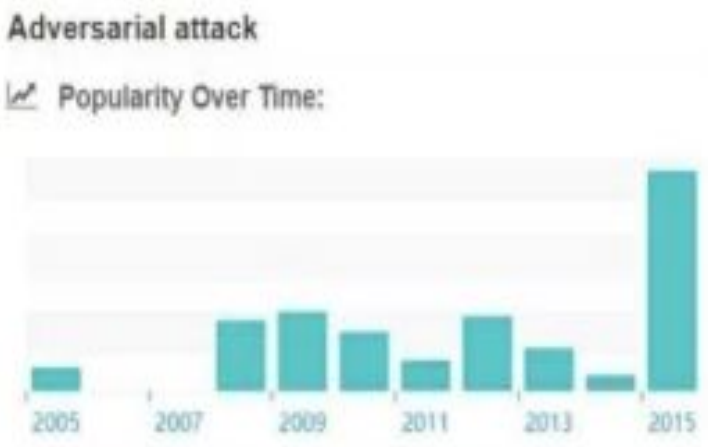
# Semantic Adversarial Deep Learning

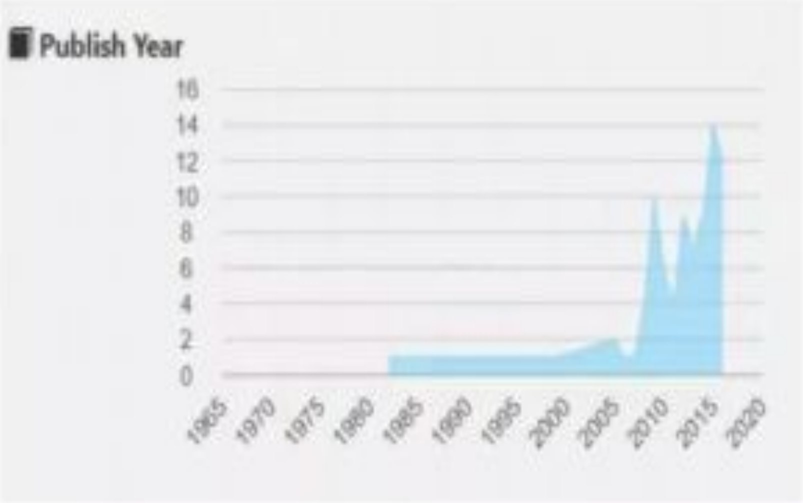
A paper of Tommaso Dreossi, Somesh Jha, and Sanjit A. Seshia

