Politecnico di Torino

Master of Science in Aerospace Engineering - Propulsion Systems

A.Y. 2025/2026

Graduation Session March 2026

Safety Process for AI-based Application: Regression Case

Advisor:
Manuela Battipede

Co-advisors:
Pietro Ottavio Lecis
Claudia Ranieri

Candidate:
Pietro Dalla Corte

Abstract

An ever-growing number of applications employ machine learning models for prediction, decision-making, or state estimation. This thesis proposes a specific certification process for machine learning-based systems performing regression tasks, in order to support future applications. The analysis is specially targeted to safety-critical systems, such as those involved in aviation, with the aim of showing compliance with the latest EASA safety requirements in regard to Artificial Intelligence (AI). The intent consists in providing a general framework that can be adapted to different contexts and, when possible, referred to the standard processes for traditional components, i.e., not AI-based, described in the EUROCAE/SAE guidelines ED-135/ARP4761A and ED-79B/ARP4754B. The core of the project is the definition of performance metrics requirements to ensure safety. The proposed assessment process is finally performed on a representative use case, in order to demonstrate its effective applicability: the Runway Alignment System (RAS), which embeds the Trajectory Change Predictor (TCP) machine learning constituent. Eventually, the current state and future expected developments in machine learning certification (ML) are discussed. The project was carried out in collaboration with Leonardo S.p.a. Helicopter Division.

Keywords: Machine Learning, Regression, Aviation, Safety, Certification

Table of Contents

List of Tables

List of Figures

Terminology

Certification

The legal recognition that a product complies with the applicable regulations[1].

Corner case

Relates to a situation that, considering at least two parameters of the AI/ML constituent ODD, occurs rarely on all of these parameters (i.e. low representation of the associated values in the distribution for those parameters)[2].

Data set (or dataset)

The sample of data used for various development phases of the model, i.e. the model training, the learning process verification, and the inference model verification.

- Training data set — Data that is input to an ML model in order to establish its behaviour.
- Validation data set— Used to tune a subset of the hyper-parameters of a model (e.g. number of hidden layers, learning rate, etc.).
- Test data set— Used to assess the performance of the model, independent of the training data set[3].

Development Assurance

All those planned and systematic actions used to substantiate, to an adequate level of confidence, that errors in requirements, design, and implementation have been identified and corrected such that the system satisfies the applicable certification basis[4].

---

[1]Source: ARP4761 Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment, 2023

[2]Source: adapted from (EASA Concept Paper, 2024)

[3]Source: adapted from (EASA Concept Paper, 2024)

[4]Source: adapted from (EASA Concept Paper, 2024)

Explainability

The EASA Concept Paper gives the following preliminary definition: capability to provide the human with understandable and relevant information on how an AI/ML application is coming to its results.

Failure

An occurrence which affects the operation of an aircraft, system, equipment, item, or piece-part such that it can no longer function as intended (this includes both loss of function and malfunction)[5].

Intended behaviour

Developed and expected functionalities, enabling meeting the objectives for which the ML/DL constituent is designed[6].

Machine Learning

Scientific field rooted in statistics and mathematical optimization that studies algorithms and mathematical models that aim at achieving artificial intelligence through learning from data. This data might consists of samples with labels (*supervised learning*), or without (*unsupervised learning*)[7].

Machine Learning Constituent

A defined and bounded set of either (one or more) hardware item(s) and/or software item(s) that implement one or more ML model(s) and the data processing required to execute the implemented ML Model(s).

Nominal (data)

Data points that are inside the ODD and are not inliers, singular points, or data points corresponding to edge cases or corner cases.

Reliability

The probability that an item will perform a required function under specified conditions, without failure, for a specified period of time[8].

---

[5]Source: ARP4761 Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment, 2023

[6]Source: adapted from (MLEAP, 2024)

[7]Source: adapted from (CoDANN I, 2020)

[8]Source: ARP4761 Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment, 2023

Robustness

The ability of the system to perform the intended behaviour in the presence of abnormal or unknown inputs and to provide an equivalent response within the neighbourhood of an input.

