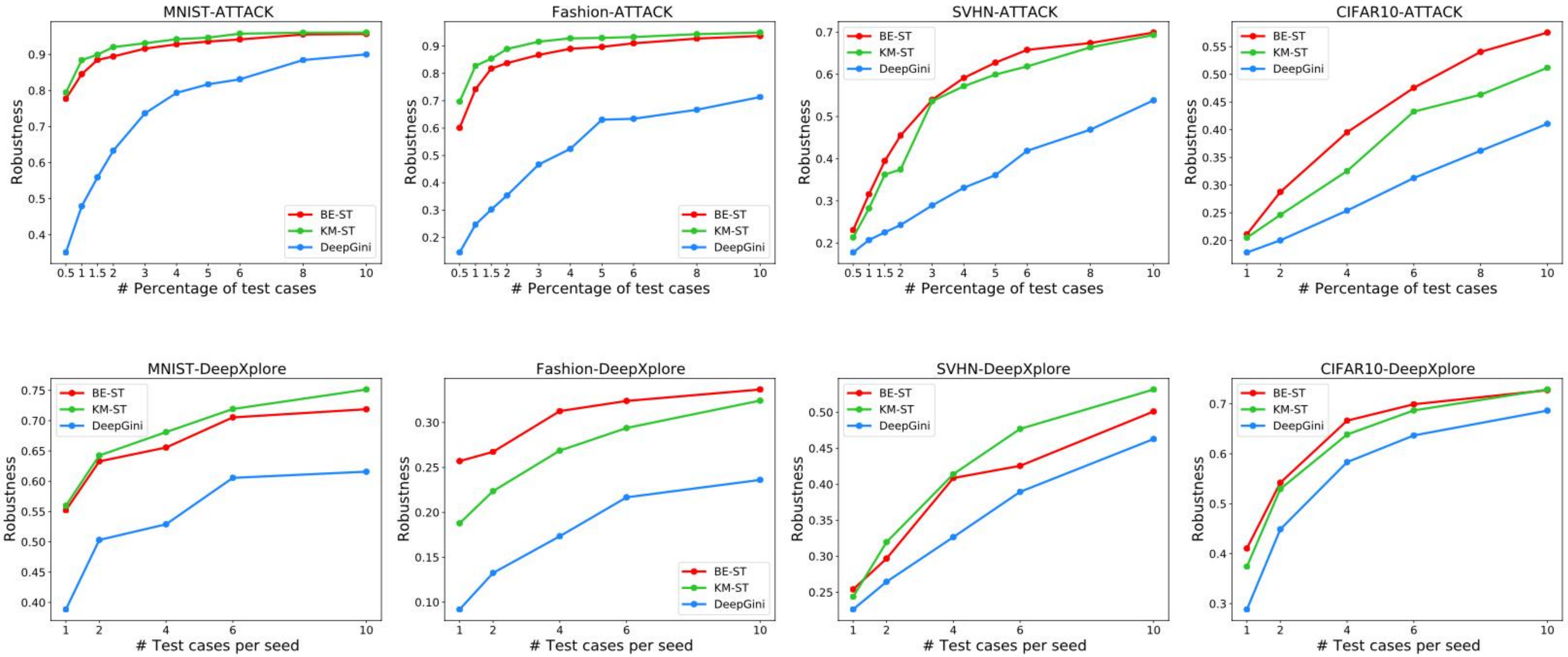
FOL distribution of adversarial examples for models with different robustness



FOL distribution of adversarial examples from FGSM and PGD for CIFAR10 model.

