Towards Certification of a Reduced Footprint ACAS-Xu System: a Hybrid ML-based Solution

Mathieu Damour, Florence de Grancey, Christophe Gabreau, Adrien Gauffriau, Jean-Brice Ginestet, Alexandre Hervieu, Thomas Huraux, Claire Pagetti, Ludovic Ponsolle, Arthur Clavière

Towards Certification of a Reduced Footprint ACAS-Xu System: a Hybrid ML-based Solution

Mathieu Damour[1,2], Florence De Grancey[3,2], Christophe Gabreau[4,2], Adrien Gauffriau[4,2], Jean-Brice Ginestet[5], Alexandre Hervieu[5], Thomas Huraux[1,2], Claire Pagetti[6], Ludovic Ponsolle[7,2], Arthur Clavière[8]

[1]*Scalian,* [2] *IRT Saint Exupéry,* [3]*THALES,* [4]*Airbus,* [5]*DGA,* [6]*ONERA,* [7]*Apsys,* [8]*Collins Aerospace*

September 27, 2021

Abstract

Approximating while compressing lookup tables (LUT) with a set of neural networks (NN) is an emerging trend in safety critical systems, such as control/command or navigation systems. Recently, as an example, many research papers have focused on the ACAS Xu LUT compression. In this work, we explore how to make such a compression while preserving the system safety and offering adequate means of certification.

1 Introduction

Due to the intensive flights traffic, the risk of collision is increasing. During the last decade, a standardization group has defined a new competitive and effective anti-collision system named ACAS X (for *Next-Generation Airborne Collision Avoidance System*) [16]. The purpose is to keep any intruder outside of the desired envelope of the ownship.

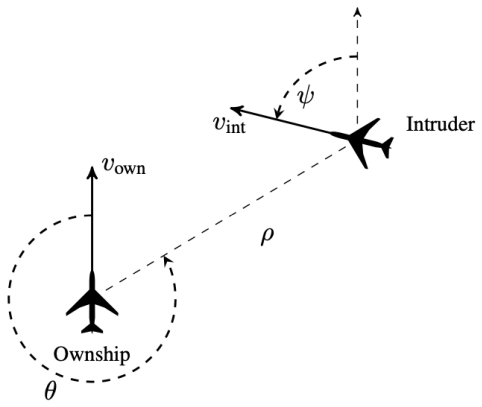

Figure 1: ACAS Xu geometry [14]

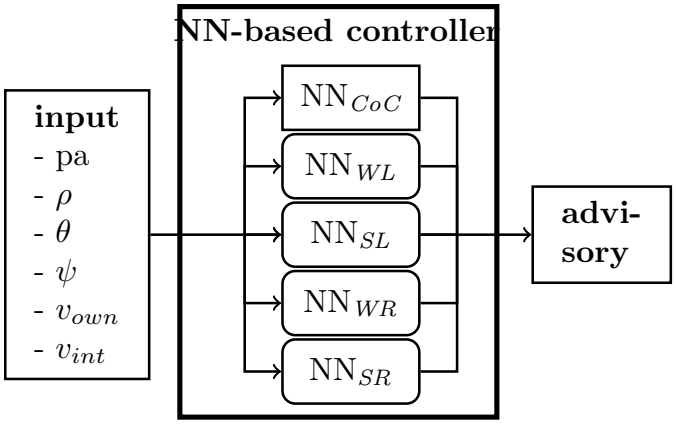

Figure 2: NN-based architecture [14]

1.1 ACAS Xu Overview

Among the family of ACAS X, we will focus on the ACAS Xu [10] dedicated to drone, Urban Air Mobility and Air Taxi with horizontal automatic resolution. The system is based on a set of lookup tables (LUT) that are used in real-time to resolve conflicts. Those LUT have been computed off-line and their size has been chosen in order to fulfil real-time (decisions must be taken every second) and safety level (there should not be any collision) requirements. The ownship computes six parameters (listed below) that enable to access the tables which give an estimation of the probability to have a collision for the each possible advisory and the chosen advisory is the one that minimizes this probability. The geometry of the system is given in figure 1, and the definition of the parameters stands as:

- ρ (ft): Distance from ownship to intruder
- θ (rad): Angle to intruder relative to ownship heading
- ψ (rad): Heading angle of intruder relative to ownship heading direction
- v_{own} (ft/s): Speed of ownship
- v_{int} (ft/s): Speed of intruder
- τ (s): Time until loss of vertical separation.

The 23 LUT provide the transitions costs between the previous advisory (pa) and the next advisory. There are five advisories: COC (Clear Of Conflict); SR (Strong Right); SL (Strong Left); WR (Weak Right) and WL (Weak Left). In particular, when the ownship is in the COC state, it can continue its mission. When the ownship is in one of the other states, it has to initiate a turn with a rate that depends of the computed state. In practice, a single table is composed of 2 sub-tables: the first contains definition of parameters values and the second contains the costs that are half-integer (16 bits). More information on the ACAS Xu system can be found in [19].

1.2 Purpose of the Work

Several universities have worked on replacing the LUT by neural networks (NN), the objective being to reduce the size of the embedded code and improve the anti-collision performance. The authors of [14] have replaced the LUT by 45 neural networks leading to an impressive reduction of the memory footprint (4GB to 150 MB), see the figure 2 for horizontal advisory (i.e. when $\tau = 0$). In this work, we want to explore how to compress the LUT with neural networks while preserving the system safety and offering adequate means of certification. We only focus in the sequel on the horizontal resolution of conflict.

