Formal Specification for Machine Learning Systems

Sanjit A. Seshia

Professor

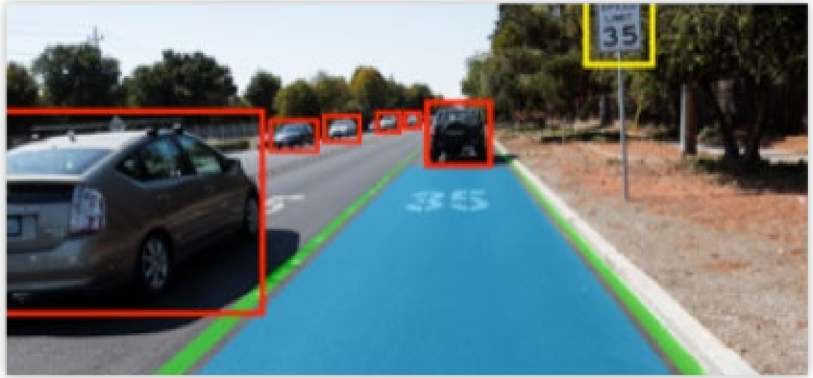
EECS Department, UC Berkeley

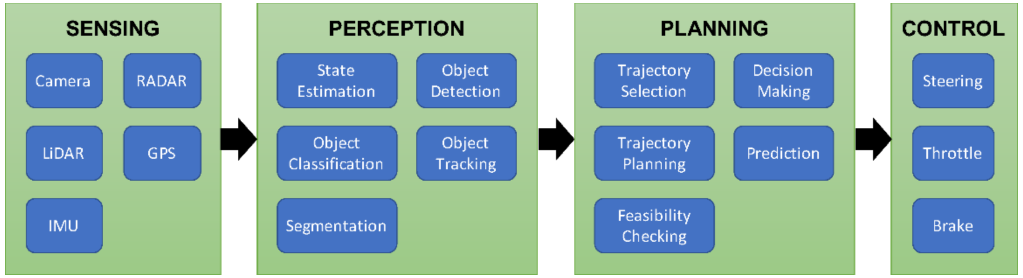
http://learnverify.org/VerifiedAI

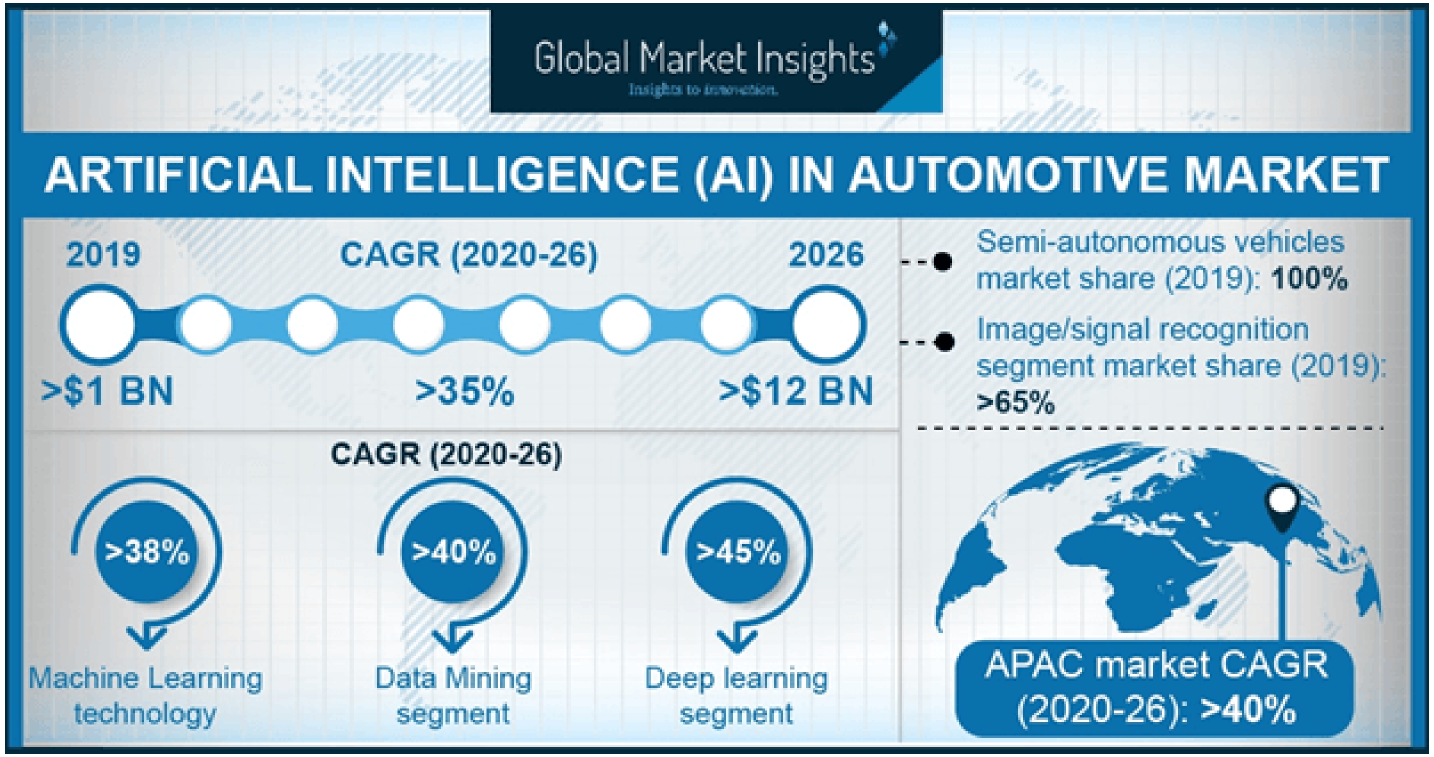
HYPER 2021
October 18, 2021

http://vehical.org

Growing Use of Machine Learning/Artificial Intelligence in Safety-Critical Autonomous & Semi-Autonomous Systems







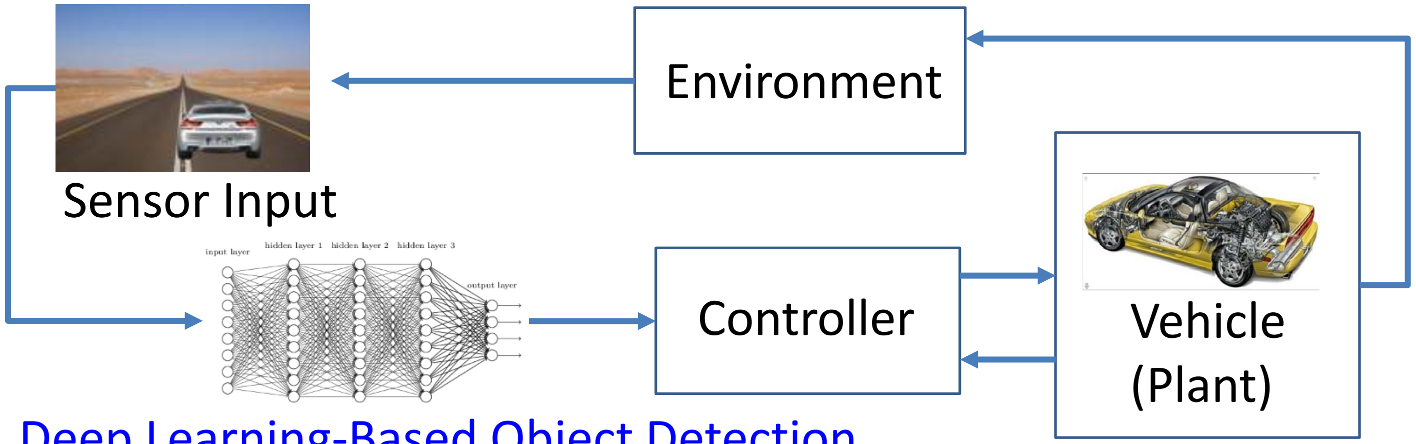
Source: gminsights.com

Growing Concerns about Safety:

- Numerous papers showing that *Deep Neural Networks can be easily fooled*
- *Accidents*, including some *fatal,* involving potential failure of AI/ML-based perception systems in self-driving cars

Can we address the Design & Verification Challenges of AI/ML-Based Autonomy with Formal Methods?

Example: Automatic Emergency Braking System (AEBS) using *Deep Learning for Perception*



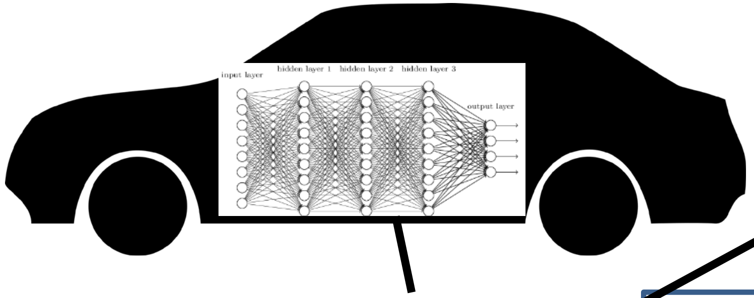
Deep Learning-Based Object Detection

- Goal: Brake when an obstacle is near, to *maintain a minimum safety distance*
- Modeling: Closed-Loop system modeled in a *software-in-the-loop simulator* (Matlab/Simulink, Udacity, Webots, CARLA, …)
- Perception: Object detection/classification system based on *deep neural networks*
  - Inception-v3, AlexNet, … trained on ImageNet
  - squeezeDet, Yolo, … trained on KITTI

[Dreossi, Donze, Seshia, "Compositional Falsification of Cyber-Physical Systems with Machine Learning Components", NASA Formal Methods (NFM), May 2017.]

