tinyML® Talks

Enabling Ultra-low Power Machine Learning at the Edge

"Verification of ML-based AI systems and its applicability in Edge ML"

Alessio Lomuscio - Imperial College of London

October 5, 2021

www.tinyML.org

tinyML Talks Sponsors and Strategic Partners

tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

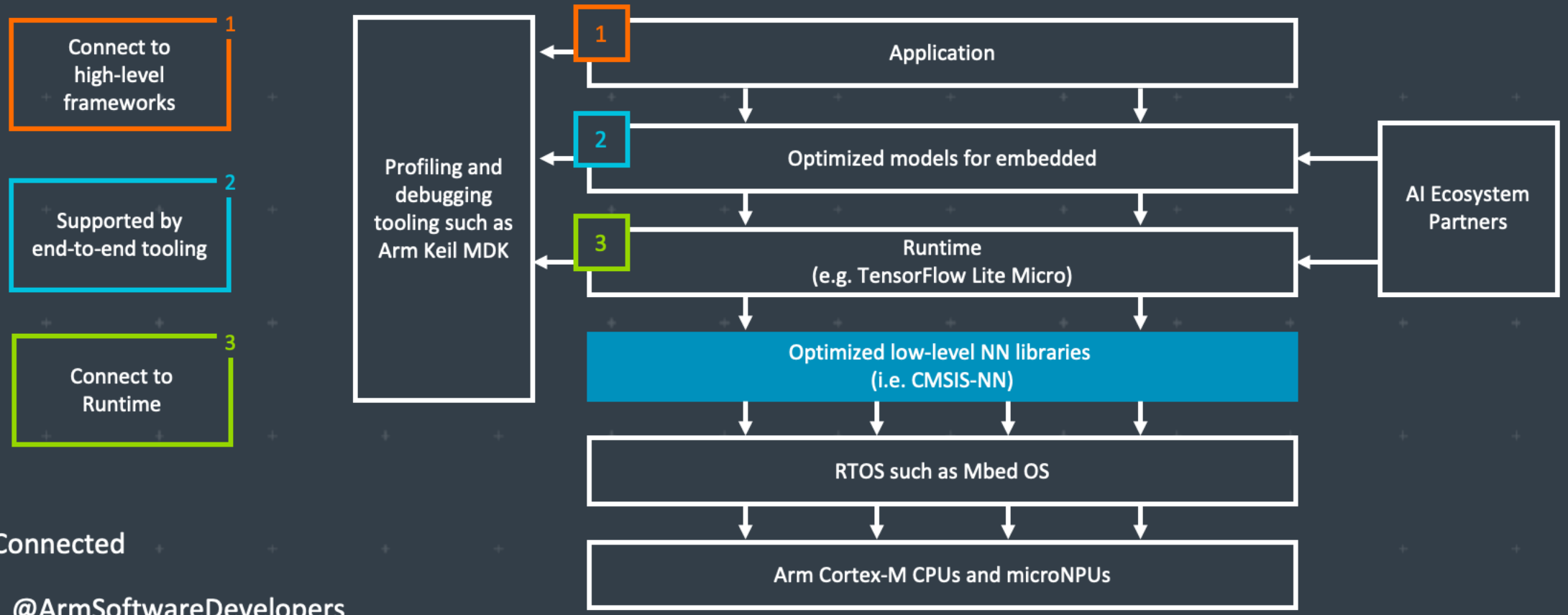
tinyML Strategic Partner

tinyML Strategic Partner

tinyML Strategic Partner

Additional Sponsorships available – contact Olga@tinyML.org for info

Arm: The Software and Hardware Foundation for tinyML



Stay Connected

@ArmSoftwareDevelopers

@ArmSoftwareDev

Resources: developer.arm.com/solutions/machine-learning-on-arm

arm

Deeplite

WE USE AI TO MAKE OTHER AI FASTER, SMALLER AND MORE POWER EFFICIENT

Automatically compress SOTA models like MobileNet to <200KB with **little to no drop in accuracy** for inference on resource-limited MCUs

Reduce model optimization trial & error from weeks to days using Deeplite's **design space exploration**

Deploy more models to your device without sacrificing performance or battery life with our **easy-to-use software**

BECOME BETA USER **bit.ly/testdeeplite**

mobilityXlab arm AI 100 CBINSIGHTS

TinyML for all developers

C++ library

Arduino library

WebAssembly

Dataset

Acquire valuable training data securely

Enrich data and train ML algorithms

Edge Device

Real sensors in real time

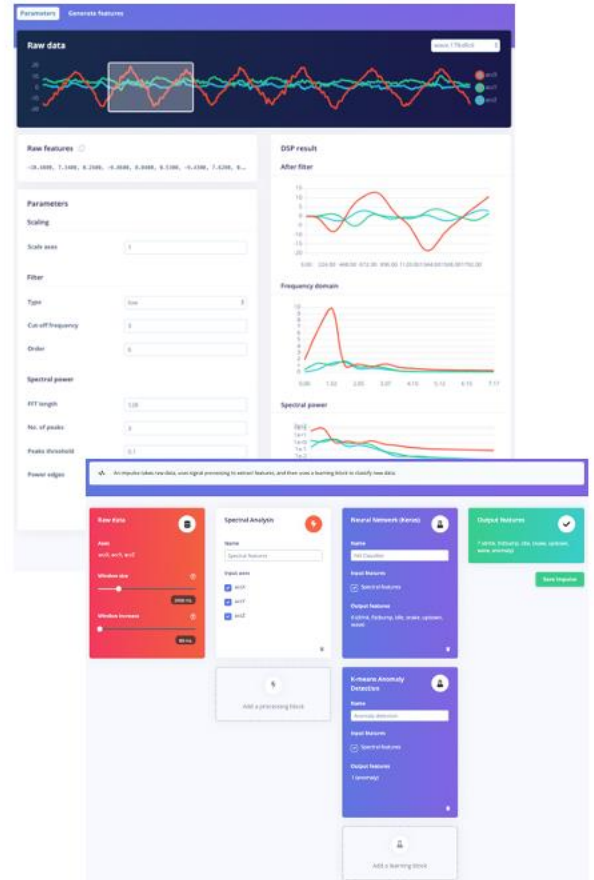
Open source SDK

Impulse

Embedded and edge compute deployment options

Test impulse with real-time device data flows

Test

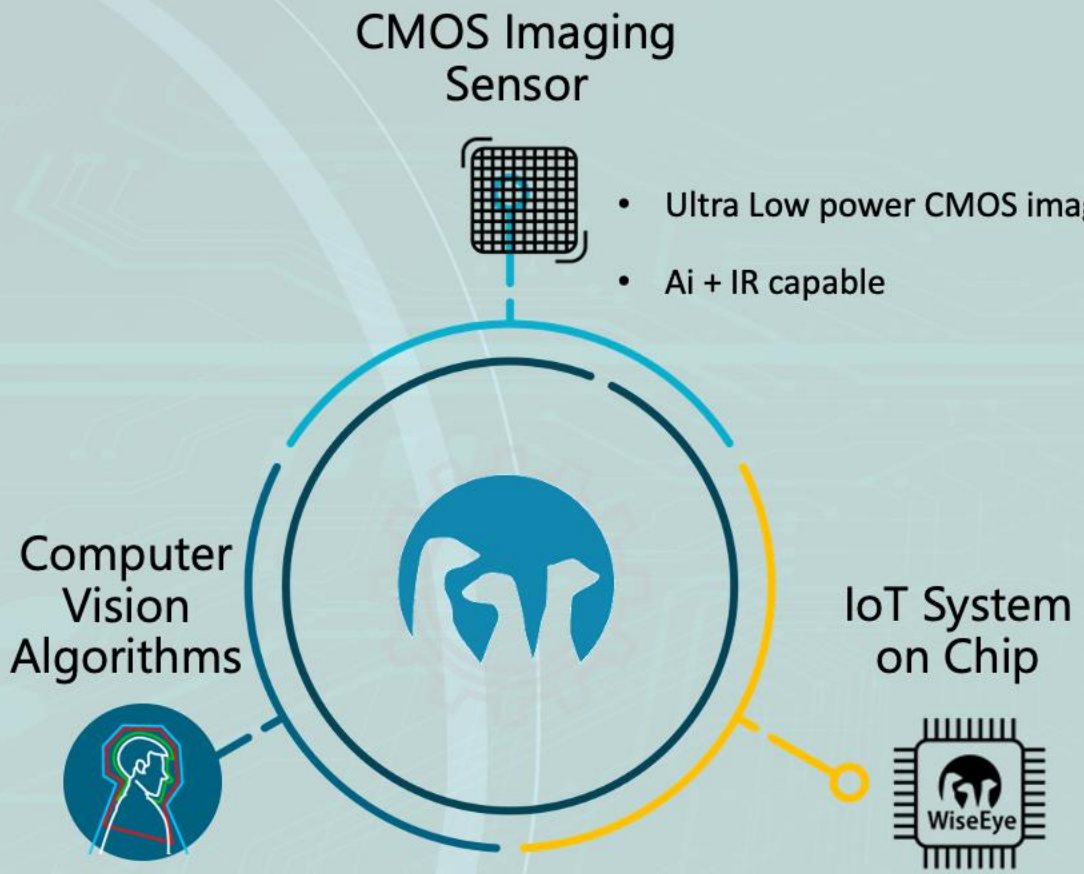


www.edgeimpulse.com

emza
visual sense

The Eye in IoT

Edge AI Visual Sensors



- Ultra Low power CMOS imager
- Ai + IR capable

- Machine Learning algorithm
- <1MB memory footprint
- Microcontrollers computing power
- Trained algorithm
- Processing of low-res images
- Human detection and other classifiers

- Machine Learning edge computing silicon
- <1mW always-on power consumption
- Computer Vision hardware accelerators

info@emza-vs.com

Enabling the next generation of Sensor and Hearable products to process rich data with energy efficiency

Visible Image

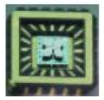
Sound

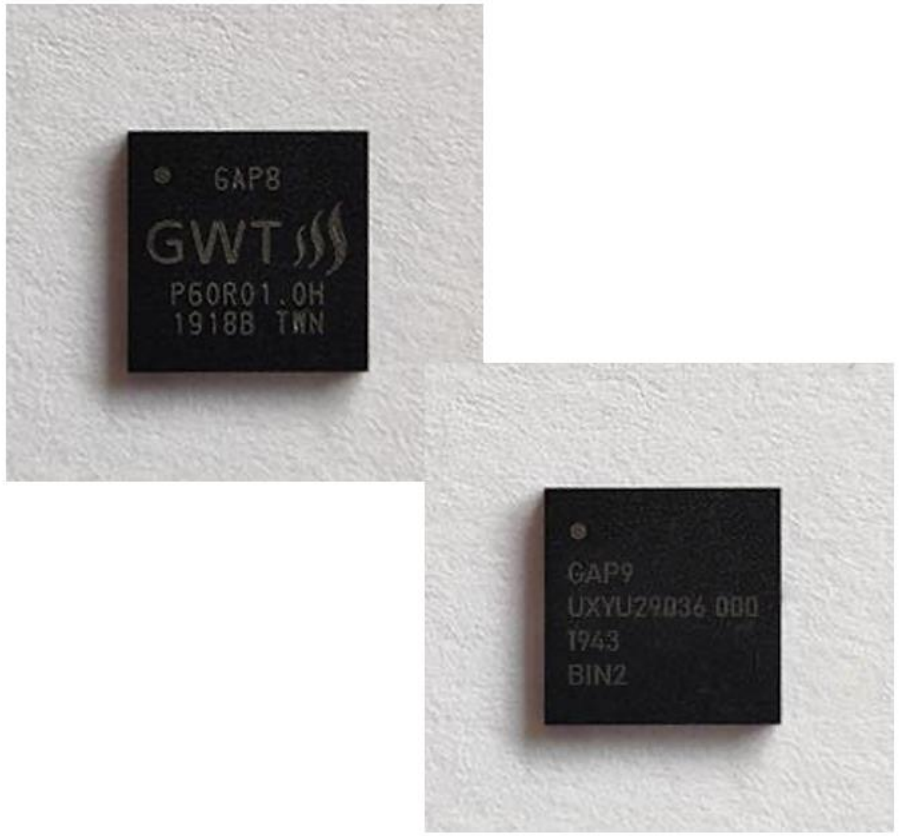
IR Image

Radar

Bio-sensor

Gyro/Accel

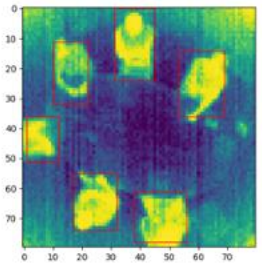




Wearables / Hearables

Battery-powered consumer electronics

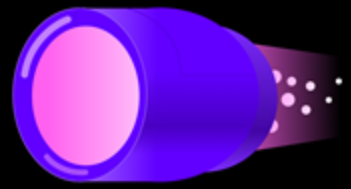
IoT Sensors



GREENWAVES TECHNOLOGIES

Distributed infrastructure for TinyML apps

HOTG
Decoupling intelligence



Develop at warp speed

Automate deployments

Device orchestration

HOTG is building the distributed infrastructure to pave the way for AI enabled edge applications