Certification Problem Statement. For any safety-critical system embedded in an aircraft, airframers (applicants) have to demonstrate to regulation authorities that their product is compliant with certification specifications. To this purpose, applicants use a set of standards that are recognized as acceptable means of compliance. Existing development assurance standards are not adapted to the data-driven paradigm of the ML technique, though such development may introduce errors that could jeopardize a safe operational use of the system (e.g. such probabilistic approach may introduce unforeseen errors). Currently a joint working group, the EUROCAE WG-114/SAE G-34 [9] (WG-114 for short in the sequel) is preparing the next standard to fill this gap.
Contributions. The use of NNs to approximate the LUT may lead to unexpected behaviors that should be mitigated to guarantee that the ML-based item will not alter the safety of the system. For this purpose, the ACAS Xu subsystem is designed as an hybrid controller: a non ML item is introduced to guarantee the safety in all the operational domain (safety net).

In addition, a new certification strategy is investigated to provide sufficient guaranties to authorities. The figure 3 instantiates the WG-114 development process workflow to the ACAS Xu use case. There are three levels of engineering: the *System and Subsystem Level* as proposed by ARP4754A [25] standard which provides guidance for the system development process and are complementary to the product requirements (WG 75-1 [10] for the ACAS Xu technical requirements); the *item Level* where ML activities are not covered by any known standard while the non-ML parts are supported by classical guidance for the implementation process of the software items (DO-178C [8]) and hardware items (DO-254 [23]).

To tackle the objectives of the system development guidance, three aspects were developed. The first concerns the learning assurance activity of the ML element (or MLM-Machine Learning Models), which aims at ensuring that the MLM requirements (covering functional, performance and robustness aspects) have been captured and correctly designed. The elements supporting this activity are detailed in section 2. The second aspect (dashed green arrow) targets the item validation activity in order to check that captured requirements fit the system needs. The third aspect (solid green arrow) targets the subsystem verification to check that the ACAS-Xu Hybrid controller safely performs its intended use. Activities and relationships covering the two first aspects are part of a constructed argumentation (see section 3).

2 ACAS Xu Hybrid Architecture

We propose to replace the LUT with a hybrid architecture (shown in figure 4) composed of a neural networks based controller part (as proposed by [14]) together with a safety net to ensure a correct behavior. The idea, that will be detailed hereafter, is to 1) apply the best practices of learning assurance to well approximate the tables; 2) identify off-line the zones where the NN-based system differs from the LUT advisories and where it may jeopardize the safe behavior of the system; 3) compute on-line in which zone the system is, call the NN-based system if it behaves similarly to the LUT or switch to the safety net if not in order to always be safe (this step is performed by *check module*). The safety net consists of the extract of the LUT for these zones.

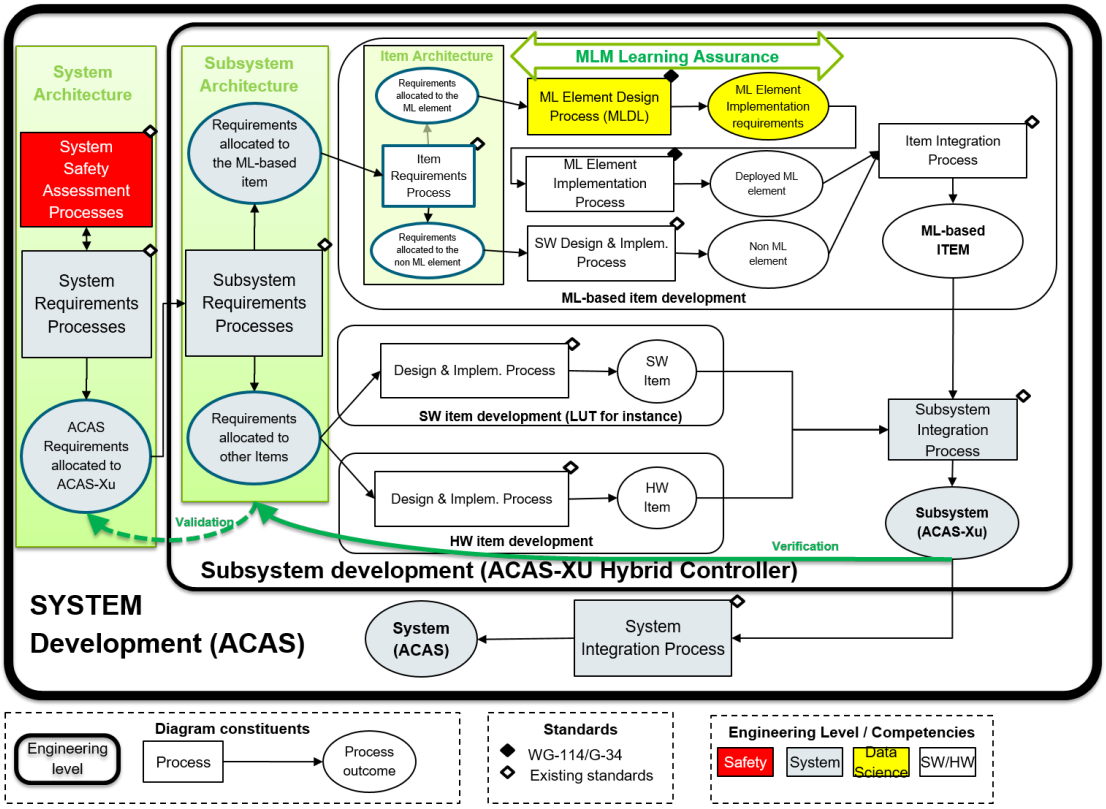

Figure 3: ACAS Xu subsystem development workflow using ML

2.1 Learning Process

The objective of the learning process is to build a model for advisory computation which performs a trade-off between reducing the memory footprint and preserving fidelity to LUT. Ideally, this model should reconstruct the original (but unknown) cost function. NNs are quite good universal approximators as long as the cost function has similar bounded derivative on the whole domain which is unfortunately not the case here. The cost function shows two local offsets where derivative reaches high value: one (figure 5) is observed for each cost function when the range is below 5000 ft, where the cost functions switches quickly from 0 to 15000, and the second (figure 6) concerns the case of COC advisory cost value when previous advisory is not COC, where a local offset of +4000 is added. This issue could be resolved by either augmenting the size of the NNs (not explored here) or finding the most suitable NN architecture.