Challenges for Verified AI

S. A. Seshia, D. Sadigh, S. S. Sastry.
Towards Verified Artificial Intelligence. July 2016. https://arxiv.org/abs/1606.08514.



System **S** →
Environment **E** →
Specification **φ** →

Does S || E satisfy φ?

→ YES [+ proof]
→ NO [+ counterexample]

Need to Search Very High-Dimensional Input and State Spaces

Design Correct-by-Construction?

Need Principles for Verified AI

Challenges

1. Environment (incl. Human) Modeling
2. Formal Specification
3. Learning Systems Representation
4. Scalable Training, Testing, Verification
5. Design for Correctness

Principles

?

S. A. Seshia, D. Sadigh, S. S. Sastry. *Towards Verified Artificial Intelligence*. July 2016. https://arxiv.org/abs/1606.08514.

http://learnverify.org/VerifiedAI

Outline

- Challenges for Formal Specification of AI/ML Systems
- Component-Level Specification
- Robustness
- System-Level Specification
- Environment Modeling
- Principles for Verified AI

Challenges for Formal Specification of AI/ML Systems

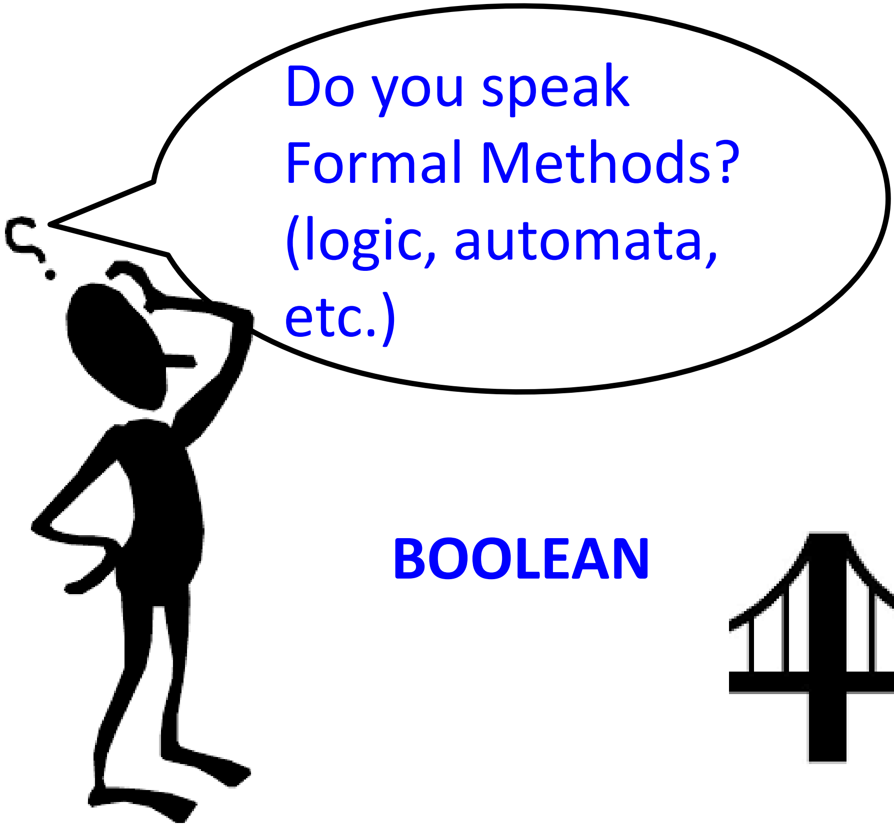
Challenge 1: Hard to Formalize Tasks

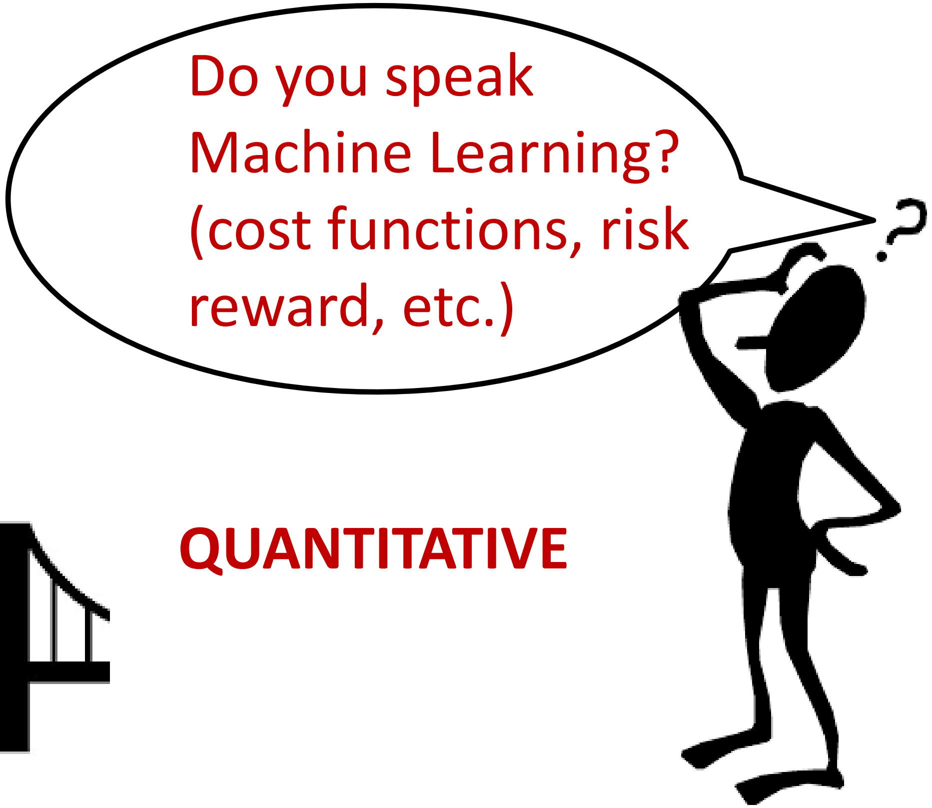
Many of the impactful applications of ML/Deep Learning are in **Perceptual Tasks**

How do you specify "a car"?

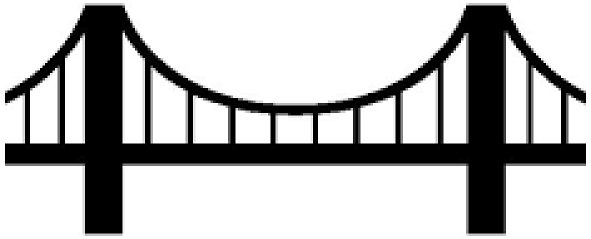
What Specifications are Meaningful for Hard-to-Formalize Tasks?

Challenge 2: Boolean vs. Quantitative Specifications





BOOLEAN



QUANTITATIVE

- More Composable
- Fit with Formal Tools

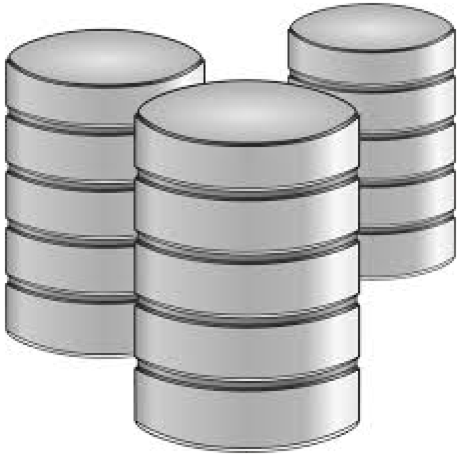
How do we bridge the gap?

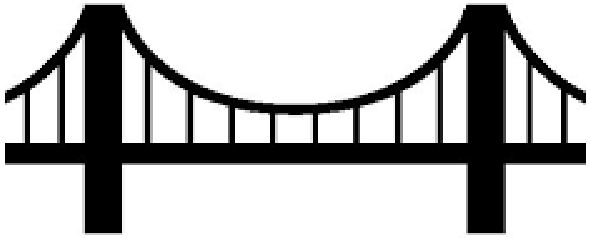
- More Flexible
- Fit with Optimization

Challenge 3: Data vs. Formal Specification

MACHINE LEARNING

FORMAL METHODS





$\varphi_1 \vee (\varphi_2 \wedge \varphi_3)$

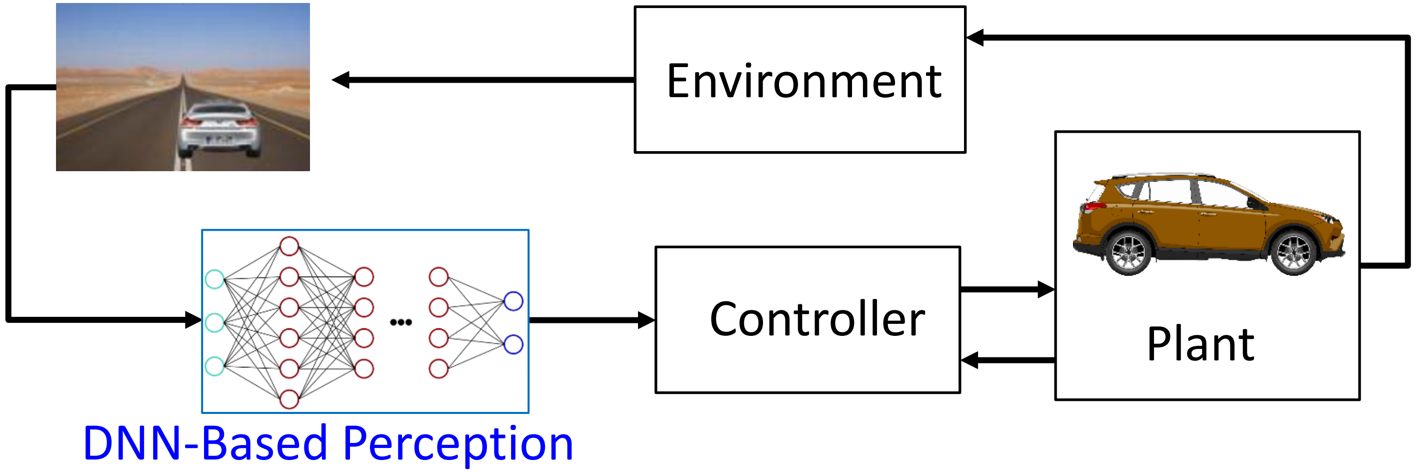
How do we bridge the gap?