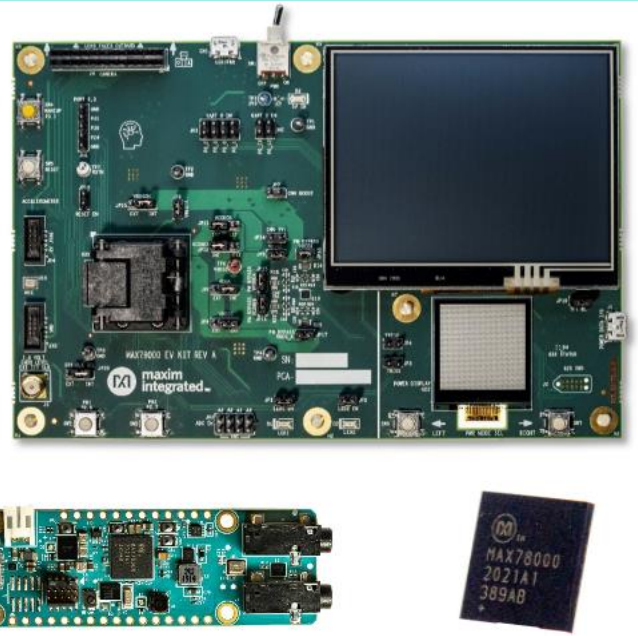
LatentAI

Adaptive AI for the Intelligent Edge

Latentai.com

Maxim Integrated: Enabling Edge Intelligence

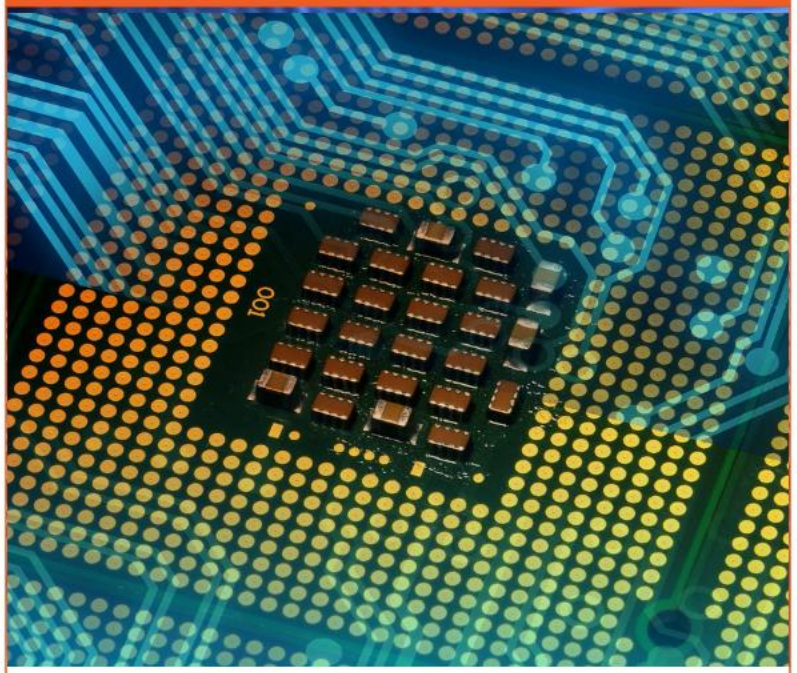
Advanced AI Acceleration IC



The new MAX78000 implements AI inferences at low energy levels, enabling complex audio and video inferencing to run on small batteries. Now the edge can see and hear like never before.

www.maximintegrated.com/MAX78000

Low Power Cortex M4 Micros



Large (3MB flash + 1MB SRAM) and small (256KB flash + 96KB SRAM, 1.6mm x 1.6mm) Cortex M4 microcontrollers enable algorithms and neural networks to run at wearable power levels.

www.maximintegrated.com/microcontrollers

Sensors and Signal Conditioning

Health sensors measure PPG and ECG signals critical to understanding vital signs. Signal chain products enable measuring even the most sensitive signals.

www.maximintegrated.com/sensors

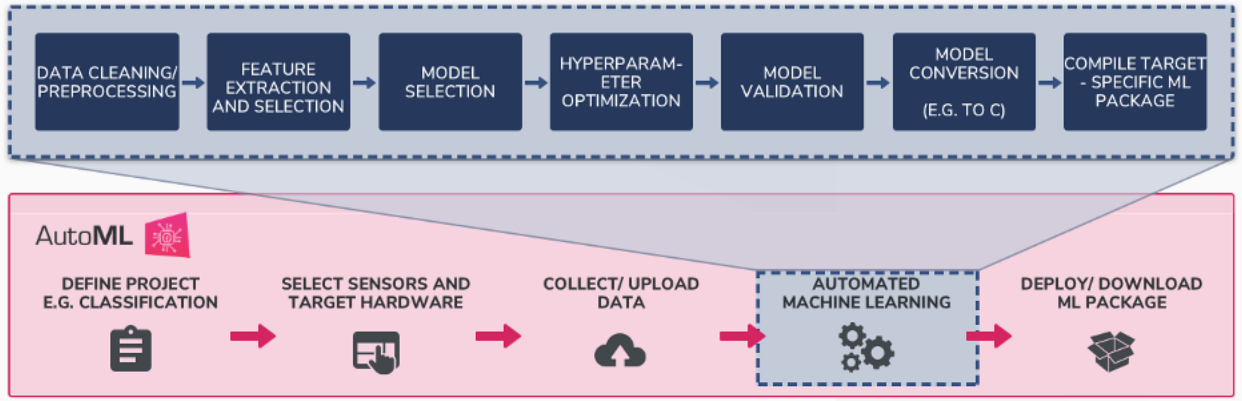
Qeexo AutoML

Automated Machine Learning Platform that builds tinyML solutions for the Edge using sensor data

Key Features

- Supports 17 ML methods:
  - Multi-class algorithms: GBM, XGBoost, Random Forest, Logistic Regression, Gaussian Naive Bayes, Decision Tree, Polynomial SVM, RBF SVM, SVM, CNN, RNN, CRNN, ANN
  - Single-class algorithms: Local Outlier Factor, One Class SVM, One Class Random Forest, Isolation Forest
- Labels, records, validates, and visualizes time-series sensor data
- On-device inference optimized for low latency, low power consumption, and small memory footprint applications
- Supports Arm® Cortex™- M0 to M4 class MCUs

End-to-End Machine Learning Platform



For more information, visit: www.qeexo.com

Target Markets/Applications

- Industrial Predictive Maintenance
- Smart Home
- Wearables
- Automotive
- Mobile
- IoT

Qualcomm
AI research

Advancing AI research to make efficient AI ubiquitous

Power efficiency

Model design, compression, quantization, algorithms, efficient hardware, software tool

Personalization

Continuous learning, contextual, always-on, privacy-preserved, distributed learning

Efficient learning

Robust learning through minimal data, unsupervised learning, on-device learning

A platform to scale AI across the industry

Perception

Object detection, speech recognition, contextual fusion

Reasoning

Scene understanding, language understanding, behavior prediction

Action

Reinforcement learning for decision making

Edge cloud

Cloud

IoT/IIoT

Automotive

Mobile

Reality AI®

Add Advanced Sensing to your Product with Edge AI / TinyML

https://reality.ai info@reality.ai @SensorAI Reality AI

Pre-built Edge AI sensing modules, plus tools to build your own

Reality AI solutions

Prebuilt sound recognition models for indoor and outdoor use cases

Solution for industrial anomaly detection

Pre-built automotive solution that lets cars "see with sound"

Reality AI Tools® software

Build prototypes, then turn them into real products

Explain ML models and relate the function to the physics

Optimize the hardware, including sensor selection and placement

SensiML™

Build Smart IoT Sensor Devices From Data

SensiML pioneered TinyML software tools that auto generate AI code for the intelligent edge.

- End-to-end AI workflow
- Multi-user auto-labeling of time-series data
- Code transparency and customization at each step in the pipeline

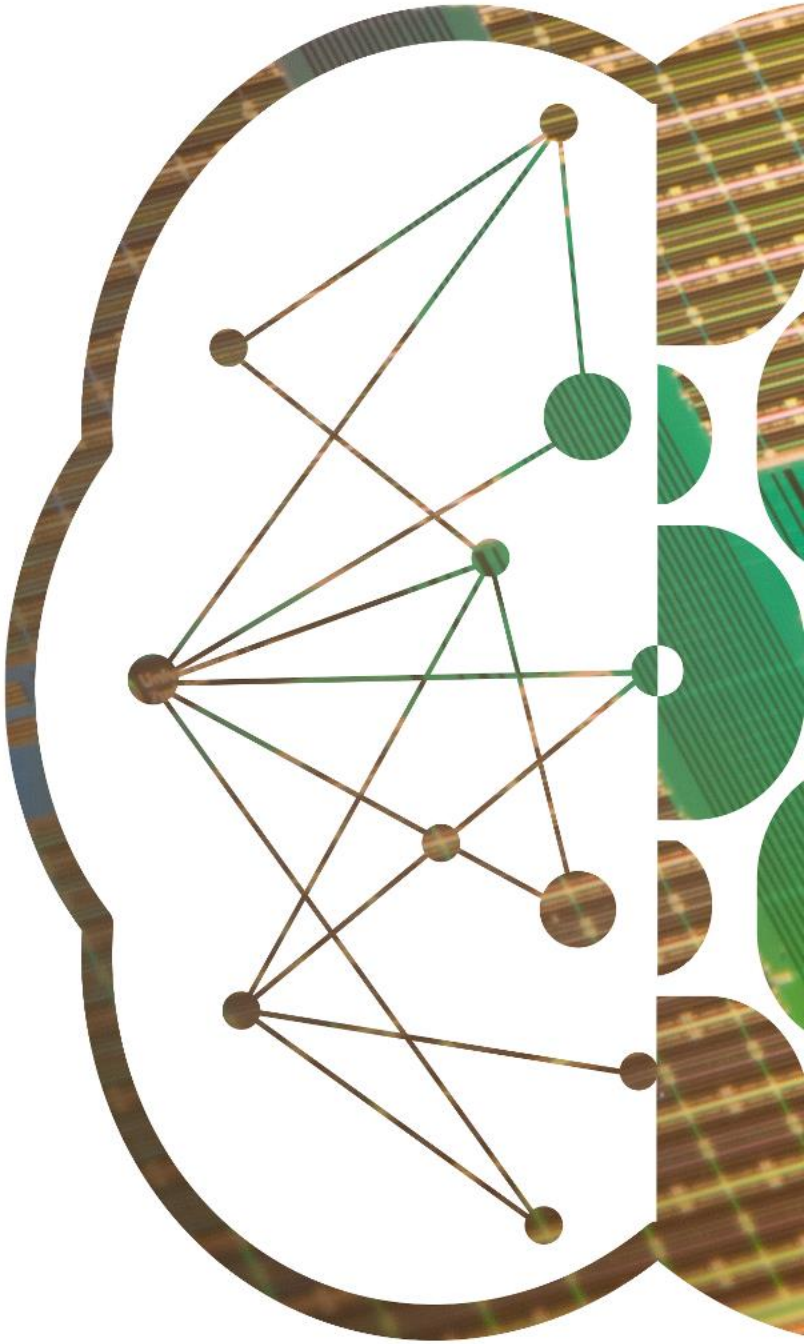
We enable the creation of production-grade smart sensor devices.

sensiml.com

SynSense

SynSense builds **sensing and inference** hardware for **ultra-low-power** (sub-mW) **embedded, mobile and edge** devices. We design systems for **real-time always-on smart sensing**, for audio, vision, IMUs, bio-signals and more.