Thanks to Somesh Jha and Sanjit A. Seshia for some of the slides

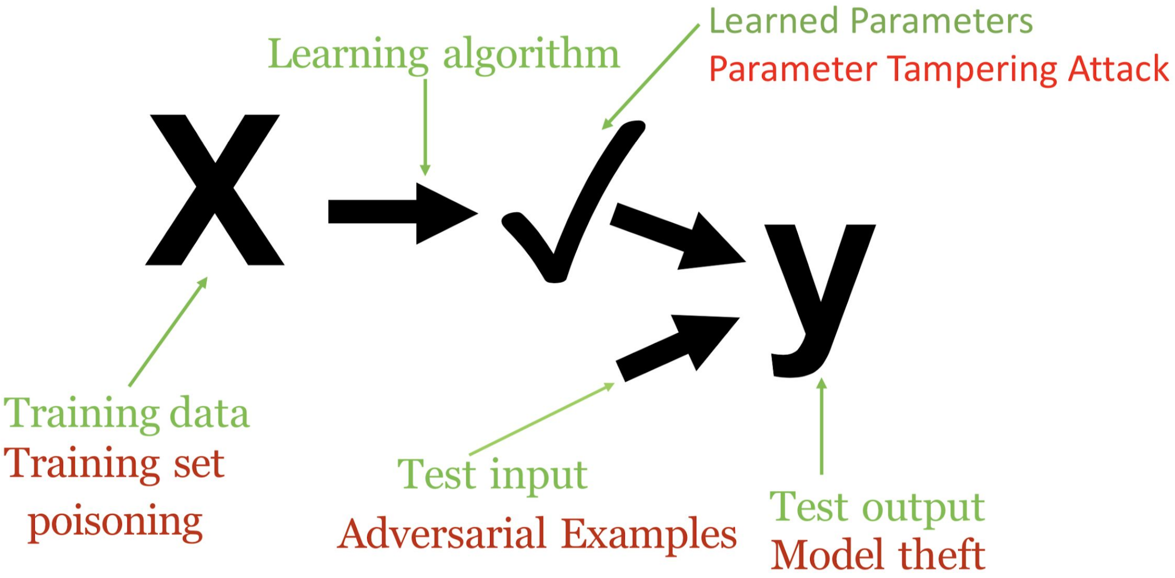
Presented by Yilin Han

# Background


Adversarial attack
Popularity Over Time:
2005
2007
2009
2011
2013
2015


Publish Year
16
14
12
10
8
6
4
2
0
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020

# Adversarial Attacks on ML



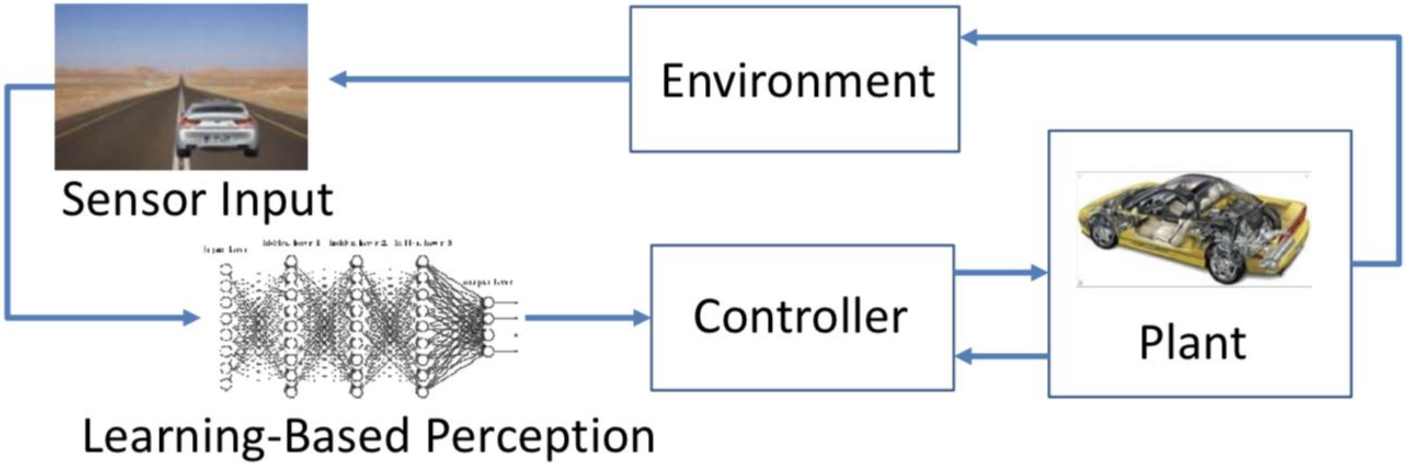
# Definition

“Adversarial examples are inputs to machine learning models that an attacker has intentionally designed to cause the model to make a mistake” (Goodfellow et al 2017)

“Adversarial examples are inputs to machine learning models that an attacker has intentionally designed to cause the entire system to make a mistake”

# Automatic Emergency Braking System(AEBS)

Goal: Brake whenever an obstacle is detected



# Semantic Adversarial Analysis and Training

Goal: DNN analysis must be more semantic

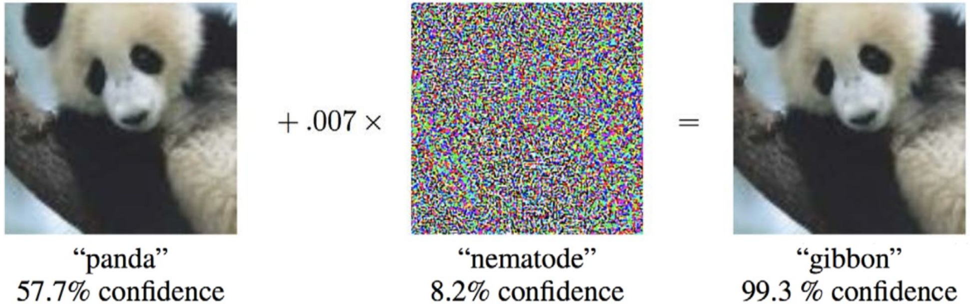
- Semantic modification
- System-level specification
- Semantic (re-)training
- Confidence-based analysis

# Semantic Modification

- Allow “Noise”: Add a vector δ
- Allow richer transformations:

  Translation, Cloudy background…..