Acronyms

ADU Air Data Unit

AGL Above Ground Level

AFHA Aircraft Functional Hazard Assessment

AHRS Attitude & Heading Reference System

AI Artificial Intelligence

AL Assurance Level

AMC Acceptable Means of Compliance

aMOC Anticipated-Means of Compliance

ASA Aircraft Safety Assessment

ATM/ANS Air Traffic Management/Air Navigation Services

ATS Air Traffic Service

CEA Cascading Effects Analysis

CMA Common Mode Analysis

CoDANN Concepts of Design Assurance for Neural Networks

ConOps Concepts of Operations

CP Concept Paper

CS Certification Specification

CI Confidence Interval

DAL Development Assurance Level

DB Database

DBN Dynamic Bayesian Network

DD Dependence Diagrams

DL Deep Learning

EASA European Union Aviation Safety Agency

EMA Exponential Moving Average

EMS Emergency Medical Services

EUROCAE European Organization for Civil Aviation Equipment

ExR Exceedance Rate

EWMA Exponentially Weighted Moving Average

FAA Federal Aviation Administration

FDA Food and Drug Administration

FDAL Function Development Assurance Level

FF Functional Failure

FHA Functional Hazard Assessment

FMEA Failure Modes and Effects Analysis

FMES Failure Modes and Effects Summary

FMS Flight Management System

ForMuLA Formal Methods for Learning Assurance

FPGA Field Programmable Gate Arrays

FTA Fault Tree Analysis

GM Guidance Material

GNSS Global Navigation Satellite System

GPS Global Positioning System

GPU Graphic Processing Unit

GRU Gated Recurrent Unit

GS Ground Speed

H/C Helicopter

HAT Human-AI teaming

HW Hardware (see also H/W)

IDAL Item Development Assurance Level

ILS Instrument Landing System

IPC Innovation Partnership Contracts

IR Implementing Rules

IoU Intersection over Union

MA Markov Analysis

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

MCC Most Critical Condition

MBSA Model-Based Safety Analysis

MFD Multi-Function Display

ML Machine Learning

MLC Machine Learning Constituent

MLEAP Machine Learning Application Approval

MOC Means of Compliance

MOPS Minimum Operational Performance Standard

MSE Mean Squared Error

MVE Mean Variance Estimation

NASA National Aeronautics and Space Administration

NN Neural Network

OD Operational Domain

ODD Operational Design Domain

OOD Out of Distribution

OS Operating Space

PAC Probably Approximately Correct

PASA Preliminary Aircraft Safety Assessment

PCA Principal Component Analysis

PBN Performance-based Navigation

PFD Primary Flight Display

PI Prediction Interval

PRA Particular Risk Analysis

PSSA Preliminary System Safety Assessment

QFU Magnetic Orientation of Runway

RA Radar altitude

RAE Relative Absolute Error

RadAlt Radar-altimeter

RAS Runway Alignment System

RMSE Root Mean Squared Error

RNN Recurrent Neural Network

RNP Required Navigation Performance

RTCA Radio Technical Commission for Aeronautics

SAE Society of Automotive Engineers

SFHA System Functional Hazard Assessment

SSA System Safety Assessment

SW Software (see also S/W)

SWAL Software Assurance Level

TCP Trajectory Change Predictor

UQ Uncertainty Quantification

VFR Visual Flight Rules

VLS Visual Landing System

VMC Visual Meteorological Conditions

VS Vertical Speed

ZSA Zonal Safety Analysis

1 | Introduction

1.1 Background

Artificial intelligence (AI) represents a groundbreaking technology in aviation that will revolutionise numerous products and services. The aviation sector is increasingly being driven towards the application of Machine Learning (ML) into new products with the intent to aid human operators or to augment automation capabilities. These products, especially safety-critical ones, need to be certified and must ensure a high level of reliability, demonstrating no unintended behaviours. This is achieved by providing design assurance, i.e., evidence that specific guidelines and verification procedures have been followed during the design process or that the product includes essential safety elements (such as redundancy and runtime monitors).