https://SynSense.ai

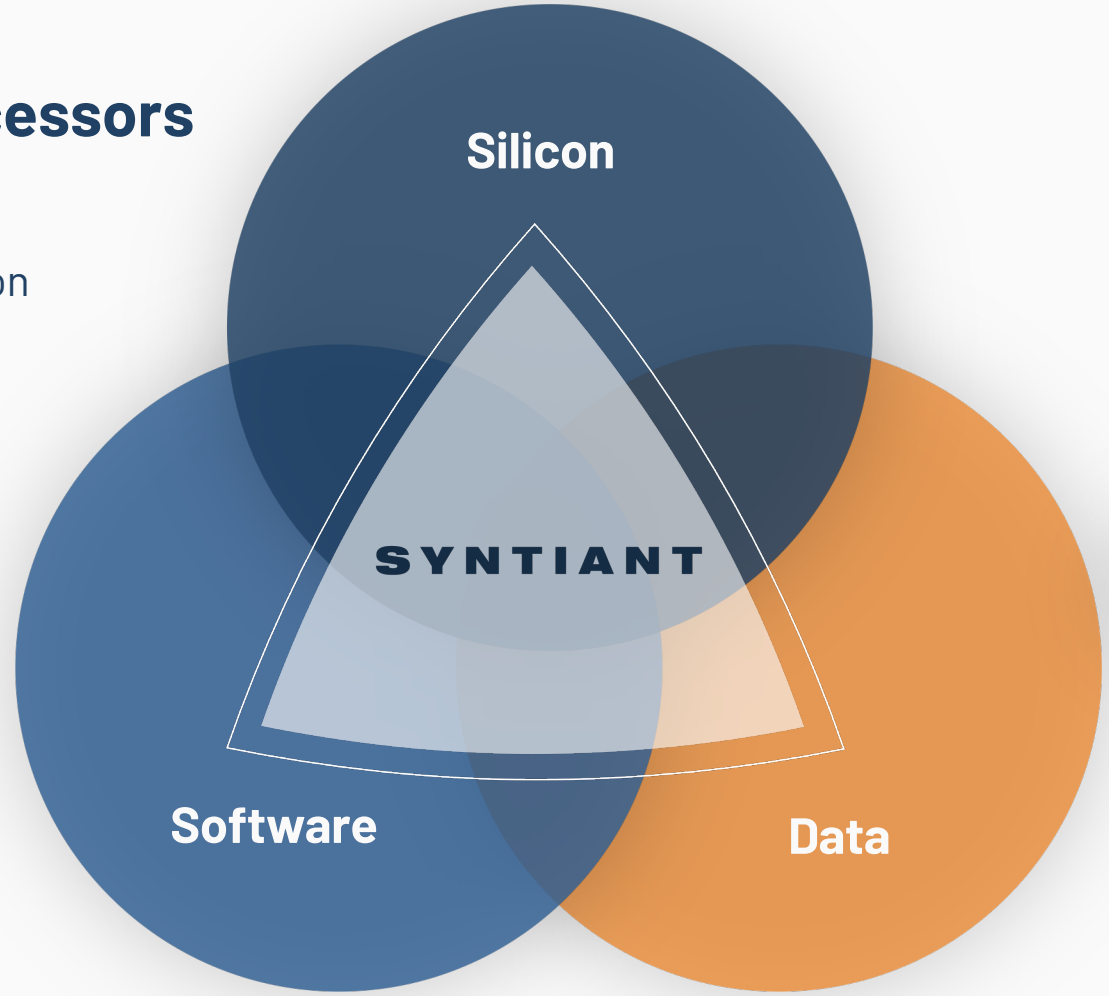


SYNTIANT

End-to-End Deep Learning Solutions for TinyML & Edge AI

Neural Decision Processors

- At-Memory Compute
- Sustained High MAC Utilization
- Native Neural Network Processing



ML Training Pipeline

- Enables Production Quality Deep Learning Deployments

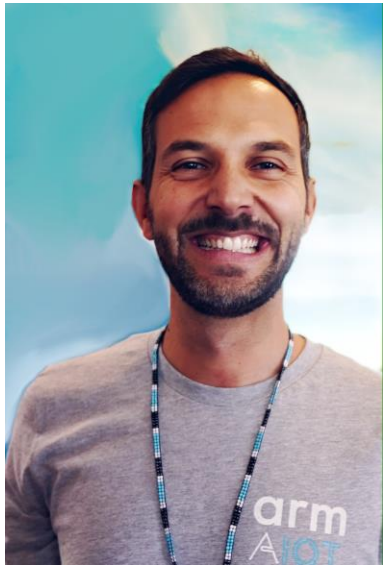
Data Platform

- Reduces Data Collection Time and Cost
- Increases Model Performance

partners@syntiant.com

www.syntiant.com

tinyML UK Committee



Alessandro Grande
Director of Technology,
Edge Impulse

Dominic Binks
VP Technology,
Audio Analytic

Gian Marco Iodice
ML Techlead,
Arm

Neil Cooper
VP Marketing,
Audio Analytic

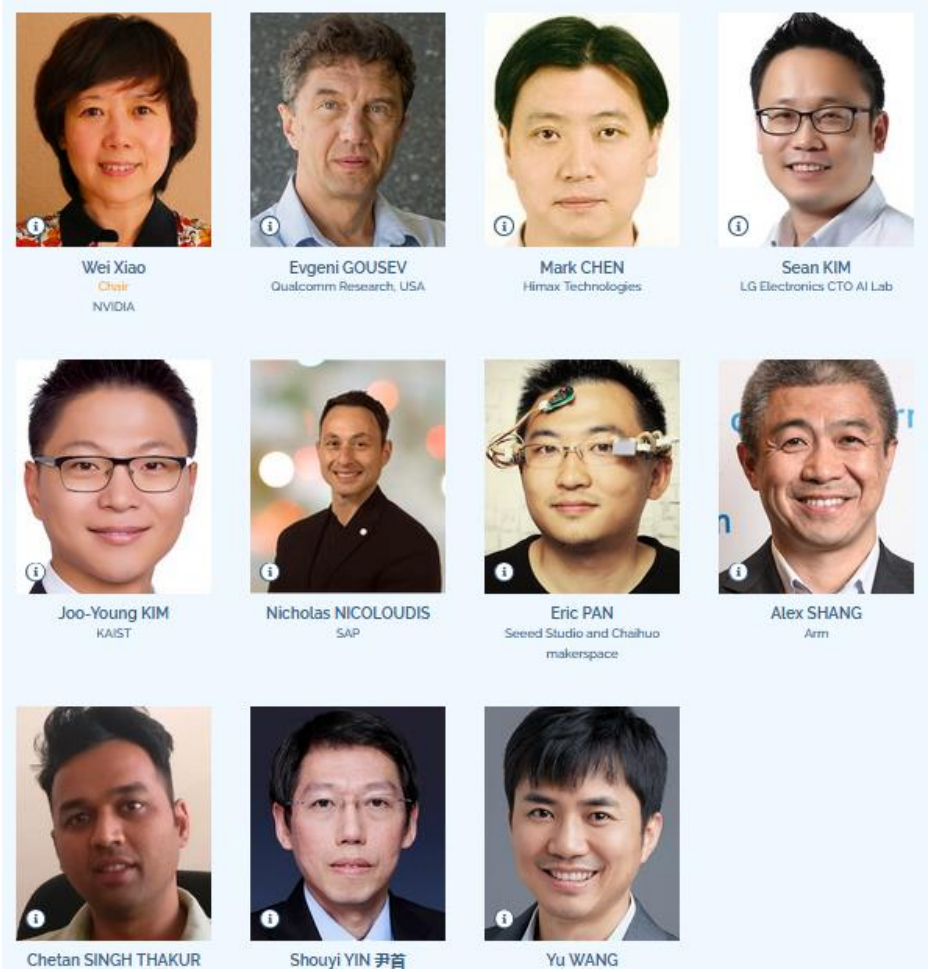
TINY ML ASIA

LIVE ONLINE **November 2-5, 2021**

(9-11:30 am China Standard time)

https://www.tinyml.org/event/asia-2021/

Technical Programm Committee



Register today!

Free event courtesy of our sponsors and strategic partners

arm · EDGE IMPULSE · emza visual sense · GREENWAVES TECHNOLOGIES

HOTG · imagimob · LatentAI Adaptive AI for a Smarter Edge · Qualcomm

Qeexo · RealityAI · seeed studio The IoT Hardware Enabler · SensiML

SynSense · SYNTIANT

More sponsorships are available: sponsorships@tinyML.org

TINY ML FOUNDATION

collaboration with hackster.io

AN AVNET COMMUNITY

Focus on:
(i) developing new use cases/apps for tinyML vision; and (ii) promoting tinyML tech & companies in the developer community

Submissions accepted until September 17th, 2021

Winners announced on October 5th, 2021 ($6k value)

Sponsorships available: *sponsorships@tinyML.org*

https://www.hackster.io/contests/tinyml-vision

TINY ML TALKS webcast

Next tinyML Talks

Date	Presenter	Topic / Title
Thursday, October 7	**Ravishankar Sivalingam, Ph.D.** Sr. Staff Engineer/Manager, Qualcomm AI Research	Enabling Ultra-low Power Always-On Computer Vision at Qualcomm

Webcast start time is 8 am Pacific time

Please contact talks@tinyml.org if you are interested in presenting

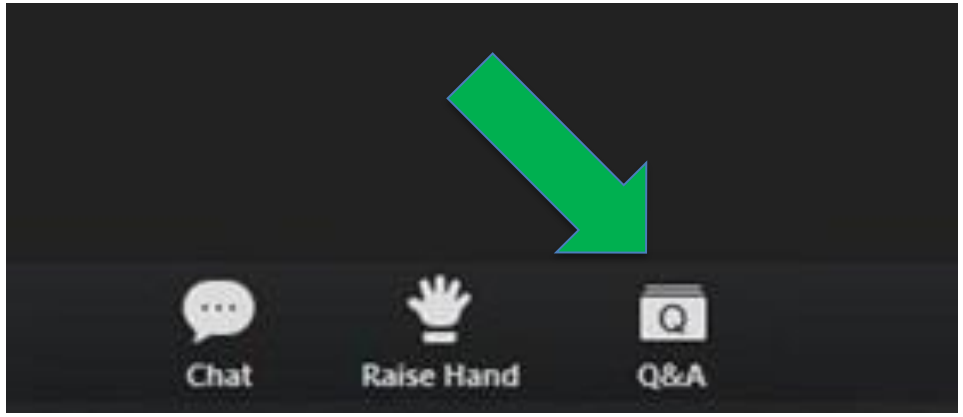
Reminders

Slides & Videos will be posted tomorrow

tinyml.org/forums

youtube.com/tinyml

Please use the Q&A window for your questions



TINY ML TALKS webcast

Alessio Lomuscio

Alessio Lomuscio (http://www.doc.ic.ac.uk/~alessio) is Professor of Safe Artificial Intelligence in the Department of Computing at Imperial College London (UK), where he leads the Verification of Autonomous Systems Lab. He serves as Deputy Director for the UKRI Doctoral Training Centre in Safe and Trusted Artificial Intelligence. He is a Distinguished ACM member, a Fellow of the European Association of Artificial Intelligence and currently holds a Royal Academy of Engineering Chair in Emerging Technologies.

Verification of ML-based AI systems and its applicability in Edge ML

E. Botoeva, P. Henriksen, F. Leofante, P. Kouvaros, A. Lomuscio

Verification of Autonomous Systems Lab
Imperial College London, UK

5 October 2021

5W1H on Verifying Edge ML

- **Who** we are and **where** we come from.
- **Why** we'd like to verify ML before deployment.
- **What** we're verifying in ML.
- **How** we're verifying neural systems.
- **Which** ML systems we're verifying
- **When** you'll be able to verify Edge ML applications.

Plan for the talk

- Verification of Autonomous Systems Lab at Imperial.
- Path towards safe AI.
- (Some!) difficulties with AI in high-reliability systems.
- Novel CS approaches to verify neural systems.
- Usecases with a major aircraft manufacturer.
- Conclusions.

Who - VAS@Imperial

Group of 15 researchers working on Safe Artificial Intelligence.

- Passionate about AI.
- Belief in robust CS/AI and engineering.
- Devoted to safety and correctness.
- Worried about consequences of poorly engineered AI for society and AI adoption.

Work presently funded by UK Research and Innovation Council, Royal Academy of Engineering, and DARPA.