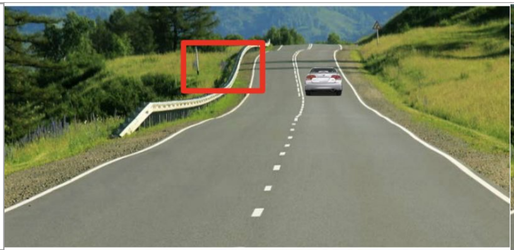
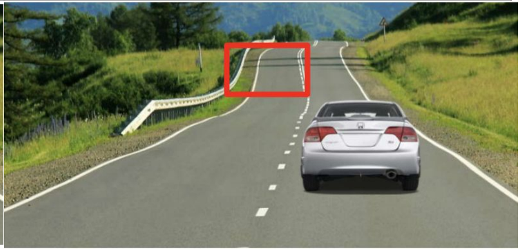
- DSL that relates to applications




Non-semantic perturbation (i.e., noise)

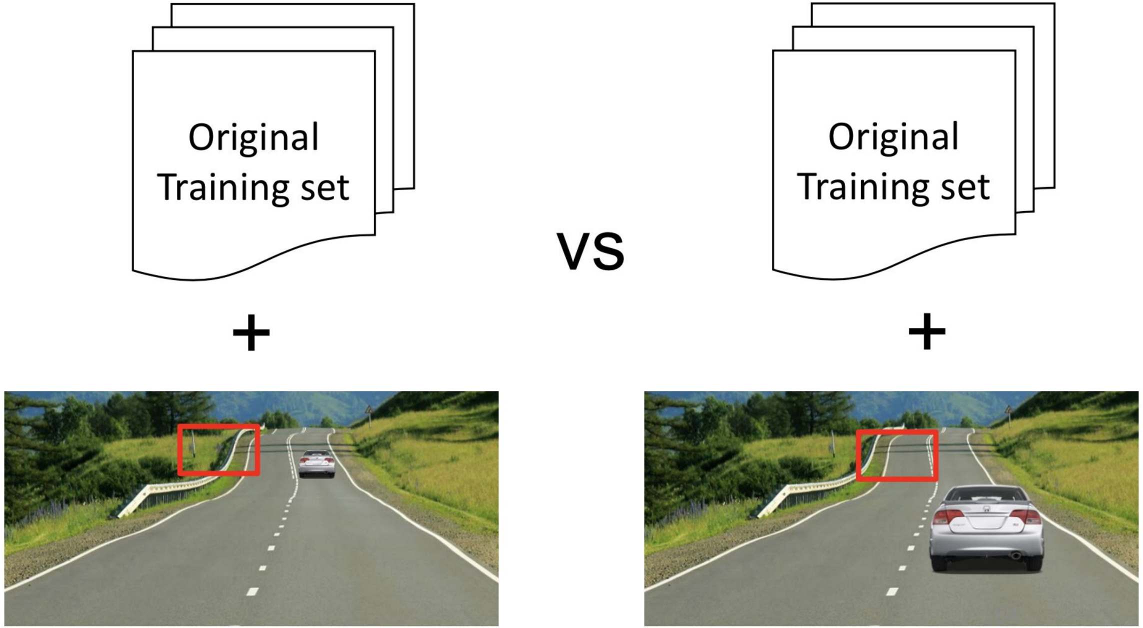
Semantic perturbation (i.e., translation)

# System-level Specification

| |  |  |
|---|---|---|
| Perception-level-spec: “Detect cars” | NO | NO |
| Perception-level-spec: “Do not crash” | YES | NO |

