DeepXplore: Automated Whitebox Testing of Deep Learning Systems

Kexin Pei[1], Yinzhi Cao[2], Junfeng Yang[1], Suman Jana[1]

[1]Columbia University, [2]Lehigh University

Deep learning (DL) has matched human performance!

- Image recognition, speech recognition, machine translation, intrusion detection...
- Wide deployment in real-world systems

Amazon Echo

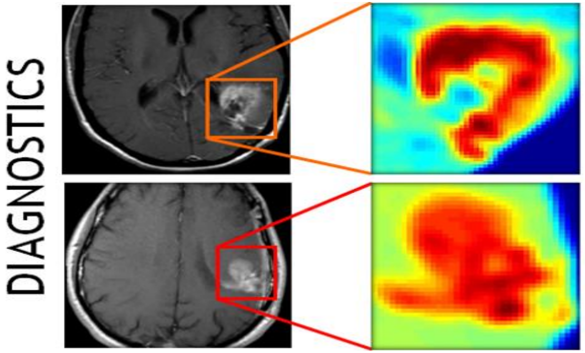
Iphone X

Google Earphone

Deep learning is increasingly used in safety-critical systems

- Deep learning correctness and security is crucial

Self-driving car



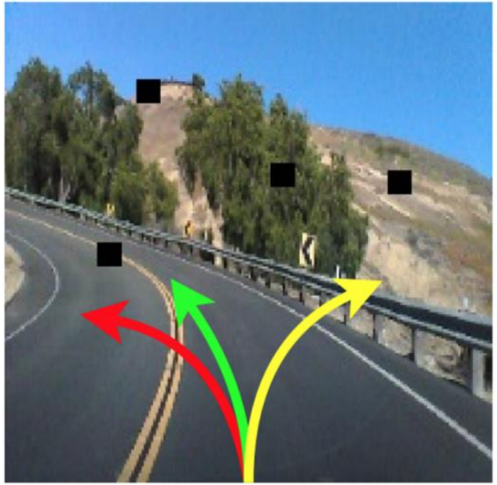
Medical diagnosis

Malware detection

Unreliable deep learning: fooling self-driving cars

DRV_C1:right

DRV_C2:right



DRV_C3:right

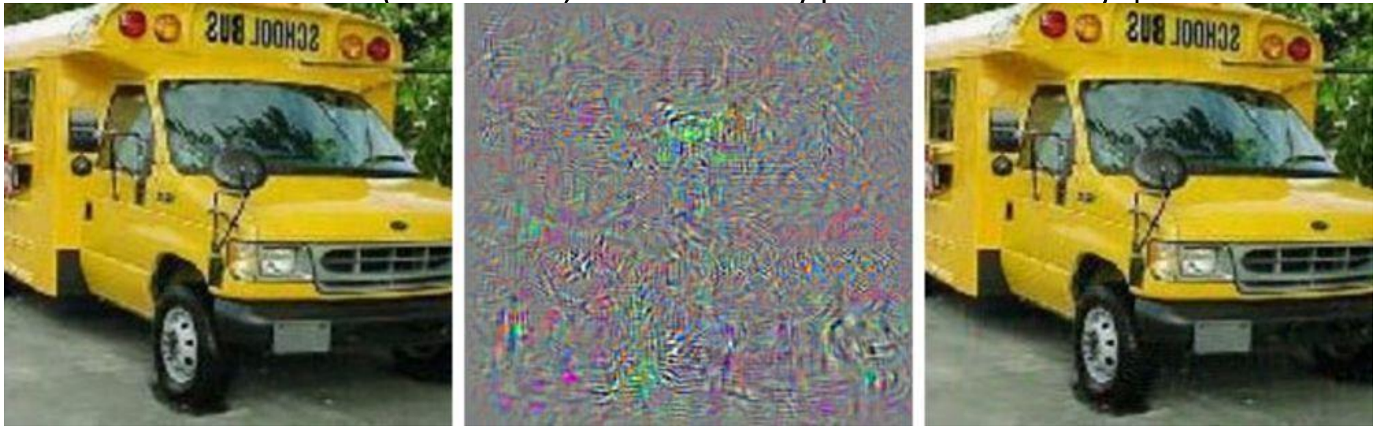
Unreliable deep learning: fooling face recognition

Existing DL testing methods are seriously limited

- Common practice: measure accuracy on a test input set of randomly chosen data
- Problem 1: how good is the coverage of the test set?
  - DL decision logic is incredibly complex
  - More fundamentally, what is testing coverage metric for DL?
- Problem 2: it requires expensive labeling effort
  - Data in test set must be manually labelled
  - To enlarge the test set, we need to manually label more data

Existing DL testing methods are seriously limited (cont.)