Verifying the Autsub6000 [2010]

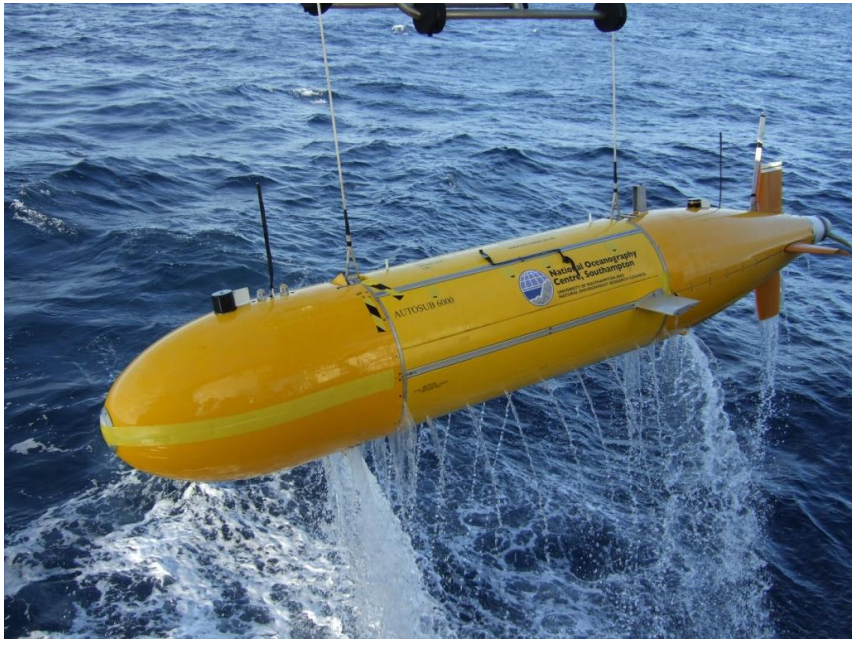


Figure: The NOC Autosub6000.
Weight: 5tons; Length: 5m; Range: 1000km;
Max Depth: 6,000m; Endurance: 10days.

- Engineering design translated and discretised from Simulink/Stateflow into ISPL.
- Different sub-modules encoded as different cooperating agents.
- Several missions verified before deployment.
- Not just correctness but also fault-tolerance.

Assessing the Autsub6000's Reliability

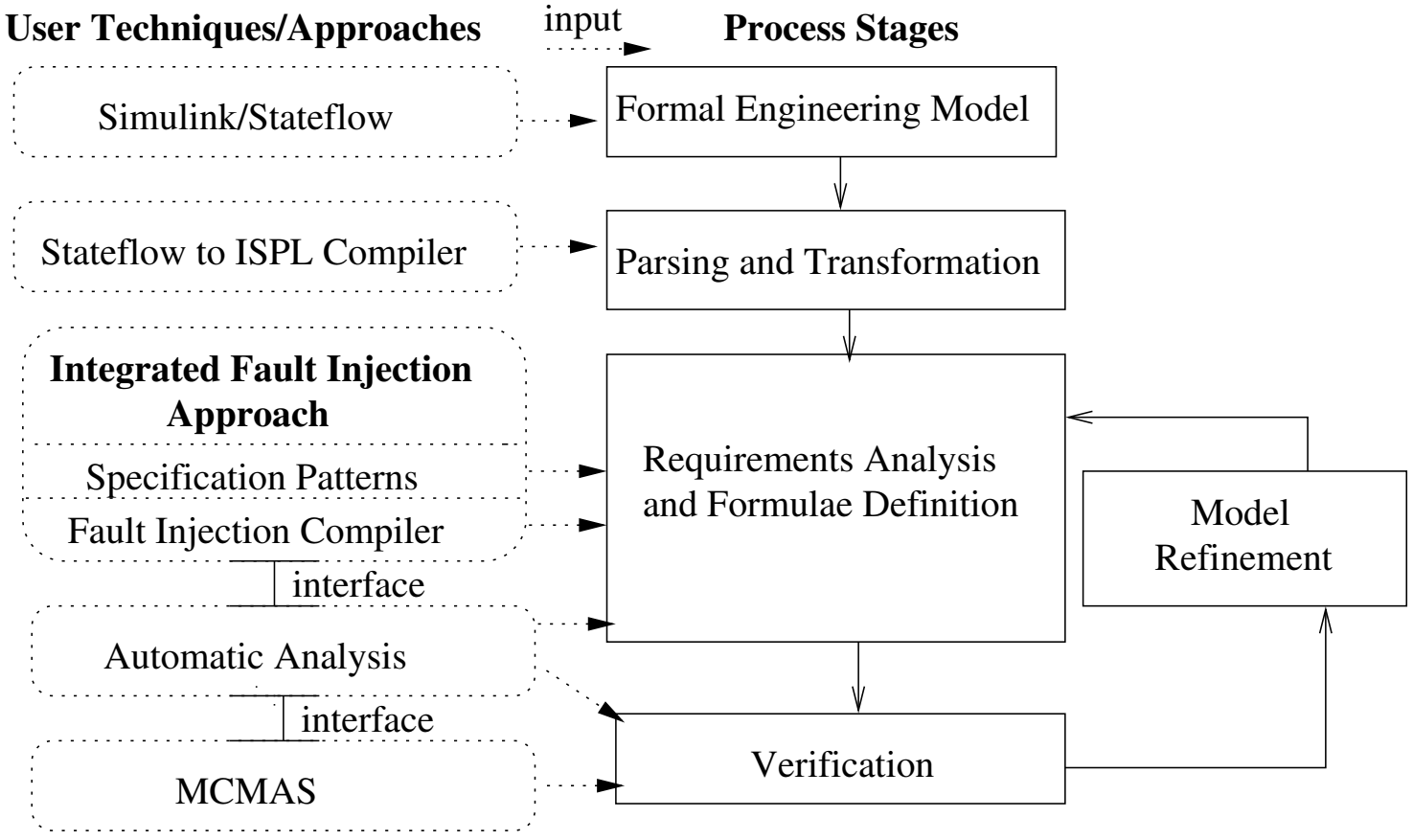
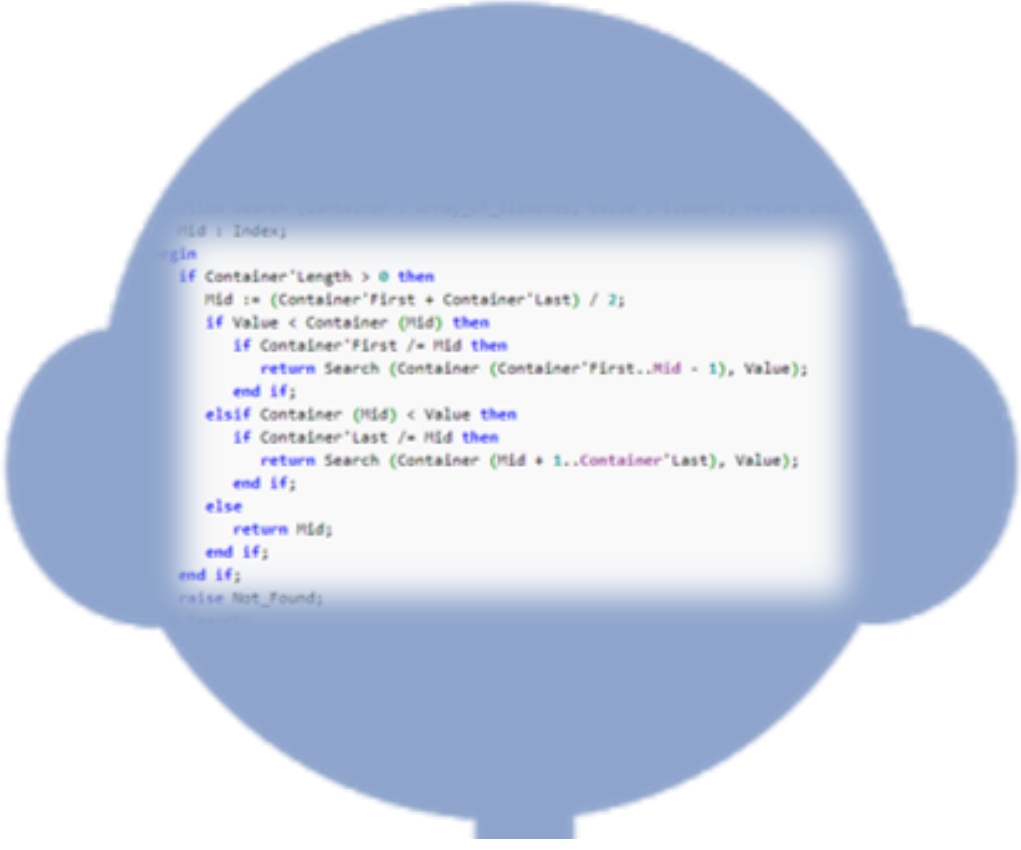


Figure: The IFAS methodology.

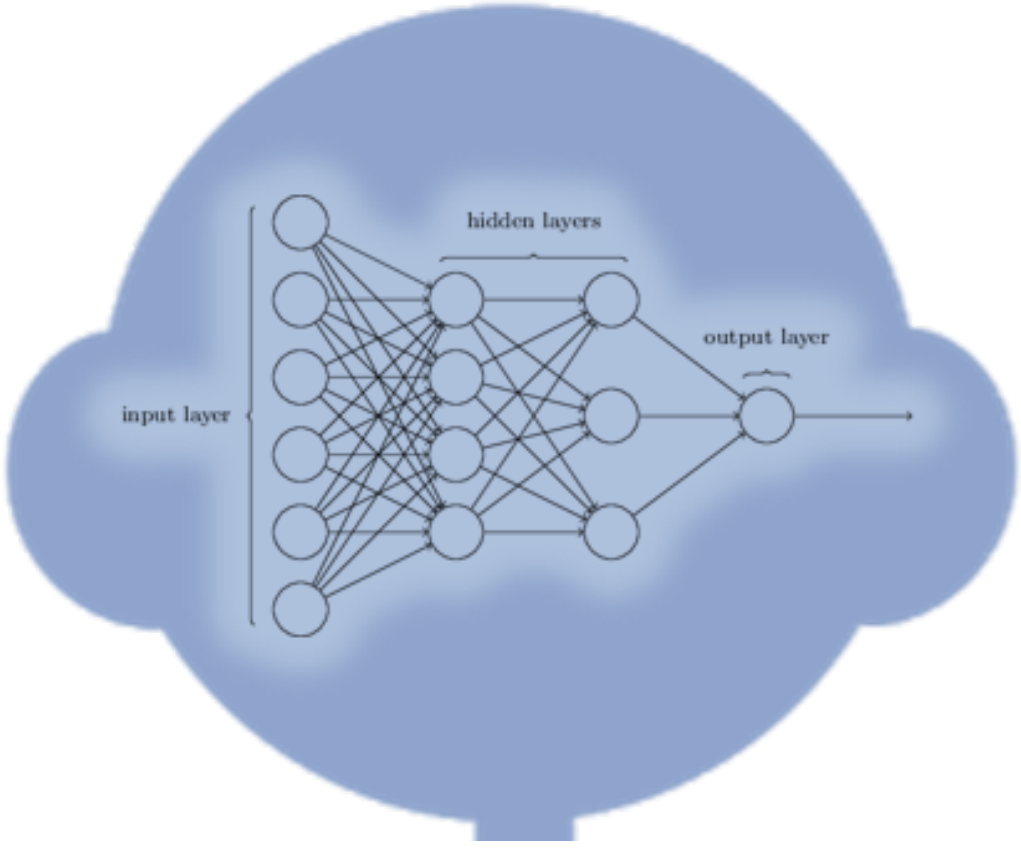
Overall assessment of which faults are diagnosable and recoverable.

Verification of autonomous systems - solved?

There is one **key** assumption in the above.



Neural-Symbolic Agents



Higher performance. Relatively easy/cheap to synthesise.
Paradigm shift. What are the implications of this?

Consequences?

Why - Difficulties with neural networks

- Poor performance on OOD data.
- Often fragile on ID data.
- Often strong confidence on incorrect classifications.
- Lack of explanations.
- ...

Hard to validate before deployment

Inherent risk of poor-performance in several applications leading to non-adoption (or worse).

New methods required

Note that failure of adversarial search does **not** show correctness.

Urgent need of verification methods targetting ML systems.

This would result in benefits towards:

- Safety, security and trustworthiness;
- auditability;
- fairness and accountability.

What - Verifying neural networks: $S \vdash \phi$

Various specifications can be formulated for neural networks (e.g., object detectors).