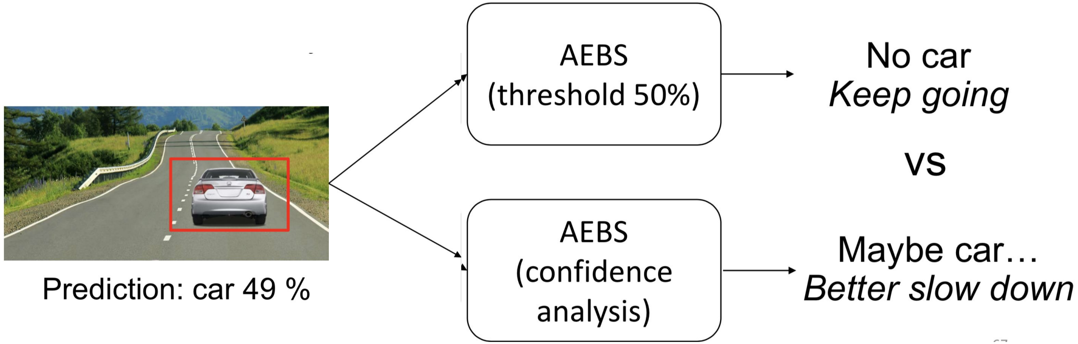
# Semantic Training

Semantic augmentation



# Confidence-based Analysis

They argue that confidence levels must be used within the design of ML-based system.



# Problem

Can we generate adversarial examples that cause system-level failure?

Can we use formal method to verify CPS?

# Compositional Falsification

Statement:

given a formal specification φ (say in a formalism such as signal temporal logic) and a CPS+ML model M, find an input for which M does not satisfy φ.

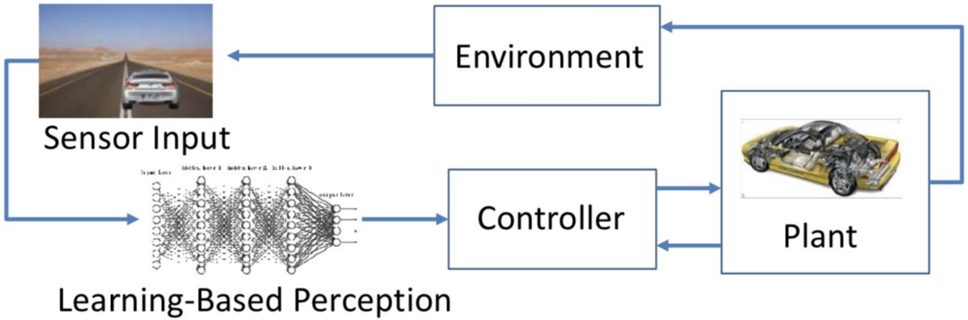
Example: $\mathtt{G}_{0,T}(\|\mathbf{x}_{\text{ego}} - \mathbf{x}_{\text{obs}}\|_2 \geq 2)$.

Problem:

How to deal with ML component?

# Compositional Falsification

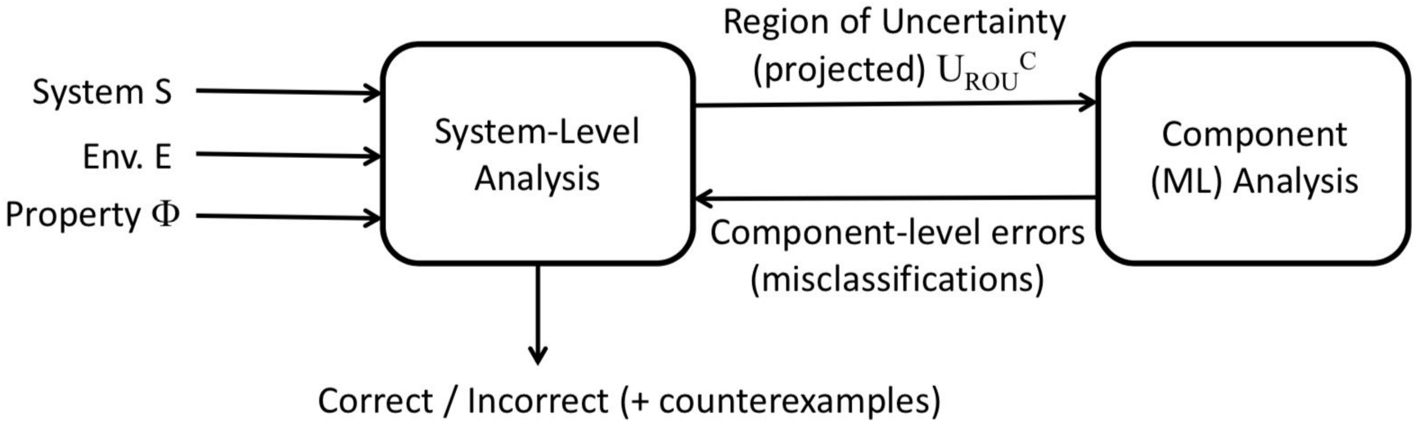
- Standard solution: use compositional verification (Modular)
- However, no formal specification for neural network component.