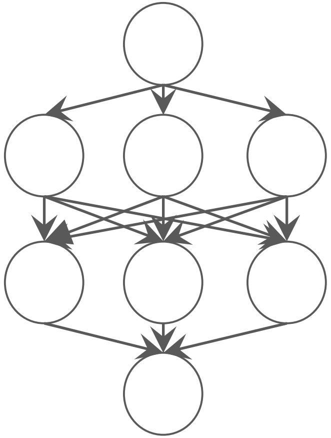
- Adversarial testing (Szegedy et al. ICLR’14): find corner-case inputs imperceptible to human but induce errors
    - Problem 1: how good is the coverage of the test set?
    - Problem 2: it requires expensive labeling effort
    - Problem 3: Not realistic. (Theoretical, assumes a very powerful adversary. [Sharif et al. CCS’16])



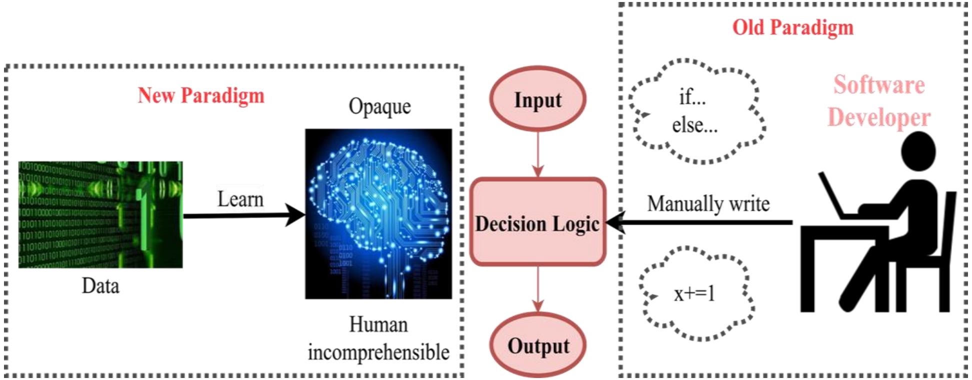
School bus | Carefully crafted noise | Ostrich

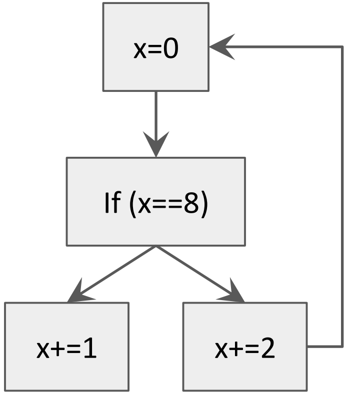
Many traditional software testing techniques don’t apply to DL

- DL decision logic is embedded in neurons and layers, not in code



Neural network

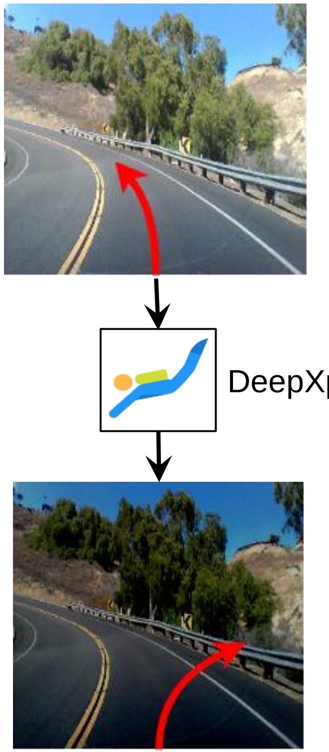




Traditional program
(control flow graph)

Quick Summary of DeepXplore

- The first step towards systematic testing of Deep Neural Nets (DNNs)
- Neuron coverage: first testing coverage metric for deep nerual net
- Automated: cross-check multiple DNNs
- Realistic: physically realizable transformations
- Effective:
    - 15 State-of-the-art DNNs on 5 large datasets (ImageNet, Self-driving cars, PDF/Android malware)
    - Numerous corner-case errors
    - 50% more neuron coverage than existing testing