Traditional design assurance processes are based on various assumptions which are not always applicable to machine learning constituents (MLC), not allowing aviation industry to exploit the knowledge acquired over the years. In particular, ED-12C/DO-178C provides guidance to produce traditional (non-ML) *software that performs the intended function with a level of confidence in safety that complies with airworthiness requirements* [1]. The standard emphasizes the need to start from functional and non-functional requirements to eventually transform them into the software code. This code shall be traced to and verified against the requirements to ensure it is correct and, above all, does not expose behaviours that are unintended by the designer or unexpected by operators.

Moreover, traditional certification procedures commonly combine mathematical principles with empirical evidence: for example, they may rely on formal analysis, path-based analysis, or event testing. These processes examine every potential execution path of the system to ensure correctness and take into account a significant, yet limited, range of inputs and events. This certification frequently relies on mathematical proofs that demonstrate the system's correctness. These proofs can be supported by assumptions related to the physical system, which can be derived from first principles, or validated through empirical evidence [2].

Unfortunately, learning-based systems are data-driven and many of these approaches are not applicable. A subfield in which this problem is even more accentuated is deep learning (DL): modern deep learning-based systems often encode little prior knowledge of the underlying principles that govern the relationships between inputs X and outputs Y. Rather, they exemplify a category of machine learning techniques capable of accurately capturing the statistical dependencies between X and Y.

Furthermore, while traditional models represent intermediate computations in a structured form, neural networks (NNs) use a large number, often millions, of unstructured intermediate computations that are linked by simple functions. In practice, this results in significant certification issues. Due to these difficulties, machine learning is one of the most researched scientific topics, with the objective of leveraging mathematical proofs to verify the correctness of these models. At the moment, completely different methods, e.g., formal [3], statistical [4], empirical, or hybrid ones [5] are being explored, but despite these efforts, there exists a significant gap between achievements in the academic domain and certification requirements emerging for real-world safety-critical applications.

Several regulatory agencies, namely the European Union Aviation Safety Agency (EASA), Federal Aviation Administration (FAA) and Food and Drug Administration (FDA), have recently proposed timelines and invested resources towards certification and regulation of AI-based methods in safety-critical systems. However, at the actual state it is not clear which methodologies could satisfy both the strong safety standards required by the domain and are practically applicable at the same time.

It is also worth noting that there are also a great deal of challenges with regard to the properties of the models and the development process itself. As reported in [6], currently main ML challenges with respect to traditional software engineering requirements concern:

- Robustness - Models can have unpredictable behaviours when used on inputs with different distribution if compared to the training one. Robustness characterises the resilience of the model to this phenomenon;

- Uncertainty - Not all outputs from a ML model are necessarily well-calibrated in terms of statistical inference, as it is challenging to estimate the model's knowledge or confidence. Uncertainty involves techniques related to calibrating the margin of error that a model can exhibit;

- Explainability - It would be ideal to have a backtrack to the root cause of any error (*traceability*) and a full understanding of the model's inner logic (*interpretability*), in order to allow assurance and improvements. Explainability covers all the methods that make this process easier;

- Verification - Verifying if a software algorithm performs as intended is already part of the traditional certification. However, in the case of ML software, due to the high dimensionality of the data and the large number of parameters to be learned, verifying the exact inner process is quite complex.

1.2 Goals and scope of the thesis

The goal of this work is to investigate current machine learning methodologies aimed towards certification and to identify the enablers needed to support the future introduction in aviation of products embedding ML-based systems.