We explored several architectures (with ReLu activation only) with the same learning process where 1) input and output data are normalized between -1 and 1 (as suggested in [17]), 2) batches size is set to 8912, 3) Adam optimizer [5] was used, 3) initial learning rate is set to 0.002:

1. regression (cost inference) versus classification (decision inference);
2. regular hidden layer size versus decreasing layer size. We have tested *regular50* with 8 layers and 5-50-50-50-50-50-50-5 neurons per layer; *decreasing128* as 5-128-64-32-16-5 and *decreasing256* as 5-256-128-64-32-16-5.

The criteria for evaluation, that we called *accuracy*, is the agreement rate between advisory computed by the NN and the ground truth LUT value. Both training and evaluation were performed in the whole data-set which is not classical but the point is to be as close as possible to the LUT. Thus, for once overfitting is encouraged to improve accuracy.

Training type	**Network shape**	**Accuracy**	**Accuracy mid range**	**Accuracy short range**	**Nparam**
Regression	[14]	93,22	82,424	68,25	13305
Regression	regular50	95,42	87,44	71,52	15855
Regression	decreasing128	95,81	89,18	75,62	28229
Regression	decreasing256	96,33	90,82	79,74	111173
Classification	decreasing256	76,06	76,08	86,43	111173

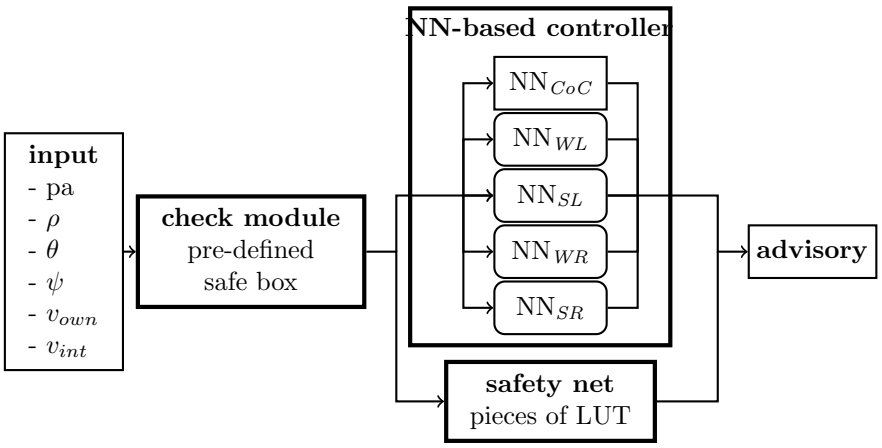

Figure 4: Architecture of the neural network based ACAS Xu

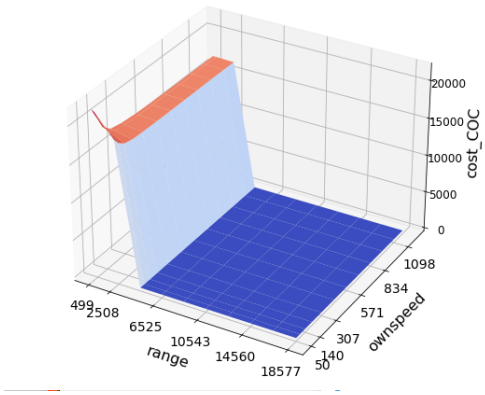

Figure 5: Cost function CoC → CoC

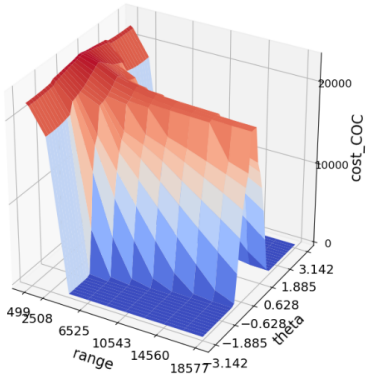

Figure 6: Cost function WL → CoC

We completed the evaluation with accuracy measurement on range restricted subsets which represent the most critical situations. At long-range, as there is no risk of collision and the advisory is most of the CoC (95 % of advisory for ranges above 10000 ft). For ranges below 5000ft, the advisory is spread with 15% CoC, 38% SL and 36% SR. We set empirically two subsets: a mid-range subset with range below 20000 ft, and a short range subset with ranges below 500ft. The table above shows the measured accuracy.

Overall, as expected we observe that accuracy decreases when the range is reduced. Furthermore, at very short ranges, we observe in LUT that the cost difference between decision is very low, then it is more difficult for neural network to infer the exact cost. We observe that the regression approach is more effective than classification. This result could be explained by the fact that the decision boundary have "square" shape, not suitable for shallow neural networks. We also observe that the decreasing architecture performs better than the regular architecture. We can suppose that this decreasing architecture favours representation of more complex functions in the first layers. Such complex representation is more suitable to represent offset effects. After this study we have selected regular50 and decreasing128 since they reaches high accuracy with limited memory (parameters) footprint.

