Qualification of Avionic Software based on Machine Learning : Challenges and Key Enabling Domains

Guillaume Vidot, Christophe Gabreau, Ileana Ober, Iulian Ober

Qualification of Avionic Software based on Machine Learning : Challenges and Key Enabling Domains

Guillaume Vidot *
Airbus Opération S.A.S and IRIT, Université de Toulouse

Christophe Gabreau†
Airbus Opération S.A.S

Ileana Ober‡
IRIT, Université de Toulouse

Iulian Ober§
ISAE-SUPAERO, Université de Toulouse

Advances in machine learning (ML) open the way to innovating functions in the avionic domain, such as navigation/surveillance assistance (e.g. vision-based navigation, obstacle sensing, virtual sensing), speech-to-text applications, autonomous flight, predictive maintenance or cockpit assistance. Current standards and practices, which were defined and refined over decades with classical programming in mind, do not however support this new development paradigm. This article provides an overview of the main challenges raised by the use of ML in the demonstration of compliance with regulation requirements (*i.e.* software qualification), and a overview of literature relevant to these challenges, with particular focus on the issues of robustness, provability and explainability of ML results.

I. Introduction

Safety critical systems, and avionics in particular, represent an application field unenthusiastic in applying new software development methods [1], as shown, for instance, by the fact that some aspects of the object-oriented programming, such as polymorphism and dynamic biding, never made their way to safety critical systems, mostly because of the inconvenient balance between added value and increased development and validation costs. Nevertheless, the recent advances in machine learning triggered genuine interest, as machine learning offers promising preliminary results and opens the way to a wide range of new functions for avionics systems, for instance in the area of autonomous flying. In this paper we investigate on how existing certification and regulation techniques, can (or cannot) handle software development that includes parts obtained by machine learning.

*Software Development Assurance Engineer/Ph.D Student, Airbus Operation S.A.S, Toulouse, France, eric-guillaume.vidot@airbus.com
†Software Process Expert, Airbus Operation S.A.S, Toulouse, France, christophe.gabreau@airbus.com
‡Professor, IRIT, Université de Toulouse, Toulouse, France. E-mail: ileana.ober@irit.fr
§Professor, ISAE-SUPAERO, Université de Toulouse, France. E-mail: iulian.ober@isae-supaero.fr

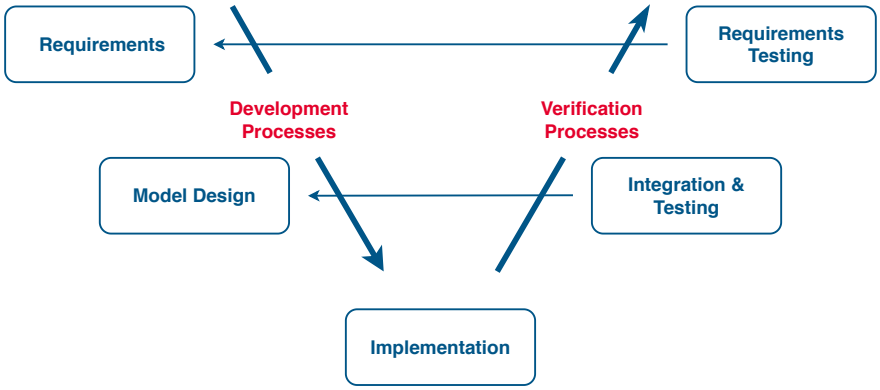

Fig. 1 Classical item development workflow

Nowadays a large aircraft cockpit offers many complex avionic functions: flight controls, navigation, surveillance, communications, displays, etc. Their design has required a top down iterative approach from the aircraft level downward, thus the functions are performed by systems of systems, with each system decomposed into subsystems (or equipment) that may contain a collection of software and hardware items. Therefore, any avionic development considers 3 levels of engineering: *(i)* Function, *(ii)* System/Subsystem and *(iii)* Item. The development process of each engineering level relies on several decades of experience and good practices that keep on being adapted today. These methods have been standardized through EUROCAE/SAE standards for system development (ED-79A/ARP4754A), EUROCAE/RTCA standards for software items (ED-12C/DO-178C and supplements) and hardware items (ED-80/DO-254). They are recognized as applicable means of compliance with regulation requirements by worldwide Avionic Authorities (AAs). In this context, we consider the development of an ML task hosted on a single item denoted as *ML item*. This paper focuses on the *item level* and more precisely the software items. Note that at the item level, the correct terminology for the demonstration of conformity to standards is "qualification". However, this term is commonly used for tooling. To avoid confusion in the following, we will use either "software aspects of certification" or "demonstration of conformity", depending on the context.

Figure 1 is the V-cycle representing the item's development workflow. Today, the usual development paradigms are Requirement-based Engineering (RBE) or Model-Based Engineering (MBE). Indeed, avionic items are developed using *classical programming*, *i.e.* the developers explicitly code algorithms to implement the requirements (or the model) and thus can control the behavior of each line of code of the embedded software item.

Recently, new avionic functions have emerged, aiming at developing new flight experiences: navigation/surveillance assistance (e.g. vision-based navigation, obstacle sensing, virtual sensing), speech-to-text applications, autonomous flight, predictive maintenance, cockpit assistance, etc. Contrary to *classical programming*, which can hardly support these functions, Machine Learning (ML) which is a sub-domain of Artificial Intelligence (AI), is well known to show good results for most of them (e.g. [2–4]). Current industrial guidance have a strong focus on established technologies

in aeronautical applications, and thus are not appropriate to support this change of development paradigm.

Indeed, ML techniques introduce a brand new paradigm in avionic development: the data-driven design of models (including supervised, unsupervised and reinforcement learning). *Data-driven* refers to the fact that the data determine the algorithm behavior through a learning phase. We choose to restrict our research to **non-adaptive supervised learning**, *i.e.* the training is done with *labelled data* (supervised) and *before embedding the ML software* (non-adaptive) [5, 6]. As a first step to the demonstration of conformity of a ML item, these restrictions seem reasonable as they still allow handling the problems raised by the advanced avionic functions mentioned above.

Even with such restrictions, many significant issues need to be addressed regarding the current demonstration of conformity. The motivation of this paper is two-fold: *(i)* identify the main challenges to the demonstration of conformity of an ML item, and *(ii)* overview the literature to identify existing methods able to solve the problems raised by the identified challenges.

It is worth mentioning that our work is carried out in parallel with the work of EUROCAE WG-114 - a worldwide working group established by EUROCAE aiming to standardize the development and the certification of aeronautical systems implementing ML. Unfortunately, as of today, the working group has not released publicly available documents. Nevertheless, as one of the authors of this paper is member of this WG, our contribution is, to the best of our knowledge, inline with the current WG work.

This paper is structured as follows: in section II, we overview the main challenges raised by the use of ML-based software item in the demonstration of compliance with regulation requirements. In section III we introduce a classification of considerations and research areas which will help to fill the existing gaps in future software aspect of certification approaches. Section IV contains a literature survey focused on the domains that address key issues introduced by the intrinsic characteristics of ML. We end the paper with section V which recaps the identified problems for the software aspect of certification of ML items.

II. Impact of ML on the software qualification approach

The European Union Aviation Safety Agency (EASA) provides regulatory material that defines all the requirements due for developing safe avionic products [7]. Regarding software/hardware items embedded in avionic safety-related systems, the certification is described in "Certification Specification" documents (CS 2x.1301 and CS 2x.1309) that are applicable to a large range of aircraft (planes, helicopters and even some drones categories). We would summarize these paragraphs as follows: *avionic systems should safely perform their intended function under all foreseeable operating and environmental conditions.* Through supplemental documents (AMC20-115D and AMC20-152A*), EASA recognizes the current avionic standards as acceptable means of compliance to the regulation text.

*AMC stands for Acceptable Mean of Compliance

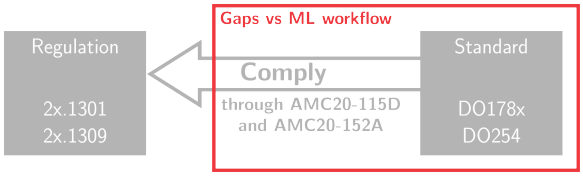

Fig. 2 Certification Process Overview

Assuming the ML techniques are reduced to non-adaptive learning and for human-assistance function[†], we do not anticipate any change to the regulation [7]. However, even if regulation requirements are unchanged, the current standards do not provide sufficient guidance to make a complete demonstration of conformity for a ML-based item, leaving open a gap schematically represented in Figure 2. Indeed, some fundamentals of the usual techniques — RBE or MBE — are jeopardized, challenging the classical safety guarantee argumentation.

Three gaps can be directly identified: the lack of *specifiability*, the lack of *traceability*, and the introduction of *unintended behavior* that traditional methods cannot handle. We elaborate on these gaps, hereafter.

Specifiability. One of the fundamentals of the RBE (or MBE) relies on the correct and complete capture of the requirements: coming from system allocated requirements (intent) or from its behavior (emerging functions). Then, the item can be verified to safely perform the intended function under all foreseeable operating and environmental conditions.

ML-based items are often used when it is very hard to fully specify the function with a classical requirement process. For instance, all the possible ways to describe a runway whatever the environmental and operational conditions (e.g. weather, light conditions, sensor bias) cannot be defined using textual requirements or a modelling technique. Another example would be the description of a pedestrian for the automotive industry; it is hardly possible to textually describe every human that might cross a road. In such a case, a ML-based approach will be preferred instead, with a training dataset of maybe millions of images governed by the Operational Design Domain (ODD) which described completely the intended function (see [5, 6]). Hence the role of the ODD and the dataset is to complete the requirement capture to allow the development of an acceptable runway detection function.

The absence of a *complete* specification inherently decreases the confidence that the model behavior will always fit the functional intent and therefore will be free of unintended behavior that may jeopardize the safety. It is however possible to mitigate this problem by approaching it from two angles:

- By ensuring the completeness and the quality of the dataset. For example, for a runway detection function the idea is to describe as thoroughly as necessary the Operational Design Domain (ODD) in which the detection function is supposed to operate and then to validate that the collected data correctly and sufficiently represent this ODD.
- By formally specifying safety properties that the ML model should satisfy for any combination of inputs. For example, for a braking distance estimation function constructed by ML, the idea is to formally specify certain

[†]as per level 1 defined in [5, 6]

properties such as bounded variation or monotony w.r.t. some parameters (weight, speed). The satisfaction of such properties of the ML model can in some cases be verified, e.g., using an SMT solver, as we will see in SectionIV.C.

Traceability. The current standard for software item development (DO-178C) requires the traceability between the requirements and the code (in both directions) so that each line of embedded code can be traced and then justified as implementing captured requirements. In a ML development context, this relationship that makes the design a white-box process, is lost. Actually the trained algorithm is a complex parametric mathematical expression, where the values of the *learned parameters* are not traceable to any upward functional requirement. Thus it becomes infeasible to demonstrate the completeness of implementation using traceability.

This lack of traceability results in the loss of transparency of the model, making the link from the input data to the output predictions not understandable by a human. This lowers the confidence that the safety properties of the intended function are preserved in its operational domain. In this context, explainable AI is seen as a means to enhance the confidence in these algorithms when safety is highly critical (see sections III and IV.A).