More precisely, this thesis intends to:

- Present a first set of guidelines for ML-based systems facilitating future compatibility with the agency regulatory framework (e.g., CS-25 [7]/CS-27 [8]/CS-29 [9]) using the specific use case proposed: the Runway Alignment System (RAS);
- Propose an innovative procedure, based on a new safety-related metric, the *Exceedance Rate*, to evaluate the performance aspects of ML-based systems and assure their behaviour to be safe;
- Derive a modified safety process to provide compliance with latest EASA AI safety requirements.

While hardware aspects were briefly considered, the thesis primarily focuses on software-related analysis and implementation.

This thesis opportunity was offered by *Leonardo Helicopters*, so all the examples, documents, and products mentioned will refer to company's established assets, reflecting its operational experience and internal practices.

1.3 Thesis Outline

Chapter 2 investigates current regulations, standards, and major reports on the use of machine learning-based systems in safety-critical domains.

Chapter 3 depicts an overview of the traditional safety assessment process and introduces the new *Learning Assurance*, specially suited for AI-based systems.

Chapter 4 includes necessary background knowledge required for the proposed guidelines related to Learning Assurance. Furthermore, a series of activities deemed to be essential for ensuring the safe application of neural networks are outlined. Each activity is examined in depth and supported by theoretical reasoning. This

framework is conceived to be flexible so that the specific implementation of each activity can be adapted to any particular needs of future applicants.

Subsequently, Chapter 5 presents the use case of neural networks in aviation applications, which will be the reference application for the remaining part of the document. The aim is to use it to illustrate the findings with a real-world example.

Finally, Chapter 6 concludes the exposition, reviews the proposed guideline and sets the first conceptual steps for future works.

1.4 Applicability

Machine learning is an extremely vast subject and, in some of its most complex implementations, researchers are still not able to completely understand and formally justify how the model reaches its results (i.e., *it is difficult to provide high-level descriptions of the numerous computations carried out on data to yield an output* [10]). For instance, one should consider that NNs are usually over-parametrised [6], as they have many more parameters than training samples. This is the reason why a ML model is typically regarded as a "**black-box**".

Due to this complex nature, the processes shown throughout this thesis are addressed to a specific group of models which satisfy some crucial assumptions. In other words, the idea is to firstly consider models with inherent characteristics that guarantee more human control along the whole development process, making their certification process easier; then, once these initial categories have been mastered and enough expertise has been acquired, applicants will be progressively provided with a more comprehensive framework, encompassing more challenging architectures.

In detail, the applicability of the proposed guideline is limited as follows:

- Covering *Level 1* and *Level 2 AI applications*, but not covering *Level 3 AI applications* yet (see Section 2.3);
- Covering *supervised learning* or *unsupervised learning*, but not other types of learning such as *reinforcement learning* (see Section 4.1);
- Covering *offline learning* processes where the model is "frozen" at the time of approval, but not *online learning* processes. In this way, the algorithm is not supposed to learn from the operative environment, which can be source of perturbation, adversarial and poisoning attacks [11];
- Covering only *AI trustworthiness* analysis, more precisely *initial safety assessment* exclusively, but not *AI assurance* or *ethic-based/human factors*;

- Taking in consideration only ML *regression problems*, since classification task have already been covered in the previous project (see [12]).

Furthermore, depending on the safety criticality of the application, and on the aviation domain, an assurance level is allocated to the AI-based system, e.g., Development Assurance Level (DAL) for initial airworthiness or air operations, Software Assurance Level (SWAL) for air traffic management or air navigation services (ATM/ANS).

This guideline only considers airborne applications where AI/ML constituents do not include DAL A or B.
To be more precise, current papers on this matter usually refer to the development assurance level as a link to the corresponding failure severity classification (e.g., DAL C for *Major* or DAL D for *Minor* failure severity).
Indeed, that represents the *minimum* quality level to guarantee, and having a higher one is only a safer and more desired option. Thus, it is more appropriate to use the terminology associated to the classification of the functional failure severity of the system, from the most dangerous to the least dangerous one: Catastrophic, Hazardous, Major, Minor, No Safety Effect.
For this reason, this thesis, according to this clarification, will focus mainly on systems whose failure can lead at most to Major effects.

