Formal Security Analysis of Neural Networks using Symbolic Intervals

Shiqi Wang et. al.

杨宇清

April 11, 2019

Table Of Contents

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

- Existing adversarial testing models:
  - No guarantee of non-existence of adversarial examples
  - My conjecture:
    - Tend to overestimate
    - The example might not be applied to real life
- High overhead of SMT
  - especially for non-linear, non-convex function

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

Goal:

A system for **formally** checking **security properties** of **Relu-based DNNs**

- High efficiency: ”200 times on average”
- High accuracy: ”a variety of optimizations to improve accuracy”

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

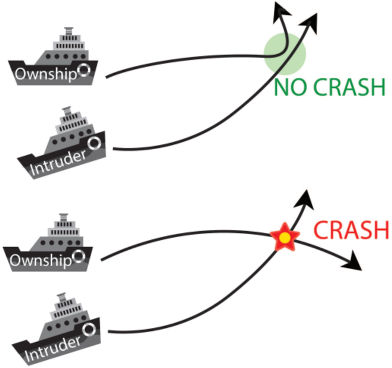
6 Appendix

- Target system: ACAS Xu
- Security property: input-output-based

To security property

- Attacker model: similar to adversarial examples:

given a computer vision DNN f, the attacker solves following optimization problem: $min(L_p(x'-x))$ such that $f(x) \neq f(x')$, where $L_p(\cdot)$ denotes the p-norm and $x'-x$ is the perturbation applied to original input x. In other words, the security property of a vision DNN being robust against adversarial perturbations can be defined as: for any x' within a L-distance ball of x in the input space, $f(x) = f(x')$.



Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

- Main method: interval analysis
- *Optimization*$_1$: symbolic Interval
- *Optimization*$_2$: iterative refinement:
  (existence of Lipschitz Consistency)

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

A Working Example: aiming to verify whether safe or not

$Distance: \ x,$

$Approaching\ angle: \ y$

$Safe\ property: \ x \in [4,6] \ y \in [1,5]$

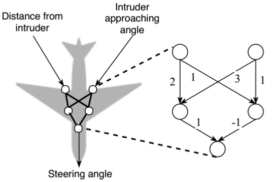
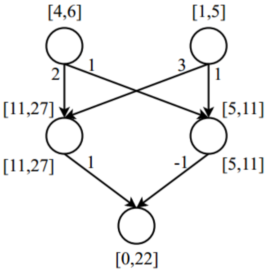


Figure 2: Running example to demonstrate our techniques.



(a) Naive interval propagation

Dependency error

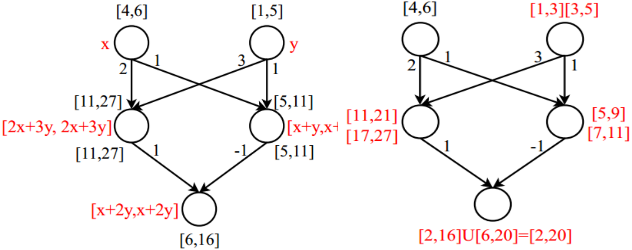
- Naively computing output intervals in this way suffers from high errors as it computes extremely loose bounds.
- Only a highly conservative estimation of the output range, too wide to be useful for checking any safety property.

Symbolic Interval and Iterative Refinement

- Symbolic interval propagation
  - explicitly represent the intermediate computations of each neuron in terms of the symbolic intervals that encode the interdependency of the inputs to minimize overestimation
- Iterative refinement
  - The dependency error for Lipschitz continuous functions decreases as the width of intervals decreases
  - Therefore, we can bisect the input interval by evenly dividing the interval into the union of two consecutive sub-intervals and reduce the overestimation

Symbolic Interval and Iterative Refinement

- Symbolic interval propagation
  (algebraic operand preservation)
- Iterative refinement
  (even interval division)



(b) Symbolic interval propagation (c) Iterative bisection and refinement

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

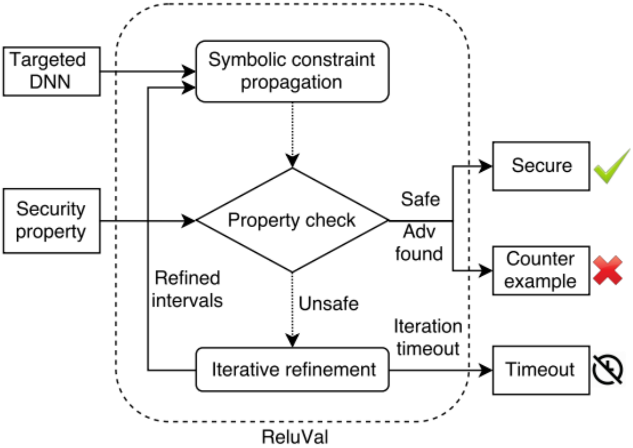
3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