Unintended behaviour. All the life-cycle processes (requirements capture, model design, implementation, integration and requirement-based testing) of the item development workflow (see Figure 1) are mainly used to demonstrate the consistency with the intended function. In traditional software development, methods for meeting the unintended behavior related objectives [8], rely on the use of preventive methodological approaches as well as structural coverage metrics. In addition to the problems mentioned above, the learning phase could introduce some unexpected behavior, such as high sensitivity to perturbations, that we cannot measure or prevent by traditional testing, verification and validation activities. Consequently, one cannot guarantee the absence of unintended behavior during item operation and specifically those affecting aircraft safety.

This may be due to several factors such as a low-quality data management process (e.g. bias, mislabeled data) or inadequate learning process. We will see in section IV.B that there are methods to limit or to formally verify the absence of such effects.

III. Key domains involved in ML software qualification

The software aspect of certification of ML items in the frame of avionic developments offers many research challenges in various domains (see Figure 3). In this paper, we do not aim at providing an exhaustive overview of possible techniques to solve these challenges. The originality of our work stands in the perspective of the conformity to the regulation requirements inherent to the development of safety-critical software for the avionic domain and the literature survey motivated by the aeronautical regulation. The EASA released in 2021 its first guidance regarding the

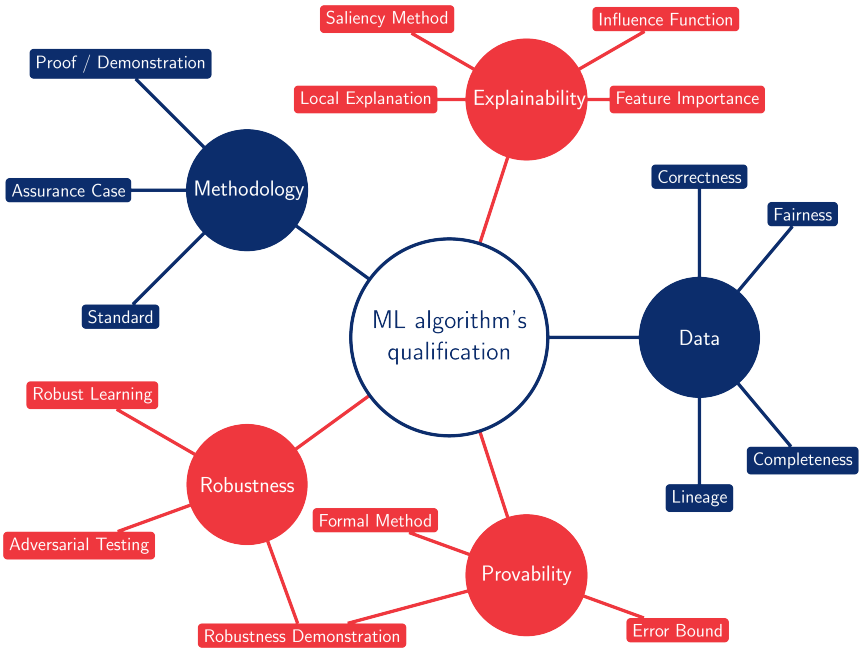

Fig. 3 Key domains involved in ML model software aspect of certification. (The domains that we address in detail in the literature survey are represented in red, those that we present briefly are in blue.)

certification of ML applications [5, 6][‡].

Due to the large number of domains involved in the demonstration of conformity of ML items, our scope is reduced to the software part and we address the novel issues raised by the introduction of Machine Learning. Thus, issues such as embeddability are not addressed, nor resiliency issues such as fault tolerance, which may be handled at the system level. We spotlight five different domains which are promising to contribute significantly to support the software aspect of certification of ML-based items: explainability, provability, robustness, data management, and *methodology* (see Figure 3).

Explainability. Explainability is a topic inherent to ML techniques. It is not needed for classical programming since the item development can be fully explained through the joint requirement and traceability processes which enable the interpretation of each line of embedded code. ML development breaks this understanding chain and makes room for an undesirable black box effect. This is the reason why one can expect that, when required by the safety level of the implemented function, explainability would be requested to add the necessary confidence to support the demonstration of conformity [5, 6]. Phillips et al. [9] introduce four principles of explainable AI: explanation, meaningfulness, explanation accuracy and knowledge limits. Basically, these principles state that an explainable AI must: *(i)* provide an "evidence or reason for all outputs" *(ii)* be meaningful with respect to the audience, *(iii)* be representative of the way that the algorithm produce the output and *(iv)* be aware of the domain of usage of the algorithm, *i.e.* it should not give an

[‡]this document gathers a first set of objectives in order to anticipate future EASA guidance and requirements

explanation when the input is out of scope, since the algorithm is not designed to work with it.

In the aeronautical context, four stakeholders could need explanation: the designers, the authorities, the pilots and the forensic investigators. Considering the "meaningfulness" principle, each of them could receive a different explanation. Indeed, the needs for explanation would be different for an engineer who knows the system and for an end-user who has no prior or inner knowledge of the system. The explanations could also differ whether it is used for debugging purposes during the development (for the designer), for investigation purposes in case of in-flight issues (for authorities or investigator) or for a user assessment of the model predictions when the system is deployed, e.g. a pilot who requests justification of the decision to enhance his confidence in the system before acting. In the literature, we find two kinds of explanation: local explanation [10–17] and global explanation [18–20] (see Section IV.A).

In a nutshell, an explanation might be given depending on the target audience and the level of the embedded AI (see Chapter B, section 5 of [5, 6]). For example, there are already some objectives given by the authority, *e.g.*, EXP-01 and EXP-02, which ask, respectively to identify all the stakeholders that needs explainability and to characterise the need for explainability to be provided for each stakeholders. The reader may refer to Chapter B section 6.2, Chapter C, section 3.2 and Chapter C section 4.1 of the issue 2 of the first guidance proposed by the EASA [5, 6] to have a more thorough view on the need for explainability expressed by the authority.

Adversarial Robustness. Robustness comprises several sub-domains (*e.g.* adversarial examples, distributional shift, unknown classes, *physical* attacks), but we focus on *adversarial robustness* which is defined as the capability of algorithms to give the same outputs considering variation of the inputs in a region of the state space. The issues addressed by the adversarial robustness domain are at the heart of the software aspect of certification process; it addresses partially the demonstration that there is no unintended function. The aim is to assess or enhance the behavior of the algorithm when dealing with *abnormal inputs* (*e.g.* noise, corner cases, sensor malfunction). The literature is two sided: one side works at enhancing the adversarial robustness of the algorithm [21–29], the other side at its verification [30–39]. Improving the adversarial robustness consists in finding robust learning procedures that are resilient against crafted examples made to defeat the ML algorithm.

In the first guidance (issue 2) [5, 6], the EASA states some objective regarding the robustness. Basically, we should capture robustness requirements (among others, LM-02) and then perform their verification (LM-13). The way to verify it is not imposed by the objective and one can either use formal method or empirical testing.

Provability. The provability aspect is the capability to formally demonstrate that a model's properties are preserved. Formal methods provide mathematical evidence to support such demonstrations. Since robustness can be expressed as a property, provability comprises adversarial robustness demonstration. Indeed, formal methods are already used to verify well-defined properties of models, such as the robustness [31, 32, 35, 37, 40] but also safety properties [33, 41].

Regarding the aeronautical context, verifying the model's properties, whether functional or safety-related, is an asset for its demonstration of conformity since it may avoid time-consuming testing and provide formal proof that the property holds.

The error (or generalization) bound gives guarantees on generalization of the algorithm. An algorithm generalizes well when it maintains its performance on unseen data, *i.e.* the theoretical error is close to the empirical error. The idea is to bound the gap between the theoretical error and its empirical counterpart using probabilistic bounds [42–50].

Again, we find already some objective associated to this challenges [5, 6]. Regarding the requirement-based engineering, one should perform requiremen-based verification (LM-10). Regarding the generalization bound, one should provide generalization guarantees (LM-04) and should test its reliability (LM-14)

Data management. In the avionic context, data has been used for a long time, with systems making heavy use of databases or configuration files. However ML techniques are bringing a totally new aspect in the software aspect of certification approach. Contrary to *classical programming*, ML design techniques are data-driven, thus the data management process is essential to the demonstration of conformity. As already stated, building a ML algorithm of good quality, requires data of good quality (no erroneous or mislabelled data) but this is not sufficient. Indeed the data representativeness plays also a significant role. Representativeness comes from statistics and is a quite challenging problem: it allows to check if the data is a correct snapshot of the phenomenon to be learned. In other words, it tries to measure whether there is sufficient data to learn the intended function and whether the data contains the correct information regarding the needs.

The quality of the data can be measured using metrics[51, 52] such as: accuracy, consistency, integrity, timeliness, traceability, and fairness. **Accuracy** checks if the data is well measured and stored. **Consistency** checks for expected relationship between parameters regarding the ODD. For example, we could never have an aircraft with zero speed at 40,000 feet. **Integrity** guarantees that data is not corrupted or removed from the source to the final user. **Timeliness** concerns the availability of the data over time. **Traceability** verifies the reliability of the data source and if all activities for the transmission do not alter the data integrity. **Fairness** is about avoiding undesirable bias in the dataset.

To have more details on what the EASA is currently requiring regarding data management, the reader may refer to the objective with the prefix "DM" in [5, 6].

Methodology. The methodology structures the assurance activities needed to support the software aspects of certification, *i.e.* the demonstration to the Authorities that the item development complies with the standardized guidance. The AMCs and the industrial standards may require changes (Figure 2); therefore it is necessary to find alternative means to comply with the regulation material, either by adapting the current software aspect of certification approach or by building a new one.

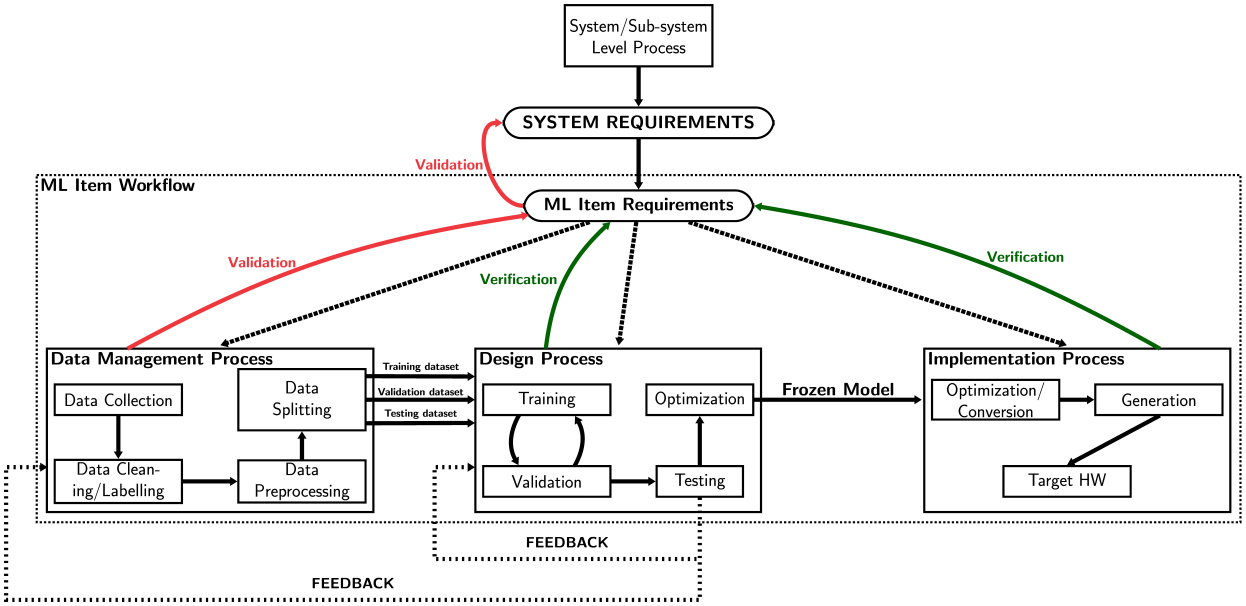

Fig. 4 Machine Learning development process.