# Approach: Use a System-Level Specification and Combine CPS Falsifier with ML Analyzer

Do not verify the DNN Object Detector.

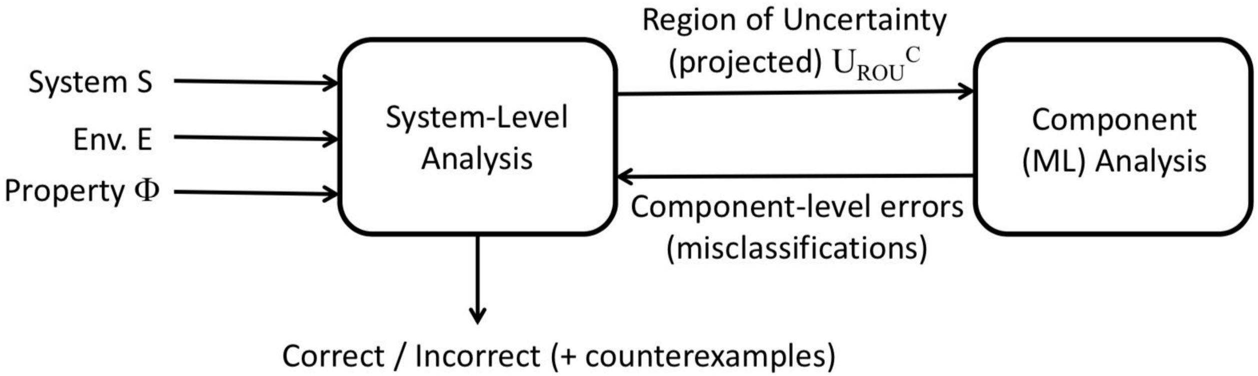
Verify the system containing the DNN model



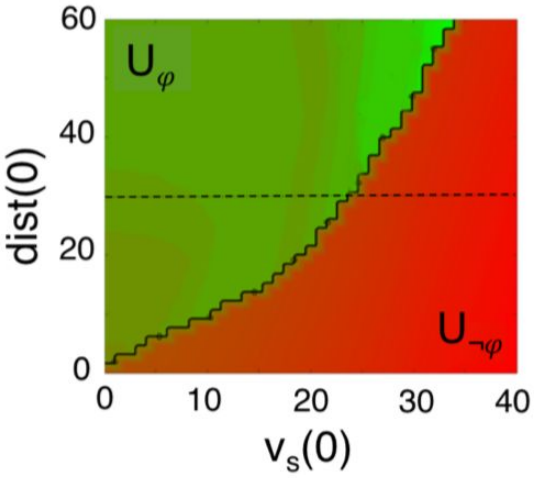
# System-Level Analysis

CPS Falsifier uses abstraction of ML component:

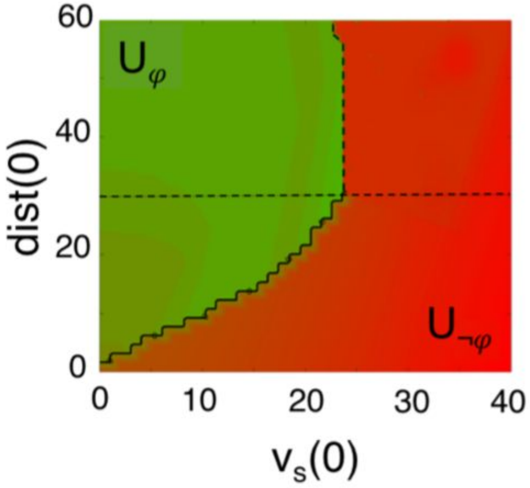
- Optimistic analysis: assume ML classifier is always correct
- Pessimistic analysis: assume it is always wrong
- Difference is the region of uncertainty where output of the ML component “matters”