Analysis Procedure



Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

Statistics

Source	Properties	Networks	Reluplex Time (sec)	ReluVal Time (sec)	Speedup
Security Properties from [25]	ϕ_1	45	>443,560.73*	14,603.27	>30×
	ϕ_2	34*[2]	123,420.40	117,243.26	1×
	ϕ_3	42	35,040.28	19,018.90	2×
	ϕ_4	42	13,919.51	441.97	32×
	ϕ_5	1	23,212.52	216.88	107×
	ϕ_6	1	220,330.82	46.59	4729×
	ϕ_7	1	>86400.0*	9,240.29	>9×
	ϕ_8	1	43,200.01	40.41	1069×
	ϕ_9	1	116,441.97	15,639.52	7×
	ϕ_{10}	1	23,683.07	10.94	2165×
Additional Security Properties	ϕ_{11}	1	4,394.91	27.89	158×
	ϕ_{12}	1	2,556.28	0.104	24580×
	ϕ_{13}	1	>172,800.0*	148.21	>1166×
	ϕ_{14}	2	>172,810.86*	288.98	>598×
	ϕ_{15}	2	31,328.26	876.80	36×

* Reluplex uses different timeout thresholds for different properties.

Table 1: ReluVal's performance at verifying properties of ACAS Xu compared with Reluplex. ϕ_1 to ϕ_{10} are the properties proposed in Reluplex [25]. ϕ_{11} to ϕ_{15} are our additional properties.

Statistics

# Seeds	CW	CW Miss	ReluVal	ReluVal Miss
50	24/40	40.0%	40/40	0%
40	21/40	47.5%	40/40	0%
30	17/40	58.5%	40/40	0%
20	10/40	75.0%	40/40	0%
10	6/40	85.0%	40/40	0%

Table 2: The number of adversarial inputs CW can find compared to ReluVal on 40 adversarial ACAS Xu properties. The third column shows the percentage of adversarial properties CW failed to find.

P	Adv Range	Adv	Timeout	Non-adv
S_1	$[6402.36, 10000]$	98229	1	163915
S_2	$[-0.2, -0.186]$ and $[-0.103, 0]$	18121	2	14645
S_3	$[-0.1, 0.0085]$	17738	1	15029

Table 3: The second column shows the input ranges containing at least one adversarial input, while the rest of ranges are found by ReluVal to be non-adversarial. The last three columns show the number of total sub-intervals checked by ReluVal with a precision of $e-6$.

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

- **Lack of Guarantee**:None of the existing attacks can provide any provable guarantees about the non-existence of adversarial examples
- ReluVal can provide a provable security analysis of given input ranges, systematically narrowing down and detecting all adversarial ranges

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

Customized STM solvers

- **Significant overhead**: Customized SMT solvers for verifying security properties of DNNs are mostly limited by the scalability of the solver
- ReluVal uses interval-based techniques and significantly outperforms the state-of-the-art solver-based systems like Reluplex

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

- **Lack of Concrete Results**: Focus on simply over-approximating the total number of potential adversarial violations without trying to find concrete counterexamples problem
- ReluVal can find concrete counterexamples as well as verify security properties of pre-trained DNNs

Outline

1 Motivation
- Existence of Problems
- Goal

2 Model
- Target

3 Method Details
- Overview
- A Working Example
- Analysis Procedure

4 Evaluation
- Statistics

5 Related Works
- Adversarial machine learning
- Customized SMT solvers
- Convex-problem transformation
- Discreteness analysis

6 Appendix

Verivis: Transform the verification problem into a convex optimization problem using relaxations to over-approximate the outputs of ReLU nodes

- **Lack of Verification**: Leveraging the discreteness of image pixels cannot verify non-existence of norm-based adversarial examples
- ReluVal can.

An example of Security Property Definition

Property ϕ_1***:*** If the intruder is distant and is significantly slower than the ownship, the score of a COC advisory will always be below a certain fixed threshold.

Tested on: all 45 networks.

Input ranges: $\rho \geq 55947.691$, $v_{own} \geq 1145$, $v_{int} \leq 60$.

Desired output: the output of COC is at most 1500.

Property ϕ_2***:*** If the intruder is distant and is significantly slower than the ownship, the score of a COC advisory will never be maximal.

Tested on: model_x_y, $x \geq 2$, except model_5_3 and model_4_2

Input ranges: $\rho \geq 55947.691$, $v_{own} \geq 1145$, $v_{int} \leq 60$.

Desired output: the score for COC is not the maximal score.

Back to Model