Improving the Reliability and Safety of Systems

Toward Scalable Deep Neural Network Verification

ThanhVu (Vu) Nguyen

GEORGE MASON UNIVERSITY

CEC P&T Seminar, Nov 12 2023

My Background

Academic

- 2013: PhD in CS, Univ of New Mexico–Albuquerque
- 2014: Postdoc, Univ of Maryland-College Park
- 2016: Assistant Prof., Univ of Nebraska-Lincoln
- 2021: Assistant Prof., **George Mason University**

Govt and Industry

- 2005–2006, 2012: Naval Research Lab, Washington DC
- 2007: Lockheed Martin, New Jersey

My Research

Software Engineering, Formal Methods, Programming Languages

- **Invariant Generation and Automatic Program Repair**
  - since '08, during PhD study
- **Highly-Configurable and Build System Analysis**
  - since '15, during postdoc
- **AI Verification**
  - since '22, new research direction

My Research

Software Engineering, Formal Methods, Programming Languages

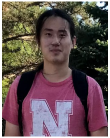
- **Invariant Generation and Automatic Program Repair**
  - since '08, during PhD study
- **Highly-Configurable and Build System Analysis**
  - since '15, during postdoc
- **AI Verification**
  - since '22, new research direction

Sponsor

- NSF (4x): CRII'20, Med Collab. '21, CAREER'23, FMIT'23
- Defense (1x): Army Research '18
- Industry (2x): Facebook'23 and Amazon'23
- Internal (1x): UNL Seed'20

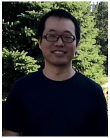
DynaROARS
dynaroars.cs.gmu.edu

Didier



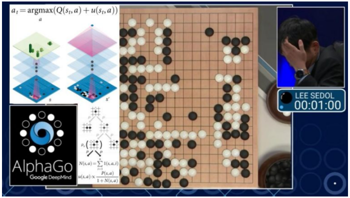
Linhan

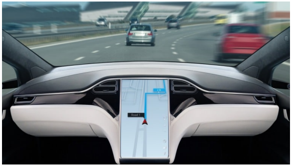
Hai



Guolong
PhD'22 at UNL

Outline





DNN EVERYWHERE

DNN Problems

Nicolas Kayser-Bril
@nicolaskb

Black person with hand-held thermometer = firearm.
Asian person with hand-held thermometer = electronic device.

Computer vision is so utterly broken it should probably be started over from scratch.

Screenshot from 2020-03-31 11-23-45.png

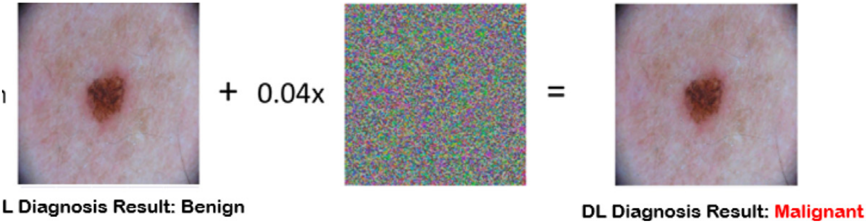
Gun	88%
Photography	68%
Firearm	65%
Plant	59%

Screenshot from 2020-03-31 11-27-22.png

Technology	68%
Electronic Device	66%
Photography	62%
Mobile Phone	54%

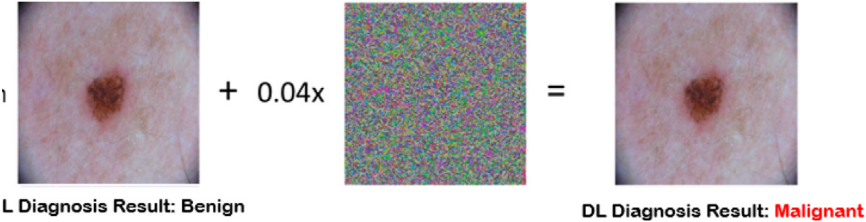

Google
Happening NOW
GOOGLE SELF-DRIVING CAR GETS INTO
AN ACCIDENT INVOLVING INJURIES
#NEWS
GOOGLE SELF DRIVING CAR
CRASHES INTO A BUS
EXCLUSIVE
TAKING ACTION FOR YOU
INVESTIGATION FOCUSED ON TESLA AUTOPILOT
abc ACTION NEWS
Mark Beach
TEMPE
DEADLY CRASH WITH SELF-DRIVING UBER
abc 15
11:01
NEW VIDEO
TAKING ACTION
DRIVERLESS UBER CAR INVOLVED IN CRASH IN TEMPE
POLICE SAY OTHER DRIVER FAILED TO YIELD
abc 15
ARIZONA
6:06 81°