It will be crucial for the future AMCs and the industrial standards to clearly define the development process of ML items, starting from existing best practices [53–56], to *(i)* ease their development and maintainability, *(ii)* identify the validation and verification activities and *(iii)* guarantee their certifiability. Figure 4 describes this development process: requirements from the system/subsystem level are refined into ML item level fitting the three main stages of the workflow:

- **Data Management Process**: First data are collected with respect to the problem to be solved. Then, the data should be cleaned and labelled[§], *i.e.* inaccurate data samples are removed and each remaining data sample is assigned with a true label (known as "ground truth"). Finally, data are preprocessed and split. Preprocessing encompasses all the tasks that transform the data into a more suitable format for the design phases (e.g., feature engineering or data normalization). The preprocessing effort may vary depending on the data collection phase (e.g., data comes from different sources) and the type of data (e.g. images, time series, tabular, sound, text). After the preprocessing step, the data is split into a training, a validation and a testing dataset.
- **Design Process**: The design process takes as input the three datasets and outputs a frozen ML model. The processes at this stage are iterative. Iterations between the training and the validation phase are performed to tune model's hyper-parameters. Once it is done, you test your model: if the performance is worse than stated in the ML item requirements, it is possible to loop back either to the data management process or the design process.
- **Implementation Process**: Within the implementation stage, the specificity of the hardware is taken into consideration: the frozen model is optimized and converted w.r.t the host platform constraints and the operational requirements cascaded from the system level. Finally, a binary code is generated and loaded in the hardware target.

[§]as the scope is reduced to non-adaptive supervised learning, labelling the data is necessary.

Since its creation in the 80's, the ED-12/DO-178 standard gathers the *industrial best practices* that impose the necessary rigor of development to avoid introducing errors that may lead to a system failure. Historically, these practices undeniably brought efficient results in terms of safety. Thus methodological considerations are key to build a correct software aspect of certificationn approach. An essential element in building relevant safety argumentation is the concept of safety case [57] or assurance case [58]. Indeed, Rushby [57] argues that the introduction of this kind of methodology (*i.e.* not prescriptive approaches) in the industries helps to significantly reduce the number of accidents and deaths. Assurance cases, which are a generalization of safety cases, are defined by MITRE [58] as follows: *a documented body of evidence that provides a convincing and valid argument that a specified set of critical claims regarding a system's properties are adequately justified for a given application in a given environment*. Hence, assurance cases could be used during the development process of a ML-based item to build, share and discuss sets of structured arguments to support the demonstration of conformity based on outcomes of specific assurance techniques [59].

IV. Literature overview

This section presents a literature overview focusing on *explainability*, *adversarial robustness* and *provability*, because they address key issues introduced by the intrinsic characteristics of ML models: the lack of transparency and the lack of reliability when inputs are slightly perturbed within the Operational Design Domain (ODD). Other surveys which do not focus specifically on the needs of the aeronautical field have been published; interested readers can refer to [60–63].

A. Explainability

Explainability is a very new constraint for embedded ML-based items. Implicitly contained in the traditionally programmed components, it has become challenging to demonstrate that the ML item's outcomes are trustworthy. Improving this confidence level seems inescapable to meet the acceptance criteria of the ML application user (*e.g.*, designer, authorities, investigators or pilot). It has been identified as a means of acceptance in the EASA first guidance [5, 6].

We identify two types of explanations : local explanations and global explanation. Both subdomains suit different needs: *(i)* explanation at the prediction level (local) and *(ii)* explanation of the behavior of the model regarding inputs evolution (global). In each subdomain, we face two types of explanation techniques: *model agnostic* [10–14, 18, 20] and *model-specific* [15–17, 19, 34]. Model agnostic methods provide explanation regardless of the type of ML algorithm while model-specific methods works with only one type of ML algorithm; most model-specific methods focus on DNN.

1. Local Explanation

Computer Vision is a complex domain because of the nature of the high dimensional data: images or videos. Since the development of DNN, it has become easier to reach good performance on vision tasks such as image classification

Table 1 Summary of explanation methods.

	Method name	Papers	Model targeted
Local Explanation	MASKING MODEL	Dabkowski and Gal [13]	ALL
	MPSM	Fong and Vedaldi [14]	ALL
	LORE	Guidotti et al. [10]	ALL
	GVE	Hendricks et al. [15]	DNN
	INFLUENCE FUNCTION	Koh and Liang [34]	DNN
	LIME	Ribeiro et al. [11]	ALL
	ANCHORS	Ribeiro et al. [12]	ALL
	DECONVNET	Zeiler and Fergus [17]	DNN
Global Explanation	SHAP	Lundberg and Lee [18]	ALL
	DEEPLIFT	Shrikumar et al. [19]	DNN
	NPSEM	Zhao and Hastie [20]	ALL

where ALL = model agnostic and DNN = deep neural network specific.

or object detection. However, the reason for the high efficiency of DNN is not yet fully understood. Hence researchers develop methods to explain, assess, and increase the confidence in the decision taken by DNN for computer vision tasks.

Model agnostic local explanation. LIME (Local Interpretable Model-agnostic Explanations) [11], ANCHORS [12] and LORE[¶] (LOcal Ruled-based Explanation) [10] locally approximate the behavior of the ML algorithm using inputs and outputs. The same principle applies to the three systems: explore a neighborhood around a sample and deduce from it an explanation for the sample. To go further, Garreau and von Luxburg [64] provide theoretical explanations of LIME.

The generation of the neighborhood differs according to the system. LIME and ANCHORS use an interpretable representation and apply perturbations on it to generate the neighborhood. An interpretable representation, which is defined according to the problem that needs to be solved, could be a binary vector where the ones are the relevant information and a perturbation could be the modification of the vector (0 switches to 1). Ribeiro et al. [11] define metrics to measure the distance between the neighbors and the original sample. LORE uses a genetic algorithm to generate a balanced neighborhood, which is tuned to create the best possible neighbors for the desired sample.

Since the neighborhood is generated, it remains to provide the explanation. In LIME linear models are used while in ANCHORS decision trees are used. For the ANCHORS explanation, "anchors" are used, *i.e.* the minimum number of features that lead to the right prediction. Note that, Ribeiro et al. [12] define several bandit algorithms in order to find the best anchors. An explanation given by LIME corresponds to the area of the image that helps to make the decision. ANCHORS and LORE provide explanation in the IF-THEN form. Guidotti et al. [10] (LORE) extract explanations from the decision tree while Ribeiro et al. [12] (ANCHORS) only check the presence or absence of anchors.

A popular method to explain a decision on images is the *saliency map* [13, 14, 65]. Intuitively, a saliency map highlights the features, *i.e.* the pixels on images, that the model considers to make its decision. Fong and Vedaldi [14]

[¶]Note that LORE is not applicable to images.

propose an agnostic gradient-based method which considers explanation as meta-predictors. A meta-predictor is a rule that is used to explain the prediction of the model. One advantage of using meta-predictors as explanation is that one can measure the "faithfulness" of your explanation by computing its prediction error, *i.e.* it represents the number of times that the model and the rule disagree on the prediction. Then, Fong and Vedaldi [14] claim that a good explanatory rule (*i.e.* meta-predictor) to produce a saliency map rely on the local explanation principle. Indeed, the authors of [14] want to study the behavior of the model using the neighborhood of the original example. This neighborhood is given by perturbing the original example; that is why they introduce the notion of "meaningful pertubations". They argue that the explanation would be better if the perturbations applied to build the neighborhood of the original example mimic *natural image effect*. Based on meaningful perturbations[‖], the authors define an optimization problem which tries to find the smallest area of the original image that must be perturbed in order to reduce prediction probability of the true class of the original image; This method then outputs the saliency map which gives the explanation of the model's prediction. In table 1, we labelled their method Meaningful Perturbation Saliency Map (MPSM).

Dabkowski and Gal [13] develop a MASKING MODEL that learns how to generate interpretable and accurate saliency maps for ML model. The authors craft a new objective function which ensures the quality of the saliency map: it ensures that the region is necessary to the good classification while its absence leads to low probability to pick the good class and at the same time it penalizes large and non-smooth regions. The MASKING MODEL comprises features filter (input) given by a trained network and upsampler layer which upscale the input into the original image dimension in order to provide a mask. Dabkowski and Gal [13] train the MASKING MODEL by minimizing its objective function.

DNN specific local explanation. Adebayo et al. [65] develop a methodology to test the usefulness of a saliency method for explanation. It is based on two tests: model parameter randomization test and data randomization test. A saliency method would fail the tests if it shows the same result for the randomized case and the trained case. A failure points out the independence of the saliency method regarding the architecture parameters or labeled data. With the methodology proposed by Adebayo et al. [65], we can test model-agnostic and model-specific methods.

A few years ago, Zeiler and Fergus [17] introduced a method that allows the visualisation the internal workings of convolutional neural networks (CNN). Their motivation was to understand how and why CNN work well by studying the hidden layer of CNN. The principle is to sample the features learned by a neural network, back to the original image dimension. Contrary to the methods discussed above, the goal was not to provide a saliency map that explains the model's predictions but only to visualize the features that a neural network learned for classifying. The system developed by Zeiler and Fergus [17] is known as multi-layered Deconvolutional Network (DECONVNET). One advantage of this technique is the possibility to start from any layer in order to see at this point the features learned by the neural network.

Another way of explaining an image is literally to provide written explanations: Hendricks et al. [15] propose a

[‖] the authors used blur, noise as meaningful perturbation

visual explanation for images**. Visual explanations are defined as *class discriminative* and *accurately described*, *i.e.* the textual explanation highlights elements of the image which are specific to the class. The authors use VGG, a CNN for image classification, and add to it an LSTM (Long Short Term Memory) that generates visual explanations; the efficiency of this system relies on the loss function. More precisely, Hendricks et al. [15] defines two loss functions for their network: one for the accurate description (relevance loss) that handles the probability of word occurrence in the sentence and the other for the class discrimination (discriminative loss) which is based on a reward function.

Koh and Liang [34] increase the explanability of DNN through the use of INFLUENCE FUNCTIONS. Unlike the other methods, where explanability is increased by providing evidence directly on the examples, Koh and Liang [34] provide the training data that leads to a prediction in order to explain that prediction. The principle of their method is to compute the impact on the predictions while modifying the training dataset. For explanability purposes, they study the effect of the removal of training data. The removed training data which strongly decrease the probability of getting the right class of a given example are actually the data that leads to the prediction. Actually, the training data given as an explanation of a prediction provide insight on the behavior of the model. Indeed, by looking at the data that influence towards prediction, it is possible to detect abnormal behavior of the model.

2. Global Explanation

Global explanation techniques manage to explain the evolution of the model outputs according to the trends of the inputs. A popular method is known as "feature importance". The principle of this method is to find out which input features play a significant role in the decision of the ML algorithm [18–20].