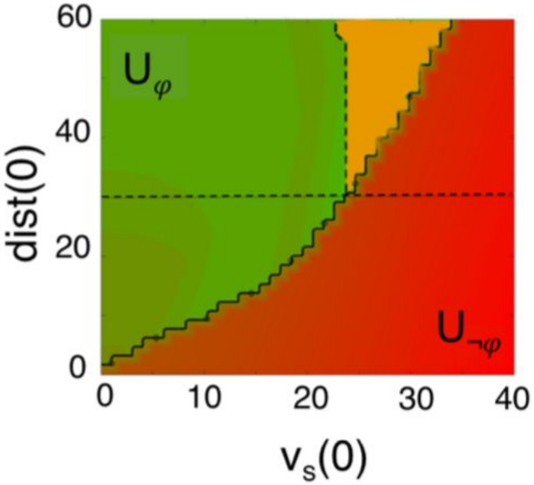
# Identify Region of Uncertainty




ML always correct

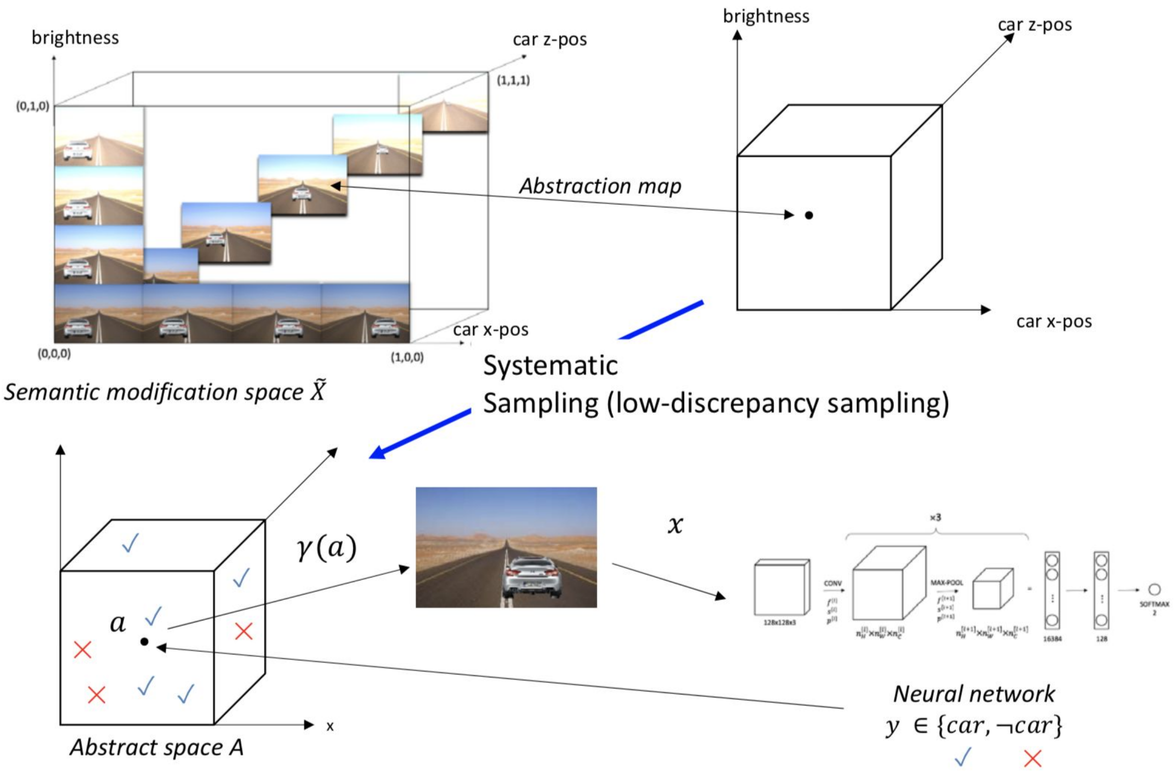



ML always wrong

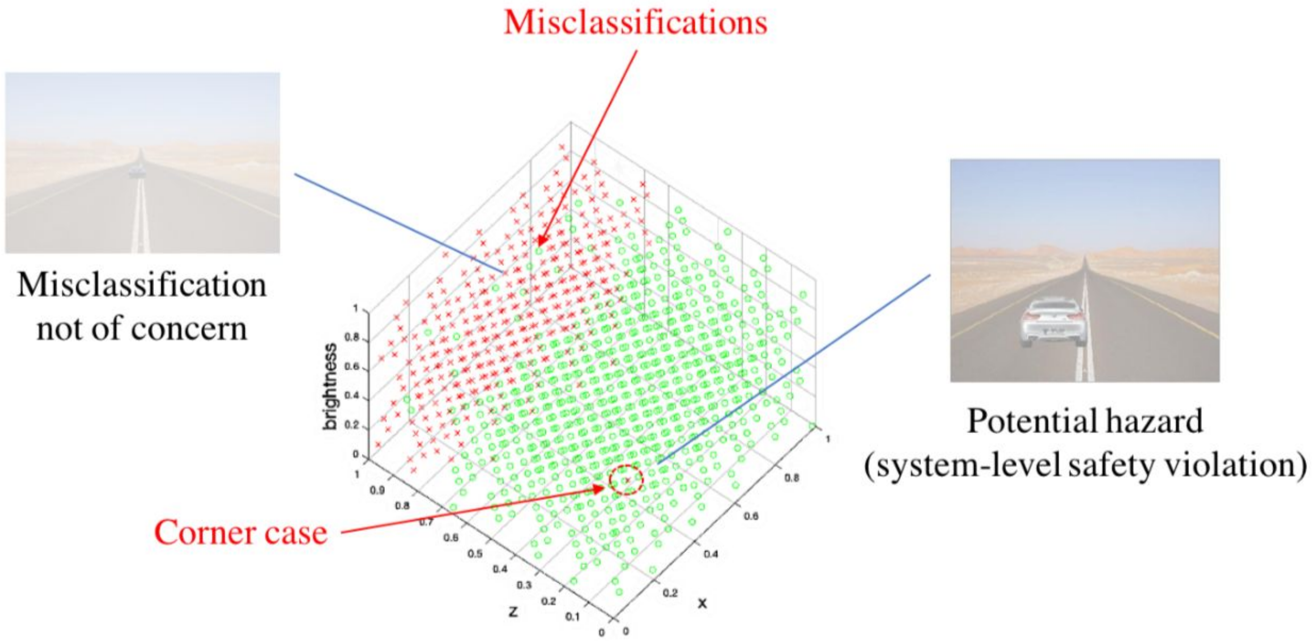



Region of uncertainty
(yellow)

# Machine Learning Analyzer



# Sample Results



# System Training

- Train the model with both original and counterexample test sets.
- That is they trained a model with hinge loss with a bias factor k, which means no penalty for a misclassification if confidence level + k < truth label

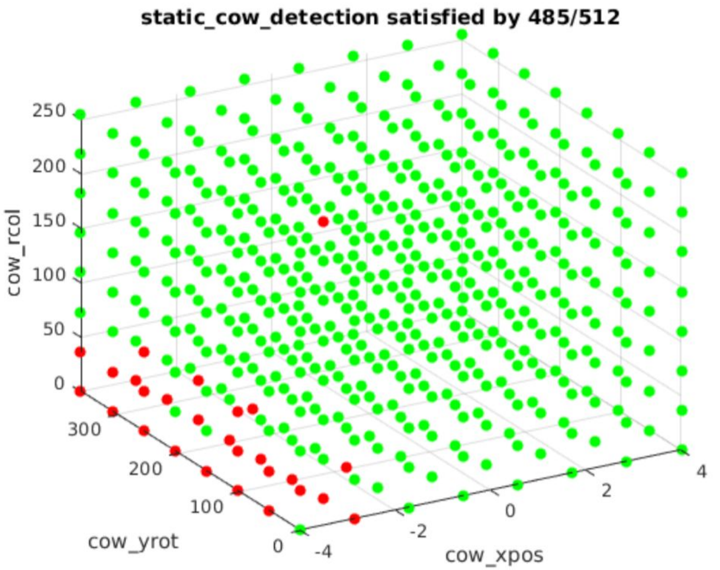
$$l(\hat{y}) = \max(0, k + \max_{i \neq l}(\hat{y}_i - \hat{y}_l))$$

Result:

- Accuracy increases on counterexamples testing set.
- Accuracy decreases on original testing set.

# System Training

- Train the model with both original and counterexample test sets. Where the counterexamples are generated from  composition falsification framework.
- Red dots are Semantic counterexamples




Result:

Still specification violation

However, obstacles get detected earlier.

# Conclusion

- Formal Methods can apply to cyber-physical system with high assurance.
- Compositional falsification framework can also be used for verification