Robustness Properties



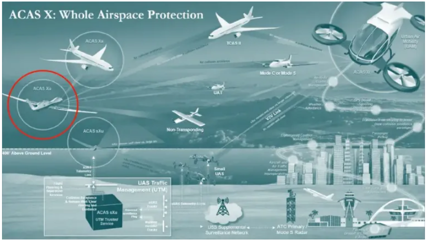
$$\forall i \in \{0 \ldots |X|-1\}.\ X_i - Y_i \leq 0.1 \ \Rightarrow\ class(X) \equiv class(Y) \qquad (1)$$

Robustness Properties

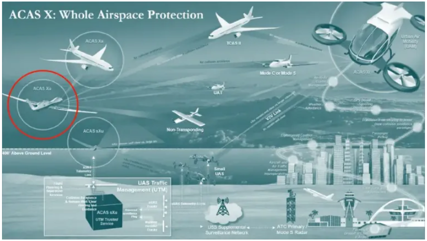


$$\forall i \in \{0 \ldots |X| - 1\}.\ X_i - Y_i \leq 0.1 \ \Rightarrow \ class(X) \equiv class(Y) \qquad (1)$$

if corresponding pixels of two images X and Y are not different by more than 0.1, then X and Y should have the same classification


ACAS X: Whole Airspace Protection
ACAS Xa
TCAS II
Mode C or Mode S
UAT
ACAS Xu
ACAS sXu
Non-Transponding
V2V Link
400' Above Ground Level
Small UAS
UAS Traffic Management (UTM)
ACAS sXu
UTM Trusted Service
USS Supplemental Surveillance Network
ATC Primary/ Mode S Radar

Safety Properties



ACAS: air traffic collision system, detects intruder and decides action.

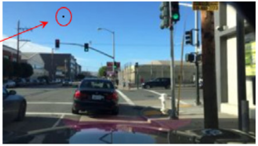
$$d_{intru} \geq 55947 \wedge v_{own} \geq 1145 \wedge v_{intru} \leq 60 \;\Rightarrow\; r_{nothing} \leq \tau$$

if intruder is distant and significantly slower than us, then we do nothing (i.e., below a certain threshold)

DL Classification: Green Light

Changing one pixel here

Text



DL Classification: Red Light

- Well-trained, e.g., 97% accuracy, DNNs are fine for most tasks
  - But not enough for mission-critical tasks, e.g., self-driving cars, air traffic collision control
- Testing can find counterexamples (e.g., adversarial attacks)
  - Testing shows the existence of errors, not its absence (*Dijkstra*)

DL Classification: Green Light   DL Classification: Red Light

- Well-trained, e.g., 97% accuracy, DNNs are fine for most tasks
  - But not enough for mission-critical tasks, e.g., self-driving cars, air traffic collision control
- Testing can find counterexamples (e.g., adversarial attacks)
  - Testing shows the existence of errors, not its absence (*Dijkstra*)

Formal Verification Can Help!

Software Verification

- Provide formal guarantee that a system really has no specific type of errors
- Mature field in CS/Logics with lots of powerful techniques and tools
  - Automated Theorem Proving
  - Constraint Solving (e.g., SAT/SMT solving)
  - Model Checking
  - Abstract Interpretation, ...
- Employed in mission-critical systems, e.g., avionics, medical devices, Windows, Clouds system (AWS)

The problem of Deep Neural Network verification

Question: Given a network N and a property p, does N have p?

- p often has the form $P \Rightarrow Q$ (precondition P, postcondition Q)

Answer: Yes / No

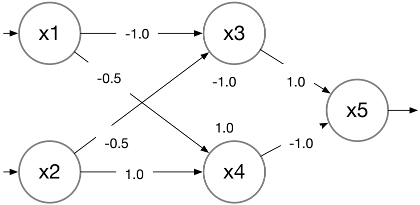
The problem of Deep Neural Network verification

Question: Given a network N and a property p, does N have p?

- p often has the form $P \Rightarrow Q$ (precondition P, postcondition Q)

Answer: Yes / No

Simple DNN with **ReLU**



- E.g., $x_3 = \max(-1x_1 + -0.5x_2, 0)$

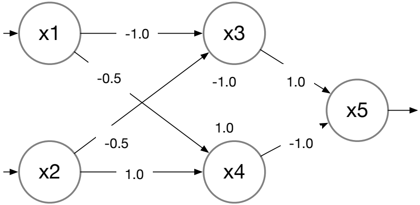
The problem of Deep Neural Network verification

Question: Given a network N and a property p, does N have p?