Model agnostic global explanation. One way of exploiting the feature importance principle is to use the Shapley value which comes from cooperative game theory. The idea is to find the fair share of the gain between players in a cooperative game regarding the involvement of each player in the coalition. Lundberg and Lee [18] develop SHAP (SHapley Additive exPlanations) that use the Shapley value in order to provide explanations for the predictions of ML algorithm. In this context, the features are the players and the prediction is the payout. Then, SHAP computes the contribution of each feature to the prediction. Since the computation of the Shapley value requires all the possible permutations of the players (features), the problem becomes intractable for inputs with numerous number of features. However, the authors provide an efficient way to compute an approximation of the Shapley value. Zhao and Hastie [20] study the importance of the features through a causal model, called Non-Parametric Structural Equation Model (NPSEM), and using Partial Dependence Plot to visualize the results. Thanks to this approach, they go further than only computing the impact of features on the prediction and manage to catch the prediction's tendency according to the features' evolution.

**their method is denoted "GVE" (Generating Visual Explanation) in table 1.

Table 2 Summary of adversarial robustness method. The ℓ_p-norm column refers the norm considered in the papers[††]. "-" means that the method do not rely on an ℓ_p-norm. The color code indicate whether the norm is used for attack or defense; black means it is used for both.

Method Name	Attack/Defense Papers	Attack	Defense	ℓ_p-norm
C&W	Carlini and Wagner [66]	✓		$\ell_0/\ell_2/\ell_\infty$
PARSEVAL TRAINING	Cissé et al. [30]		✓	ℓ_2
FGSM	Goodfellow et al. [23]	✓	✓	ℓ_∞
IFGSM	Kurakin et al. [24]	✓	✓	ℓ_∞
GRADIENT-BASED ATTACK+ILA ([67])	Li et al. [68]	✓*		ℓ_∞
PGD	Madry et al. [25]	✓	✓	ℓ_∞/ℓ_2
DEEPFOOL	Moosavi-Dezfooli et al. [69]	✓		ℓ_2
JSMA	Papernot et al. [26]	✓		ℓ_0
DKNN	Papernot and McDaniel [27]	✓*	✓	ℓ_∞/ℓ_2
L-BFGS	Szegedy et al. [28]	✓	✓	ℓ_2
LMT	Tsuzuku et al. [39]		✓	ℓ_2
GDA	Zantedeschi et al. [70]		✓	-
YOPO	Zhang et al. [29]		✓	ℓ_∞

* The authors adapt existing attacks to target the weakness of their defense.

DNN specific global explanation. Shrikumar et al. [19] develop a new way of handling features importance with DEEPLIFT (Deep Learning Important FeaTures). The principle is to compare inputs to an input reference and outputs to an output reference. These references represent a kind of *neutral* behavior of the neural network. Since these references are considered as baseline, the authors backpropagate the differences between the references and the actual sample through the network by comparing the activation values. Then, they define a contribution score system that derives the features importance from the differences. Hence, at the end of the backpropagation, we end up with the importance of each feature for a given prediction. Intuitively, we would say that the greater the difference is, the greater is the impact on the feature importance. The efficiency of DEEPLIFT depends on the references that relies on domain-specific knowledge.

In EASA [5, 6] first guidance, several objectives are dedicated to explainability. It is highlighted that the given explanation should match the audience; global explainability is related to the ML item itself and hence will be useful for engineers during development and post operation while local explainability is related to the output of the ML item and hence will be useful for any stakeholders (engineers, end users, authorities).

B. Adversarial Robustness

Szegedy et al. [28] was one of the first to point out adversarial examples as a weakness of ML. This weakness gave rise to new researches around adversarial attacks and defenses [22–29, 34, 71, 72] that will be discussed in this section. With regard to the fundamentals of the aviation regulation, which mainly requires demonstration that the system safely performs the intended function, the key role of robustness is twofold. On one hand and according to ED-12C/DO-178C, it is *the extent to which software can continue to operate correctly despite abnormal inputs and conditions*. On the other

[††]It is important to note that the techniques can be adapted with different ℓ_p-norm

hand and more specifically to ML application, EASA [5, 6] states that ML system is robust when it *produces the same outputs for an input varying in a region of the state space*. Perturbations can be natural (e.g. sensor noise, bias, etc.), variations due to failures (e.g. invalid data from degraded sensors) or maliciously inserted (e.g. pixels modified in images) to fool the model predictions. When perturbed examples fool the ML algorithm we talk about *adversarial examples*. It is commonly defined as noises on inputs that are imperceptible or that do not exceed a threshold. We state in Definition 1 two possible manners of generating adversarial examples, *i.e.* attacking an ML model.

Definition 1 *Adversarial Example. Let $(x, y) \in \mathcal{X} \times \mathcal{Y}$ be a benign example and the true label, δ be a perturbation, ϵ be the maximum allowed perturbation, L be a loss function and f be the trained ML model. we obtain an adversarial example $x + \delta$ by solving one of the following optimization problems:*

$$\min_{\delta} \|\delta\|_p \quad \textit{such that} \quad f(x+\delta) \neq y \quad \textit{and} \quad x+\delta \in \mathcal{X} \tag{1}$$

$$\textit{or} \qquad \max_{\delta} L(f(x+\delta), y) \quad \textit{such that} \quad \|\delta\|_p \leq \epsilon \quad \textit{and} \quad x+\delta \in \mathcal{X} \tag{2}$$

where $\|\cdot\|_p$ is an ℓ_p-norm and Domain(x) depict the allowed values for the example x.

As stated in Definition 1 the optimization problem leads to get an adversarial example with a different label than the original one; we refer to it as "untargeted attack". Few changes of the optimization problem produce a "targeted attack", *i.e.* the possibility to choose the label of the adversarial examples. Let $t \in \mathcal{Y}$ be the targeted label, we have

$$\min_{\delta} \|\delta\|_p \quad \text{such that} \quad f(x+\delta) = t \quad \text{and} \quad x+\delta \in \mathcal{X} \tag{3}$$

$$\text{or} \qquad \min_{\delta} L(f(x+\delta), t) \quad \text{such that} \quad \|\delta\|_p \leq \epsilon \quad \text{and} \quad x+\delta \in \mathcal{X} \tag{4}$$

Moreover, the attacks could be done in either a white-box or black-box setting. White-box setting means that the *attacker* has full access to the model, its parameters and the training dataset while black-box setting means that the *attacker* can only query the model with data.

Finally, to overcome adversarial attacks, adversarial defenses are deployed. One of the most efficient defense is the Adversarial Training (AT) [23–25]. It consists in augmenting the training dataset with adversarial examples. However, it is often observed that the AT based on a specific ℓ_p-norm are less effective against attacks based on different ℓ_p-norm.

Nevertheless, the defenses only enhance the adversarial robustness of ML algorithms whereas for the demonstration of conformity, one will need proof that systems based on the ML algorithm behave safely and as expected. Actually, there exist methods in the literature that verify that an ML algorithm is adversarially robust. They bring the guarantee needed to demonstrate the conformity to the regulation. Thus, to handle the adversarial robustness issue, we review adversarial attacks/defenses and verification methods, and we summarize the information in table 2. For the attack/defense methods, we report the ℓ_p-norm used in the paper. However the methods could be adapted to other norms.

1. Adversarial attacks

To better understand the principle behind the adversarial attacks, we may briefly recall how a model training based on the gradient descent works. It consists of iteratively computing the gradient of the loss w.r.t its weights and biases where at each step, the weights and biases are updated such that the loss becomes smaller by applying the following rule:

$$w^{(t)} = w^{(t-1)} - \lambda \nabla_w L(h, (x, y))$$

where w is the weights of your model, $(x, y) \in \S$ and $\lambda \in \mathbb{R}^+$ is the parameter controlling the descent known as the *learning rate*. The choice of the value of λ will determine the speed of convergence of the minimization. Knowing that, for crafting an adversarial example, instead of updating the weights leveraging the gradient loss (leading toward the minimization of the loss), the sample x is updated using the direction of the gradient to guide the search. In this case only the sign of the gradient is used, as its properties (the fact that its derivative is 0 or undefined) help in better fooling the model, as explained in Goodfellow et al. [23].

Goodfellow et al. [23] develop an efficient attack, called Fast Gradient Sign Method (FGSM), consisting in crafting the adversarial examples x_{adv}, by adding a fraction, ϵ, of the loss gradient's sign w.r.t the input to the original example x:

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x L(f(x), y))$$

Besides, the authors show that adversarial examples are invariant to the learning and the architectures, *i.e.* a different architecture trained on different subsets of the dataset misclassifies the same adversarial example. Later, Kurakin et al. [24] proposed an iterative version of FGSM which we denote IFGSM where at each iteration, a perturbation is added to x by applying FGSM to finally obtain x_{adv} after the desired number of iterations:

$$x^0_{adv} = x \qquad x^i_{adv} = x^{i-1}_{adv} + \epsilon \cdot \text{sign}(\nabla_x L(f(x^{i-1}_{adv}), y))$$

Since perturbations are iteratively added to x in IFGSM, the ϵ chosen is thus smaller than in FGSM. The attack based on Projected Gradient Descent (PGD) [25] is similar to IFGSM except that PGD randomly initialize x before the optimization:

$$x^0_{adv} = x + noise \qquad x^i_{adv} = x^{i-1}_{adv} + \epsilon \cdot \text{sign}(\nabla_x L(f(x^{i-1}_{adv}), y))$$

Carlini and Wagner [66] also develop a gradient based method but with a slightly different formulation of the optimization problem (cf. Definition 1). It is similar to Equation 1 but at the same time they minimize the margin of the model. In addition, the authors provide the formulation of their method for three distance metrics (ℓ_0, ℓ_2 and ℓ_∞). Moosavi-Dezfooli et al. [69] tailor their attack framework, called DEEPFOOL, for finding the minimum perturbation necessary with respect to the ℓ_2-norm[‡‡] to fool the algorithm. They first simplify the optimization problem to a linear classifier, derive the optimal solution for it and finally adapt the optimal solution found for the linear classifier to a neural network. They provide the algorithm of DEEPFOOL, which describes their iterative approach to estimate the smallest perturbation to create an adversarial example (cf. Algorithm 2 of [69]). Their results show that the adversarial example

[‡‡]Note that the authors explain how to adapt their method to other ℓ_p-norm

crafted by DEEPFOOL is often closer to the original example than the ones crafted by FGSM and L-BFGS [28].

Papernot et al. [26] leverage the "forward derivative" of a network f, to design the Jacobian-based Saliency Map Attack (JSMA). To obtain the forward derivative, they compute, from the input layer to the output layer, the derivative of the network f, instead of the derivative of its loss function, w.r.t. the input x. It basically corresponds to the Jacobian of the function learned by the network f. Then, the authors derive an "adversarial saliency map" based on the Jacobian of the network which points out the input features that should be modified to significantly impact the network's output.

Kurakin et al. [72] assess the adversarial robustness of models deployed in "real-world conditions", *i.e.* where the only way to communicate with the systems is through its sensors. Indeed this is a legitimate question since an attacker will not necessarily have access to the ML model. In their experiments, the authors feed an image classification algorithm through a camera. However, they still use the model to generate adversarial examples. They demonstrate that models are still vulnerable against adversarial attack even in "real-world conditions".