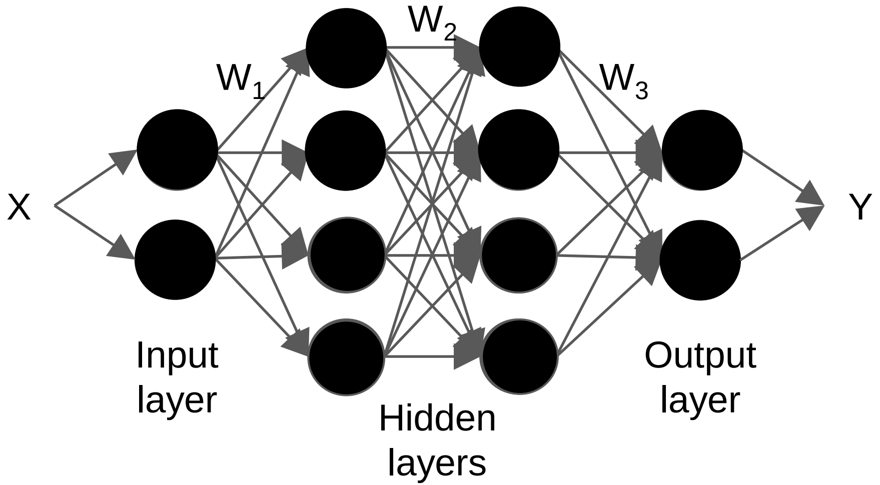


Outline

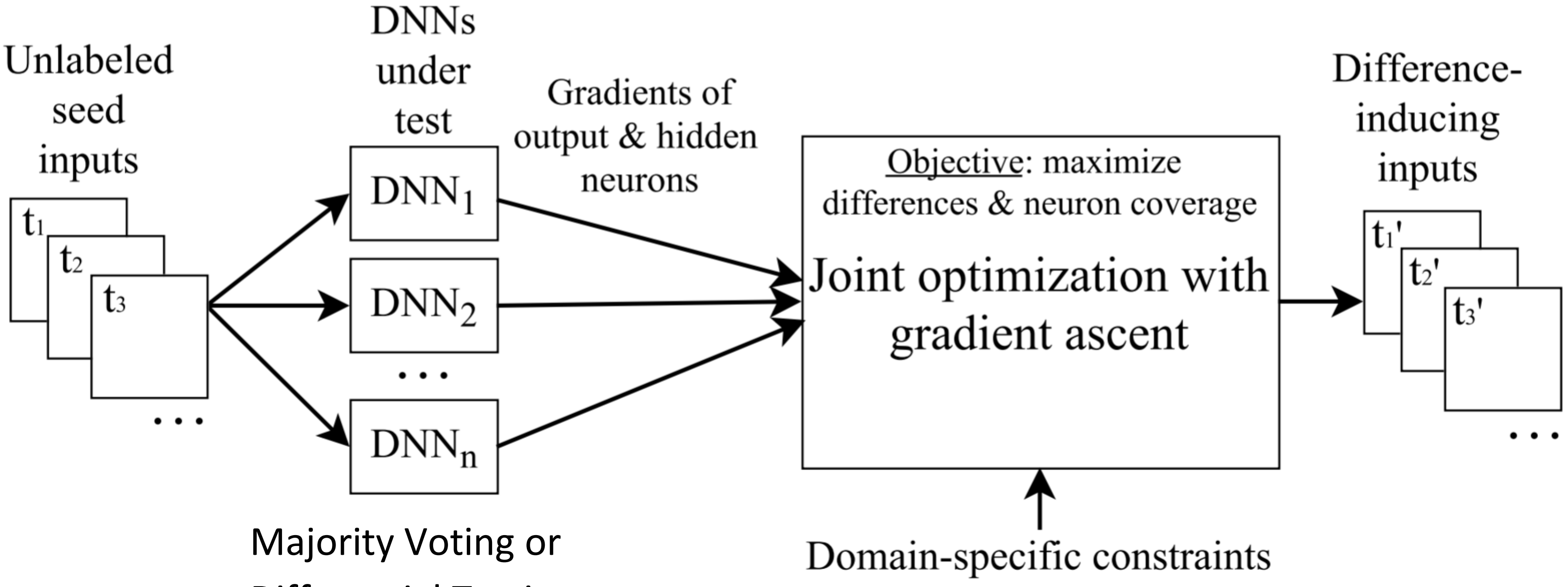
- Quick deep learning primer
- Workflow of DeepXplore
  - Design
  - Detail of Neuron coverage
- Implementation
- Evaluation setup and results summary