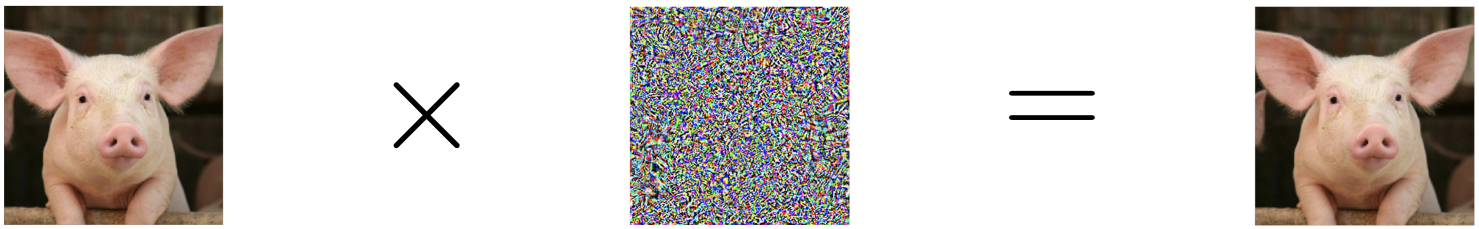
1. **Local** correctness wrt a spec: Given an input i, S is provably correct wrt the spec on i.
2. **Global** correctness wrt a spec: Given an input region I, S is provably correct wrt the spec on all $i \in I$.

Which correctness specifications?

Adversarial Robustness

- CNNs are known to be susceptible to adversarial attacks.
- Slight perturbations of an image (even imperceptible to the human eye) can cause the misclassification of the image.

pig noise airplane



Robustness (folklore notion)

An image classifier is **robust** whenever it assigns the same label to every pair of images that human perception finds indistinguishable.

[Example from http://mpawankumar.info/tutorials/vmcai2019/index.html.]

Local adversarial robustness

Local adversarial robustness is a property of NNs whereby two images are classified in the same way if the distance between them (as measured in some norm L_P) is below a certain threshold.

Definition (Local adversarial robustness)

A NN is *locally adversarially robust* for an input x and $\epsilon > 0$ if it is the case that for all x' s.t. $\|x - x'\|_\infty < \epsilon$ we have $NN(x) = NN(x')$.

Local adversarial robustness can be used to analyse the model's fragility wrt white noise.

General Verification problem

Note that local adversarial robustness is an instance of the general verification problem formulated as:

Verification problem

Given a set of inputs and a set of outputs Y, determine whether

$$\forall x \in X : f(x) \in Y.$$

For example, $X = \{x \mid l_i \leq x_i \leq u_i; l_i, u_i \in \mathbb{R}\}$, where l_i is the lower bound and u_i is the upper bound of component x_i.

The local adversarial robustness problem is an instance of the above where $X = \{x' \mid \|x' - \overline{x}\|_\infty \leq \epsilon\}$ and Y defining the class label.

(What - ctd) Other specifications of interest

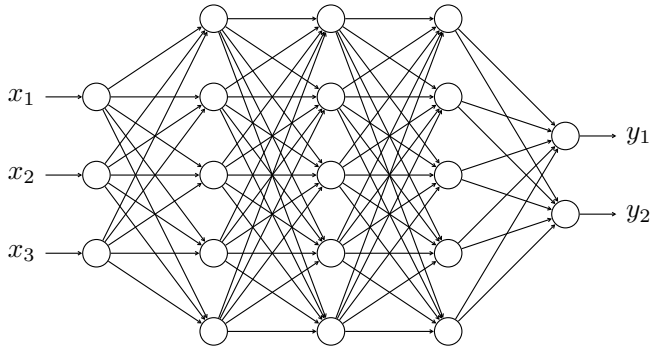
Robustness wrt:

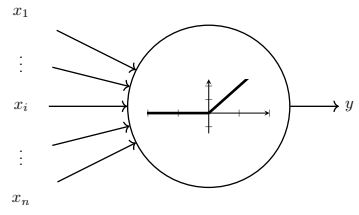
- Not just L_∞ norm, but other norms.
- $X = f(x)$ for some x encoding geometric transformations (translation, scaling, rotations), colour, luminosity, contrast, etc.
- Rescaling governed by more complex functions, e.g., illumination changes.
- Occlusion, 3D pose changes.

More complex functions governing Y or the relation between X and Y.

(How -) Verifying deep ReLU networks

Consider network as computing a function.





$$y = \max(0, \sum_i x_i)$$

Reason in terms of possible outputs when varying inputs.

Difficulties: activation functions, mapping into computationally feasible problems.

Two approaches to verify neural systems

- Complete (MILP-based) verification for ReLU-based NNs.
- Abstraction-based (symbolic interval propagation) verification for ReLU-based NNs.

MILP Encoding for Deep ReLU FF

ReLU activation function

$$\bar{x}_j^{(i)} = \max\left(0, W_j^{(i)}\bar{x}^{(i-1)} + b_j^{(i)}\right),\ j = 1 \cdots |L^{(i)}|$$

- Active phase: $\bar{x}_j^{(i)} = W_j^{(i)}\bar{x}^{(i-1)} + b_j^{(i)}$ (set $\bar{\delta}_j^{(i)} = 0$)
- Inactive phase: $\bar{x}_j^{(i)} = 0$ (set $\bar{\delta}_j^{(i)} = 1$)
- Value of $\bar{\delta}_j$ forces two of the four constraints to become vacuously true, and the other two correspond exactly to inactive/active phase of neuron:

$$\begin{aligned}
\bar{x}_j^{(i)} &\geq W_j^{(i)}\bar{x}^{(i-1)} + b_j^{(i)} \\
\bar{x}_j^{(i)} &\leq W_j^{(i)}\bar{x}^{(i-1)} + b_j^{(i)} + M\bar{\delta}_j^{(i)} \\
\bar{x}_j^{(i)} &\geq 0 \\
\bar{x}_j^{(i)} &\leq M(1 - \bar{\delta}_j^{(i)})
\end{aligned}$$

L., Maganti. An approach to reachability analysis for feed-forward relu neural networks. ArXiV 1706.07351.

Equivalence between MILP encoding and NN computation

Consider the MILP problem P_{NN} defined by $obj_{NN} = 0$ and $C_{NN} = \cup_{i=2}^{m} C_i$, where m is the depth of NN.

Theorem

P is feasible for $\overline{x}^{(1)} = \overline{x}, \overline{x}^{(m)} = \overline{y}$ iff $f_{NN}(\overline{x}) = \overline{y}$.

Robustness verification recast as solving a MILP program.

However, solving MILP problems is NP-Hard.

Approach above enabled the verification of a few hundred nodes.

Scalability?

Dependency-based methods

Branch and bound is the leading approach for solving MILP programs

Branch-and-bound procedure

1. Relax all integrality constraints to linear ones.
2. Solve the resulting linear program.
3. If all integrality constraints are satisfied then terminate.
4. Otherwise, branch on an integral variable and repeat for each resulting sub-problem.

Branching causes exponential blow-up.

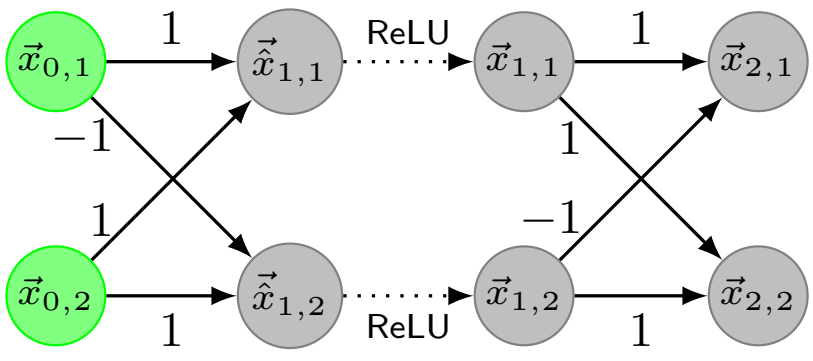
Botoeva, Kouvaros, Kronqvist, L., Misener. Efficient Verification of Neural Networks via Dependency Analysis. Proceedings of the AAAI20.

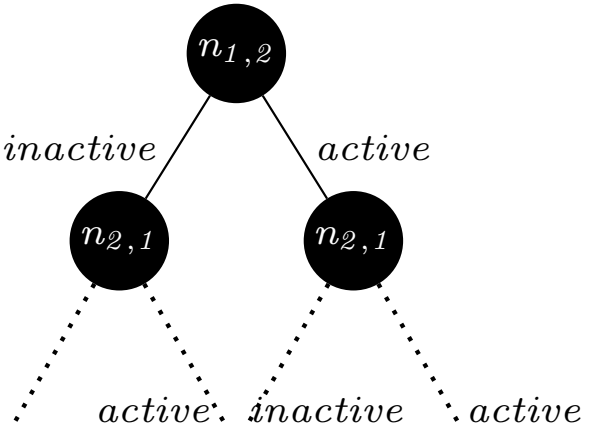
Dependency analysis

Aims at reducing the search space during branch-and-bound.

Definition (Dependency between ReLU nodes)

An unstable (i.e., neither active nor inactive) node $n_{i,q}$ depends on an unstable node $n_{j,r}$ if whenever $n_{j,r}$ is stable, $n_{i,q}$ is stable as well.

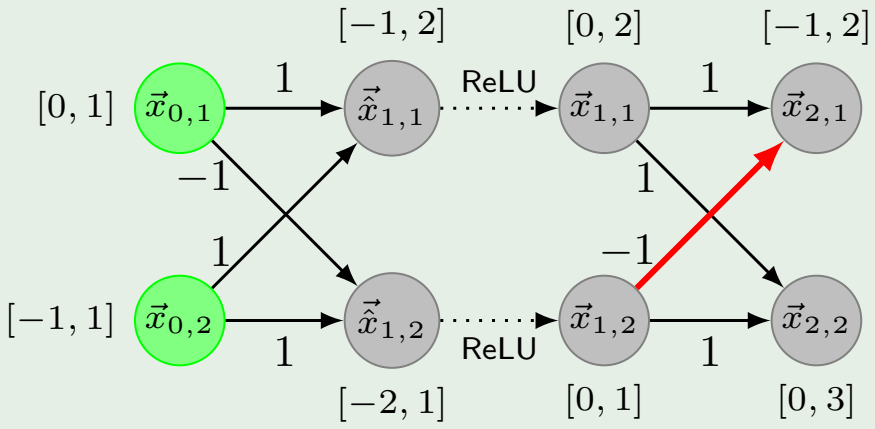


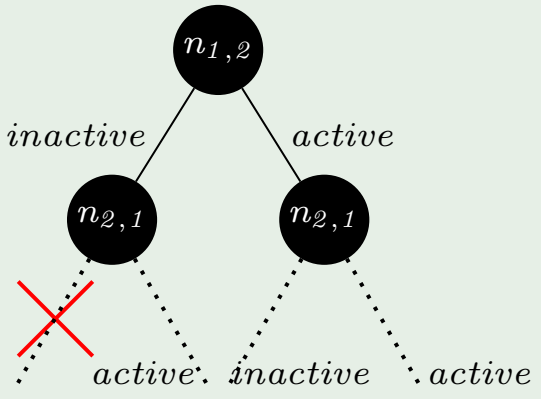


Dependency analysis

Example (Inactive-active dependency)

If node $n_{1,2}$ becomes inactive, then node $n_{2,1}$ has to be active; $n_{2,1}$ **depends on** $n_{1,2}$.





This gives raise to a number of possible dependencies.

Dependency analysis procedure

Dependency analysis procedure

1. Stop the branch-and-bound procedure at runtime.
2. Compute the pre-activation bounds.
3. Compute the dependencies.
4. Express each dependency as an MILP constraint.
5. Add the constraint to the MILP program being solved.