- p often has the form $P \Rightarrow Q$ (precondition P, postcondition Q)

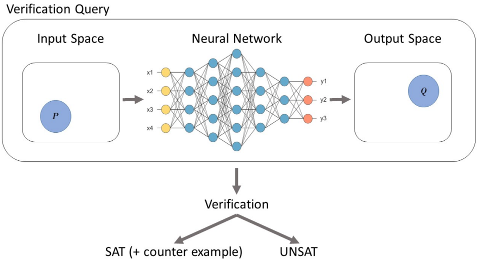
Answer: Yes / No

Simple DNN with **ReLU**

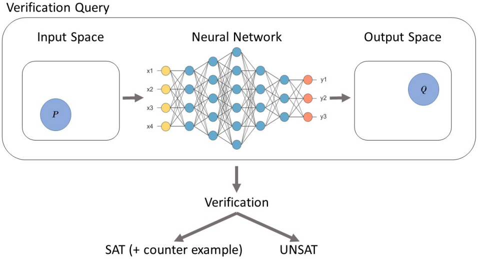


- E.g., $x_3 = \max(-1x_1 + -0.5x_2, 0)$
- Valid: $x_1 \in [-1, 1] \wedge x_2 \in [-2, 2] \Rightarrow x_5 \leq 0$
- Invalid: $x_1 \in [-1, 1] \wedge x_2 \in [-2, 2] \Rightarrow x_5 > 0$

Constraint Solving Techniques

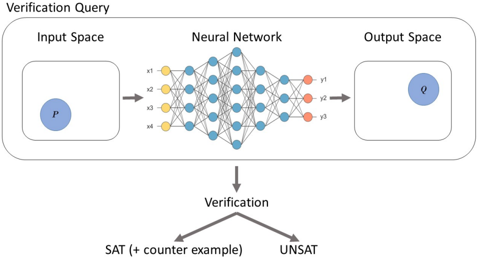


Constraint Solving Techniques



- Transform DNN verification into a constraint (satisfiability) problem
  - UNSAT: p is a property of N
  - SAT: p is not a property of N (also provide counterexamples)
  - TIMEOUT

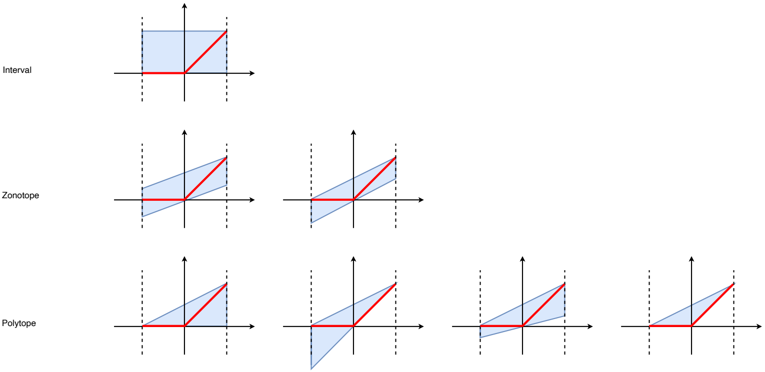
Constraint Solving Techniques



- Transform DNN verification into a constraint (satisfiability) problem
  - UNSAT: p is a property of N
  - SAT: p is not a property of N (also provide counterexamples)
  - TIMEOUT
- Solve the constraint, e.g., using MILP solvers
- Scalability is a Huge problem (many TIMEOUTs)
  - Complexity $O(2^N)$, where N is the number of neurons

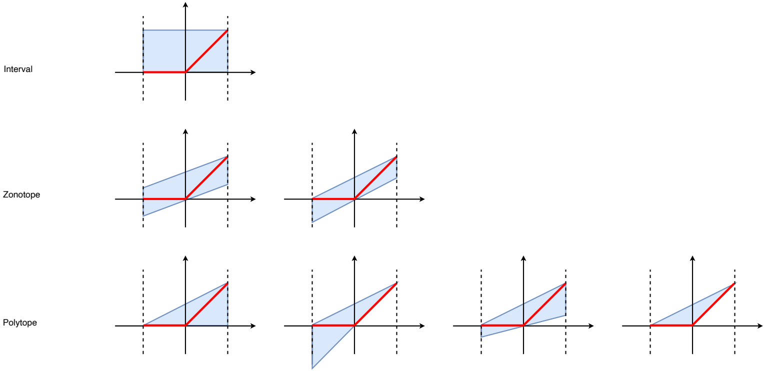
Abstraction Techniques

- Overapproximate computation (e.g., ReLU) using abstract domains
  - interval, zonotopes, polytopes



Abstraction Techniques

- Overapproximate computation (e.g., ReLU) using abstract domains
  - interval, zonotopes, polytopes

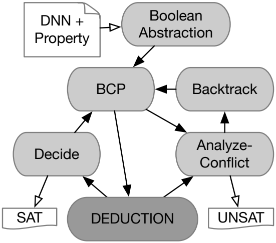


- Scale well, but loose precision (producing spurious cex's)
  - Claiming a property is violated when it is not