Deep learning primer

- A neural network is a function f(X) → Y
  - Trainable parameters (W_i) on each edge and nonlinear activation function at each neuron
  - DNN learns the weights during training
- <u>Training:</u> Given training set (X,Y), adjust W to minimize the prediction error (slow)
- <u>Inference</u>: Simply propagates X through layers (fast)

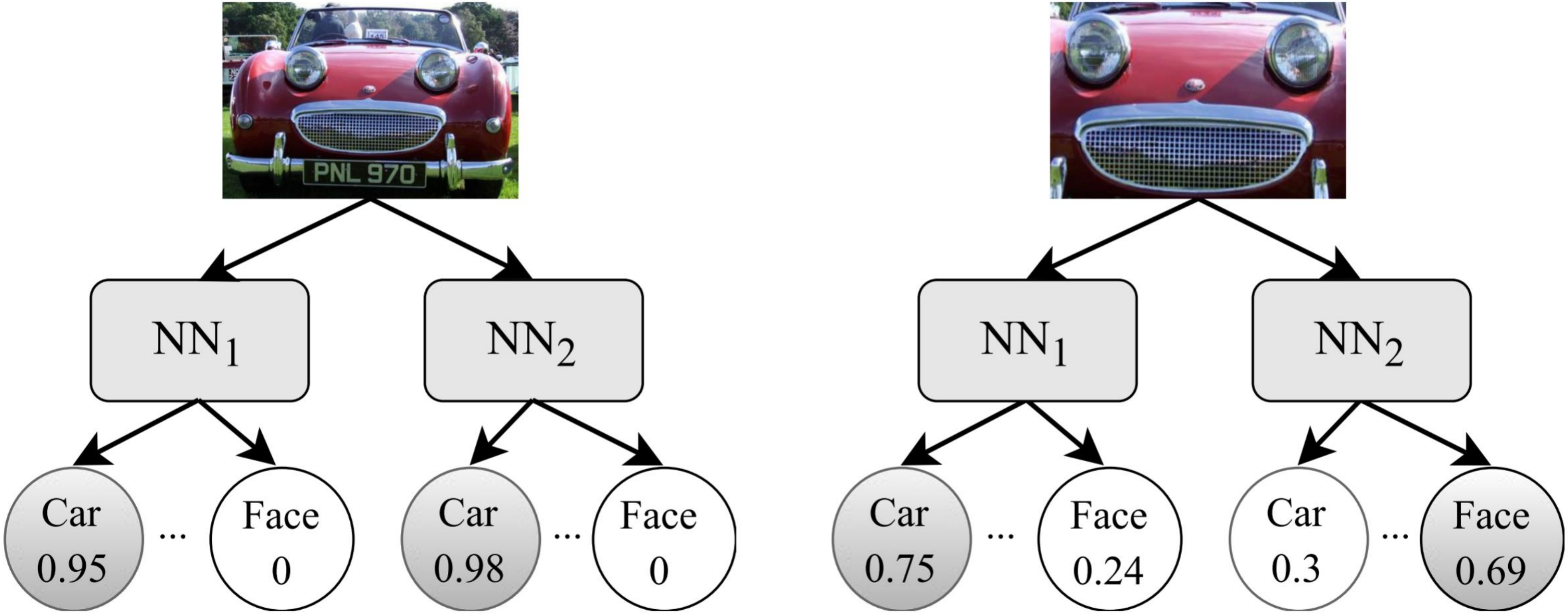


How DeepXplore Works?



Majority Voting or
Differential Testing

How To Maximize Differences?



How To Maximize Differences?