2.2 Design of the Hybrid Architecture

The objective of the safety net is to take over when the NN does not take similar advisory as the LUT in the same situation. To determine the zones where the NN differ from the LUT, we use formal verification techniques. More precisely, we decompose the space as a set of p-dimensional boxes (short as p-box). Definition[p-box] Let $p \in \mathbb{N}$, a p-dimensional box $[b]^p$ is a set of $\mathbb{R}^p$ defined as the cartesian product of p intervals:

$$[b]^p = \underset{1 \leq i \leq p}{\times} [l_i, u_i]$$

wherein $l_i \in \mathbb{R}$ (resp $u_i \in \mathbb{R}$) is the lower bound (resp the upper bound) of the i^{th} interval composing the box $[b]^p$. The boundaries of the p-boxes come from the parameters values of the LUT. In practice, the input state space is split in $36,784.10^6$ 5-boxes. For each box, we compute the possible decisions obtained either from the

LUT and the NN, and we check that they are *similar*, which is formalized in the property below. Property[NN-based architecture compliant with specification] We define by *decisions* $f(l) \subseteq \{CoC, WL, SL, WR, SR\}$ the set of reachable advisories by f from any point of l where $f \in \{NN, LUT\}$ and l is a p-box. We consider that a NN behaves similarly to the LUT on an p-box l if

$$decisions\ NN(l) \subseteq decisions\ LUT(l)$$

To determine which p-box satisfies the property 2.2, we use the verification tools DEEPPOLY [28], RELUPLEX [14] and PLANET [7]. First the property 2.2 is checked on the p-boxes with DEEPPOLY. As DEEPPOLY computes an over-approximation, thus either DEEPPOLY provides a positive answer (i.e. property holds) or an unknown answer. Then RELUPLEX/PLANET is called on the remaining boxes (those for which DEEPPOLY provided unknown).

In this work, the safety net is designed with the onwship and intruder having a constant speed of 438 ft/s and 414 ft/s respectively. This corresponds most of the time to the worst case situation, i.e. if an aircraft flights slower, decisions are similar. This improves the compression of the hybrid architecture and speeds up the safety net design. In effect, we would need to design the safety so that it covers all situations. Because of our hypotheses ($\tau = 0$ for horizontal resolution and constant speed), the space is split in 304 000 3-boxes and thus one 3-box is defined by (ρ, θ, ψ). The results are given in the table below.

Method	DEEPPOLY		RELUPLEX/PLANET	number of
	time	success	time	failed boxes
Regression [14]	14min	77.2%	63h	24057
regular50	15 min	78.7%	48h	6912
decreasing128	16 min	78.2%	56h	1664

We plot (in figure 7) for each 3-box (ρ, θ, ψ) the lower bound l_i with on the left a color indicating the solver used (RELUPLEX/PLANET or DEEPPOLY) and on the right a color indicating the advisory. More precisely, right plot of the figure shows advisory for an intruder aircraft located at each point on the plot, whose coordinates indicate the slant range (ρ) and angle to intruder (θ) and with the own-ship located at the center of the plot. Since 87% of LUT(l) decisions are unique, it entails that the NNs take exactly the same decision as the LUT most of the time.

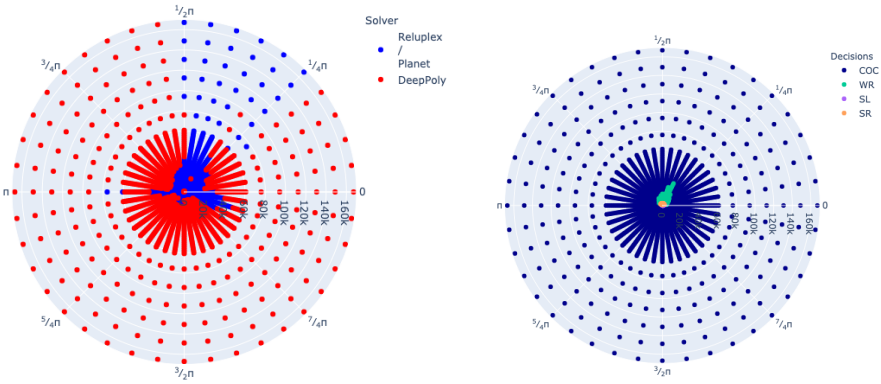

Figure 7: Solver used for proving properties over 3-box - Polar coordinates (ρ, θ)

We can see that DEEPPOLY is able to quickly prove properties in area where the cost functions are very different, whereas it does not reach a proof in areas where cost functions are very close. Difficult verification needing RELUPLEX/PLANET are the areas where ACAS Xu system gives avoidance orders.

2.3 Why a New Hybrid Architecture

Current implementation of ACAS Xu should embed 4 Gbytes LUT and executes at 1 Hz. Such implementation with avionic constraints is rather challenging, see for instance [21]. In particular, there is not much such large memory available on the market that is compatible with avionics constraints. Compression is therefore a strategic approach but not at the cost of reduced safety. This is the reason why we approximate them as NN together with a safety net. Using an approved fallback to mitigate safety risks is regularly used in the avionic system architectures. This is also a strong recommendation of the AVSI report [2] to bound the behavior of ML algorithms and prevent any unintended behaviour that may challenge the system safety.

In the table below, we have computed the size needed by the neural networks as well as the one for managing the switch and the safety net.

Network shape	Nb of Parameters in NN	NNs Size (MB)	Failed boxes (kB)	Full memory footprint (MB)
[14]	598,725	102.6	564.0	103.2
regular50	713,475	122.4	162.0	122.7
decreasing128	1,270,305	217.8	39.0	217.8

For the safety net and *check module*, we need to store p-boxes and exactly the same since *check module* identifies when to switch in the safety net. Each unsafe box will be stored in the memory using the lower and upper points. Using float32, the size needed by a 3-boxes is 24 bytes. This leads to a size of 974 kB for [14], 4.1 kB for regular50 and 4.0 kB for decreasing128. Footprint of unsafe boxes is one order of magnitude below the networks footprint and is decreasing with the size of the network. The compression of NN could also be improved using pruning and quantization techniques [11].