NeuralSAT: Our DNN Constraint Solver

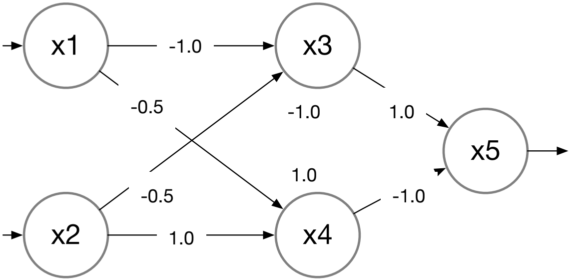
To prove $N \Rightarrow (P \Rightarrow Q)$

- Call NeuralSAT($N \wedge P \wedge \neg Q$)
- Return UNSAT or SAT (and counterexample)



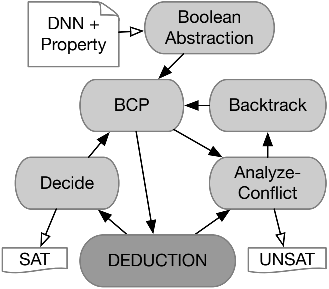
1. Abstract as a boolean satisfiability problem
2. Iteratively search for satisfying assignment
   - Use heuristics to make decision
   - Use propagation to communicate learn information
   - Analyze conflicts, learn conflict information, and backtrack
   - Use a theory solver to quickly deduce unsatisfiability (UNSAT)

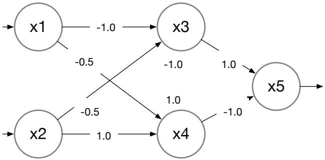
Example: Simple DNN with ReLU activation



To prove $f : x_1 \in [-1, 1] \wedge x_2 \in [-2, 2] \Rightarrow x_5 \leq 0$:

- Use NeuralSAT to check if $\neg f$ is satisfiable
- NeuralSAT($N \wedge x_1 \in [-1, 1] \wedge x_2 \in [-2, 2] \wedge x_5 > 0$)
- NeuralSAT returns UNSAT, indicating f is valid

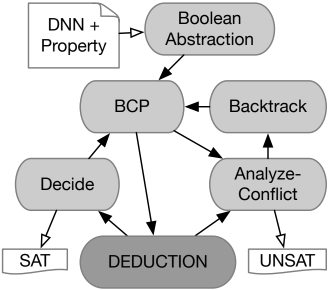


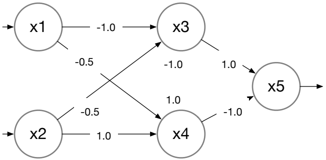


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

Boolean Abstraction

- Create 2 boolean variables v_3 and v_4 to represent *activation status* of x_3, x_4
  - $v_3 = T$ means x_3 is active, $-x_1 - 0.5x_2 - 1 > 0$

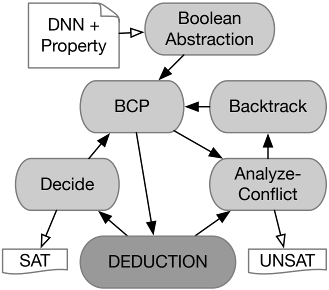


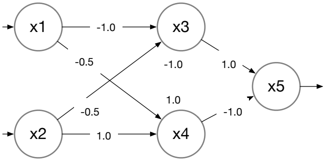


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

Boolean Abstraction

- Create 2 boolean variables v_3 and v_4 to represent *activation status* of x_3, x_4
  - $v_3 = T$ means x_3 is active, $-x_1 - 0.5x_2 - 1 > 0$
- Form two clauses $\{v_3 \vee \overline{v_3}\ ;\ v_4 \vee \overline{v_4}\}$
- Find boolean values for v_3, v_4 that satisfies the clauses and their implications

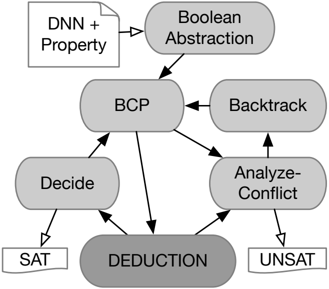


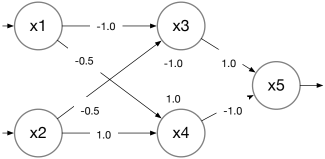


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

Iteration 1

- Use **abstraction** to approximate upperbound $x_5 \leq 0.55$ (from $x_1 \in [-1, 1], x_2 \in [-2, 2]$)