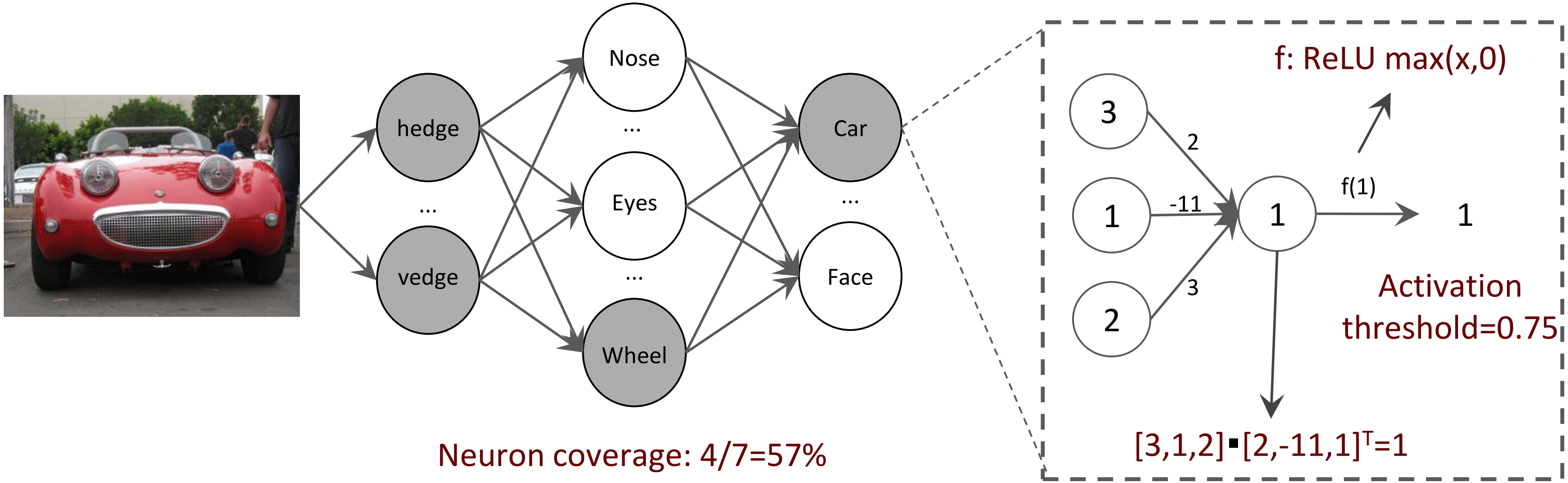
- MAXIMIZE

{sum of classification probabilities of other networks - classification probability of the debugging network}

$$obj_1(\boldsymbol{x}) = \Sigma_{k \neq j} F_k(\boldsymbol{x})[c] - \lambda_1 \cdot F_j(\boldsymbol{x})[c]$$

How To Maximize Neuron Coverage?

- Neuron coverage = # neurons activated at least once/ # total neurons
- A neuron is activated if it is output is larger than a threshold, e.g., 0.



How To Maximize Neuron Coverage?

- Randomly select an inacitivated neuron and maximze its output.

$$obj_2(\boldsymbol{x}) = f_n(\boldsymbol{x})$$

Joint Optimization

- Weighted sum of the above two objectives.

$$obj_{joint} = (\Sigma_{i \neq j} F_i(\boldsymbol{x})[c] - \lambda_1 F_j(\boldsymbol{x})[c]) + \lambda_2 \cdot f_n(\boldsymbol{x})$$

- Use **Gradient Ascent** to solve the above problem, the **variable is *x***, i.e., the input of the neural network.
- Given an original input of the network **x**, we may find a modified input ***x'*** *that causes different behaviors, but may not.*

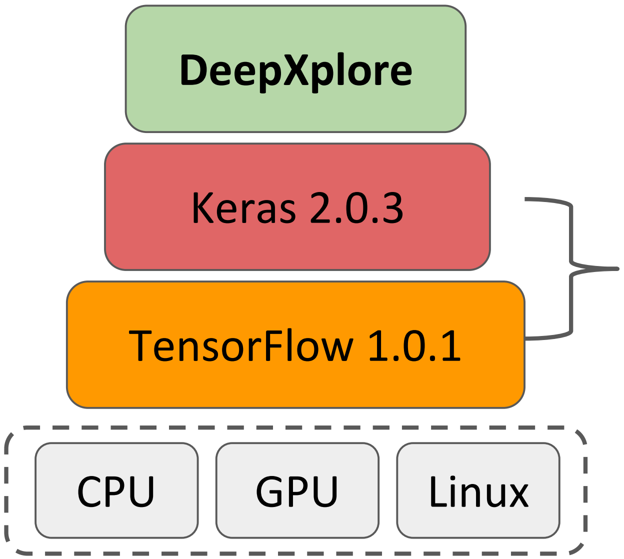
How to achieve multiple goals simultaneously