Dependency-based branching

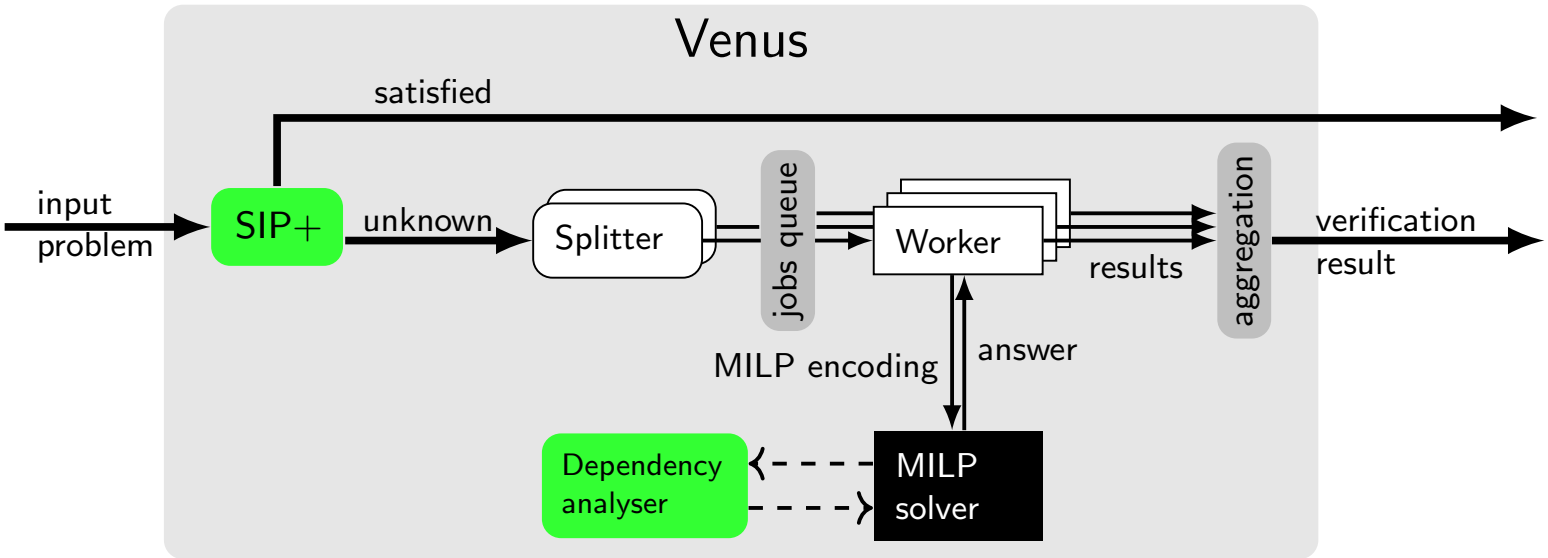
Dependencies are also used as a heuristic to divide the verification problem into sub-problems with simpler MILP-formulations (i.e., with fewer binary variables).

Dependency-based Branching

1. Identify the node that has the most dependent nodes.
2. Divide the verification problem into two sub-problems:
   - one where the selected node is stabilised into the active state
   - and one where it is stabilised into the inactive state.
3. Repeat 1. and 2. until the required *branching depth* is reached.
4. Solve the resulting verification sub-problems in parallel.

Kouvaros, L.. Towards Scalable Complete Verification of ReLU Neural Networks via Dependency-based Branching. Proceedings of IJCAI21.

Venus

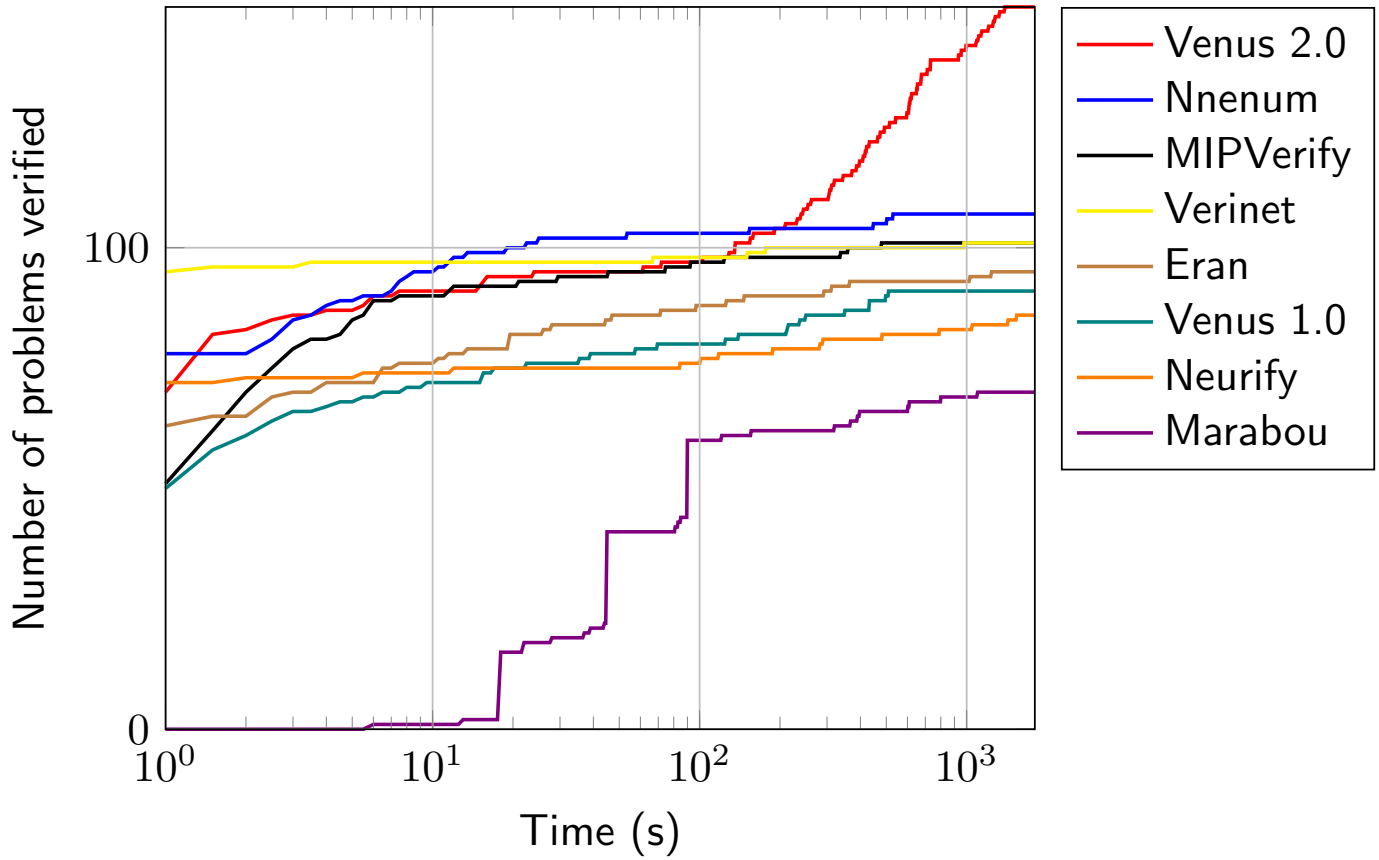


- Venus is a SoA MILP-based verification tool.
- Its performance relies on a combination of novelties in dependency analysis, input domain splitting, symbolic bound propagation.

---

https://github.com/vas-group-imperial/venus

Venus — Comparison with the SoA



Results from VNN20 w.r.t the fully connected networks.

VeriNet: A complete abstraction verification method

Henriksen, L.. Efficient Neural Network Verification via Adaptive Refinement and Adversarial Search. Proceedings of ECAI20.
Henriksen, L. DEEPSPLIT: An Efficient Splitting Method for Neural Network Verification via Indirect Effect Analysis. Proceedings of IJCAI21.

VeriNet-Properties

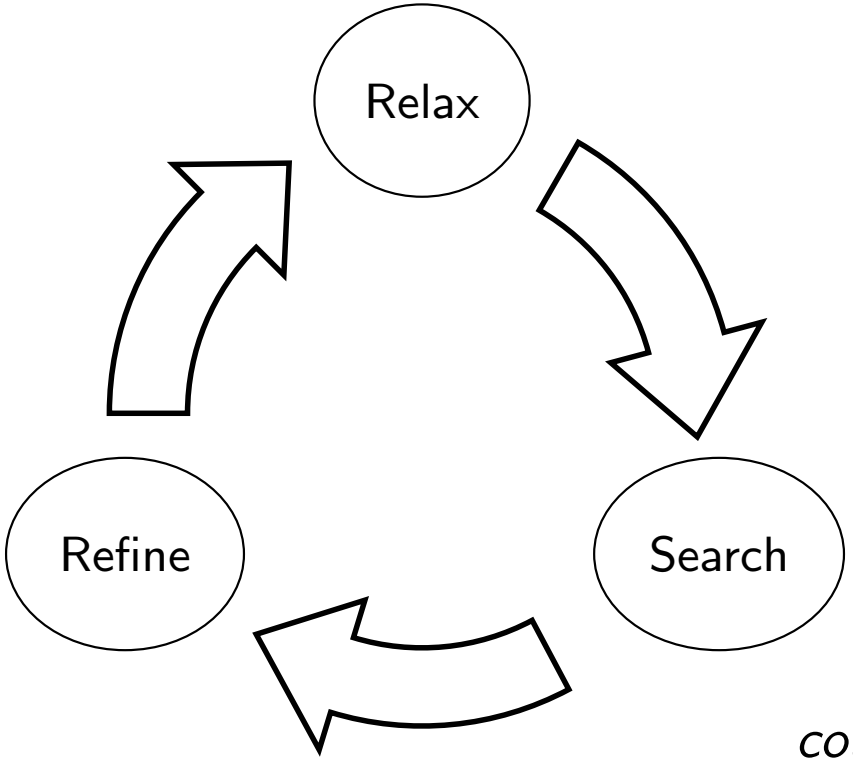
Supported Verification Problems

- Input-sets defined via box-constraint (including $l_\infty < \epsilon$).
- Supported layers: Fully connected, Convolutional 2D, Batch Normalization, Average Pooling, Max Pooling, Reshape, Transpose, Add, multiply by constant, Mean, Crop.
- Activations: Feed-Forward ReLU, sigmoid and tanh.

VeriNet computes an abstraction of the model and determines whether this abstraction meets the specification constraints for the output.

Verification Loop

Linear relaxation with Symbolic Interval Propagation

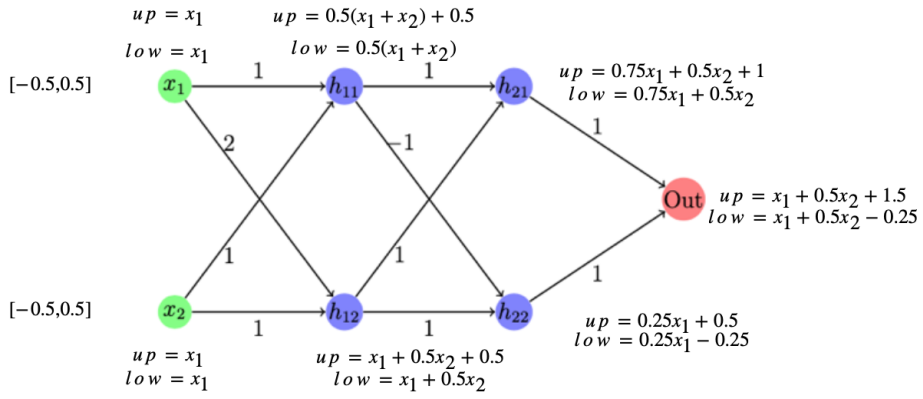


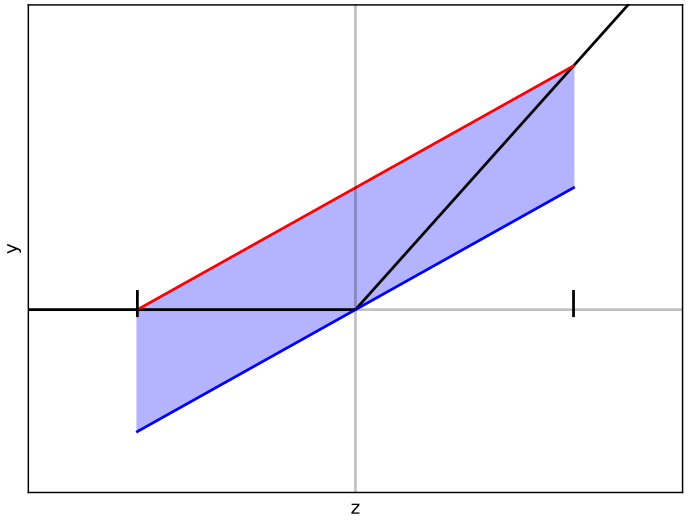
Refinement via branching

Search for counterexamples

Symbolic Interval Propagation (SIP)

- Linear lower and upper bounds are propagated through the network.
- Non-linearities are linearly relaxed.
- The resulting bounds at the network's output are passed to the verification phase.