What Properties must we Verify?

Taxonomy of Properties: Multiple Dimensions

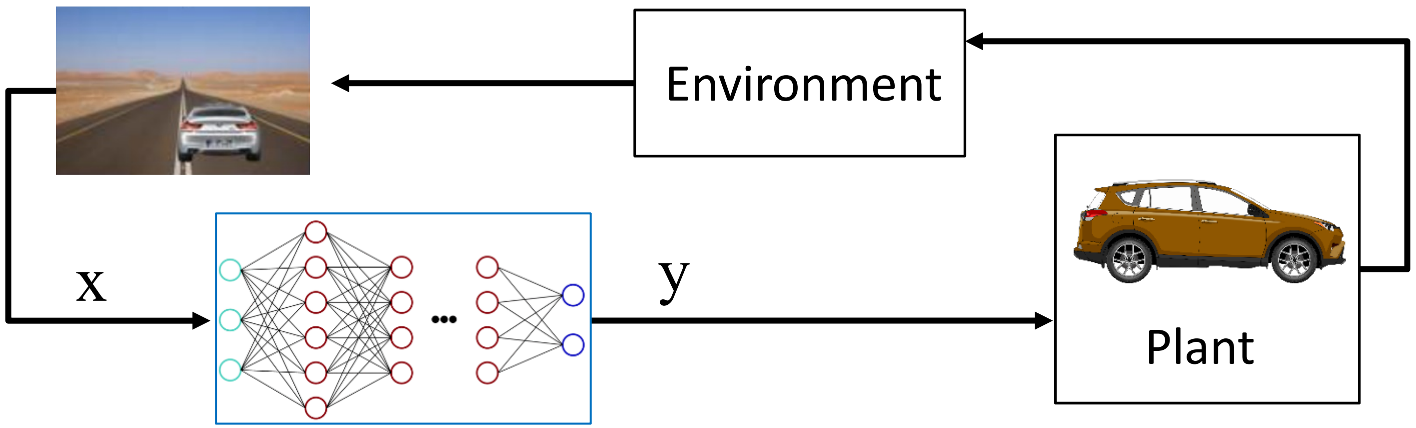
1. System-level vs. Component-level
2. Trace Properties vs. HyperProperties
3. Boolean vs. Quantitative
4. Purpose: Robustness, Safety, Fairness, etc.

System-Level, Boolean Trace Property



$$\mathbf{G}\ [\mathrm{AV\_moving} \Rightarrow \mathrm{dist}(\mathbf{x}_{\mathbf{AV}}, \mathbf{x}_{\mathbf{env}}) > \mathbf{\Delta}]$$

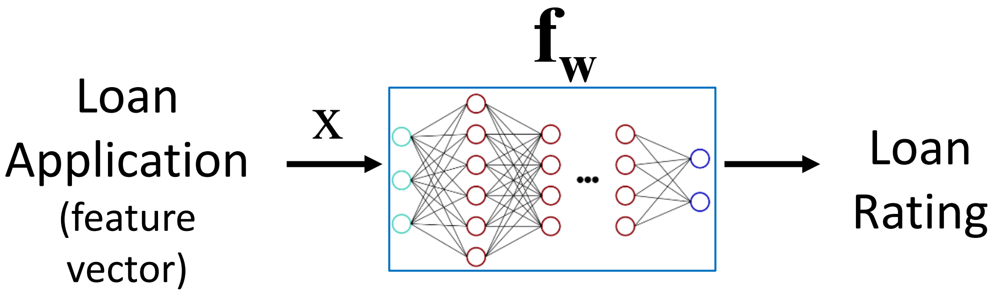
Component-Level, Boolean Trace Property



For a given $\mathbf{x_1}$, ε, δ and for all $\mathbf{x_2}$:

$$d_i(\mathbf{x_1}, \mathbf{x_2}) \leq \epsilon \implies \mathbf{d_o}(\mathbf{y_1}, \mathbf{y_2}) \leq \delta$$

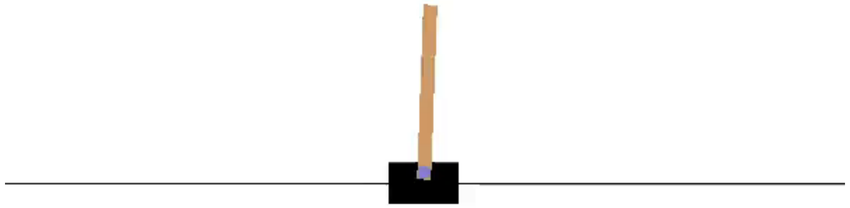
Component-Level, Boolean HyperProperty



DNN-based Loan Decision Risk Rating System

$$\forall \mathbf{x_1}, \mathbf{x_2}. \mathbf{x_{1,sal}} \leq \mathbf{x_{2,sal}} \implies \mathbf{f_w}(\mathbf{x_1}) \leq \mathbf{f_w}(\mathbf{x_2})$$

System-Level, Quantitative Trace/Hyper Property



Cart-pole Balancing
[Barto et al., '83] (from OpenAI Gym)

Reward $r(t)$=1 each step it is upright

Reward for every finite-time horizon trace τ

$$R_\tau(T) = \sum_{t=1}^{T} r(t)$$

For the set of all traces $\mathcal{T}$

$$\bar{R}_{\mathcal{T}}(T) = \inf_{\tau \in \mathcal{T}} R_\tau(T)$$

Formal Specification for ML: Classification by Purpose

[S. A. Seshia, et al., "Formal Specification for Deep Neural Networks", ATVA 2018]

- System Level (for ML based systems)
  - Similar to other systems (safety, liveness, stability, etc.)
- Component Level (for ML models as components)
  - Robustness: local vs. global, syntactic vs. semantic
  - Input-Output Relations
  - Monotonicity
  - Fairness
  - Coverage
  - Semantic Invariance (e.g. output invariant to geometric transformations)
  - Distributional Assumptions & Corresponding Guarantees
  - …
- Properties of ML Algorithms: e.g., for Stochastic Gradient Descent: "stochastic backpropagation procedure yields unbiased estimates of the true mathematical gradients" [Selsam et al, '17]

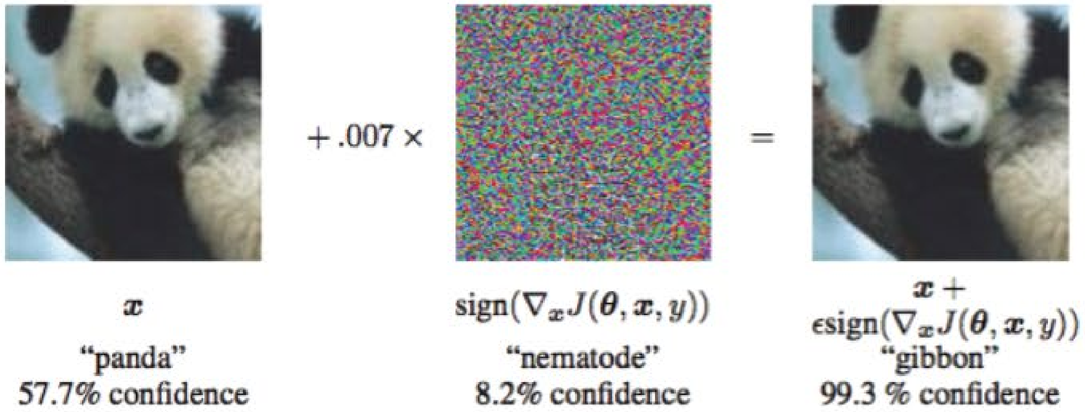
Fairness of ML Models: 3 Broad Flavors

[S. A. Seshia, et al., "Formal Specification for Deep Neural Networks", ATVA 2018]