- Goal 1: systematically find corner cases
  - Generate inputs that maximize neuron coverage
- Goal 2: find DNN errors without manual labels
  - Differential testing: use multiple DNNs as cross-referencing oracles
- Goal 3: generate realistic test inputs
  - Domain-specific constraints, e.g., a pixel value must be an integer within [0,255]

Outline

- Quick deep learning primer
- Workflow of DeepXplore
  - Design
  - Detail of Neuron coverage
- Implementation
- Evaluation setup and results summary

Implementation



Efficient gradient computation &
support for intercepting outputs of intermediate neurons
for calculating neuron coverage

Outline

- Quick deep learning primer
- Workflow of DeepXplore
  - Design
  - Detail of Neuron coverage
- Implementation
- Evaluation setup and results summary

Evaluation setup and results summary

Dataset	Description	DNNs	Original random testing accuracy	Avg. neuron coverage improvement over random/adversarial	Avg. Violations found by DeepXplore (2000 seeds)
MNIST	Handwritten digits	LeNet variants	98.63%	30.5% → 70%	1,289
ImageNet	General Images in 1000 categories	VGG16, VGG19, ResNet15	93.91%	1% → 69%	1,980
Driving	Udacity self-driving car competition dataset	Nvidia Dave-2 variants	99.94%	3.2% → 59%	1,839
Contagio/ VirusTotal	PDF malware	Fully connected	96.29%	18% → 70%	1,048
Drebin	Android malware	Fully connected	96.03%	18.5% → 40%	2,000

Sample corner-case errors for images

Driving:


Go straight

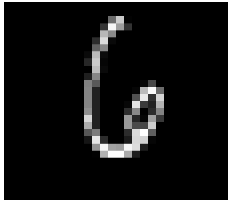

Turn right

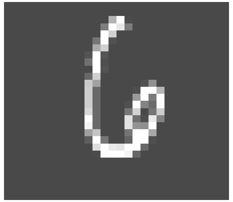
ImageNet:


Light cardigan


Diaper

MNIST
:


6


2

Driving:


Go straight

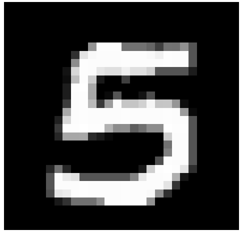

Turn left

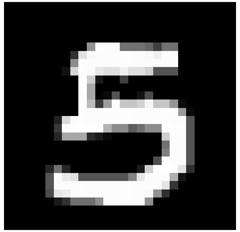
ImageNet:


bolete


buckeye

MNIST
:


5


3

Performance

- For all tested DL models, on average **DeepXplore generated one test input demonstrating incorrect behavior within one second** while running only on a commodity laptop.

- Given a set of DNNs, we can use majority voting to automatically label the exceptional cases generated by DeepXplore. Using these as new labeled training examples improves accuracy by 1-3%.

Conclusions

- Systematically testing DL for realistic corner cases is a hard problem
- DeepXplore is the first step for systematic DL testing
  - Neuron coverage: first testing coverage metric for deep nerual net
  - Automated: differential testing by cross-checking multiple DNNs
  - Realistic: physically realizable transformations
  - Effective: find neumerous unexpected corner-case errors

- A lot of exciting new research problems!
  - Build analysis tools for testing and verification of ML
  - Build better debugging support for opaque ML

Comments

- I like the ideas:
  - Apply traditional software debugging techniques (code coverage, differential testing) to debug DNNs.
  - Formulate the testing problem as an joint-optimization problem and solve it using gradient-based method.
- This paper focuses on **classification problem** in supervised learning.
- Neuron coverage can not cover all possible inputs.
- The constraints are not flexible enough and the resulting inputs can only be the input used for training, plus some minor modifications.
- DL systems probably can’t handle all safety issues alone, we need non-DNN “safety-wrapper”.

Thank you!

Questions?