In the avionic context, security measures will be taken to restrict the access to the models and the datasets to undermine the white-box and black-box attacks. Nevertheless, the threat still exists since Li et al. [68] propose an attack in a more restricted setting than black-box, known as "no-box" setting [73] where the attacker has only a small number of examples that are not training examples. They succeed to efficiently fool the models trained on the imagenet dataset by developing an auto-encoder called "prototypical reconstruction" that manages to learn well with very few data. An auto-encoder consists of two parts: an encoder and a decoder. The encoder, encodes the input by learning a new representation of the original input and the decoder strives to reconstruct the input from the encoding as close as possible to the original input. The authors introduce a new loss for their auto-encoder that is well suited for gradient based attack. Their attack consists in mixing ILA [67], which improves the transferability of the attack, and a gradient-based method.

2. Adversarial defense

As stated earlier, one of the most effective defenses, called Adversarial Training (AT), consists in replacing the original training dataset by an "adversarial dataset" crafted with an attack. However, this method is time consuming, since it requires to generate adversarial examples at each step of the learning phase. Nevertheless, Goodfellow et al. [23] overcome this issue with FGSM and propose an adversarial training where they replace a part of the training dataset with perturbed examples. Following this principle, other works with stronger attacks propose adversarial training based on these attacks [24, 25]. Besides, Madry et al. [25] show that the adversarial training principle boils down to solve the following min-max optimization (or saddle-point problem):

$$\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \left[\max_{\|\delta\|_p \leq \epsilon} L(f_\theta(x+\delta), y) \right], \tag{5}$$

where $\mathcal{D}$ is the unknown distribution of the dataset, f_θ is the model whose parameters are θ, L is the loss and ϵ is maximum allowed perturbations. Indeed, the equation 5 comprises two parts: maximization and minimization. On the one hand, the equation maximizes the loss L regarding the noises δ, *i.e.* it makes $x + \delta$ more likely to be an adversarial

example. On the other hand, the equation minimizes the expectation of the loss regarding the model's parameters. To the best of our knowledge, AT based on PGD is one of the most efficient defense against attacks using the ℓ_∞-norm.

Though the methods presented above make AT feasible, it still increases the learning time of a model. Zhang et al. [29] aim at improving the computation cost of AT by reformulating it as a differential game and then derive the Pontryagin's Maximum Principle (PMP)[§§]. The PMP reveals that AT are closely linked with the first layer of neural networks. Therefore they develop YOPO (You Only Propagate Once) which leverage this fact by limiting the number of forward and backward propagation without worsening the network's performance. Their experiments show that YOPO learns 4 to 5 time faster to achieve as good results as adversarial training based on PGD.

Papernot and McDaniel [27] find another means to enhance the robustness of ML algorithms. The authors develop a method called Deep k-Nearest Neighbors (DKNN), *i.e.* a hybrid classifier that mixes Deep Neural Network (DNN) and kNN. Their motivation is to improve the confidence estimation, the model interpretability, and the robustness. The principle of DKNN is to find the nearest neighbors (from the training set) of an input x at each layer of the DNN and find the classes of each neighbor. This procedure allows the analysis of the evolution of classes in the neighborhood of x throughout the network. That is why, the authors introduce several metrics such as *nonconformity* and *credibility*. The nonconformity metric is used to measure the discrepancy between the labels of the neighborhood and the predicted label of an input x. A high nonconformity value means that the labels predicted for the neighborhood of x is different from the one predicted for x. The computation of the credibility measure is based on a "calibration dataset" whose examples were not used for the training. Then, the credibility of an input x is the ratio of nonconformity measures of the examples from the calibration dataset that are greater than the input's nonconformity measure. DKNN increases the *confidence* in predictions thanks to the *credibility measure* which is used to assess and select the model's predictions. Moreover, this algorithm is more interpretable, because the generated neighborhood provides some insight of how the model works. Papernot and McDaniel [27] claim that DKNN would prevent adversarial examples by assigning to them a low credibiility measure. Their experiments show encouraging results against the attack proposed by Carlini and Wagner [66].

The drawback of the above techniques lies on their dependence on the ℓ_p-norm. Indeed, even if it brings robustness against attacks relying on the same metric distance, the defense could become ineffective when the attacks depend on a different metric distance. Zantedeschi et al. [70] propose to consider all the local perturbations (drawn from a gaussian distribution) around each examples and thus avoiding the use of ℓ_p-norm:

$$\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \mathop{\mathbb{E}}_{\delta\sim\mathcal{N}(0,\sigma^2)} L(f_\theta(x+\delta), y),$$

where $\mathcal{D}$ is the unknown distribution of the dataset, f_θ is the model whose parameters are θ and L is the loss function of the model. They claim that their method, known as Gaussian Data Augmentation (GDA), instead of crafting adversarial example only in the direction of the gradient, explore many more directions around the examples. GDA outperforms AT considering adversarial accuracy measure (*i.e.* accuracy on an attacked dataset) on MNIST [74] and CIFAR-10 [75].

[§§]PMP is used in optimal control theory to find the best solution regarding input and constraint.

A hypothesis that could explain the adversarial phenomenon is the high expressiveness of the models, *i.e.* the the capacity of the model to learn complex behavior. Cissé et al. [30] and Tsuzuku et al. [39] explore this lead by constraining the Lipschitz constant of neural networks to enhance their robustness. Considering all possible pairs of points of the input space of a function f, the Lipschitz constant is the smallest value that upperbounds the absolute value of the slopes given by each pair of points. Intuitively, constraining the Lipschitz constant of a function f to a small value would smooth the function and thus reduce its expressiveness. Cissé et al. [30] develop specific training called PARSEVAL TRAINING which consists in ensuring that the Lipschitz constant of all the network's layers is smaller than 1 by having Parseval tight weight's matrices. We refer interested readers to the paper [30] for the mathematical details. Tsuzuku et al. [39] propose another training method, called Lipschitz Margin Training (LMT), constraining the Lipschitz constant of a network. They first link the margin of network to the Lipschitz constant. From that, they derive an algorithm relying on the relation between the margin and the Lipschitz constant that enhances the robustness of the network. In addition, LMT provide a "certified" lower bound on the smallest perturbation that can defeat the network. This latter property could be desirable for the demonstration of conformity to the regulation requirement of an embedded system based on ML. Indeed, theoretical results are great assets for the software aspect of certification process. For example, Gourdeau et al. [71] leverage the **PAC theory** (Probably Approximately Correct) to provide theoretical proof of the feasibility of robust learning from the perspective of computational learning theory. The authors focus on the setting where the input space is the boolean hypercube $\mathcal{X} = \{0, 1\}^n$ and show whether robust learning is feasible for different classes of models.

From a methodological standpoint, Carlini et al. [21] provide advice and good practices about the evaluation of the adversarial robustness. Especially, they claim that one that develops a new defense mechanism must think about an "adaptive attack" to evaluate the efficiency of their defense mechanism. In other words, to evaluate the defense efficiency of your new defense mechanism, you have to test the worst attack against it. Gilmer et al. [22] conduct experiments showing that improving adversarial robustness also improve corruption robustness, *i.e.* the robustness of the model against distributional shift. The goal is to show that both type of robustness is the manifestation of the same phenomenon. Nevertheless, it is encouraging that improving adversarial robustness has a positive impact on corruption robustness.

Robustness verification is part of the objectives stated in the first guidance delivered by the EASA [5, 6]: the methods seen in this section improve the robustness of ML models and hence, help to satisfy the objective stated by the EASA. Though those methods might be efficient and are a good first step, they do not provide any guarantee of robustness.

C. Provability

In the context of ML item demonstration of conformity, provability will help to demonstrate that the model preserves its properties. There are frameworks, denoted as verifiers, that can verify any types of properties as long as they are expressible in the format expected by the verification framework. Some of them were initially developed to verify robustness.

Table 3 Summary of verification methods. The column "# parameters" reports the size of the neural network in terms of numbers of parameters used in the papers; it give an insight on the scalability of the associated method.

	Verification			
Method Name	Papers	Sound	Complete	# paramaters
AI^2	Gehr et al. [31]	✓		> 238,000
DLV	Huang et al. [32]	✓	✓	> 138,000,000
RELUPLEX	Katz et al. [33]	✓	✓	~ 13,000
MARABOU	Katz et al. [41]	✓	✓	~ 13,000
REFINEZONO	Singh et al. [37]	✓		> 1,000,000
RELUVAL	Wang et al. [40]	✓		~ 670, 000
PRIMA	Müller et al. [76]	✓		> 250,000
CNN-ABS	Ostrovsky et al. [77]	✓	✓	~ 850
DEEPCERT	Paterson et al. [78]	✓	✓	> 1, 600, 000
β-CROWN BAB	Wang et al. [79]	✓	✓	> 210, 000
LIRPA	Xu et al. [80]	✓		> 19, 000, 000
MIP VERIFY	Tjeng et al. [38]	✓	✓	> 4, 000, 000

Completeness and *soundness* are two important criteria for a verifier. *Completeness* means that the verifier can decide the validity of all properties[¶¶] that hold, whereas *soundness* means that the verifier cannot prove any wrong property. All the methods presented here are sound, otherwise we could not obtain any guarantee from their outputs. Complete methods suffer from scalability issues while incomplete one suffer from the lack of precision. Despite these drawbacks, formal methods are a worthy asset, as they provide proofs and may save testing time.

Katz et al. [33] develop a method that uses the Simplex algorithm –a method to solve optimization problem of linear programming– and Satisfiability Modulo Theories (SMT) solvers –solvers for formulas of first-order logic with respect to some background theories such as arithmetic, arrays, etc.– which the authors adapt to work with Neural Network using the ReLU[***] activation function ($y = max(0, x)$). Their method is called RELUPLEX and the authors assess its performance on the ACAS XU case study [81]. The framework uses quantifier-free formulas to the formalization of properties which consists of constrained domains for inputs and outputs. The verification problem is reduced to a constraint satisfiability problem (CSP). The paper presents ten properties for which verification with RELUPLEX is attempted, two of them terminating with a timeout, *i.e.* RELUPLEX was not able to end the verification within the given time. The longest verification took almost five days while the fastest took less than eight minutes. MARABOU [41], which is the successor of RELUPLEX, is based on the same principles. The framework, now supports piecewise linear layer and activation and has a "divide and conquer mode" which improves the computation time of the verification.

There exists a large number of verifiers in the literature [63], each coming with its own specificity. Shriver et al. [82] highlights that this may be a burden for users who want to compare verifiers and proposes a framework for Deep Neural Network Verification (DNNV) aiming to ease the use of verifiers and benchmarks.

Many verification works described below use robustness properties as benchmark or application, but most of them

[¶¶] that is, all properties expressible in the logical formalism supported by the verifier

[***] ReLU stands for Rectified Linear Unit

are not restricted to robustness properties. Robustness verification is an emerging field [30–32, 37, 39, 76–80, 82]. Verifying robustness means ensuring that the algorithm outputs the same label y for an example x and its neighborhood whose the computation usually relies on a metric distance (*e.g.*ℓ_p-norm). This property could be stated as follow:

$$\forall x' \text{ s.t. } \|x - x'\|_p \leq \epsilon, \quad f(x) = f(x'), \tag{6}$$

where $\|x - x'\|_p \leq \epsilon$ represent the neighborhood around x such that the distance between x and any neighbor x' does not exceed ϵ and f is the ML model. Indeed, the existence of adversarial examples could be seen as the negation of Equation 6. Thus, verifying that Equation 6 holds boils down to check the adversarial robustness of a model.