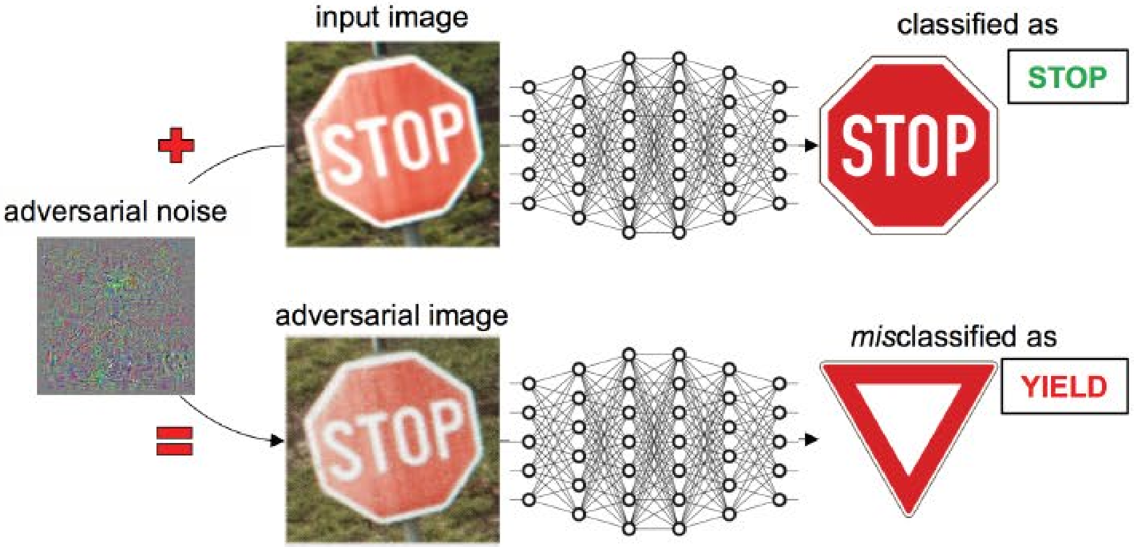
- Individual / Similarity-Based
  - View of ML model operating on individual inputs
  - E.g., Similar inputs mapped to similar outputs (cf. robustness)
- Group / Population-Based
  - View of ML model operating on population of data
  - E.g., Probability of getting a particular output is independent of certain features
- Counterfactual
  - Decision of ML model same in actual world and a counterfactual world

Robustness

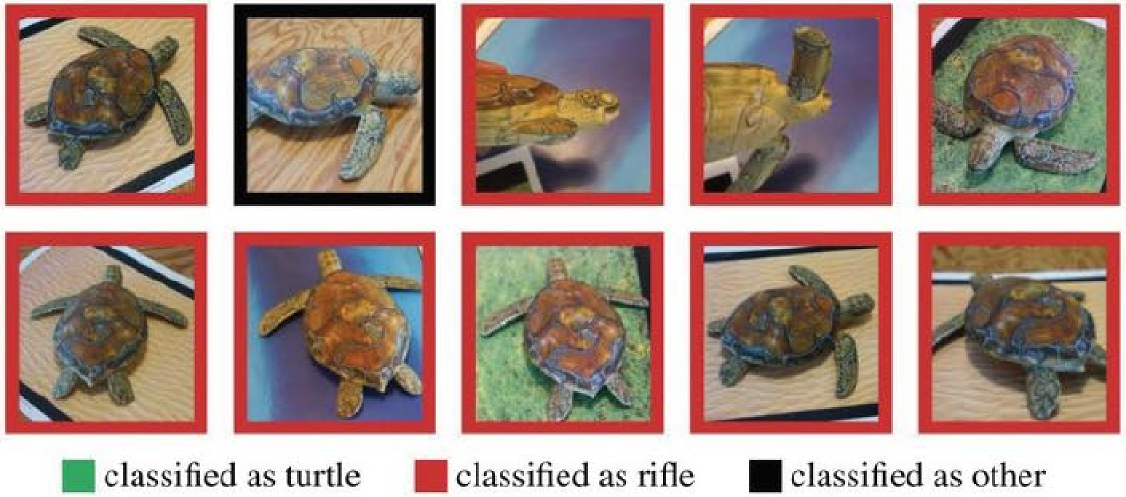
Robustness to Adversarial Inputs/Mutations

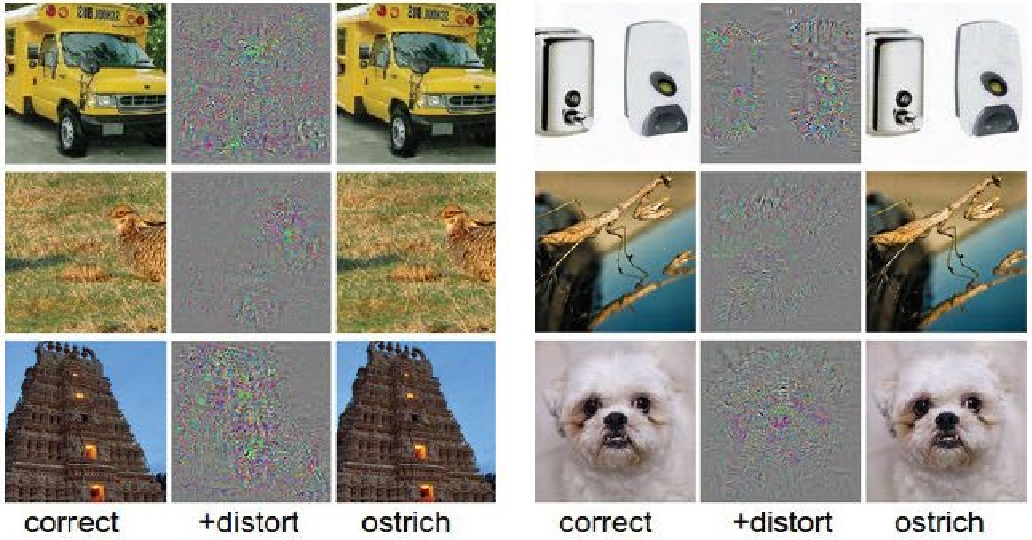


Explaining and Harnessing Adversarial Examples, Goodfellow et. al



Slides by Andrej Karpathy





Intriguing Properties of Neural Networks, Szegedy et. al

(Local) Robustness – Adversarial ML Test-Time Attacks

- Given a specific input **x** to an ML model (e.g. deep neural network), find a small perturbation **x*** of that input that produces an "incorrect" output

- If no such perturbation is possible, the ML model is **robust** (to the test-time attack)
  - *Locally robust* around input **x**

- **Problem:** No uniform way to define adversary or attacks!

[see, for example, Goodfellow et al, article in CACM 2018]

(Local) Robustness – A General Formulation

Given: Input $x \in X$

NN $f : X \rightarrow Y$

DECISION PROBLEM FORMULATION

Find: Adversarial example x^* which satisfy,

(1) Admissibility constraint: $x^* \in \tilde{X}$ ← Only "Valid" Perturbations allowed

(2) Distance constraint: $D(\mu(x, x^*), \alpha)$ ← Perturb within a specified "distance"

(3) Target behavior constraint: $A(x, x^*, \beta)$ ← Adversarial goal

[Dreossi, Ghosh, Sangiovanni-Vincentelli, Seshia, "A General Formalization of Robustness for Deep Neural Networks", VNN'19]

(Local) Robustness – A General Formulation

DECISION PROBLEM

Given: Input $x \in X$

NN $f : X \to Y$

Find: Adversarial example x^* which satisfy,

(1) Admissibility constraint: $x^* \in \tilde{X}$

(2) Distance constraint: $D(\mu(x, x^*), \alpha)$

(3) Target behavior constraint: $A(x, x^*, \beta)$

OPTIMIZATION FORMULATION

Minimizing Perturbation

$$x^* = \underset{x' \in \tilde{X}}{\text{argmin}} \ \alpha \quad \text{s.t.} \quad \mu(x, x') \leq \alpha$$
$$A(x, x', \beta)$$

Maximizing Loss

$$x^* = \underset{x' \in \tilde{X}}{\text{argmax}} \ \beta \quad \text{s.t.} \quad L(f(x), f(x')) \geq \beta$$
$$\mu(x, x') \leq \alpha$$

[Dreossi, Ghosh, Sangiovanni-Vincentelli, Seshia, "A General Formalization of Robustness for Deep Neural Networks", VNN'19]

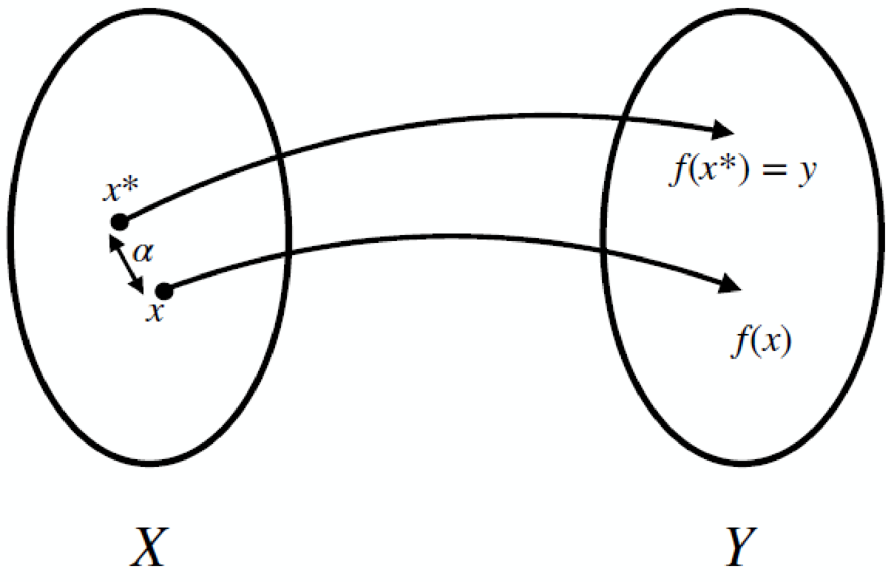
Ex 1: Minimum Perturbation, Targeted Attacks