Lastly, no DAL reduction is considered.

2 | Foundational documents for safety-critical systems development

In the last few years the aviation regulation agencies have been proposing the approach to follow the existing Safety Assessment process as closely as possible even for AI-based systems [13]. However, even if it revealed to be the right way to deal with this challenge, important adjustments would be necessary. For this purpose, this chapter includes the existing aviation guidelines and the most relevant works towards the safety assurance of AI-based technologies.

2.1 Existing guidelines and standards

Aeronautical software items development and systems certification are based on three main documents following the recommended practices by EUROCAE, and its SAE (or RTCA) counterpart. They consist of the following regulations:

1. *ED-79B/ARP4754B* "Guidelines for Development of Civil Aircraft and Systems";
2. *ED-135/ARP4761A* "Guidelines and Methods for conducting the Safety Assessment Process on Civil Airborne Systems and Equipment";
3. *ED-12C/DO-178C* "Software Considerations in Airborne Systems and Equipment Certification".

Together, these guidelines form a comprehensive, though evolving, foundation for software assurance in the safety-critical aviation context.

ED-79B/ARP4754B

The *ED-79B/ARP4754B* standard, defines guidelines for the development of civil aircraft systems with a focus on safety and certification compliance. It introduces

a structured top-down methodology that begins with system-level requirements and flows down to hardware and software components.

The document promotes traceability, validation, and verification throughout the system lifecycle, ensuring that safety objectives are met from concept through certification [14][15].

ED-135/ARP4761A

The *ED-135/ARP4761A* offers comprehensive methods for executing the Safety Assessment process in civil aviation. It builds upon the foundational principles of ED-79B/ARP4754B by focusing specifically on the identification, classification, and analysis of failure conditions and their associated risks [16].

This document will be extensively examined in Section 3.2.

ED-12C/DO-178C

The *ED-12C/DO-178C*, published in 2011, remains the milestone for airborne software certification, providing guidance to ensure that aviation software performs its intended functions with a level of safety adequate to airworthiness requirements.

The standard emphasises the principles of requirements flow-down and bidirectional traceability, ensuring consistency across all levels of software development. Additionally, it outlines five Software Levels (A–E), where the meticulousness of assurance activities increases based on the potential impact of software failure on aircraft safety [1].

However, the standard's traditional assurance framework faces challenges when applied to machine learning–based systems, due to issues such as data-centric development, lack of predictability and limited explainability of algorithmic behaviour. To address evolving technologies, the document is complemented by supplements on Model-Based Development (*DO-331* [17]), Object-Oriented Technology (*DO-332* [18]), and Formal Methods (*DO-333* [19]). The latter supplement contains valuable approaches that may help verifying the robustness of novel models.

2.2 Current works on machine learning in aviation

This section contains all latest notable documents that concern guidelines and advancements towards AI certification.

EASA AI Roadmap (2.0) - 2020/2023

Led by the common European goal of obtaining a human-centric development of AI in order to maximise the benefits of AI-based systems and, at the same time, to prevent and minimise the associated risks, EASA published the "AI Roadmap" [20] in 2020. This represents one of the most ambitious public studies with regard to ML certification and integration into aviation. In 2023, the same document was updated ("AI Roadmap 2.0" [21]) setting new goals, analysing the impact of AI on aviation, and offering a clearer three-phase timeline, spanning from 2019 to 2050, for the total incorporation of AI-based systems, as shown in Figure 2.1.