3 Certification Methodology

Assurance cases (AC) are gaining more and more consideration as valuable methodologies for development and certification. John Rushby [24] defines them as: *Assurance cases are a method for providing assurance for a system by giving an argument to justify a claim about the system, based on evidence about its design, development, and tested behavior.*

3.1 Notations

The idea is to detail the argumentation leading to a certain *conclusion* or *claim*. In the context of certification, a claim is an objective to be fulfilled by the applicant. In practice, the demonstration is based on the elicitation of requirements that correspond to the justifications that the objective is achieved. There exist several notations, either textual or graphical, to support the design of an assurance case, such as GSN (Goal Structuring Notation) [15]. All of them are relying on the Toulmin work [29].

Among the existing notations, we will use subsequently in the paper a graphical adaptation of Toulmin notation proposed by the RESSAC [22] project. RESSAC was a European project that coordinated European industry efforts to contribute to the FAA initiative called "Overarching Properties", which promotes an alternative certification approach to ease the introduction of next generation systems. This notation relies on (see figure 8): the Claim C is either the upper conclusion or intermediate conclusions (or sub-claims), the Evidence E is a leaf that consists of a V&V documentation that supports some claim, the Reasoning R explicitly describes the argument and the Backing B supports the reasoning. The Backing is a kind of endorsement of the reasoning, a guarantee that the reasoning is reliable. The defeater D allows for expressing that in some circumstances the conclusion may not be true. Such a notation is very helpful as it offers simplicity and the possibility to challenge the reasoning steps. The context contains additional information needed to provide definitions or descriptions of terms constraining the applicability of the assurance case to a particular environment or set of conditions.

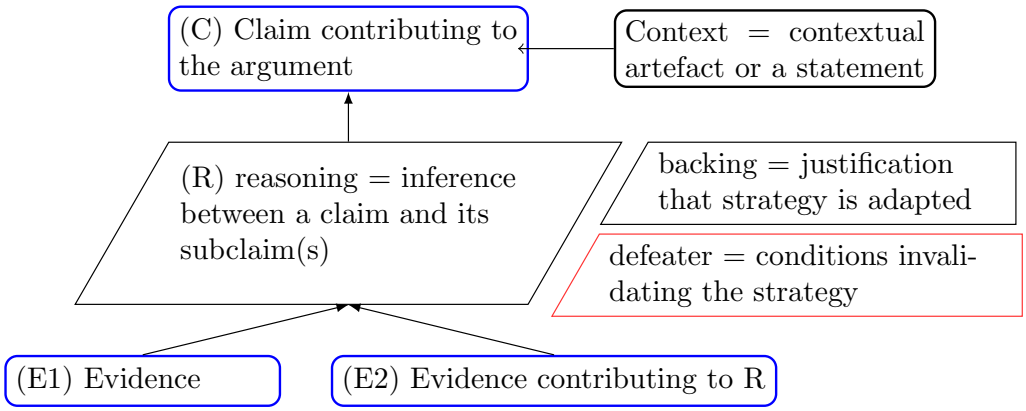

Figure 8: Graphical RESSAC notation

3.2 Assurance Case for the Hybrid Controller

The objective of the certification approach is two-fold: first demonstrate the completeness and the correctness of the ML-based item implementation with respect to the system and safety requirements; and second reinforce the

confidence that the ML-based item has been developed in a sufficiently disciplined manner to limit the likelihood of development errors that could impact system safety. This approach covers the "Learning Assurance" and "AI safety risk mitigation" building blocks, pillars of the trustworthiness concept introduced by the EASA AI Roadmap 1.0 [6].

The overall assurance case for the hybrid architecture is quite large as it covers the full ACAS system development and contains 120 elements (claims, reasoning, context, backing and evidences) addressing the objectives of the ARP4754A [25]. Due to space limit, we cannot detail everything and we chose to focus on some objectives.

ACAS-Xu Subsystem Requirement Capture. The figure 9 shows the reasoning to demonstrate that the ACAS Xu specification process meets the ARP4754A [25] guidance concerning the definition of system requirements and interfaces. The argument is based on 2 sub-claims: the capture of the functional and performance requirements. The performance requirements are not further detailed.