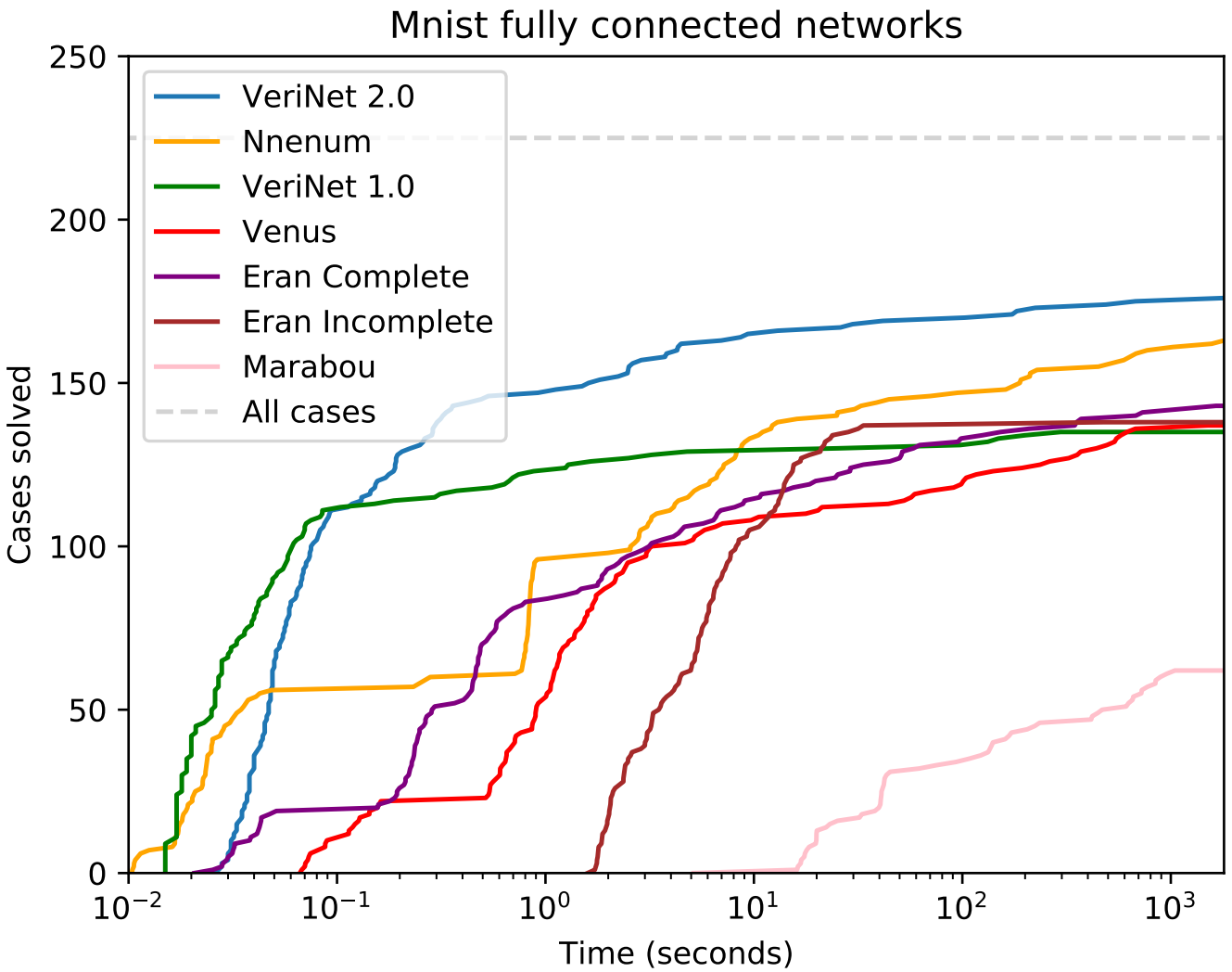
Abstract verification via LP

- Verification problem on **relaxed (abstract) network** is passed to LP solver.
- If no counterexample is found, the network satisfies the specification.
- If a counterexample is found, we test its validity. If spurious we launch a gradient descent search to find a valid one.

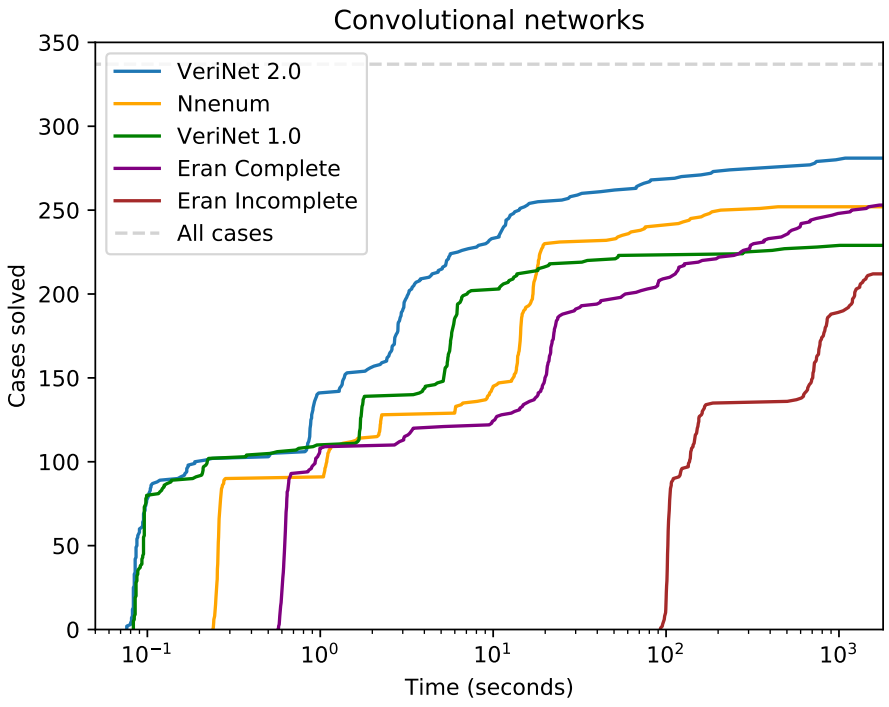
Automated refinement

- If no conclusion can be drawn, we refine the abstraction by splitting appropriate ReLU nodes.
- Nodes to be split derived by accounting for direct and indirect effects of splitting on the formulation.

Experimental Results – MNIST FC



Experimental Results – CIFAR Convolutional



Experimental Results – CIFAR Convolutional

Network	Size	Layers	Defence	ϵ
MNIST	512	$2 \times FC$	None	0.05
MNIST	1024	$4 \times FC$	None	0.03
MNIST	1536	$6 \times FC$	None	0.02
CIFAR10	17k	$2 \times Conv + 2 \times FC$	PGD	$8/255$
CIFAR10	50k	$3 \times Conv + 2 \times FC$	PGD	$\approx 2/255$
CIFAR10	56k	$7 \times Conv$	None	0.002
CIFAR10	110k	$6 \times Conv$	None	≈ 0.002

Table: Reported epsilons are the largest among the tested perturbations for which at least $1/3$ of the test-cases where verified as robust.

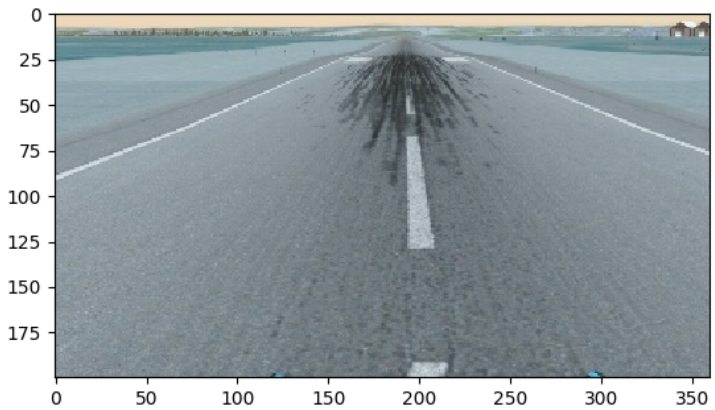
Verinet was classified 2nd in VNN-COMP21, the annual international competition for NN verifiers.

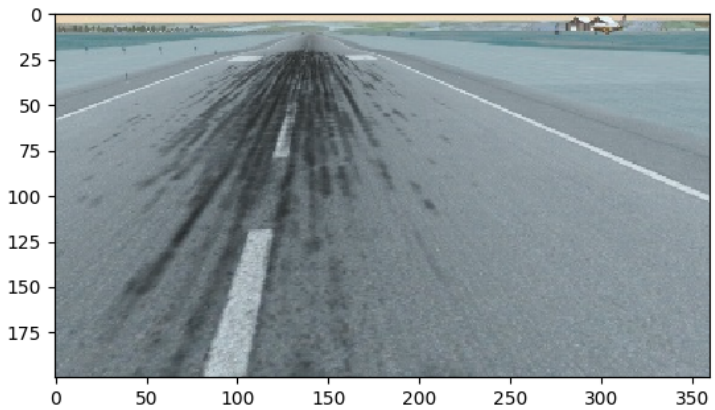
https://sites.google.com/view/vnn2021/home

Which - Usecase 1: Verifying a distance estimator

Taxinet

- Neural network to estimate cross track error of plane taxiing on the runway
  - **in**: snapshot of the runway
  - **out**: cross track error (offset from centre line)
  - ∼10K ReLU nodes

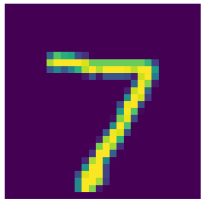




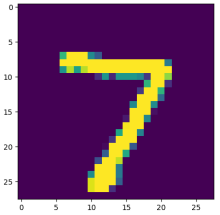
Kouvaros, Kyono, Leofante, L., Margineantu, Osipychev, Zheng. Formal Analysis of Neural Network-based Systems in the Aircraft Domain. Proceedings of FM21.

Specification

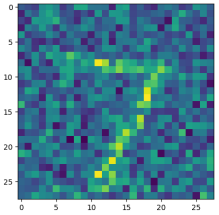
- **Objective**: assess the robustness of the model against variable environmental conditions.
- **Specification**: determine whether the cross-track error for a given image is always less than 1.5 meters under all transformations.
  - (**local robustness**)
- Examples of transformations we considered:



Original



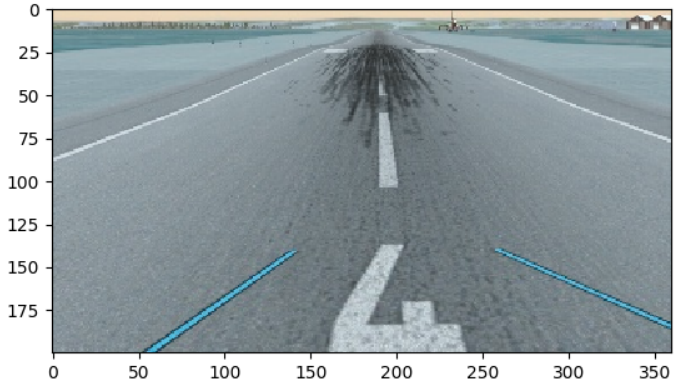
Photometric



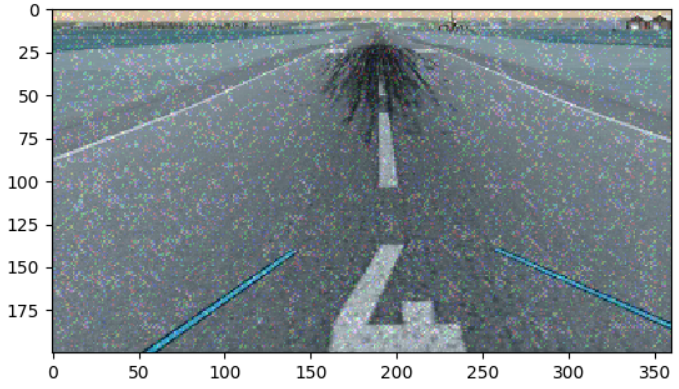
Noise

Experimental results with Venus

VENUS discovered several unwanted features of the models.



Original (0m track error)



Counterexample (1.6m track error)

Experiment above against noise; similar results can be obtained for contrast, luminosity, etc.

Note: VENUS is complete, in principle all counterexamples can be found.

Usecase 2: Open Category Detection

- Neural object detectors used to assist pilots during landing
- Novel objects may appear during deployment, such as novel types of ground vehicles.
- **Goal**: detect these novel objects with a small number of false alarms

Vehicle

Person

Airplane

Robustness wrt perturbations with VENUS

Given an individual image with label "Vehicle"

- Noise transformation: each pixel may be perturbed up to $\pm\epsilon = 1.0 \times 10^{-4}$
- Is there any counterexample within such a perturbation?