Tjeng et al. [38] propose MIP VERIFYa tool that use Mixed Integer Linear Programming (MILP) Solver in order to check the adversarial robustness of ML model. The principle is to express ML model as linear program comprising the encoding of the robustness property and the solution of the linear program indicates whether the property holds. The encoding of the activation function and the bounds on each neuron determine the efficiency of such method. MIP VERIFY uses an asymmetric formulation for the activation function and includes a procedure named progressive bound tightening, which seems to improve the running time [38]. Singh et al. [37] develop REFINEZONO, a verification framework mixing optimization techniques (MILP, LP/MILP relaxation) and abstract interpretation – a method mainly used in static analysis that leverages overappromixation to analyse the behavior of computer programs. Mixing those techniques allows better scalability for complete verifiers and improves the precision of incomplete ones. The principle of REFINEZONO is to compute the bounds of neuron outputs using optimization techniques and abstract interpretation. As in [33], a property is expressed as domains for inputs and outputs. A robustness property, expressed as *same label predicted within a region around* x, can be easily transformed into domains for input and output: the region around x is the domain for the input and the domain for the output is given to have the right class. For example, Figure 5 shows a robustness property verifying that the model outputs only the desired class within the neighborhood of $x = (0.9, 0.7, 0.4)$. Ostrovsky et al. [77] propose to improve scalability of complete verifiers by using an abstraction-refinement approach and propose a framework named CNN-ABS. The principle is to find sound over-approximation of the the original network (abstraction) to perform the verification on: it corresponds to a smaller network created by removing neurons from the original network so that if the property holds for the smaller network it also holds for the original one. However, due to the abstraction, the verifier may output a spurious counter-example, *i.e.* a counter example for the smaller network but not for the original one. In this case, the smaller network is refined by adding a subset of removed neurons before running again the verification. At a higher level, CNN-ABS alternate between abstraction and refinement step until reaching the level of abstraction that allows it to output a correct answer.

Instead of trying to improve scalability of complete verifiers, such as MILP with abstract interpretation [37], it is possible to make abstract interpretation as precise as possible [31]. The benefit of applying only abstract interpretation is to verify properties on large neural networks (e.g. CNN). Gehr et al. [31] develop the AI^2 framework which implements abstract interpretation techniques to verify neural networks. Their framework also focus on the verification of the

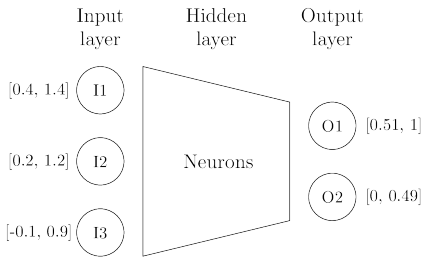

Fig. 5 Robustness property example - We consider a hypercube (of size 1) as a neighborhood around $x = (0.9, 0.7, 0.4)$ and the output must always be the first class, *i.e.* the lower bound of the output neuron O1 must be greater than the upper bound of the neuron O2.

robustness property (Equation 6). In AI^2[31], the input is replaced by an abstract domain then it is propagated through the network thanks to abstract transformers until the output layer. Property verification is applied to the abstract domain obtained at the output layer. The method is sound but incomplete: the verifier output can either be true or inconclusive. Hence, the choice of the abstract domain for the input is important because it influences the precision of the verifier.

Most incomplete techniques approximate the activation function neuron wisely [31, 37, 40, 79, 80]. Several works state that "multiple neurons over-approximation" may tighten the precision of the techniques [36, 76, 83–85]. Particularly, Müller et al. [76] propose PRIMA, a framework that leverage convex hull approximation of convex polyhedra to compute the abstraction of several neurons. The method used in PRIMA allows tighter abstraction and thus managed to outperform the other incomplete method regarding the number of inconclusive answer.

Huang et al. [32] reduce the proof of robustness to a search of adversarial examples. They focus on the robustness of image classification which is quite a hard problem considering high dimension of an image and all possible perturbations. The first assumption is that *images are discrete*. Hence, Huang et al. [32] explore completely the region of an image x, searching for an adversarial example, by defining a set of *minimum* manipulations. To develop a verification procedure that ensures the safety of a point, the authors first formally define a *safe point*. Based on their assumption of "discrete images", the authors develop a sound and complete method using SMT solvers, named Deep Learning Verification (DLV), for *safety verification*. Then, if the algorithm does not find any adversarial example, it means that there are none for the region and the manipulation set considered. Moreover, Huang et al. [32] show that their algorithm is scalable to big neural networks by doing experiments on neural networks with more than one hundred million parameters.

Xu et al. [80] provide a framework, named LIRPA, that unifies the methods of linear relaxation whose goal is to find affine functions that upperbound and lowerbound the output of the network within a sub-domain of the input. For example, to prove a robustness property (as stated in Equation 6) with LIRPA, one must show that the lower bound of the true label is greater than all upper bounds of all other classes; this would mean that even in the worst case, the model will always output the correct class. More recently, Wang et al. [79] propose a method called β-CROWN also based on linear relaxation that gives more tight bounds than other existing linear relaxation-based methods. The novelty of [79]

stands in the formulation of the bound which relies on Lagrange functions and ReLU splits encoding. As is, β-CROWN is an incomplete method. However, Wang et al. [79] add a Branch and Bound mechanism to β-CROWN, namely β-CROWN BAB, in order to make it complete and show through experiments that β-CROWN BAB is time efficient.

The adversarial robustness verifiers mentioned above are promising for ensuring the absence of unintended behavior of the ML model but these guarantees hold for small perturbations which are not necessarily meaningful. In the aeronautical context, for use-cases dealing with images, robustness against meaningful perturbation, such as weather conditions, might be relevant. [78] takes a step towards it and propose DEEPCERT, a tool verifying meaningful perturbation, namely "contextually relevant perturbations". DEEPCERTencodes three perturbations: blur, haze and contrast variation.

Another side of *provability* domain deals with generalization bounds. To obtain these generalization bounds, the theory rely on metrics that measure the model complexity such as the VC-dimension [45], the Rademacher complexity [46] or the KL-divergence with the PAC[†††]-Bayesian framework [43, 44]. Particularly, it has been shown that the PAC-Bayesian theory is applicable to neural networks [49] and is also able to provide tight bounds[47].

Generalization bounds provide guarantees on the performance of ML model on unseen data. Recent works empowered the idea of giving guarantees on the performance of the ML model under adversarial attacks, *i.e.* generalization bounds on the so-called adversarial risk [86–89]. As for the classical bounds, some of them are relying on the rademacher complexity [86, 87], the VC-dimension [88] or the KL-divergence using the PAC-bayesian theory [89].

Robustness verification as well as generalization guarantees are objectives stated in the first guidance of the EASA [5, 6]. We saw in this section methods that could be means to reach both of these objectives.

V. Conclusion

In this paper, we have investigated the compatibility between recent advances in ML and established qualification procedures used in civil avionic software. ML proves to be an interesting approach for developing functionalities that are difficult or impossible to implement using classical programming, such as vision-based navigation (see for example the ATTOL project from Airbus[‡‡‡]). This technology opens the way to advanced features in autonomous flight systems, and this needs to be done while keeping a high level of system safety. However, the intrinsic characteristics of ML-based software, such as the use of alternative requirements specification techniques or the difficult traceability of the results, seriously challenge established software aspect of certification procedures. In this work, we consider that the ML function is hosted in a single item; however current discussion at the EUROCAE WG-114 implies that a ML function could be hosted on multiple items, meaning that the ML function development would then be between the system level and the item level. Nevertheless, the issues and potential solutions stated in this work remains valid even when considering this change of context.

[†††] Probably Approximately Correct

[‡‡‡] https://www.airbus.com/en/newsroom/press-releases/2020-01-airbus-demonstrates-first-fully-automatic-vision-based-take-off

As a first step towards the demonstration of conformity of embedded software using ML, we contribute by (i) establishing a taxonomy of the involved key domains, (ii) providing a review of the literature, focused on explainability, adversarial robustness and provability and (iii) highlighting how these methods will eventually support the compliance to regulation requirements.

We address the safety issue through adversarial robustness, which is a problem raised by the data-driven paradigm and particularly the well-known neural networks, hence inducing the necessary management of unintended system behavior. With *explainability* and *provability*, we address the *trustworthiness* issue in ML algorithms which are still suffering from their black-box aspect as well as their lack of guarantees. For these three domains, our review of the state of the art shows promising methods that address those issues. It is interesting to notice that most of verification methods (cf. Section IV.C) are based on formal methods that have already been used for the demonstration of conformity in the aviation domain. Assuming that verification is the most time and cost consuming part of the development process, formal methods are promising means to handle the robustness aspects because they provide mathematical evidence and avoid a huge amount of empirical verification. The ultimate goal of qualifying embedded ML-based software, will require addressing all the various issues identified in Figure 2 and thus demands further research in all these domains.

References

[1] Krodel, J., "Technology Changes In Aeronautical Systems," *Proceeding of the 4th European Congress on Embedded Real Time Software (ERTS)*, 2008. URL `https://hal.science/ERTS2008/hal-02269834v1`.

[2] Bahdanau, D., Cho, K., and Bengio, Y., "Neural Machine Translation by Jointly Learning to Align and Translate," *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, edited by Y. Bengio and Y. LeCun, arXiv e-prints, 2015, p. arXiv:1409.0473. https://doi.org/10.48550/arXiv.1409.0473.

[3] Devlin, J., Chang, M., Lee, K., and Toutanova, K., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *NAACL-HLT*, Association for Computational Linguistics, 2019, pp. 4171–4186. https://doi.org/10.18653/V1/N19-1423.

[4] Redmon, J., Divvala, S. K., Girshick, R. B., and Farhadi, A., "You Only Look Once: Unified, Real-Time Object Detection," *CVPR*, IEEE Computer Society, 2016, pp. 779–788. https://doi.org/10.1109/CVPR.2016.91.

[5] EASA, *EASA Concept Paper: First usable guidance for Level 1 machine learning applications*, 2021. URL https://www.easa.europa.eu/downloads/134357/en.

[6] EASA, *EASA Concept Paper: First usable guidance for Level 1& 2 machine learning applications*, 2023. URL https://www.easa.europa.eu/en/document-library/general-publications/easa-artificial-intelligence-concept-paper-proposed-issue-2#group-easa-downloads.

[7] EASA, *CS-25 Amendment 22*, 2018. URL https://www.easa.europa.eu/sites/default/files/dfu/CS-25%20Amendment%2022.pdf.

[8] Winter, J., "System Integration Testing for Unintended Behaviors in Flight-Critical Aerospace Applications," *2020 IEEE International Systems Conference (SysCon)*, 2020, pp. 1–5. https://doi.org/10.1109/SysCon47679.2020.9275658.

[9] Phillips, P., Hahn, A., Fontana, P., Broniatowski, D., and Przybocki, M., "Four Principles of Explainable Artificial Intelligence (Draft)," , 2020-08-18 2020. https://doi.org/10.6028/NIST.IR.8312-draft.

[10] Guidotti, R., Monreale, A., Ruggieri, S., Pedreschi, D., Turini, F., and Giannotti, F., "Local Rule-Based Explanations of Black Box Decision Systems," , 2018. https://doi.org/10.48550/arXiv.1805.10820.