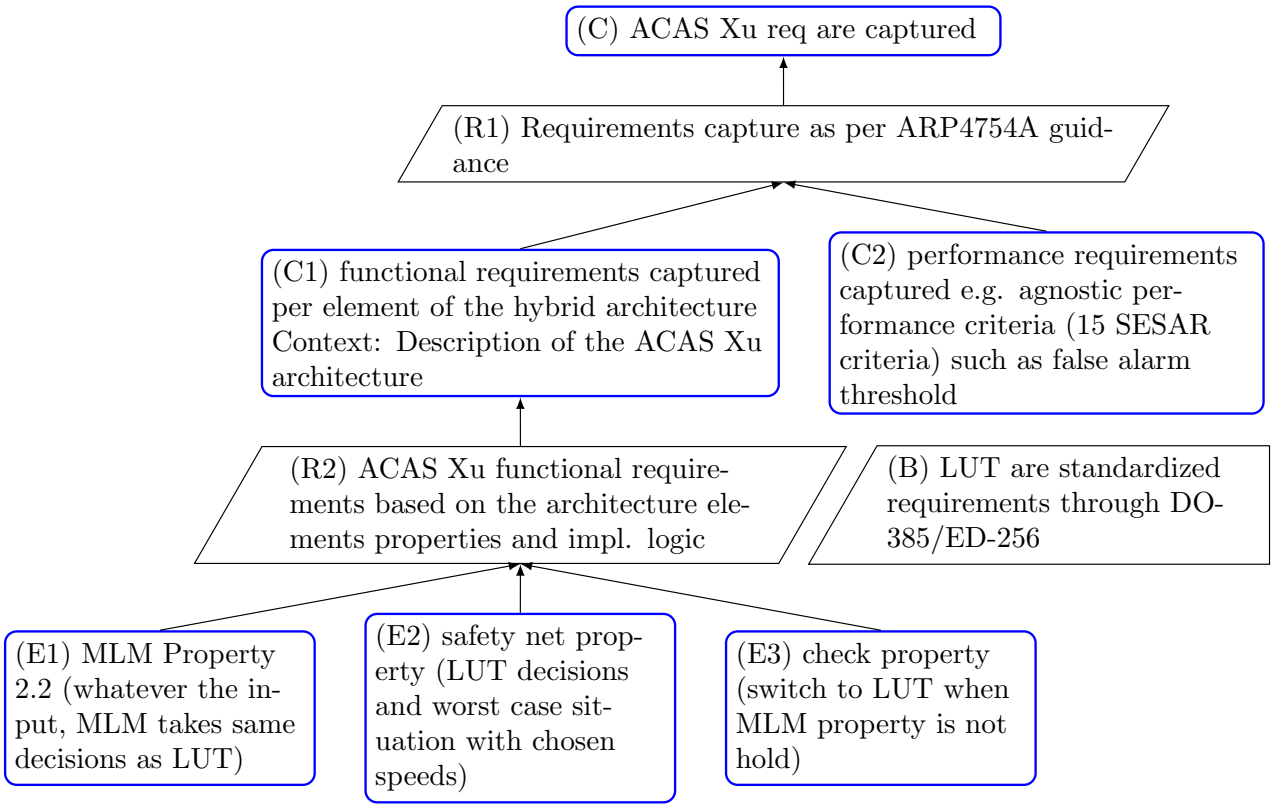

Figure 9: Assurance Case - ML subsystem requirements

The functional requirements must be refined for each item of the hybrid architecture (NNs and safety net). The reasoning is that the LUT decisions are the behaviour reference of the controller. Thus each item of the hybrid architecture should have equivalent properties and the switch logic should be appropriate. Specifically, property 2.2 is defined to guarantee the correct operation of the MLM.

ACAS-Xu Item Verification (for the ML Element Robustness Part). As per [3], one of the main premises of the robustness demonstration is "*real-world situations to which the subsystem is not robust should be identified and mitigated*" (refer to claim C in figure 10). All the situations where the MLM provides incorrect predictions (i.e. where Property 2.2 is not preserved by the MLM), are identified. The mitigation is realized by the architecture design (switch to the safety net which embeds the subset of LUT needed for mitigation). The preservation of the Property 2.2 is formally verified within the robustness analysis: the input space is divided into boxes defined by points of the LUT. When decisions associated to the top-points of a box are different from one another (frontiers of decisions), then the estimated prediction of each point of the box is considered as correct when identical to one of the box top points. Property 2.2 is verified using formal methods: when Property 2.2 does not hold, this means that the situation may be unsafe and that the hybrid controller should switch to the LUT computation to take the appropriate decision.

ACAS-Xu Subsystem Validation. Considering that LUT standardization has been recognized by the Authorities, one can think that the proper verification of the MLMs (to correctly approximate LUT predictions) would be sufficient to consider them as validated. Actually, the lack of transparency of the ML technique (no