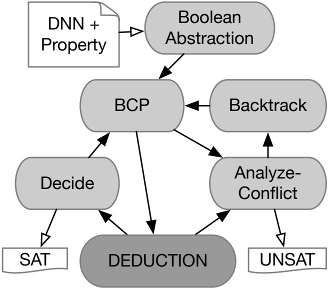


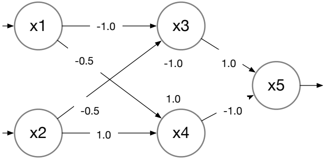


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

Iteration 1

- Use **abstraction** to approximate upperbound $x_5 \leq 0.55$ (from $x_1 \in [-1, 1], x_2 \in [-2, 2]$)
- **Deduce** $x_5 > 0$ *might be* feasible

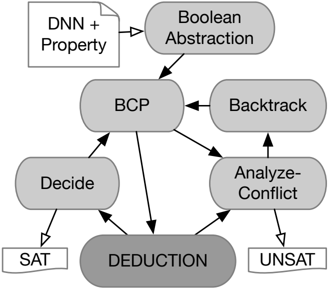


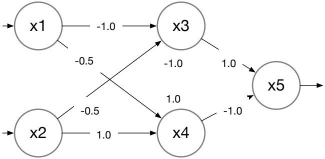


$x_1 \in [-1,1], x_2 \in [-2,2], x_5 > 0$

Iteration 1

- Use **abstraction** to approximate upperbound $x_5 \leq 0.55$ (from $x_1 \in [-1,1], x_2 \in [-2,2]$)
- **Deduce** $x_5 > 0$ *might be* feasible
- **Decide** $v_3 = F$ (randomly)
  - new constraint $-x_1 - 0.5x_2 - 1 < 0$

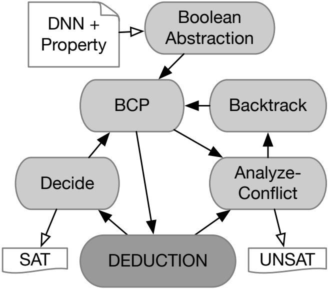


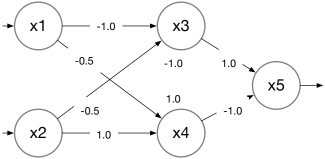


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

Iteration 2

- **Approximate** upperbound $x_5 \leq 0$ (due to additional constraint from $v_3 = F$)
- **Deduce** $x_5 > 0$ infeasible: CONFLICT

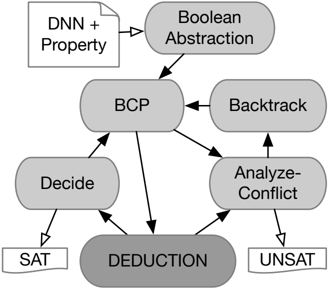


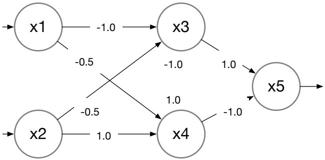


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

Iteration 2

- **Approximate** upperbound $x_5 \leq 0$ (due to additional constraint from $v_3 = F$)
- **Deduce** $x_5 > 0$ infeasible: CONFLICT
- **Analyze** conflict, **backtrack** and erase prev. decision $v_3 = F$
- **Learn** new clause v_3
  - v_3 will have to be T in next iteration

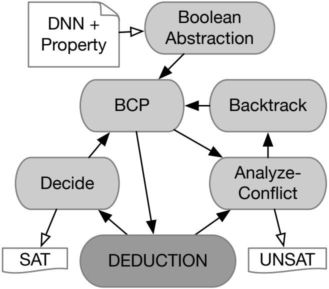


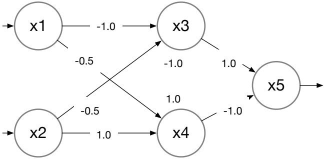


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

Iteration 3

- **Decide** $v_3 = T$ (**BCP**, due to learned clause v_3)
  - new constraint $-x_1 - 0.5x_2 - 1 > 0$

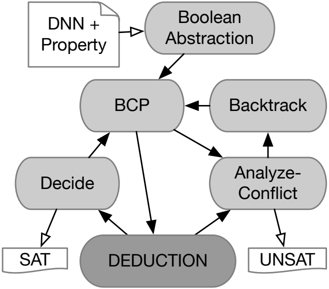


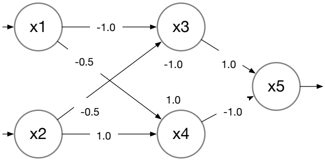


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

Iteration 3

- **Decide** $v_3 = T$ (**BCP**, due to learned clause v_3)
  - new constraint $-x_1 - 0.5x_2 - 1 > 0$
- **Approximate** new upperbound for x_5 (using additional constraint from $v_3 = T$)
- **Deduce** $x_5 > 0$ might be feasible
- **Decide** $v_4 = T$ (randomly)
- $\vdots$