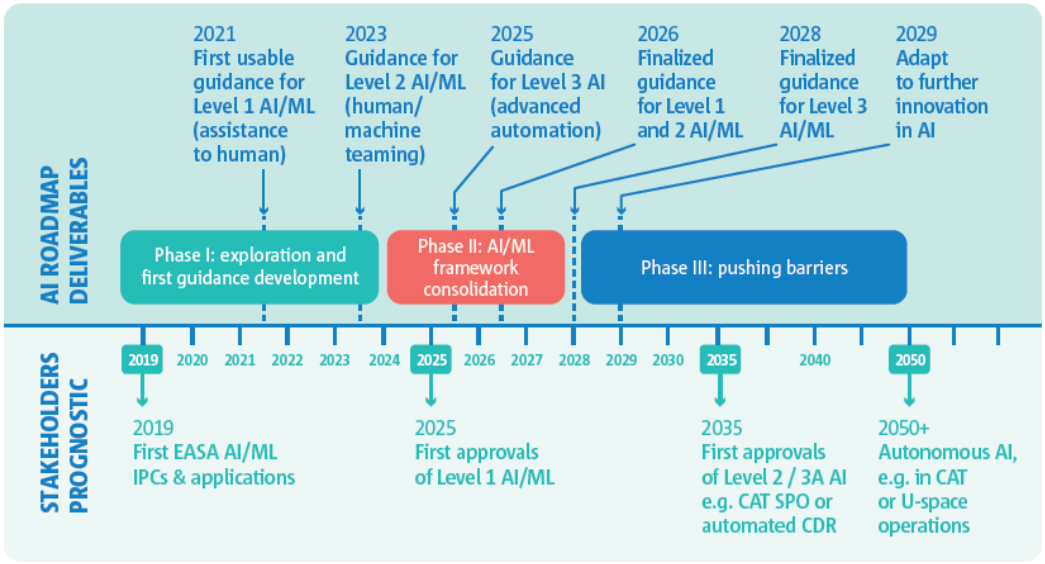

Figure 2.1: AI Roadmap 2.0 milestones.
(Source: AI Roadmap [21])

The primary objective of the Roadmap is to establish a risk-based “AI trustworthiness" framework to facilitate future AI/ML applications and to support European research and leadership in AI. This framework is designed to be adaptable, with annual updates and improvements achieved through ongoing knowledge exchange and practical AI development work. The initial version is concentrated on the ML segment of AI.

The Roadmap identifies four building blocks, still to be further researched, which are considered essential for the creation of a framework for AI/ML trustworthiness: *AI trustworthiness analysis*, *Learning assurance*, *Explainability* and *AI safety risk mitigation*. In particular, the AI trustworthiness analysis should provide, in the specific context of civil aviation, guidance to applicants on how to address each of the seven key guidelines identified in the report from the EC High Level Group of

Experts on AI, titled "Ethics and Guidelines on Trustworthy AI" [22]. Figure 2.2 shows the division mentioned above. Key elements of trustworthiness analysis are the *Safety Assessment* and the *Security Assessment*.