DECISION PROBLEM:

$$x^* := x + r \in X$$

$$D(\mu(x, x^*), \alpha) := \|r\|_p \leq \alpha$$

$$A(x, x^*, \beta) := f(x^*) = y \quad (y \in Y \backslash f(x))$$



OPTIMIZATION PROBLEM:

$$x^* = \underset{x' \in X}{\text{argmin}} \ \alpha$$

$$\text{s.t.} \quad x' = x + r$$

$$\|r\|_p \leq \alpha$$

$$f(x') = y$$

$$p \in \{0, 2, \infty\}$$

Intriguing Properties of Neural Networks, Szegedy et. al
Explaining and Harnessing Adversarial Examples, Goodfellow et. al (FGSM-Fast gradient sign method)
Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks, Papernot et. al
Towards Evaluating the Robustness of Neural Networks, Carlini and Wagner
The Limitations of Deep Learning in Adversarial Settings, Papernot et. al (JSMA- Jacobian based Saliency Map Attack)

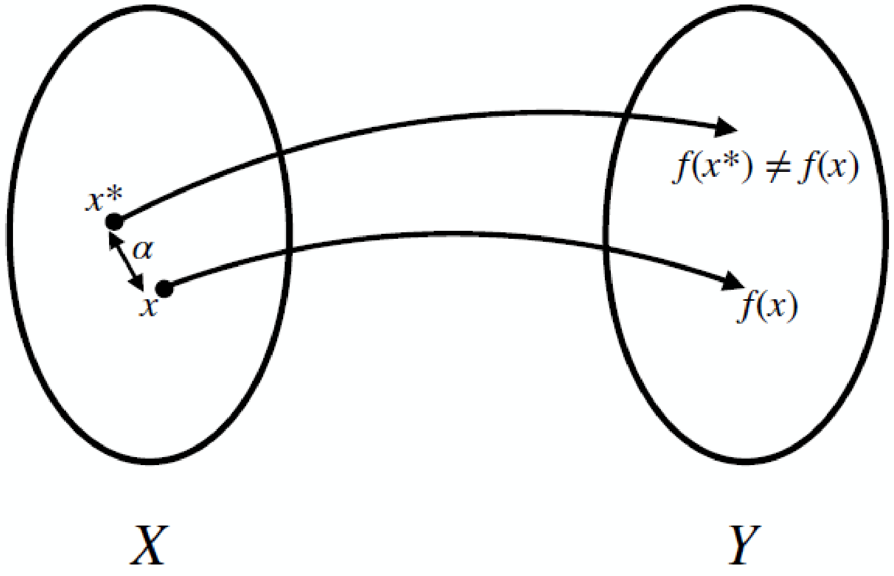
Ex 2: Minimum Perturbation, Untargeted Attacks

DECISION PROBLEM:

$$x^* := x + r \in X$$

$$D(\mu(x, x^*), \alpha) := \|r\|_p \leq \alpha$$

$$A(x, x^*, \beta) := f(x^*) \neq f(x)$$



OPTIMIZATION PROBLEM:

$$\begin{aligned} x^* = \ & \underset{x' \in X}{\text{argmin}} \ \alpha \\ \text{s.t.} \ & x' = x + r \\ & \|r\|_p \leq \alpha \\ & f(x') \neq f(x) \\ & p \in \{0, 2, \infty\} \end{aligned}$$

DeepFool: a simple and accurate method to fool deep neural networks , Moosavi-Dezfooli et. al
Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples, Athalye et. al
Towards Fast Computation of Certified Robustness for ReLU Networks, Weng et. al

Ex 3: Maximum Loss, Untargeted Attacks

DECISION PROBLEM:

$$x^* := x + r \in X$$

$$D(\mu(x, x^*), \alpha) := ||r||_\infty \leq \alpha$$

$$A(x, x^*, \beta) := L(\theta, x^*, y) \geq \beta$$

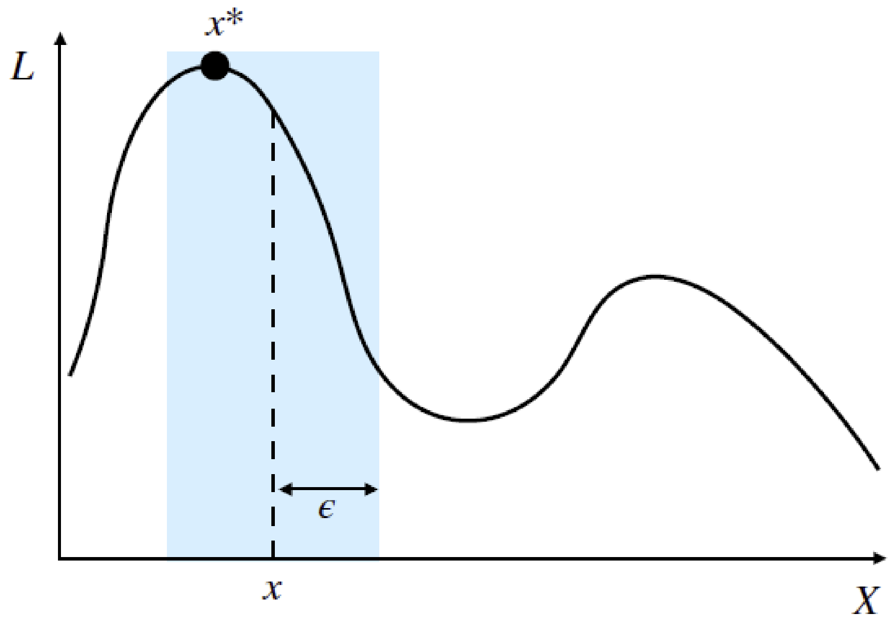
OPTIMIZATION PROBLEM:

$$x^* = \underset{x' \in X}{\text{argmax}} \; \beta$$

$$\text{s.t.} \quad x' = x + r$$

$$||r||_\infty \leq \alpha$$

$$L(\theta, x, y) \geq \beta$$



Towards Deep Learning Models Resistant to Adversarial Attacks, Madry et. al

Ex 4: Adversarial Examples Robust to Transformations

DECISION PROBLEM:

Define : Set of transformations T

x^* must remain adversarial when transformed $t(x^*) \forall t \in T$

$$x^* := x + r \in X$$

$$D(\mu(x, x^*), \alpha) := \mathbb{E}_{t \in T}[d(t(x), t(x^*))] \leq \alpha$$

$$A(x, x^*, \beta) := \mathbb{E}_{t \in T}[\log P(y_t \mid t(x^*))] \geq \beta$$

OPTIMIZATION PROBLEM:

$$x^* = \underset{x' \in X}{\text{argmax}} \ \beta$$

$$\text{s.t.} \quad x' = x + r$$

$$\mathbb{E}_{t \in T}[d(t(x), t(x^*))] \leq \alpha$$

$$\mathbb{E}_{t \in T}[\log P(y_t \mid t(x^*))] \geq \beta$$

Synthesizing Robust Adversarial Examples, Athalye et. al

Ex 5: Breaking Input-Output Relations

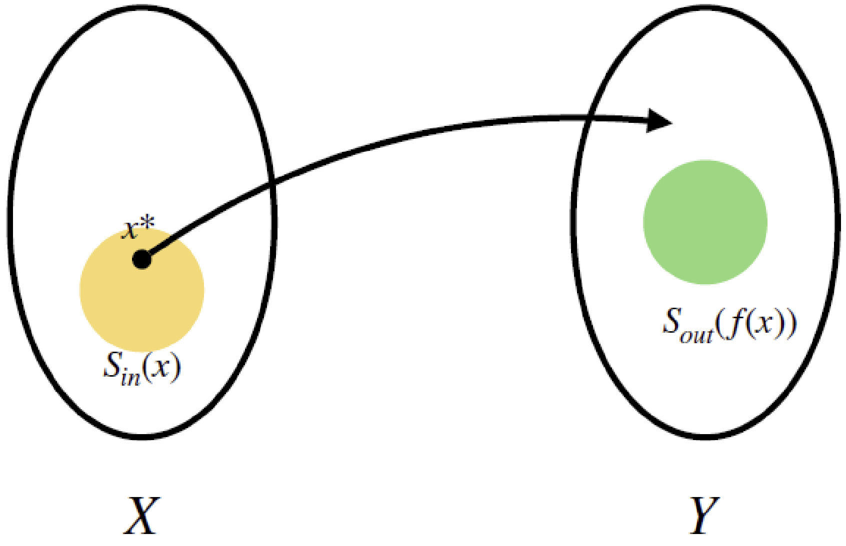
DECISION PROBLEM:

$$\boxed{\forall x' \in S_{in}(x) \Rightarrow f(x') \in S_{out}(f(x))}$$

$$x^* := x + r \in X$$

$$D(\mu(x, x^*), \alpha) := x^* \in S_{in}(x) \subseteq X$$

$$A(x, x^*, \beta) := f(x^*) \notin S_{out}(f(x))$$



A Dual Approach to Scalable Verification of Deep Networks, Dvijotham et. al
Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks, Katz et. al
Safety Verification of Deep Neural Networks, Huang et. al
Output Range Analysis for Deep Feedforward Neural Networks, Dutta et. al
Reachability Analysis of Deep Neural Networks with Provable Guarantees, Ruan et. al

Adversary Type	Admissibility Constraint $x^* \in \tilde{X}$	Distance Constraint $D(\mu(x, x^*), \alpha)$	Target Constraint $A(x, x^*, \beta)$
Minimum Perturbation Adversary, Targeted Attacks	$x^* = x + r \in X$	$\|r\|_p \leq \alpha$	$f(x^*) = y$
Minimum Perturbation Adversary, Untargeted Attacks	$x^* = x + r \in X$	$\|r\|_p \leq \alpha$	$f(x^*) \neq f(x)$
Maximize Loss Adversary, Untargeted Attacks	$x^* = x + r \in X$	$\|r\|_\infty \leq \alpha = \epsilon$	$L(\theta, x^*, y) \geq \beta$
Robust Adversarial Examples	$x^* = x + r \in X$	$\mathbb{E}_{t \in T}[d(t(x), t(x^*))] \leq \alpha = \epsilon$	$\mathbb{E}_{t \in T}[\log P(y_t \mid t(x^*))] \geq \beta$
Input Output Relations	$x^* = x + r \in X$	$x^* \in S_{in}(x)$	$f(x^*) \notin S_{out}(f(x))$
Black-Box Transferable Attacks	$x^* = x + r \in X$	$\|r\|_2 \leq \alpha$	$f_{sub}(x^*) = y, f_{sub}(x^*) \neq f_{sub}(x)$ $f_{sub}(x^*) \neq f_{sub}(x) \rightarrow f(x^*) \neq f(x)$
Neuron Coverage	$x^* \in X$	$x^* \in \{\gamma x, x + r\}$	$f_1(x) = \ldots = f_k(x) \Rightarrow f_i(x^*) \neq f_j(x^*)$ $F_n(x^*) \geq \beta$

[Dreossi, Ghosh, Sangiovanni-Vincentelli, Seshia, "A General Formalization of Robustness for Deep Neural Networks", VNN'19]

Robustness to Adversarial Inputs/Mutations

Q1: Can these mutations occur in practice in the environment?