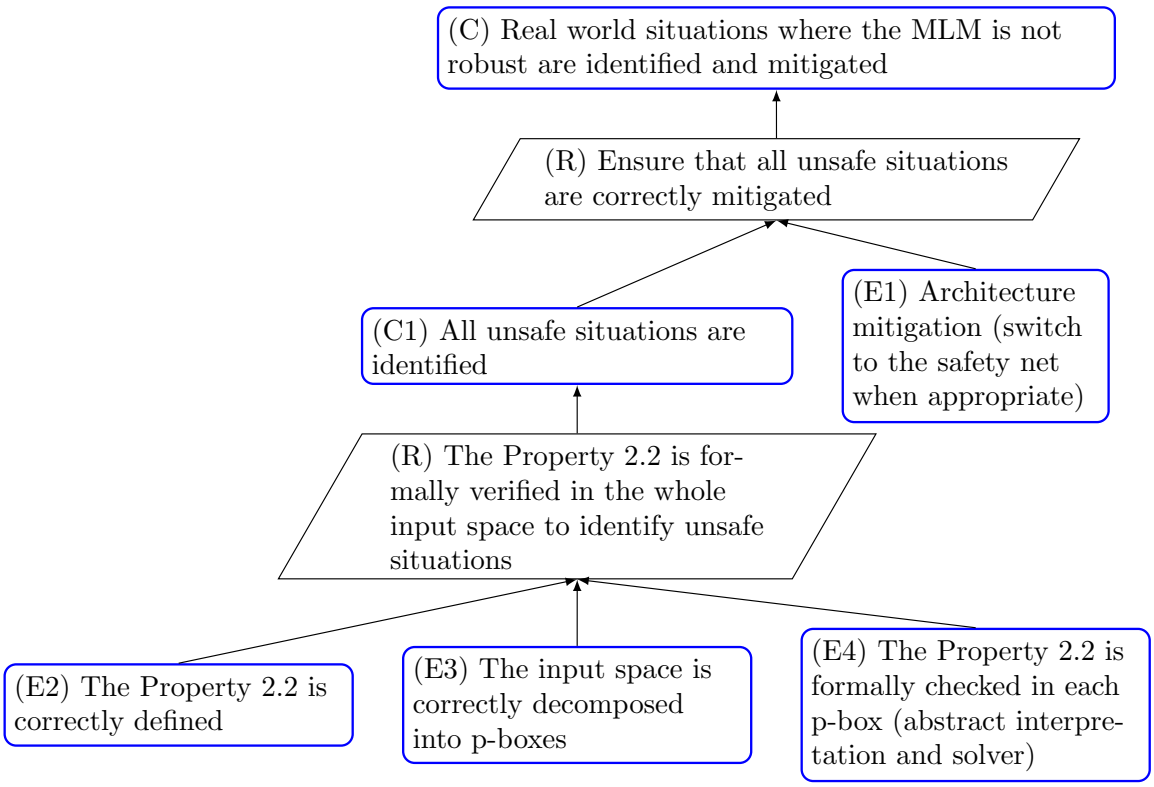

Figure 10: Assurance Case - ML item robustness

traceability capability, black box effect) may require a need for additional assurance that MLMs properties are correct and complete. For this purpose, specific tests have been developed in a simulation environment enabling the comparison between operational behaviors of ML-based design and real LUT design whatever the geometric situation. The figure 11 develops the argumentation and illustrates the use of defeater (D) to challenge the confidence that the use of standardized data may not be sufficient for demonstration of conformity.

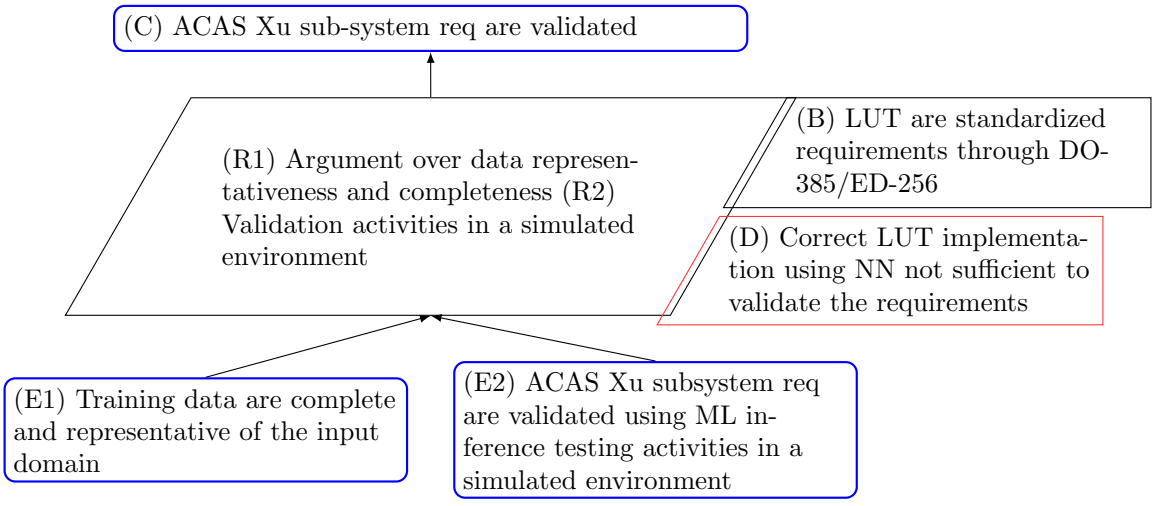

Figure 11: Assurance Case - ML subsystem validation

4 Related work

Proved ACAS Xu Up to now, the ACAS Xu compression works only proposed to replace the LUT with a set of neural networks. Even if there have been several papers on formal verification of NN, none of them has tackled the certification itself. Most of the time, papers prove some local properties on the neural network which is

not sufficient to cover certification expectation. Authors of [14] proposed to prove 10 avoidance meta-properties without any explicit link with with avoidance standard [10]. We believe that these properties are not enough for enabling the certification of an ACAS-Xu system. Our approach is different, because we consider LUT as the requirements (part of the standard) and we formally guarantee that outputs of our system will be exactly the same as LUT.

Certification Methodology There are several works offering assurance case to summarize confidence for ML components. [3] proposed a pattern to ensure the robustness of ML subsystems. We have completed and adapted this approach to integrate the specific properties (safe behaviour reference given by the LUT) and the safety net. [26] proposed a template to structure the safety argumentation part specific to DNNs. Their work is illustrated with an example use case based on pedestrian detection.

To the best of our knowledge, no assurance case approach has been proposed to tackle the respect of functional and performance objectives at system level (when ML sub-components are involved) for aircraft certification. The literature in the automotive is richer. In particular, [20] argued that assurance cases can be used for DNNs based systems. [30] goes further as it proposes GSN patterns to reason on the safety requirements of ML-based components and their integration within a system-level reasoning to show the compliance with ISO 26262. Thus, our work is complementary as we address the aeronautical sector and tackle the ARP4754A.

[1] proposed a novel concept of Dynamic Assurances Cases (DAC) that is applied to an aviation system that integrates ML-based perception function for autonomous taxiing. This concept is based on a framework of assurance methods /tools addressing safety concerns during development and extending this level of assurance to an in-flight operational use. They use both Assurance Case (with GSN notation) and architecture mitigation to develop assurance components for the DAC framework. Though we share the ARP4754A objectives and the assurance case methodology, our main objective is more to bridge the gaps of conformity of a ML-based system to the ARP4754A safety, functional and operational objectives and guarantee an acceptable means of compliance with certification requirements.

Learning a Surrogate NN of a LUT Neural networks are a new trend for approximating complex functions as a replacement of LUT, for example for control command systems [27]. Significant work has been performed in the framework of deep Q-learning, where the Q table is approximated by neural network. There have been some experiments in the context of Calibration Look-up table for the tuning of voltage-controlled circuits [18]. It was also explored for ACAS-Xu use case in [13, 12]. In [13], authors compares table compression using the origami algorithm which exploits data's redundancies and symmetries, and a method and using a neural network with a regular architecture. In [12], further exploration of the neural network approach is performed, introducing several tricks to enhance performance.