[11] Ribeiro, M. T., Singh, S., and Guestrin, C., ""Why Should I Trust You?": Explaining the Predictions of Any Classifier," *SIGKDD*, ACM, 2016, pp. 1135–1144. https://doi.org/10.1145/2939672.2939778.

[12] Ribeiro, M. T., Singh, S., and Guestrin, C., "Anchors: High-Precision Model-Agnostic Explanations," *AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, AAAI Press, 2018, pp. 1527–1535. URL https://dl.acm.org/doi/abs/10.5555/3504035.3504222.

[13] Dabkowski, P., and Gal, Y., "Real Time Image Saliency for Black Box Classifiers," *NeurIPS*, Curran Associates Inc. Red Hook, NY, United States, 2017, pp. 9525–9536. URL https://dl.acm.org/doi/10.5555/3295222.3295440.

[14] Fong, R. C., and Vedaldi, A., "Interpretable Explanations of Black Boxes by Meaningful Perturbation," *ICCV*, IEEE Computer Society, 2017, pp. 3449–3457. https://doi.org/10.1109/ICCV.2017.371.

[15] Hendricks, L. A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., and Darrell, T., "Generating Visual Explanations," *ECCV*, Lecture Notes in Computer Science, Vol. 9908, Springer, 2016, pp. 3–19. https://doi.org/10.1007/978-3-319-46493-0_1.

[16] Kendall, A., and Gal, Y., "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" *NeurIPS*, Curran Associates Inc. Red Hook, NY, United States, 2017, pp. 5574–5584. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/2650d6089a6d640c5e85b2b88265dc2b-Paper.pdf.

[17] Zeiler, M. D., and Fergus, R., "Visualizing and Understanding Convolutional Networks," *ECCV*, Lecture Notes in Computer Science, Vol. 8689, Springer, 2014, pp. 818–833. https://doi.org/10.1007/978-3-319-10590-1_53.

[18] Lundberg, S. M., and Lee, S., "A Unified Approach to Interpreting Model Predictions," *NeurIPS*, Curran Associates Inc. Red Hook, NY, United States, 2017, pp. 4765–4774. URL https://papers.nips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html.

[19] Shrikumar, A., Greenside, P., and Kundaje, A., "Learning Important Features Through Propagating Activation Differences," *ICML*, Proceedings of Machine Learning Research, Vol. 70, JMLR.org, 2017, pp. 3145–3153. URL https://dl.acm.org/doi/10.5555/3305890.3306006.

[20] Zhao, Q., and Hastie, T., "Causal Interpretations of Black-Box Models," *Journal of Business & Economic Statistics*, Vol. 39, No. 1, 2021, pp. 272–281. https://doi.org/10.1080/07350015.2019.1624293.

[21] Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I. J., Madry, A., and Kurakin, A., "On Evaluating Adversarial Robustness," *CoRR*, Vol. abs/1902.06705, 2019. https://doi.org/10.48550/arXiv.1902.06705.

[22] Gilmer, J., Ford, N., Carlini, N., and Cubuk, E. D., "Adversarial Examples Are a Natural Consequence of Test Error in Noise," *ICML*, Proceedings of Machine Learning Research, Vol. 97, PMLR, 2019, pp. 2280–2289. URL https://proceedings.mlr.press/v97/gilmer19a.html.

[23] Goodfellow, I. J., Shlens, J., and Szegedy, C., "Explaining and Harnessing Adversarial Examples," , 2015. https://doi.org/10.48550/arXiv.1412.6572.

[24] Kurakin, A., Goodfellow, I. J., and Bengio, S., "Adversarial Machine Learning at Scale," 2017. https://doi.org/10.48550/arXiv.1611.01236.

[25] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A., "Towards Deep Learning Models Resistant to Adversarial Attacks," 2018. https://doi.org/10.48550/arXiv.1706.06083.

[26] Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A., "The Limitations of Deep Learning in Adversarial Settings," *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2016, pp. 372–387. https://doi.org/10.1109/EuroSP.2016.36.

[27] Papernot, N., and McDaniel, P., "Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning," , 2018. https://doi.org/10.48550/arXiv.1803.04765.

[28] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R., "Intriguing properties of neural networks," 2014. https://doi.org/10.48550/arXiv.1312.6199.

[29] Zhang, D., Zhang, T., Lu, Y., Zhu, Z., and Dong, B., "You Only Propagate Once: Accelerating Adversarial Training via Maximal Principle," *NeurIPS*, Curran Associates Inc. Red Hook, NY, United States, 2019, pp. 227–238. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf.

[30] Cissé, M., Bojanowski, P., Grave, E., Dauphin, Y. N., and Usunier, N., "Parseval Networks: Improving Robustness to Adversarial Examples," *ICML*, Proceedings of Machine Learning Research, Vol. 70, PMLR, 2017, pp. 854–863. URL https://dl.acm.org/doi/10.5555/3305381.3305470.

[31] Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., and Vechev, M. T., "AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation," *IEEE SP*, IEEE Computer Society, 2018, pp. 3–18. https://doi.org/https://doi.org/10.1109/SP.2018.00058.

[32] Huang, X., Kwiatkowska, M., Wang, S., and Wu, M., "Safety Verification of Deep Neural Networks," *CAV*, Lecture Notes in Computer Science, Vol. 10426, Springer, 2017, pp. 3–29. https://doi.org/https://doi.org/10.1007/978-3-319-63387-9_1.

[33] Katz, G., Barrett, C. W., Dill, D. L., Julian, K., and Kochenderfer, M. J., "Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks," *CAV*, Lecture Notes in Computer Science, Vol. 10426, Springer, 2017, pp. 97–117. https://doi.org/https://doi.org/10.1007/978-3-319-63387-9_5.

[34] Koh, P. W., and Liang, P., "Understanding Black-box Predictions via Influence Functions," *ICML*, Proceedings of Machine Learning Research, Vol. 70, PMLR, 2017, pp. 1885–1894. URL https://dl.acm.org/doi/10.5555/3305381.3305576.

[35] Mirman, M., Gehr, T., and Vechev, M. T., "Differentiable Abstract Interpretation for Provably Robust Neural Networks," *ICML*, Proceedings of Machine Learning Research, Vol. 80, PMLR, 2018, pp. 3575–3583. URL https://proceedings.mlr.press/v80/mirman18b/mirman18b.pdf.

[36] Salman, H., Yang, G., Zhang, H., Hsieh, C., and Zhang, P., "A Convex Relaxation Barrier to Tight Robustness Verification of Neural Networks," *NeurIPS*, Curran Associates Inc. Red Hook, NY, United States, 2019, pp. 9832–9842. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/246a3c5544feb054f3ea718f61adfa16-Paper.pdf.

[37] Singh, G., Gehr, T., Püschel, M., and Vechev, M., "Boosting Robustness Certification of Neural Networks," *ICLR*, OpenReview.net, 2019. URL https://openreview.net/forum?id=HJgeEh09KQ.

[38] Tjeng, V., Xiao, K. Y., and Tedrake, R., "Evaluating Robustness of Neural Networks with Mixed Integer Programming," *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, OpenReview.net, 2019. URL https://openreview.net/forum?id=HyGIdiRqtm.

[39] Tsuzuku, Y., Sato, I., and Sugiyama, M., "Lipschitz-Margin Training: Scalable Certification of Perturbation Invariance for Deep Neural Networks," *NeurIPS*, Curran Associates Inc. Red Hook, NY, United States, 2018, pp. 6542–6551. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/485843481a7edacbfce101ecb1e4d2a8-Paper.pdf.

[40] Wang, S., Pei, K., Whitehouse, J., Yang, J., and Jana, S., "Formal Security Analysis of Neural Networks using Symbolic Intervals," *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, USENIX Association, 2018, pp. 1599–1614. URL https://www.usenix.org/conference/usenixsecurity18/presentation/wang-shiqi.

[41] Katz, G., Huang, D. A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljic, A., Dill, D. L., Kochenderfer, M. J., and Barrett, C. W., "The Marabou Framework for Verification and Analysis of Deep Neural Networks," *CAV*, Lecture Notes in Computer Science, Vol. 11561, Springer, 2019, pp. 443–452. https://doi.org/10.1007/978-3-030-25540-4_26.

[42] Valiant, L. G., "A Theory of the Learnable," *Commun. ACM*, Vol. 27, No. 11, 1984, p. 1134–1142. https://doi.org/10.1145/1968.1972.

[43] McAllester, D. A., "Some PAC-Bayesian Theorems," *Mach. Learn.*, Vol. 37, No. 3, 1999, pp. 355–363. https://doi.org/10.1023/A:1007618624809.

[44] Shawe-Taylor, J., and Williamson, R. C., "A PAC Analysis of a Bayesian Estimator," *COLT*, ACM, 1997, pp. 2–9. https://doi.org/10.1145/267460.267466.

[45] Vapnik, V. N., and Chervonenkis, A. Y., "On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities," *Theory of Probability & Its Applications*, Vol. 16, No. 2, 1971, pp. 264–280. https://doi.org/10.1137/1116025.

[46] Bartlett, P. L., and Mendelson, S., "Rademacher and Gaussian Complexities: Risk Bounds and Structural Results," *J. Mach. Learn. Res.*, Vol. 3, 2002, pp. 463–482. URL http://jmlr.org/papers/v3/bartlett02a.html.

[47] Parrado-Hernández, E., Ambroladze, A., Shawe-Taylor, J., and Sun, S., "PAC-Bayes Bounds with Data Dependent Priors," *Journal of Machine Learning Research*, Vol. 13, No. 112, 2012, pp. 3507–3531. URL https://www.jmlr.org/papers/volume13/parrado12a/parrado12a.pdf.

[48] Germain, P., Lacasse, A., Laviolette, F., Marchand, M., and Roy, J., "Risk Bounds for the Majority Vote: From a PAC-Bayesian Analysis to a Learning Algorithm," *J. Mach. Learn. Res.*, Vol. 16, 2015, pp. 787–860. https://doi.org/10.5555/2789272.2831140, URL http://jmlr.org/papers/v16/germain15a.html.

[49] Dziugaite, G. K., and Roy, D. M., "Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data," *UAI*, AUAI Press, 2017. URL http://auai.org/uai2017/proceedings/papers/173.pdf.

[50] Masegosa, A., Lorenzen, S., Igel, C., and Seldin, Y., "Second Order PAC-Bayesian Bounds for the Weighted Majority Vote," *NeurIPS*, Vol. 33, Curran Associates Inc. Red Hook, NY, United States, 2020, pp. 5263–5273. URL https://proceedings.neurips.cc/paper/2020/file/386854131f58a556343e056f03626e00-Paper.pdf.

[51] Cai, L., and Zhu, Y., "The Challenges of Data Quality and Data Quality Assessment in the Big Data Era," *Data Science Journal*, Vol. 14, 2015. https://doi.org/10.5334/dsj-2015-002.

[52] Picard, S., Chapdelaine, C., Cappi, C., Gardes, L., Jenn, E., Lefèvre, B., and Soumarmon, T., "Ensuring Dataset Quality for Machine Learning Certification," *2020 IEEE International Symposium on Software Reliability Engineering Workshops, ISSRE Workshops, Coimbra, Portugal, October 12-15, 2020*, IEEE, 2020, pp. 275–282. https://doi.org/10.1109/ISSREW51248.2020.00085.