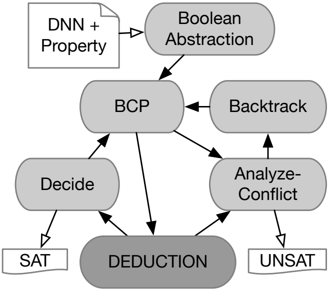


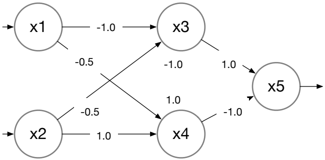


$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

After several iterations

- **Learn** clauses $\{v_3, \overline{v_3} \vee v_4, \overline{v_3} \vee \overline{v_4}\}$
- **Deduce** not possible to satisfy the clauses





$x_1 \in [-1, 1], x_2 \in [-2, 2], x_5 > 0$

After several iterations

- **Learn** clauses $\{v_3, \overline{v_3} \vee v_4, \overline{v_3} \vee \overline{v_4}\}$
- **Deduce** not possible to satisfy the clauses
- **Return** UNSAT
  - Cannot find inputs satisfying $x_1 \in [-1, 1], x_2 \in [-2, 2]$ that cause N to return $x_5 > 0$
  - Hence, $x_5 \leq 0$ holds (i.e., the original property is valid)

Benchmark	Rank	Verifier	Score	Percent	Verify	Falsify
ACAS Xu (13K)	1	NeuralSAT	1437	100.0%	**139**	**47**
	1	nnenum	1437	100.0%	**139**	**47**
	3	$\alpha\beta$-CROWN	1436	99.9%	**139**	46
	4	Marabou	1426	99.2%	138	46
	5	MN-BaB	1097	76.3%	105	**47**
MNISTFC (532K)	1	$\alpha\beta$-CROWN	582	100.0%	**56**	22
	2	NeuralSAT	573	98.5%	55	**23**
	3	nnenum	403	69.2%	39	13
	4	MN-BaB	370	63.6%	36	10
	4	Marabou	370	63.6%	35	20
CIFAR2020 (2.5M)	1	NeuralSAT	1533	100.0%	**149**	**43**
	2	$\alpha\beta$-CROWN	1522	99.3%	148	42
	3	MN-BaB	1486	96.9%	145	36
	5	nnenum	518	33.8%	50	18
RESNET_AB (354K)	1	NeuralSAT	513	100.0%	**23**	**23**
	1	$\alpha\beta$-CROWN	513	100.0%	**49**	**23**
	3	MN-BaB	363	70.8%	34	**23**
MNIST_GDVB (3M)	1	NeuralSAT	480	100.0%	**48**	0
	2	$\alpha\beta$-CROWN	400	83.3%	40	0
	3	MN-BaB	200	41.7%	20	0
Overall	1	NeuralSAT	4536	100.0%	**440**	**136**
	2	$\alpha\beta$-CROWN	4453	98.2%	432	133
	3	MN-BaB	3516	77.5%	340	116
	4	nnenum	2358	52.0%	228	78
	5	Marabou	1796	39.6%	173	66

Key Ideas

- Formalization of DNN verification
- Analyze, learn, and propagate information (significantly reduce search space)
- Dedicated DNN-specific theory solver (enable fast proving)
- *New approach; open doors to new research on heuristics, optimizations specific to DNNs*

Key Ideas

- Formalization of DNN verification
- Analyze, learn, and propagate information (significantly reduce search space)
- Dedicated DNN-specific theory solver (enable fast proving)
- *New approach; open doors to new research on heuristics, optimizations specific to DNNs*

Usability Features

- Standard: inputs (ONNX) and outputs (SAT/UNSAT/TIMEOUT)
- Versatile
  - Support Feedforward, Convolutional, Residual Networks
  - Support ReLU, Sigmoid, Tanh, Power, etc
- Scale well to large networks with millions of neurons
- Active development & frequent Updates
- Fully automatic (require little configurations from users)

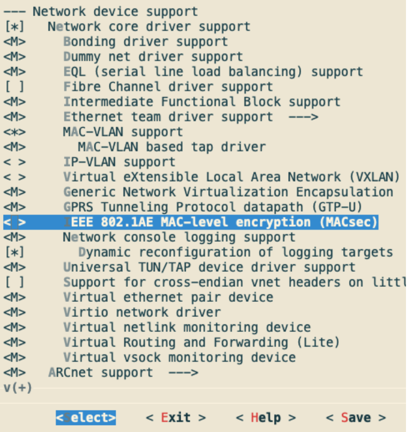
Outline

AI Safety Verification

Highly Configurable and Build Systems

Invariant Generation and Program Repair

Linux/Unix Build Systems



- Modern software are highly-configurable
  - Allow for customization and flexibility
  - Can have misconfigurations (5^{th} on OWASP most critical security risks)
- Challenge: huge search space (2^{13000} for Linux)