5 Conclusion

We have designed a safe NN-based ACAS Xu architecture and shown with an assurance case that such a way of doing could be well argued for certification to the regulation authorities. The certification evidences will be completed with sub-system level analyses (with simulation and reachability analysis [4]).

In the future, we also plan to implement the hybrid architecture on an embedded board to complete the certification proof. We will also apply our methodology for other LUT-based safety critical systems.

Acknowledgments. This project received funding from the French "Investing for the Future – PIA3" program within the Artificial and Natural Intelligence Toulouse Institute (ANITI). The authors gratefully acknowledge the support of the DEEL project[1].

References

[1] E. Asaadi, S. Beland, A. Chen, E. Denney, D. Margineantu, M. Moser, G. Pai, J. Paunicka, D. Stuart, and H. Yu. Assured integration of machine learning-based autonomy on aviation platforms. 2020. 39th Digital Avionics Systems Conference (DASC'20).

[2] AVSI. Final Report AFE 87 – Machine Learning, 2020.

[3] R. H. R. C. Chiara Picardi, Colin Paterson and I. Habli. Argument patterns and processes for machine learning in safety-related systems. 2020. University of York, York, U.K.

[4] A. Clavière, E. Asselin, C. Garion, and C. Pagetti. Safety Verification of Neural Network Controlled Systems. In *7th International Workshop on Safety and Security of Intelligent Vehicles (SSIV 2021)*, 2021.

[1] https://www.deel.ai/

[5] J. B. Diederik P. Kingma. Adam: A method for stochastic optimization. In *3rd International Conference for Learning Representations*, 2015.

[6] EASA. Artificial Intelligence Roadmap: A human-centric approach to AI in aviation, 2020.

[7] R. Ehlers. Formal verification of piece-wise linear feed-forward neural networks. *CoRR*, abs/1705.01320, 2017.

[8] EUROCAE / RTCA. DO-178C/ED-12C - Software Considerations in Airborne Systems and Equipment Certification, 2011.

[9] EUROCAE WG-114/SAE joint group. Certification/approval of aeronautical systems based on ai, 2021. on going standardization.

[10] EUROCAE WG 75.1 /RTCA SC-147. Minimum Operational Performance Standards For Airborne Collision Avoidance System Xu (ACAS Xu), 2020.

[11] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, 2016.

[12] K. D. Julian, M. J. Kochenderfer, and M. P. Owen. Deep neural network compression for aircraft collision avoidance systems. *arXiv:1810.04240*, 2018.

[13] K. D. Julian, J. Lopezy, J. S. Brushy, M. P. Owenz, and M. J. Kochenderfer. Deep neural network compression for aircraft collision avoidance systems. *35th Digital Avionics Systems Conference (DASC)*, 2016.

[14] G. Katz, C. W. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. *CoRR*, abs/1702.01135, 2017.

[15] T. Kelly and R. Weaver. The goal structuring notation /- a safety argument notation. In *Workshop on Assurance Cases*, 2004.

[16] M. Kochenderfer, J. Holland, and J. Chryssanthacopoulos. Next generation airborne collision avoidance system. *Lincoln Laboratory Journal*, 19:17–33, 2012.

[17] Y. LeCun, L. Bottou, G. Orr, and K. Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade*, chapter 2, page 546. 1998.

[18] A. Leoni, Z. Marinković, and L. Pantoli. On the introduction of neural network-based optimization algorithm in an automated calibration system. In *14th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS)*, pages 323–326, 2019.

[19] G. Manfredi and Y. Jestin. An introduction to acas xu and the challenges ahead. In *35th Digital Avionics Systems Conference (DASC'16)*, pages 1–9, 2016.

[20] M. C. O. S. Ramneet Kaur, Radoslav Ivanov and I. Lee. Assurance case patterns for cyber-physical systems with deep neural networks. 2020. SAFECOMP 2020 Workshops.

[21] L. Ren, R. Fisher, J. Lopez, J. Markham, M. Figard, R. Evans, R. Spoelhof, I. Pedan, M. Rubenstahl, S. Edwards, B. Meng, and C. Barrett. Integration and flight test of small uas detect and avoid on a miniaturized avionics platform. 2019.

[22] RESSAC. Recommendations for the use of assurance cases for demonstrating and assessing overarching properties. Technical report, LIV-S026-D4-199, 2019.

[23] RTCA, Inc. DO-254 - Design Assurance Guidance For Airborne Electronic Hardware, 2005.

[24] J. Rushby. The interpretation and evaluation of assurance cases. Technical report, 2015. Technical Report SRI-CSL-15-01.

[25] E. . SAE. Aerospace Recommended Practices ARP4754a/ed-79a- development of civil aircraft and systems, 2010.

[26] G. Schwalbe, B. Knie, T. Sämann, T. Dobberphul, L. Gauerhof, S. Raafatnia, and V. Rocco. Safety argumentation for deep neural network based perception in automotive applications. 2020. SAFECOMP 2020 Workshops.

[27] C. Seren, P. Ezerzere, and G. Hardier. Model–based techniques for virtual sensing of longitudinal flight parameters. *International Journal of Applied Mathematics and Computer Science*, 25, 03 2015.

[28] G. Singh, T. Gehr, M. Püschel, and M. Vechev. An abstract domain for certifying neural networks. *Proc. ACM Program. Lang.*, 3(POPL), 2019.

[29] S. E. Toulmin. *The Uses of Argument*. Cambridge University Press, 2003. Updated Edition, first published in 1958.

[30] E. Wozniak, C. Carlan, E. Acar-Celik, and H. J. Putzer. A safety case pattern for systems with machine learning components. 2020. SAFECOMP 2020 Workshops.