Q2: What about "big" mutations in pixel space producing "equivalent" i/p?

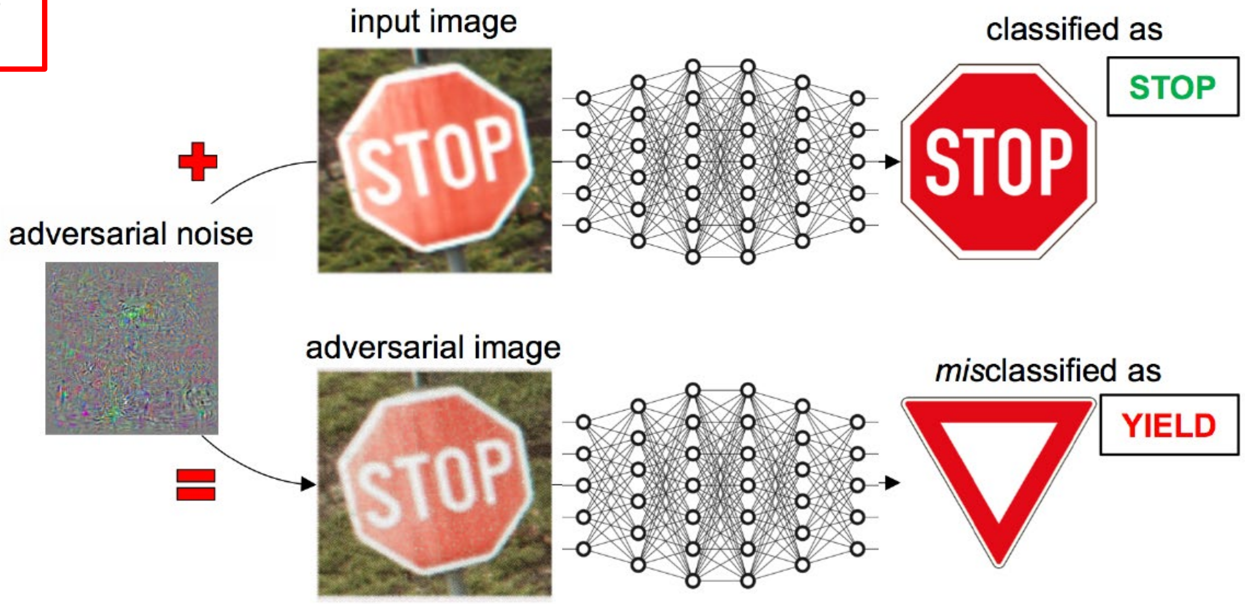
Q3: What is the impact of such an adversarial input on the system containing the NN as a component?

"Street sign"

Mutation →

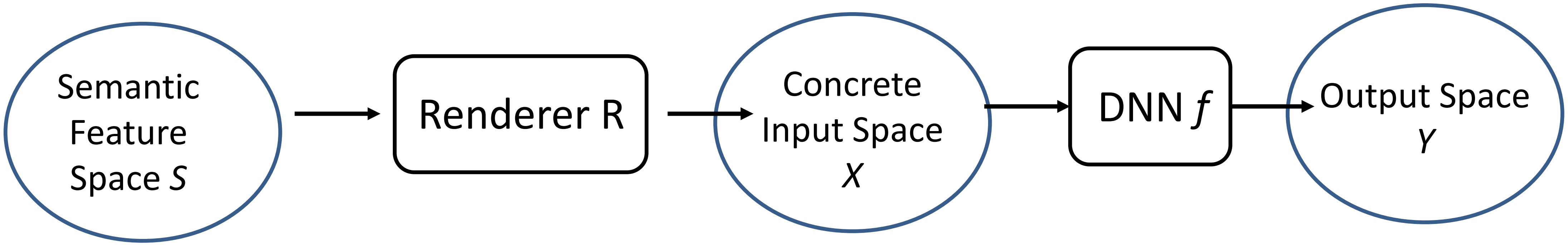
"Bird house"

[Huang et al., 2016]



Slides by Andrej Karpathy

Semantic Adversarial Analysis / Semantic Robustness



Semantic Robustness:

$$[\, s \approx_S s' \wedge \mathrm{R}(s) = x \wedge \mathrm{R}(s') = x' \,] \Rightarrow f(x) \approx_Y f(x')$$

Can apply techniques from standard adversarial analysis, provided R is differentiable

[S. A. Seshia, et al., "Formal Specification for Deep Neural Networks", ATVA 2018.]

Sample Result for Semantic Adversarial Analysis

SqueezeDet NN for object detection on Virtual KITTI data set

Uses 3D-SDN differentiable renderer [Yao et al. NeurIPS'18] and FGSM on semantic feature space

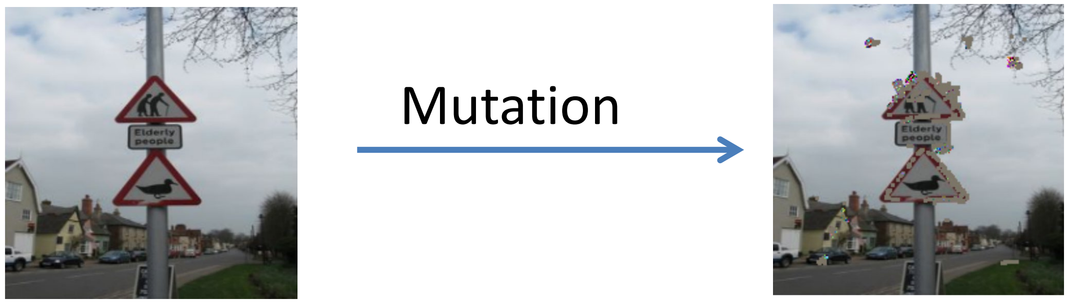
[Jain, Wu, Chandrasekharan, Chen, Jang, Jha, Seshia, 2019]

Robustness to Adversarial Inputs/Mutations

Q1: Can these mutations occur in practice in the environment?

Q2: What about "big" mutations in pixel space producing "equivalent" i/p?

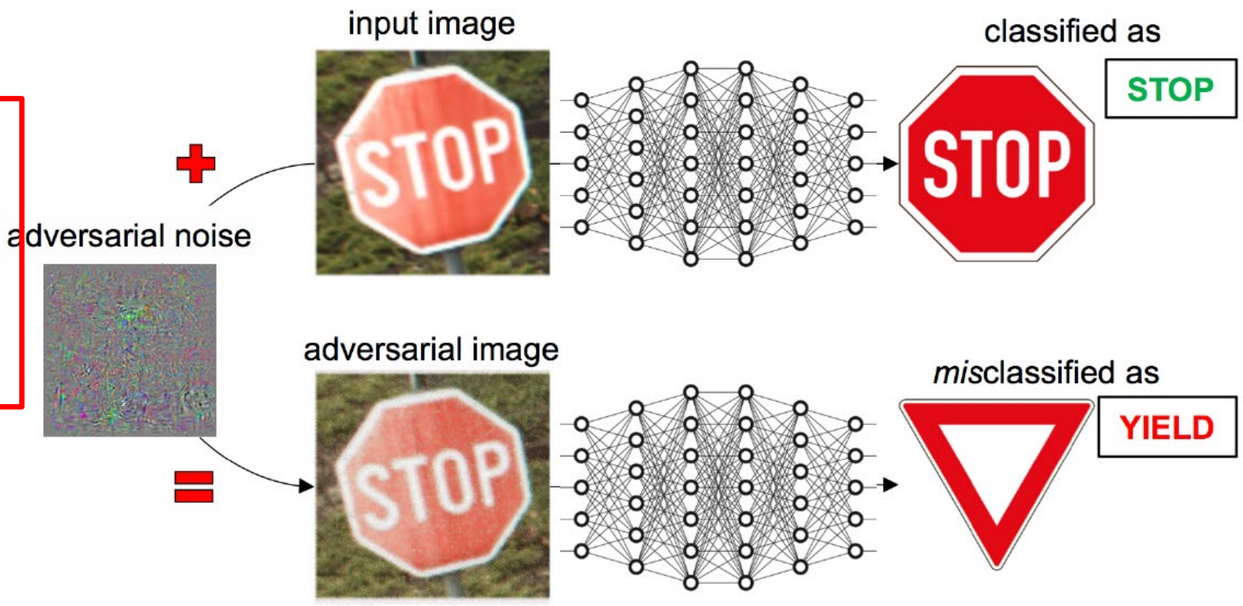
Q3: What is the impact of such an adversarial input on the system containing the NN as a component?



