CAISAR: A Platform for Characterizing Artificial Intelligence Safety and Robustness
AISafety 2022

Julien Girard-Satabin (CEA LIST): julien.girard2@cea.fr
Formal verification for machine learning
Adapted from https://github.com/nemanja-rakicevic/conference_historical_data_analysis
Non-exhaustive list of tools

Include only the latest ”version” (including extensions and rebranding)

- Marabou [Kat+19]
- Neurify
- ERAN [Sin+19; Mül+21]
- $\alpha - \beta$-Crown [Wan+21]
- Nnenum [Bak21]
- NNV (https://github.com/verivital/nnv)
- Facet-Vertex incidence (https://github.com/Shaddadi/Facet-Vertex-FFNN)
- Veritex (https://github.com/Shaddadi/veritex)
- Verinet and Venus (https://github.com/vas-group-imperial/VeriNet)
Non-exhaustive list of tools

Include only the latest "version" (including extensions and rebranding)

- Oval (https://github.com/oval-group/oval-bab)
- Libra [Urb+19]
- MIPVerify [TXT19]
- Planet [Ehl17]
- Sherlock [Dut+17]
- DNNV [SED21]
it's the Cambrian explosion

(credits: Bill Wurtz)
Lots of tools increases burden of choice

Which tool to choose?

How to encode a given problem for multiple tools?

Could we specify a verification problem independently of the tool?
A new, thriving ecosystem

Selective pressure

Short lifetime of tools: Reluplex to Marabou, $A_1^2$ to ERAN, Fast-Lin to $\alpha - \beta$-CROWN

Ecological niches: from fully-connected neural networks to state-of-the-art architectures, by way of SVMs

Collaboration

Collaborative initiatives: VNN-COMP or VNNLib

Cross-fertilization of techniques across tools: symbolic propagation [Li+19; Sun+18; HL20; Wan+21], efficient space partitionning [Urb+19; Gir+21], mixing exact solvers and fast bound propagation [Gil+18; Fer+22]
Families of properties according to the literature

local robustness: given $x_0$
\[ \forall x, \text{dist}(x - x_0) < \varepsilon \implies f(x) = f(x_0) \]

clearly defined semantics à la ACAS

global properties on low-dimensional programs
Existing benchmarks: ACAS-like and local adversarial robustness?

What about characterizing privacy? Fairness? Symmetry relations? How to phrase custom properties for provers that are not designed to?
Handling complex systems?

How to compose system components in the analysis?
Three interesting venues

1. tool-independant modelling
2. flexibility in problem statement
3. composition of components
platform for Characterizing Artificial Intelligence Safety And Robustness
A platform building from principled and industrial-tested techniques

Written in OCaml, using Why3 as backend
Overall architecture of CAISAR

Supports SMT and abstract interpretation reasoning (Marabou, PyRAT, SAVer), and soon metamorphic testing (AIMOS)
Theory T

Use Net.NNasTuple
Use SVM.SVMAsArray
Use ieee_float.Float64
Use caisar.NN

goal G: for all x1 x2 x3.
  (0.0:t). < x1 .< (0.5:t) ->
  let (y1,y2) = Net.net_apply x1 x2 x3 in
  let (z1,z2) = SVM.svm_apply y1 y2 in
  (0.0:t). < z1 .< (0.5:t)
  /
  (0.0:t). < z2 .< (0.5:t)
end
WhyML language: expressivity

first-order language with polymorphic types, pattern matching, and inductive properties capabilities
modelization freedom to define a vast set of property

\begin{verbatim}
predicate dist_linf
  (a: input_type)
  (b: input_type)
  (eps:t)
  (n: int) =
  forall i. 0 <= i < n ->
         - eps < a_i - b_i < eps

predicate robust_to
  (model: model)
  (a: input_type)
  (eps: t) =
  forall b. dist_linf a b eps
  model.num_input ->
  model.app a = model.app b
\end{verbatim}
Future work

• support more prominent verifiers (among ERAN, nnenum) as well as VNNLib/SMTLIB format
• how to compose several verifiers techniques (metamorphic tests plus formal verification)?
• how to choose the proper prover heuristics? more generally, how to refine and adapt proof strategies according to one’s need?
CAISAR maturation is partially funded by the Confiance.IA program


Julien Girard-Satabin, Aymeric Varasse, Marc Schoenauer, Guillaume Charpiat, and Zakaria Chihani. “DISCO: Division of Input Space into COnvex polytopes for neural network verification”. In: JFLA (2021) (cit. on p. 8).