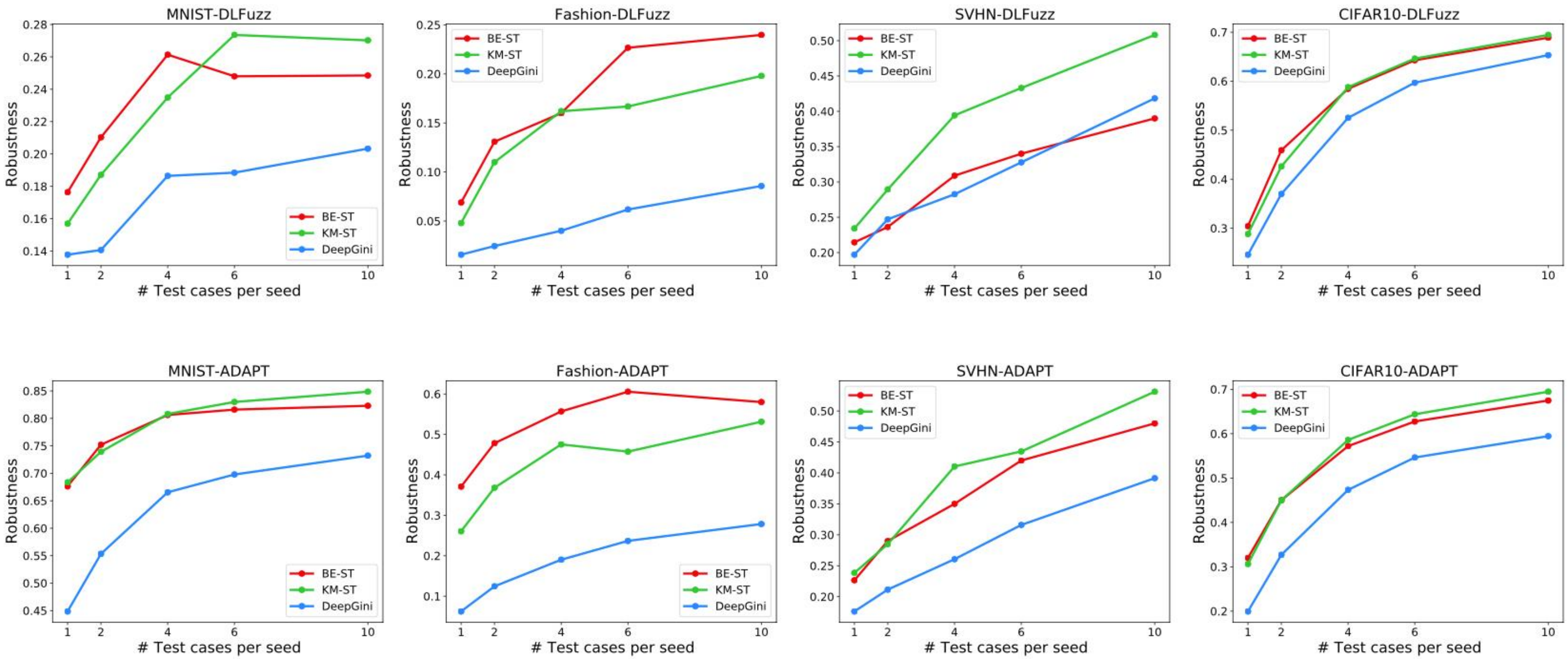
Answer to RQ1: FOL is strongly correlated with model robustness. A more robust model have smaller FOL values for adversarial examples.

Evaluation



Test case selection and robustness improvement with different strategies - I
ATTACK & DeepXplore

Evaluation



Test case selection and robustness improvement with different strategies - II
DLFuzz & ADAPT

Evaluation

Dataset	DeepXplore				DLFuzz				ADAPT			
	BE-ST	KM-ST	DeepGini	Average	BE-ST	KM-ST	DeepGini	Average	BE-ST	KM-ST	DeepGini	Average
MNIST	86.12%	80.56%	73.74%	**80.14%**	76.39%	74.73%	65.59%	**72.24%**	82.60%	75.68%	70.36%	**76.21%**
Fashion-MNIST	51.57%	47.97%	34.14%	**44.56%**	38.15%	35.44%	27.16%	**33.58%**	50.55%	47.50%	31.92%	**43.32%**
SVHN	37.10%	38.29%	27.26%	**34.55%**	32.83%	34.83%	25.34%	**31.00%**	25.71%	28.51%	19.15%	**24.46%**
CIFAR10	25.25%	20.16%	12.92%	**19.44%**	18.28%	14.20%	9.31%	**13.93%**	22.37%	18.48%	12.08%	**17.64%**
Average	**50.01%**	**46.75%**	**37.01%**		**41.41%**	**39.8%**	**31.85%**		**45.31%**	**42.54%**	**33.36%**	

Robustness performance of models (retrained using adversarial examples from attack algorithms) against test cases generated by DL testing tools.

FOL guided strategies have significantly better performance than DeepGini.

Answer to RQ2: FOL guided test case selection is able to select more valuable test cases to improve the model robustness by retraining.

Evaluation

Dataset	5 min # Test case	5 min Robustness↑	10 min # Test case	10 min Robustness↑	20 min # Test case	20 min Robustness↑
MNIST	1692/2125	33.62%/18.73%	3472/4521	48.04%/36.46%	7226/8943	68.02%/54.38%
Fashion-MNIST	4294/5485	40.75%/6.74%	8906/10433	53.88%/14.94%	18527/21872	69.03%/27.24%
SVHN	6236/8401	24.25%/21.3%	12465/17429	30.42%/27.52%	24864/33692	39.99%/34.51%
CIFAR10	1029/1911	18.62%/17.03%	2006/3722	22.07%/18.12%	4050/6947	27.36%/20.54%
Average	**3313/4480**	**29.31%/15.95%**	**6712/9026**	**38.6%/24.26%**	**13667/17864**	**51.1%/34.17%**

Comparison of FOL-fuzz and ADAPT. a/b: a is the result of FOL-fuzz and b is the result of ADAPT.

Answer to RQ3: FOL-Fuzz is able to efficiently generate more valuable test cases to improve the model robustness.

Conclusion

- We propose a novel robustness-oriented testing framework RobOT for deep learning systems towards improving model robustness.
  - A novel set of lightweight metrics that strongly correlated with model robustness.
  - A respective fuzzing strategy to automatically generate high-quality test cases for improving robustness.

- Achieving 50.65% more robustness improvement compared to the state-of-the-art DeepGini.

Thanks!