"Street sign"

"Bird house"

[Huang et al., 2016]



Slides by Andrej Karpathy

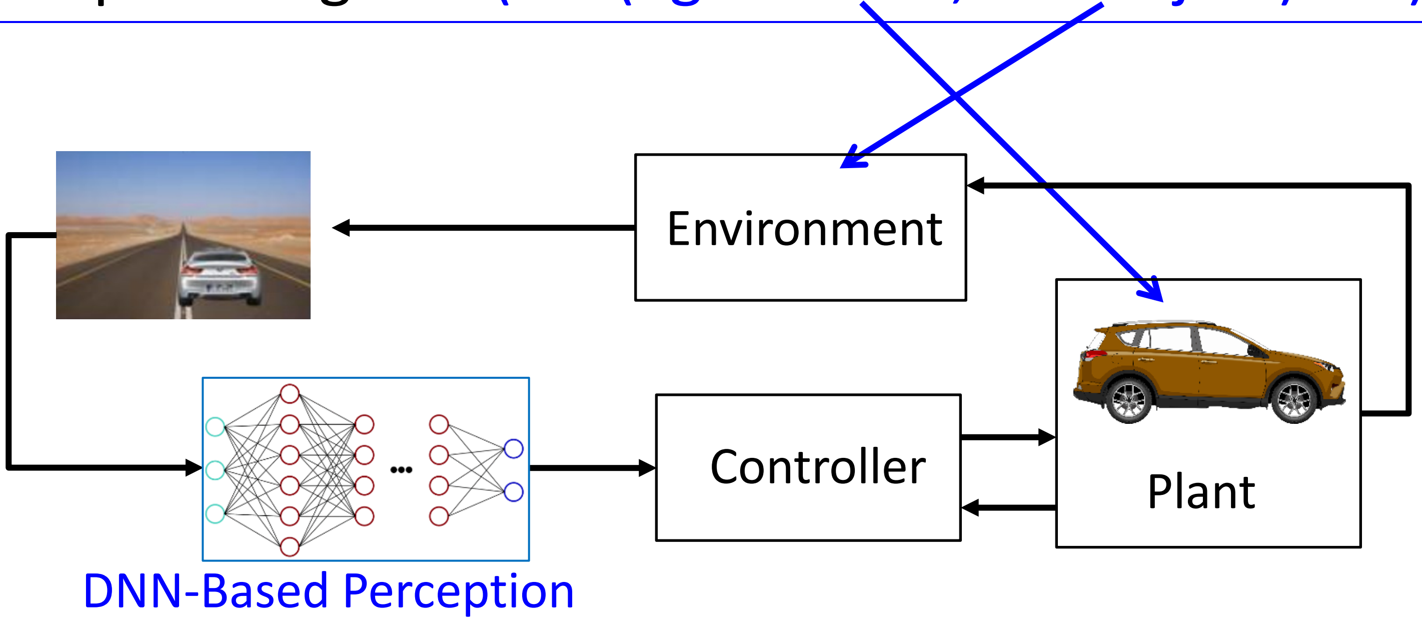
Insight: Start with **System-Level** Specification

- ✗ “Verify the Deep Neural Network Object Detector”
- ✓ “Verify the System containing the Deep Neural Network”

Formally Specify the *End-to-End Behavior* of the System

Temporal Logic: **G** (*dist*(ego vehicle, env object) > Δ)

Recall AEBS Example



Property does not mention inputs/outputs of the neural network

Compositional Falsification

Principles:

(1) Abstraction (replace DNN by simpler abstract function),

and

(2) Compositional Reasoning (component-level adversarial analysis guided by system-level analysis)

T. Dreossi, A. Donze, and S. A. Seshia. *Compositional Falsification of Cyber-Physical Systems with Machine Learning Components*, In NASA Formal Methods Symposium, May 2017.
(Extended version: Journal of Automated Reasoning, 2019.)

Result on AEBS Example

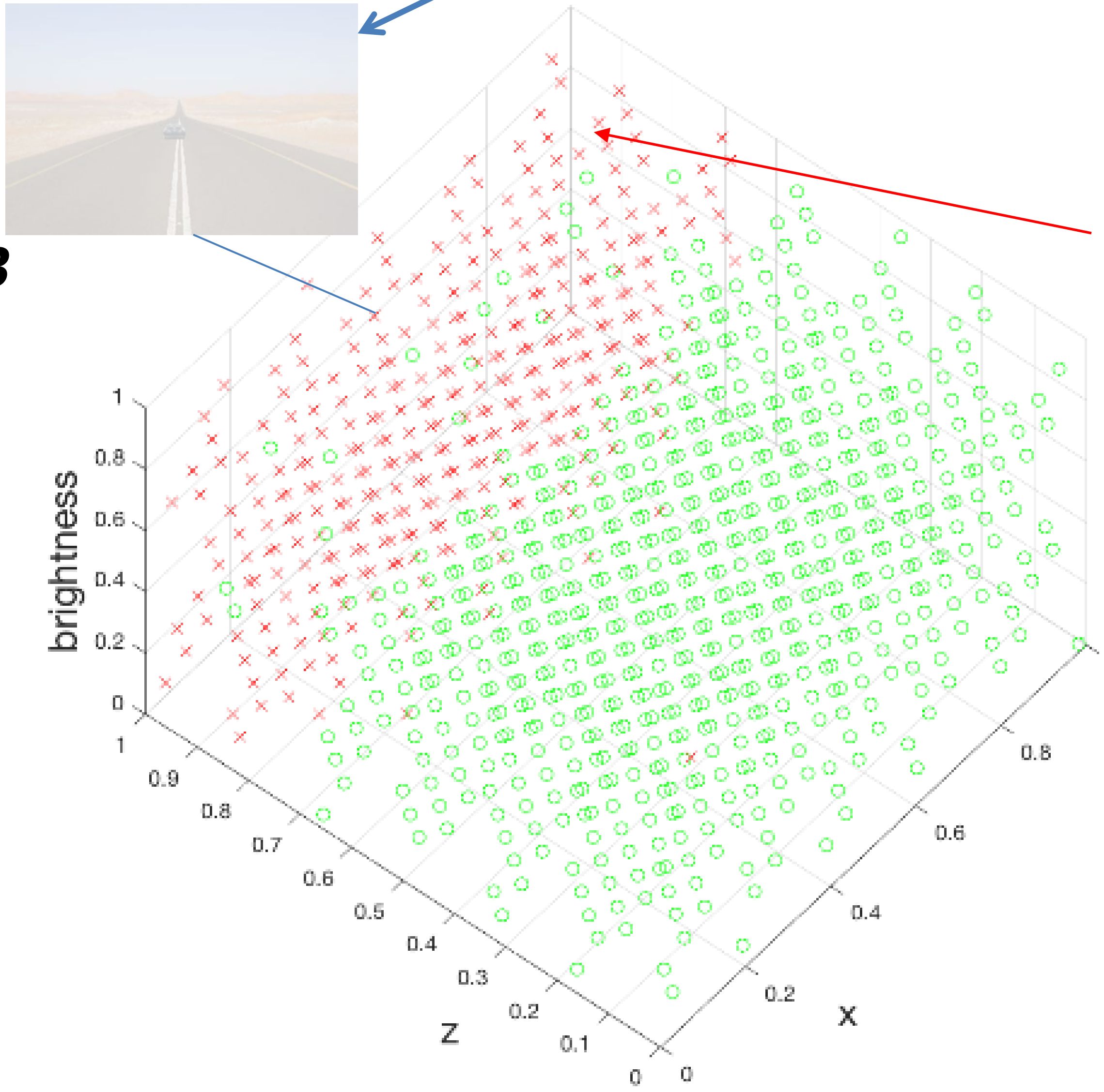
This misclassification not of concern

Inception-v3

Neural

Network

(pre-trained on ImageNet using TensorFlow)



Misclassifications

Sample image

[Dreossi, Donze, Seshia, NFM 2017; J. Automated Reasoning 2019]