Linux/Unix Build Systems

```
--- Network device support
[*]   Network core driver support
<M>     Bonding driver support
<M>     Dummy net driver support
<M>     EQL (serial line load balancing) support
[ ]     Fibre Channel driver support
<M>     Intermediate Functional Block support
<M>     Ethernet team driver support  --->
<*>     MAC-VLAN support
<M>       MAC-VLAN based tap driver
< >     IP-VLAN support
< >     Virtual eXtensible Local Area Network (VXLAN)
<M>     Generic Network Virtualization Encapsulation
<M>     GPRS Tunneling Protocol datapath (GTP-U)
< >     IEEE 802.1AE MAC-level encryption (MACsec)
<M>     Network console logging support
[*]       Dynamic reconfiguration of logging targets
<M>     Universal TUN/TAP device driver support
[ ]     Support for cross-endian vnet headers on littl
<M>     Virtual ethernet pair device
<M>     Virtio network driver
<M>     Virtual netlink monitoring device
<M>     Virtual Routing and Forwarding (Lite)
<M>     Virtual vsock monitoring device
<M>   ARCnet support  --->
v(+)

   <Select>    < Exit >    < Help >    < Save >
```

- Modern software are highly-configurable
  - Allow for customization and flexibility
  - Can have misconfigurations (5^{th} on OWASP most critical security risks)
- Challenge: huge search space (2^{13000} for Linux)
- Approach: use symbolic execution to compute *path conditions* mapping to built files
  - # of files is very small
  - Solve path conds to find build issues and misconfigurations

Outline

AI Safety Verification
Highly Configurable and Build Systems
Invariant Generation and Program Repair

Invariant Generation (DIG)

```
def intdiv(x, y):
    q = 0
    r = x
    while r ≥ y:
    a = 1
    b = y
    while [??] r ≥ 2b:
        a = 2a
        b = 2b
    r = r - b
    q = q + a
    [??]
    return q
```

- Discover invariant properties at certain program locations
- Answer the question *"what does this program do ?"*
- Approach: use *template* and *dynamic analysis*

Invariant Generation (DIG)

```
def intdiv(x, y):
    q = 0
    r = x
    while r ≥ y:
    a = 1
    b = y
    while [??] r ≥ 2b:
        a = 2a
        b = 2b
    r = r - b
    q = q + a
    [??]
    return q
```

- Discover invariant properties at certain program locations
- Answer the question *“what does this program do ?”*
- Approach: use *template* and *dynamic analysis*

Program Repair (GenProg)

```
def intdiv(x, y):
    q = 0
    r = x
                ≠
    while r ≥ y:
    a = 1
              3*y
    b = y
    while r ≥ 2b:
        a = 2a
        b = 2b
    r = r - b
                  −2*a
    q = q + a

    return q
```

- *Localize* errors and *modify* code to fix bugs
- Approach: use *dynamic* and *static* analyses to identify, create, and validate patches

Awards and Impacts

AI Verification

- NSF CAREER ('23—'28)
- Amazon Research Award'23
- featured in SIGBED
- ranked 4^{th} in VNN-COMP'23 (would be 1st now)

Awards and Impacts

AI Verification

- NSF CAREER ('23—'28)
- Amazon Research Award'23
- featured in SIGBED
- ranked 4^{th} in VNN-COMP'23 (would be 1st now)

Highly-Configurable and Build System Analysis

- NSF CISE CRII '20
- NSF Formal Methods in the Field (FMiT) '23
- Meta/Facebook unrestricted gift
- Adoption: used internally at Meta Whatsapp to analyze build issues

Awards and Impacts

AI Verification

- NSF CAREER ('23—'28)
- Amazon Research Award'23
- featured in SIGBED
- ranked 4th in VNN-COMP'23 (would be 1st now)

Highly-Configurable and Build System Analysis

- NSF CISE CRII '20
- NSF Formal Methods in the Field (FMiT) '23
- Meta/Facebook unrestricted gift
- Adoption: used internally at Meta Whatsapp to analyze build issues

Invariant Generation and Automatic Program Repair

- 10-year ACM SIGSOFT/IEEE TCSE Most Influential Paper Award'19
- 10-year ACM SIGEVO Most Impact Award'19
- NSF Medium Collaborative grant '21–'25
- Army Office of Research '18–'21
- Adoption
  - SV-COMP included benchmarks created by DIG
  - GrammaTech integrated DIG in Mnemosyne
  - Facebook and GrammaTech used GenProg in multiple projects

Future Directions

Currently

- focuses on *existing* problems (robustness, safety)
- tested with *existing* benchmarks

Future Directions

Currently

- focuses on *existing* problems (robustness, safety)
- tested with *existing* benchmarks