  Answer: No. The network always reports "Vehicle" with confidence, *i.e.*, the following property is true, verified by VENUS

$$\phi(y) > 0, \forall y = f(\boldsymbol{x}) \text{ s.t. } \|\boldsymbol{x} - \boldsymbol{x_0}\|_\infty \leq \varepsilon$$

Local robustness verification

Randomly selected 20 images

Perturbation	Robust		Non-robust		Time-out (7200 s)
	Number	Time (s)	Number	Time (s)	
1.0×10^{-4}	20	2500	0	—	0
1.0×10^{-3}	6	2720	2	4850	12
1.0×10^{-2}	0	—	10	4200	10

- Provable robustness to $\epsilon = 1.0 \times 10^{-4}$
- The model is not robust with respect to perturbation $\epsilon = 1.0 \times 10^{-2}$
- VENUS can also identify adversarial images that violate the specification

Local robustness - Counterexamples

CLICK HERE TO LAUNCH DEMO

Usecase 3: Landing Advisory System

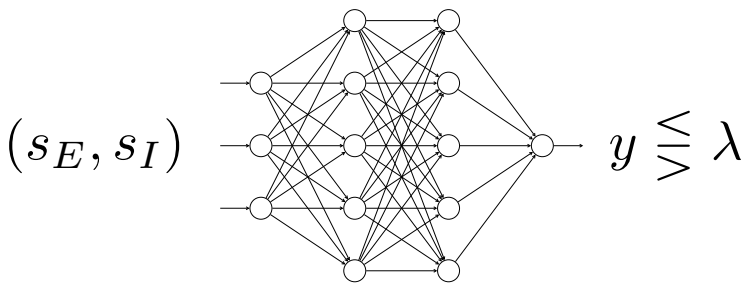
$s_E = [x_E, y_E, \ldots]$

$s_I = [x_I, y_I, \ldots]$

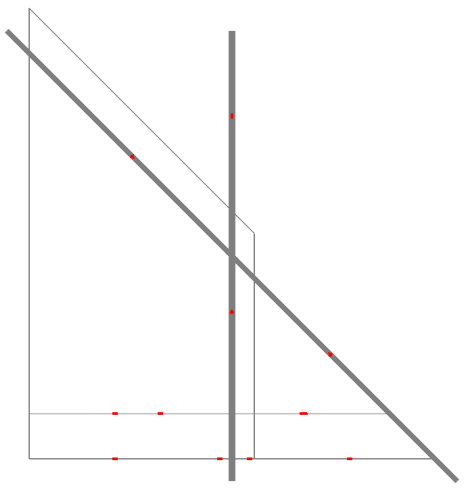
A Binary Classifier for Landing/Go-Around

The model advises whether to **land** or **reject**

Neural network

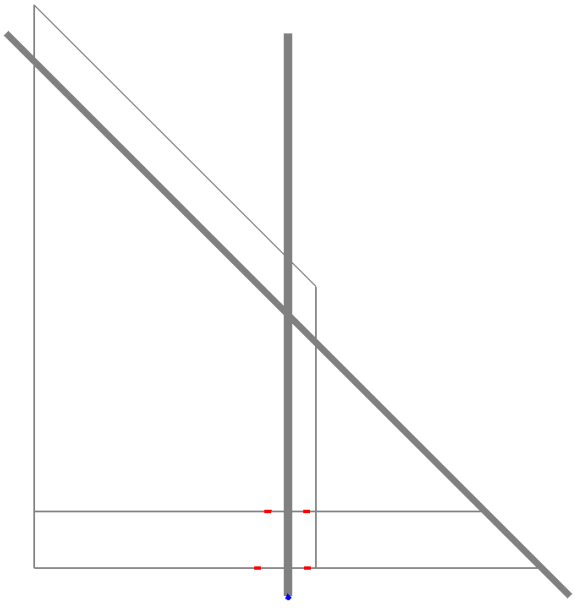


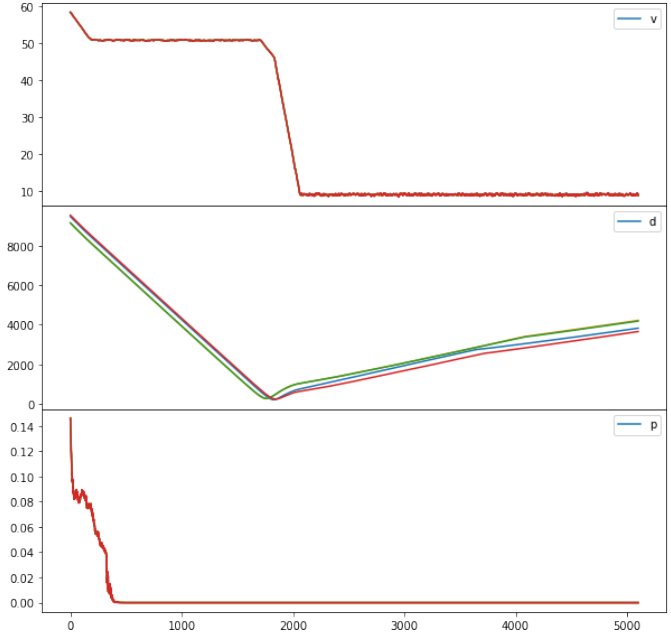
Simulator



- **Specification**: **reject** if $y \geq \lambda$
- Small model of 128 RELU nodes

Single analysis of robustness/fragility of landing decisions





Specification

Suppose we have a classification result which is **land**.

Is there an instance obtained via a noise transformation of $\pm\varepsilon$ for which the model advises to **reject**?

Multi-instance analysis of landing and reject decisions

We used the simulator to sample **land** and **reject** configurations and analysed their robustness.

Land (50 instances) $\rightarrow$

Perturbation	Robust	Average time (s)	Not robust	Average time (s)
0.05	50	0.09	0	-
0.1	50	0.09	0	-
0.2	50	0.08	0	-
0.5	50	0.08	0	-
1	50	0.08	0	-
1.5	50	0.07	0	-

Reject (50 instances) $\rightarrow$

Perturbation	Robust	Average time (s)	Not robust	Average time (s)
0.05	48	0.11	2	0.33
0.1	48	0.10	2	0.38
0.2	48	0.10	2	0.39
0.5	48	0.11	2	0.39
1	47	0.10	3	0.33
1.5	45	0.11	5	0.37

Following our sampling we observed that:

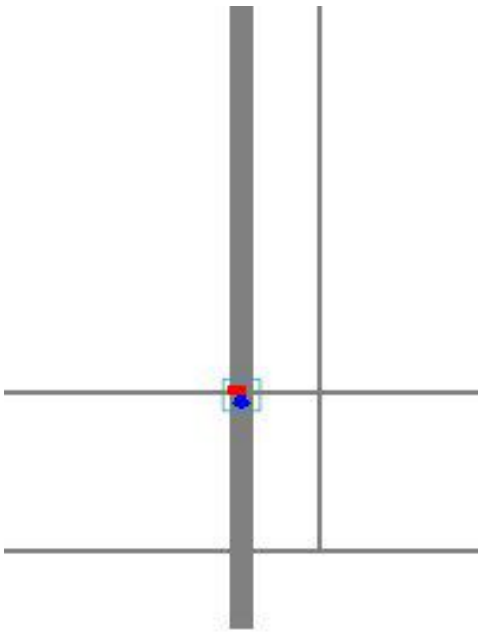
- **land**: decisions are robust irrespective of ε.
- **reject**: decisions less robust for increasing ε.

Towards realistic physical perturbations

- So far we considered perturbations that simulate a **worst-case** scenario where all sensors are subject to failures (L_∞ norm).
- More **realistic** failures involve only some sensor readings.
- VENUS supports such analysis.

Analysing the impact of corrupted localisation data

Given a possible **reject** region
Can we find an instance where the model reports **land**?



$$
\begin{aligned}
x_E &= -0.926780382551181 \\
y_E &\in [-1390, -1410] \\
\theta_E &= 2.83338226678984 \\
v_E &= 41.792804303235 \\
x_I &\in [-10, 10] \\
y_I &\in [-1390, -1410] \\
\theta_I &= 89.9449216220387 \\
v_I &= 0.0834243931549194 \\
&\implies \mathcal{N}(x) \geq 0.75
\end{aligned}
$$

In the experiments we were able to find such problematic instances.

Sensitivity wrt single input dimensions

Verification can also be used to understand whether predictions are sensitive to perturbations on a **single input dimension**.

- perturbations applied to one input component at a time
- models appear to be fragile wrt perturbations on velocity inputs:

Overflying (50 instances) with $\varepsilon = 1.5 \rightarrow$

Parameters	Not Robust	Avg time (s)
x_E	1	0.25
y_E	1	0.26
θ_E	1	0.26
v_E	1	0.25
x_I	1	0.24
y_I	1	0.24
θ_I	1	0.26
v_I	3	0.29

Usecases - Conclusions

Provided robustness guarantees against portions of the test set and identified fragilities.

- Small models (1k nodes) can be verified in seconds for large perturbations also on high-dimensional inputs.
- Larger models (100k nodes) can be verified in 2hrs on a deskptop machine for white noise perturbations of (10^{-4}).
- Rich analysis: not just noise, but various parameters.

Why – Validation

- Verification as a method for validation and certification before deployment.
- Identification of weaknesses in the model before deployment.
- As part of certification effort before deployment.
- Compliance with regulations (in some instances).

Why – Increased performance

- Verification-based augmentation. Build training pipeline by adding verification-based counter-examples.
- Verification-based repair. Use counterexamples to minimally patch models to accommodate counterexamples, without severely impacting correct classifications.

Move from plain accuracy to a multiple criteria including robustness, reliability, explainability.

When – Can I verify my Edge ML system right now?

Key parameters: system (model) size, input size, and specification.

- Present usecases show models up to 100k ReLU nodes can be verified against non-trivial noise.
- Larger models can be verified against less intensive perturbations (e.g., hue).
- Some perturbations are harder to verify (e.g., rotations).

Given the typical model size, it is likely that present/forthcoming methods may be applicable to Edge and TinyML systems.

Fast progress - Ongoing work

- White noise expensive due to high-dimensionality of the complexity of the resulting problem. Tighter relaxations, eg SDP.
- Not just white noise but geometrical transforms (translation, rotation, scaling, etc.), hue, luminosity, contrast changes. Models of 10^6 neurons (10^7 tuneable parameters can be analysed in ongoing work).
- Explainability: Verification can be used to validate and generate robust explanations (semifactuals and counterfactuals).
- Tabular data: robustness assessment on particular dimensions or combinations of them.
- Verification of (Closed-loop) Neural-symbolic multi-agent systems.

Batten, Kouvaros, L., Zheng. Efficient Neural Network Verification via Layer-based Semidefinite Relaxations and Linear Cuts. Proceedings of IJCAI21.

Selected References

- Kouvaros, Kyono, Leofante, Lomuscio, Margineantu, Osipychev, Zheng. Formal Analysis of Neural Network-based Systems in the Aircraft Domain. FM21.
- Henriksen, Lomuscio. DEEPSPLIT: An Efficient Splitting Method for Neural Network Verification via Indirect Effect Analysis. IJCAI21.
- Kouvaros, Lomuscio. Towards Scalable Complete Verification of ReLU Neural Networks via Dependency-based Branching. IJCAI21.
- Batten, Kouvaros, Lomuscio, Zheng. Efficient Neural Network Verification via Layer-based Semidefinite Relaxations and Linear Cuts. IJCAI21.
- M. Akintunde, E. Botoeva, P. Kouvaros, A. Lomuscio. Verifying Strategic Abilities of Neural-symbolic Multi-agent Systems. KR20.
- Henriksen, Lomuscio. Efficient Neural Network Verification via Adaptive Refinement and Adversarial Search. ECAI20.
- Akintunde, Botoeva, Kouvaros, Lomuscio. Formal Verification of Neural Agents in non-Deterministic Environments. Proceedings of AAMAS20.
- E. Botoeva, P. Kouvaros, J. Kronqvist, A. Lomuscio, R. Misener. Efficient Verification of Neural Networks via Dependency Analysis. AAAI20.
- Akintunde, Kevorchian, Lomuscio, Pirovano. Verification of RNN-Based Neural Agent-Environment Systems. AAAI19.
- Akintunde, Lomuscio, Maganti, Pirovano. Reachability Analysis for Neural Agent-Environment Systems. KR18.

Would like to know more?

Key messages:

- Verification will be an enabler for trust, accountability, fairness and auditability in all AI applications, including Edge and TinyML.
- Benefits include validation, understanding, and higher performance.
- Can contribute for transition from pilot study to deployment.

Edge ML and Tiny ML could be one of the initial beneficiaries both in terms of validation and increased performance/accuracy.

Comments, questions, problems, feasibility studies? Get in touch!

Get in touch!

http://vas.doc.ic.ac.uk/ a.lomuscio@safeai.co.uk

Copyright Notice

www.tinyml.org

Copyright Notice

www.tinyML.org