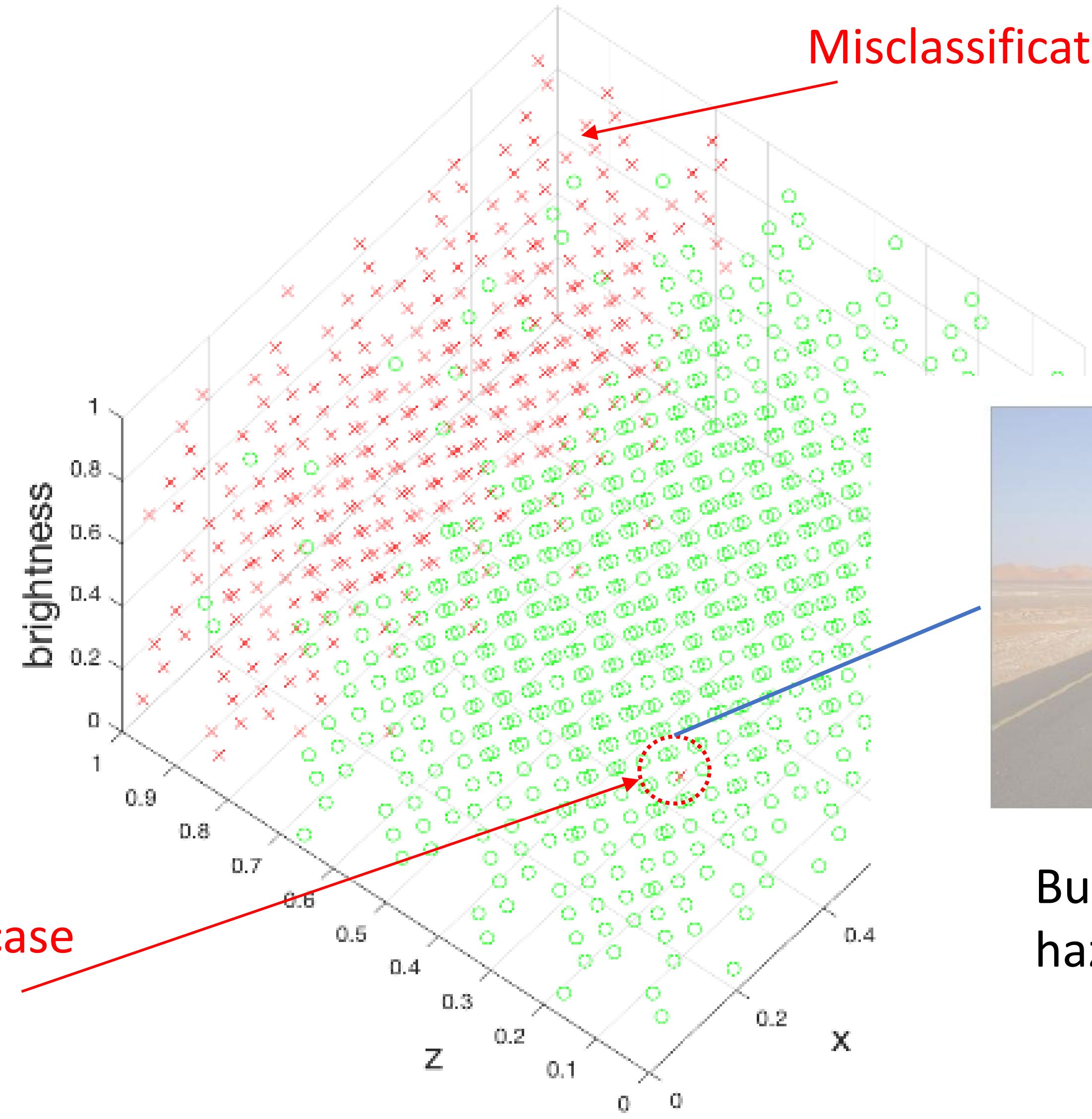
Result on AEBS Example

Inception-v3

Neural Network

(pre-trained on ImageNet using TensorFlow)

Misclassifications



Corner case Image

But this one is a real hazard!

[Dreossi, Donze, Seshia, NFM 2017; J. Automated Reasoning 2019]

Revisiting the Challenges

Challenge 1: Hard to Formalize Tasks

Many of the impactful applications of ML/Deep Learning are in **Perceptual Tasks**

How do you specify “a car”?

What Specifications are Meaningful for Hard-to-Formalize Tasks?

Principle: Start with Formalizable System-Level Specification

S. A. Seshia, D. Sadigh, S. S. Sastry. *Towards Verified Artificial Intelligence*. July 2016. https://arxiv.org/abs/1606.08514.

Challenge 2: Boolean vs. Quantitative Specifications

Do you speak
Formal Methods?
(logic, automata,
etc.)

Do you speak
Machine Learning?
(cost functions, risk
reward, etc.)

BOOLEAN

- More Composable
- Fit with Formal Tools

QUANTITATIVE

- More Flexible
- Fit with Optimization

HYBRID

Principle: Employ Hybrid Boolean-Quantitative Formalisms

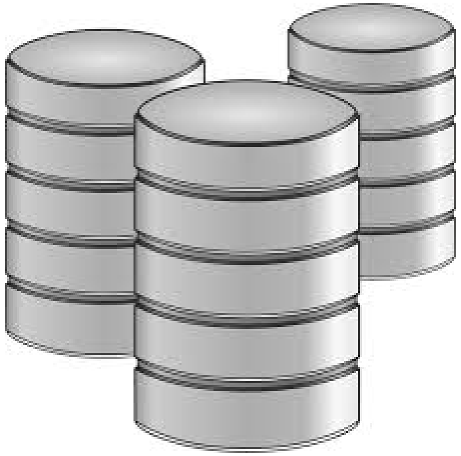
E.g., MTL, STL, Rulebooks, etc.

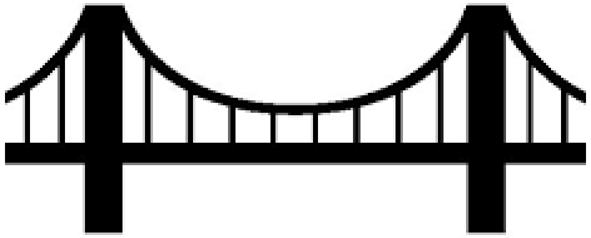
S. A. Seshia, D. Sadigh, S. S. Sastry. *Towards Verified Artificial Intelligence*. July 2016. https://arxiv.org/abs/1606.08514.

Challenge 3: Data vs. Formal Specification

MACHINE LEARNING

FORMAL METHODS





$$\varphi_1 \vee (\varphi_2 \wedge \varphi_3)$$

How do we bridge the gap?

Principle: Use Specification Mining

E.g., [Vazquez-Chanlatte et al. '17, '18; Puranic et al., 21; Belta et al., '17, …]

S. A. Seshia, D. Sadigh, S. S. Sastry. *Towards Verified Artificial Intelligence*. July 2016. https://arxiv.org/abs/1606.08514.

Specifying Environments / Distributional Assumptions

SCENIC: Environment Modeling and Data Generation

- *Scenic* is a probabilistic programming language defining *distributions over scenes/scenarios*
- *Use cases:* data generation, test generation, verification, debugging, design exploration, etc.

```
model scenic.domains.driving.model

ego = Car

spot = OrientedPoint on visible curb
badAngle = Uniform(1.0, -1.0) * Range(10, 20) deg
parkedCar = Car left of spot by 0.5,
                facing badAngle relative to roadDirection
```

```
model scenic.domains.driving.model

behavior PullIntoRoad():
    while (distance from self to ego) > 15:
        wait
    FollowLaneBehavior(lane=ego.lane)

ego = Car with behavior DriveAvoidingCollisions

spot = OrientedPoint on visible curb
badAngle = Uniform(1.0, -1.0) * Range(10, 20) deg
parkedCar = Car left of spot by 0.5,
                facing badAngle relative to roadDirection,
                with behavior PullIntoRoad
```

Example: Badly-parked car

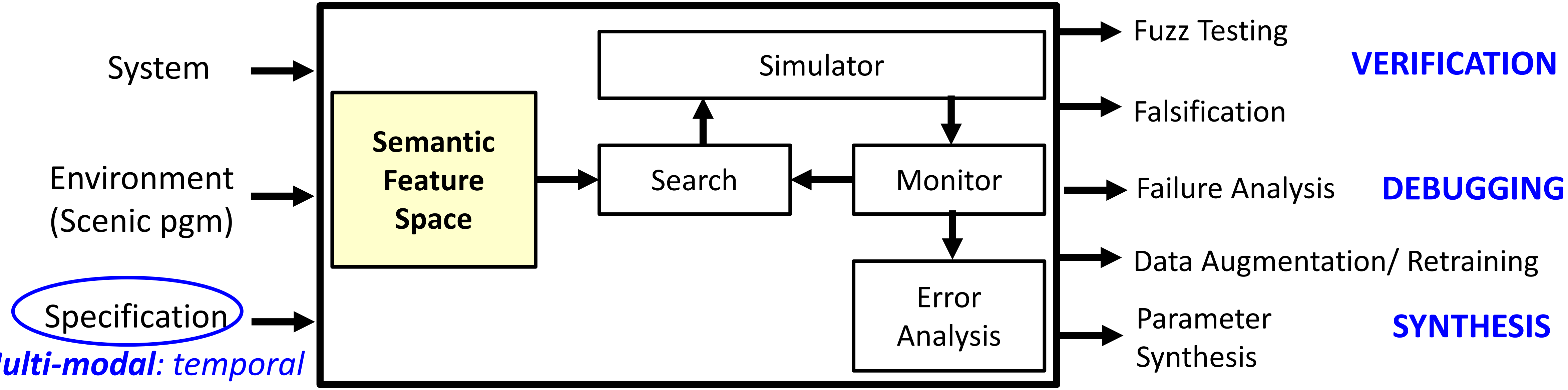
Image created with GTA-V

Video created with CARLA

[D. Fremont et al., "Scenic: A Language for Scenario Specification and Scene Generation", TR 2018, PLDI 2019.]

VERIFAI: A Toolkit for the Design and Analysis of AI-Based Systems [CAV 2019]

https://github.com/BerkeleyLearnVerify/VerifAI



***Multi-modal**: temporal logic, obj funcs, …*

Conclusion: Towards Verified AI/ML based Autonomy

Challenges		Principles
1. Environment (incl. Human) Modeling	→	Data-Driven, Introspective, Probabilistic Modeling
2. Specification	→	Start with System-Level Spec; Hybrid Boolean-Quant; Spec. Mining
3. Learning Systems Complexity	→	Abstraction, Semantic Representation, and Explanations
4. Efficient Training, Testing, Verification	→	Compositional Analysis and Semantics-directed Search/Training
5. Design for Correctness	→	Oracle-Guided Inductive Synthesis; Run-Time Assurance

Exciting Times Ahead!!! Thank you!

S. A. Seshia, D. Sadigh, S. S. Sastry. *Towards Verified Artificial Intelligence*. July 2016. https://arxiv.org/abs/1606.08514.