Challenges & Opportunities

- new problems
  - what properties should AI/ML have? (e.g., fairness, privacy, security)
  - how to formally define such specifications?
- new benchmarks (e.g., real-world, industrial data)
- new analyses (e.g., automatic property inference and repair for NNs)

Funding

- 8 grants: 4 NSF (3 sole-PI, 1 PI), 1 Defense (Co-PI), 2 industry (sole-PI), 1 internal (sole-PI)
  - Total **$2.65M**; my share $1.5M, as PI $1.3M
  - At GMU (total **$1.9M**, my/GMU share $1.1M, as PI $1.1M)
  - Young Faculty: NSF CRII'20, NSF CAREER'23, Amazon Research Award'23

Funding

- 8 grants: 4 NSF (3 sole-PI, 1 PI), 1 Defense (Co-PI), 2 industry (sole-PI), 1 internal (sole-PI)
  - Total **$2.65M**; my share $1.5M, as PI $1.3M
  - At GMU (total **$1.9M**, my/GMU share $1.1M, as PI $1.1M)
  - Young Faculty: NSF CRII'20, NSF CAREER'23, Amazon Research Award'23

Publications

- 27 journals/confs. papers since '16 (11 since joining GMU in '21)
  - 20 papers with students (9 with undergrad)
- Google: 3617 citations (h-index 17 i10-index 24)
- SIGSOFT MIP paper award, SIGEVO Impact paper award

Funding

- 8 grants: 4 NSF (3 sole-PI, 1 PI), 1 Defense (Co-PI), 2 industry (sole-PI), 1 internal (sole-PI)
  - Total **$2.65M**; my share $1.5M, as PI $1.3M
  - At GMU (total **$1.9M**, my/GMU share $1.1M, as PI $1.1M)
  - Young Faculty: NSF CRII'20, NSF CAREER'23, Amazon Research Award'23

Publications

- 27 journals/confs. papers since '16 (11 since joining GMU in '21)
  - 20 papers with students (9 with undergrad)
- Google: 3617 citations (h-index 17 i10-index 24)
- SIGSOFT MIP paper award, SIGEVO Impact paper award

Students Mentoring

- Current: 3 Ph.D RA's, 2 undergrads
- Graduated (at UNL): 1 PhD, 2 Masters, 11 undergrads (2 Outstanding Undergrad Research Awards)

Funding

- 8 grants: 4 NSF (3 sole-PI, 1 PI), 1 Defense (Co-PI), 2 industry (sole-PI), 1 internal (sole-PI)
  - Total **$2.65M**; my share $1.5M, as PI $1.3M
  - At GMU (total **$1.9M**, my/GMU share $1.1M, as PI $1.1M)
  - Young Faculty: NSF CRII'20, NSF CAREER'23, Amazon Research Award'23

Publications

- 27 journals/confs. papers since '16 (11 since joining GMU in '21)
  - 20 papers with students (9 with undergrad)
- Google: 3617 citations (h-index 17 i10-index 24)
- SIGSOFT MIP paper award, SIGEVO Impact paper award

Students Mentoring

- Current: 3 Ph.D RA's, 2 undergrads
- Graduated (at UNL): 1 PhD, 2 Masters, 11 undergrads (2 Outstanding Undergrad Research Awards)

Teaching

- At GMU (2 years): 1 grad (2x, required, SWE619), 1 undergrad (SWE419), 1 seminar (CS695)
- Developed online SWE619 course with Wiley (went live in Spring'23)

Funding

- 8 grants: 4 NSF (3 sole-PI, 1 PI), 1 Defense (Co-PI), 2 industry (sole-PI), 1 internal (sole-PI)
    - Total **$2.65M**; my share $1.5M, as PI $1.3M
    - At GMU (total **$1.9M**, my/GMU share $1.1M, as PI $1.1M)
    - Young Faculty: NSF CRII'20, NSF CAREER'23, Amazon Research Award'23

Publications

- 27 journals/confs. papers since '16 (11 since joining GMU in '21)
    - 20 papers with students (9 with undergrad)
- Google: 3617 citations (h-index 17 i10-index 24)
- SIGSOFT MIP paper award, SIGEVO Impact paper award

Students Mentoring

- Current: 3 Ph.D RA's, 2 undergrads
- Graduated (at UNL): 1 PhD, 2 Masters, 11 undergrads (2 Outstanding Undergrad Research Awards)

Teaching

- At GMU (2 years): 1 grad (2x, required, SWE619), 1 undergrad (SWE419), 1 seminar (CS695)
- Developed online SWE619 course with Wiley (went live in Spring'23)

Services

- Regularly serve in well-known confs/journals, 7 NSF panels in past 5 consec. yrs
- At GMU: program director of MS SWE; organize Virtual Open House; maintain CSRankings DB (GMU is ranked 32!)