[53] Baylor, D., Breck, E., Cheng, H., Fiedel, N., Foo, C. Y., Haque, Z., Haykal, S., Ispir, M., Jain, V., Koc, L., Koo, C. Y., Lew, L., Mewald, C., Modi, A. N., Polyzotis, N., Ramesh, S., Roy, S., Whang, S. E., Wicke, M., Wilkiewicz, J., Zhang, X., and Zinkevich, M., "TFX: A TensorFlow-Based Production-Scale Machine Learning Platform," *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, ACM, 2017, pp. 1387–1395. https://doi.org/10.1145/3097983.3098021.

[54] Ashmore, R., Calinescu, R., and Paterson, C., "Assuring the Machine Learning Lifecycle: Desiderata, Methods, and Challenges," *ACM Comput. Surv.*, Vol. 54, No. 5, 2021. https://doi.org/10.1145/3453444.

[55] Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H. C., Kamar, E., Nagappan, N., Nushi, B., and Zimmermann, T., "Software Engineering for Machine Learning: A Case Study," *Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice, ICSE (SEIP) 2019, Montreal, QC, Canada, May 25-31, 2019*, edited by H. Sharp and M. Whalen, IEEE / ACM, 2019, pp. 291–300. https://doi.org/10.1109/ICSE-SEIP.2019.00042.

[56] Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., and Müller, K.-R., "Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology," *Machine Learning and Knowledge Extraction*, Vol. 3, No. 2, 2021, pp. 392–413. https://doi.org/10.3390/make3020020.

[57] Rushby, J., "The Interpretation and Evaluation of Assurance Cases," Tech. Rep. SRI-CSL-15-01, Computer Science Laboratory, SRI International, Menlo Park, CA, 2015. URL http://www.csl.sri.com/users/rushby/papers/sri-csl-15-1-assurance-cases.pdf.

[58] Martin, R., "Assured Software - A journey and discussion," , 2017. URL https://www.his-2019.co.uk/session/cwe-cve-its-history-and-future.

[59] Damour, M., Grancey, F. D., Gabreau, C., Gauffriau, A., Ginestet, J., Hervieu, A., Huraux, T., Pagetti, C., Ponsolle, L., and Clavière, A., "Towards Certification of a Reduced Footprint ACAS-Xu System: A Hybrid ML-Based Solution," *SAFECOMP*, Vol. 12852, Springer, 2021, pp. 34–48. https://doi.org/10.1007/978-3-030-83903-1_3.

[60] Huang, X., Kroening, D., Ruan, W., Sharp, J., Sun, Y., Thamo, E., Wu, M., and Yi, X., "A Survey of Safety and Trustworthiness of Deep Neural Networks: Verification, Testing, Adversarial Attack and Defence, and Interpretability," *Computer Science Review*, Vol. 37, 2020, pp. 100–270. https://doi.org/10.1016/J.COSREV.2020.100270.

[61] Biggio, B., and Roli, F., "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recogn.*, Vol. 84, No. C, 2018, p. 317–331. https://doi.org/10.1016/j.patcog.2018.07.023.

[62] Ren, K., Zheng, T., Qin, Z., and Liu, X., "Adversarial Attacks and Defenses in Deep Learning," *Engineering*, Vol. 6, No. 3, 2020, pp. 346 – 360. https://doi.org/10.1016/j.eng.2019.12.012.

[63] Urban, C., and Miné, A., "A Review of Formal Methods applied to Machine Learning," , 2021. https://doi.org/10.48550/arXiv.2104.02466.

[64] Garreau, D., and von Luxburg, U., "Explaining the Explainer: A First Theoretical Analysis of LIME," *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, Proceedings of Machine Learning Research, Vol. 108, PMLR, 2020, pp. 1287–1296. URL http://proceedings.mlr.press/v108/garreau20a.html.

[65] Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I. J., Hardt, M., and Kim, B., "Sanity Checks for Saliency Maps," *NeurIPS*, 2018. URL https://papers.nips.cc/paper_files/paper/2018/hash/294a8ed24b1ad22ec2e7efea049b8737-Abstract.html.

[66] Carlini, N., and Wagner, D., "Towards Evaluating the Robustness of Neural Networks," *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 39–57. https://doi.org/10.1109/SP.2017.49.

[67] Huang, Q., Katsman, I., Gu, Z., He, H., Belongie, S., and Lim, S., "Enhancing Adversarial Example Transferability With an Intermediate Level Attack," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, IEEE Computer Society, Los Alamitos, CA, USA, 2019, pp. 4732–4741. https://doi.org/10.1109/ICCV.2019.00483, URL https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00483.

[68] Li, Q., Guo, Y., and Chen, H., "Practical No-box Adversarial Attacks against DNNs," *Advances in Neural Information Processing Systems*, Vol. 33, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Curran Associates Inc. Red Hook, NY, United States, 2020, pp. 12849–12860. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/96e07156db854ca7b00b5df21716b0c6-Paper.pdf.

[69] Moosavi-Dezfooli, S., Fawzi, A., and Frossard, P., "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Computer Society, Los Alamitos, CA, USA, 2016, pp. 2574–2582. https://doi.org/10.1109/CVPR.2016.282.

[70] Zantedeschi, V., Nicolae, M.-I., and Rawat, A., "Efficient Defenses Against Adversarial Attacks," *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, Association for Computing Machinery, New York, NY, USA, 2017, p. 39–49. https://doi.org/10.1145/3128572.3140449.

[71] Gourdeau, P., Kanade, V., Kwiatkowska, M., and Worrell, J., "On the Hardness of Robust Classification," *Advances in Neural Information Processing Systems*, Vol. 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Curran Associates Inc. Red Hook, NY, United States, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/8133415ea4647b6345849fb38311cf32-Paper.pdf.

[72] Kurakin, A., Goodfellow, I. J., and Bengio, S., "Adversarial examples in the physical world," *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, Chapman and Hall/CRC, 2017. URL https://openreview.net/forum?id=HJGU3Rodl.

[73] Chen, P.-Y., Zhang, H., Sharma, Y., Yi, J., and Hsieh, C.-J., "ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models," *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, Association for Computing Machinery, New York, NY, USA, 2017, p. 15–26. https://doi.org/10.1145/3128572.3140448.

[74] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, Vol. 86, No. 11, 1998, pp. 2278–2324. https://doi.org/10.1109/5.726791.

[75] Krizhevsky, A., "Learning Multiple Layers of Features from Tiny Images," *University of Toronto TR*, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[76] Müller, M. N., Makarchuk, G., Singh, G., Püschel, M., and Vechev, M., "PRIMA: general and precise neural network certification via scalable convex hull approximations," *Proc. ACM Program. Lang.*, Vol. 6, No. POPL, 2022. https://doi.org/10.1145/3498704.

[77] Ostrovsky, M., Barrett, C., and Katz, G., "An Abstraction-Refinement Approach to Verifying Convolutional Neural Networks," , 2022. https://doi.org/10.48550/arXiv.2201.01978.

[78] Paterson, C., Wu, H., Grese, J., Calinescu, R., Pasareanu, C. S., and Barrett, C. W., "DeepCert: Verification of Contextually Relevant Robustness for Neural Network Image Classifiers," *Computer Safety, Reliability, and Security - 40th International Conference, SAFECOMP 2021, York, UK, September 8-10, 2021, Proceedings*, Lecture Notes in Computer Science, Vol. 12852, edited by I. Habli, M. Sujan, and F. Bitsch, Springer, 2021, pp. 3–17. https://doi.org/10.1007/978-3-030-83903-1_5.

[79] Wang, S., Zhang, H., Xu, K., Lin, X., Jana, S., Hsieh, C.-J., and Kolter, J. Z., "Beta-CROWN: Efficient Bound Propagation with Per-neuron Split Constraints for Neural Network Robustness Verification," *Advances in Neural Information Processing Systems*, Vol. 34, edited by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Curran Associates Inc. Red Hook, NY, United States, 2021, pp. 29909–29921. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/fac7fead96dafceaf80c1daffeae82a4-Paper.pdf.

[80] Xu, K., Shi, Z., Zhang, H., Wang, Y., Chang, K.-W., Huang, M., Kailkhura, B., Lin, X., and Hsieh, C.-J., "Automatic Perturbation Analysis for Scalable Certified Robustness and Beyond," *Advances in Neural Information Processing Systems*, Vol. 33, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Curran Associates Inc. Red Hook, NY, United States, 2020, pp. 1129–1141. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/0cbc5671ae26f67871cb914d81ef8fc1-Paper.pdf.

[81] Manfredi, G., and Jestin, Y., "An introduction to ACAS Xu and the challenges ahead," *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, 2016, pp. 1–9. https://doi.org/10.1109/DASC.2016.7778055.

[82] Shriver, D., Elbaum, S., and Dwyer, M. B., "DNNV: A Framework for Deep Neural Network Verification," *Computer Aided Verification*, edited by A. Silva and K. R. M. Leino, Springer International Publishing, Cham, 2021, pp. 137–150. https://doi.org/10.1007/978-3-030-81685-8_6.

[83] Singh, G., Ganvir, R., Püschel, M., and Vechev, M., "Beyond the Single Neuron Convex Barrier for Neural Network Certification," *Advances in Neural Information Processing Systems*, Vol. 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Curran Associates Inc. Red Hook, NY, United States, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/0a9fdbb17feb6ccb7ec405cfb85222c4-Paper.pdf.

[84] Palma, A. D., Behl, H., Bunel, R. R., Torr, P., and Kumar, M. P., "Scaling the Convex Barrier with Active Sets," *International Conference on Learning Representations*, edited by arXiv e prints, 2021, p. arXiv:2101.05844. URL https://openreview.net/forum?id=uQfOy7LrlTR.

[85] Tjandraatmadja, C., Anderson, R., Huchette, J., Ma, W., PATEL, K. K., and Vielma, J. P., "The Convex Relaxation Barrier, Revisited: Tightened Single-Neuron Relaxations for Neural Network Verification," *Advances in Neural Information Processing Systems*, Vol. 33, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Curran Associates, Inc., 2020, pp. 21675–21686. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/f6c2a0c4b566bc99d596e58638e342b0-Paper.pdf.

[86] Khim, J., and Loh, P.-L., "Adversarial Risk Bounds via Function Transformation," , 2019. https://doi.org/10.48550/arXiv.1810.09519.

[87] Yin, D., Kannan, R., and Bartlett, P., "Rademacher Complexity for Adversarially Robust Generalization," *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 97, edited by K. Chaudhuri and R. Salakhutdinov, PMLR, 2019, pp. 7085–7094. URL https://proceedings.mlr.press/v97/yin19b.html.

[88] Montasser, O., Hanneke, S., and Srebro, N., "Reducing Adversarially Robust Learning to Non-Robust PAC Learning," *Advances in Neural Information Processing Systems*, Vol. 33, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin,

Curran Associates Inc. Red Hook, NY, United States, 2020, pp. 14626–14637. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/a822554e5403b1d370db84cfbc530503-Paper.pdf.

[89] Viallard, P., Vidot, E. G., Habrard, A., and Morvant, E., "A PAC-Bayes Analysis of Adversarial Robustness," *Advances in Neural Information Processing Systems*, Vol. 34, edited by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Curran Associates Inc. Red Hook, NY, United States, 2021, pp. 14421–14433. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/78e8dffe65a2898eef68a33b8db35b78-Paper.pdf.