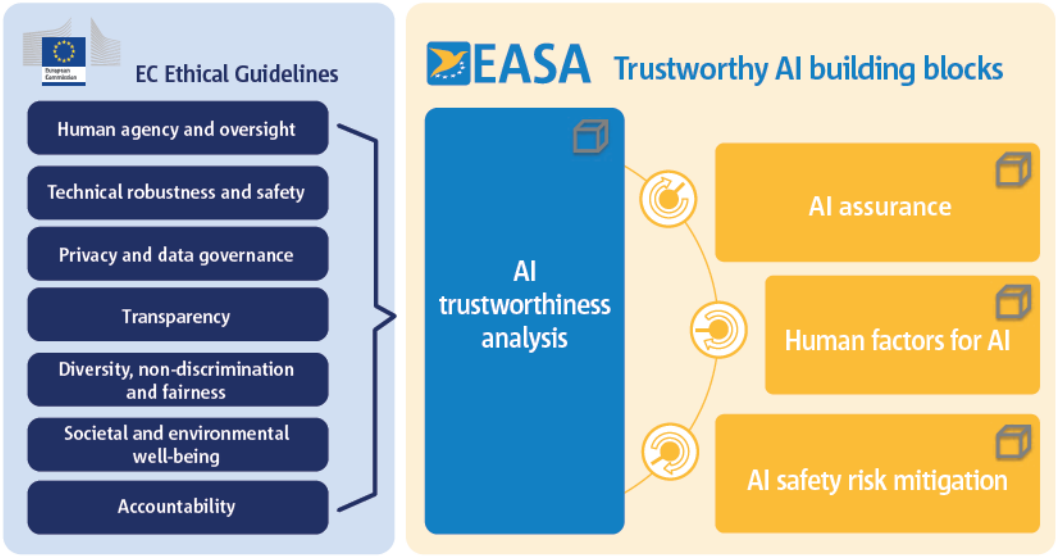

Figure 2.2: EASA AI trustworthiness building blocks.
(Source: EASA CP [13])

With this plan, EASA expects to become a leading oversight authority for AI and support the leadership of the European aviation industry in this field.

First Daedalean-EASA IPC: CoDANN - 2020

As foreseen in the first phase of the EASA AI Roadmap, a series of Innovation Partnership Contracts (IPCs) were established to take mutual advantage from the Agency's expertise and industry technologies. The "Concepts of Design Assurance for Neural Networks" (CoDANN) [23] is the first document encompassed in these collaborations; the focus was put on the Learning Assurance and AI Trustworthiness analysis building-blocks. The study yielded several important achievements, including:

1. The identification of how possible gaps in existing guidance for traditional applications can be filled.

2. The definition of the *W-shaped Learning Assurance process* (see Section 3.3) as the new future standard for the development and implementation of ML-based systems;

3. The exploration of methods to obtain generalisation bounds (see Section 4.1.3), especially for (deep) neural networks.

The importance of the document lies in the adaptability of the proposed theory to all models trained with supervised learning; particular attention is given to NNs, since they represent the most promising and, at the same time, elaborated techniques.

EASA first Concept Paper - 2021

In April 2021 EASA published a document containing a set of technical objectives crucial for the approval of Level 1 AI applications (see Section 2.3) titled "EASA Concept Paper First usable guidance for Level 1 machine learning applications" [24].

The objective was "to support applicants in the introduction of AI/ML technologies into safety-related applications in all domains covered by the EASA Basic Regulation [25]".

Second Daedalean-EASA IPC: CoDANN II - 2021

Between July 2020 and May 2021, another IPC project between EASA and Daedalean enabled the drafting of the second issue of CoDANN [26], in which they investigated what remained unresolved in the previous report. In particular, it focused on the right bracket of the W-shaped process (*integration*, *implementation* and *inference*), also dealing with hardware aspects.

Furthermore, it delved into the definition and role of explainability, trying to "open the black-box" and to simplify human-machine interaction.

FAA Neural Network Based Runway Landing Guidance for General Aviation Autoland - 2021

In November 2021, Daedalean took part in an IPC project with FAA as well and published the most practical report [27], in which they developed and tested an almost entirely complete ML-based system: the Visual Landing Guidance System. The notable project outcomes are described as follows:

1. Evaluation of NN-based technology for 14 CFR Part 91 General Aviation [28];
2. Test in practice the W-shaped process for Learning Assurance;
3. Inform specific policy for machine learning-based systems;
4. Validating the visual-based AI landing assistance as a backup system.

Formal Methods for Learning Assurance - 2023

The *Formal Methods for Learning Assurance* (ForMuLA) project, developed through an IPC between EASA and Collins Aerospace, explored the application of *formal methods* for the assurance of machine learning systems. Unlike traditional verification techniques, formal methods provide mathematically rigorous tools to analyse the stability, predictability, and safety compliance of ML-based components. This document complements the outcomes of MLEAP by providing a theoretical and methodological foundation for the certification of AI-driven systems [3].

Machine Learning Application Approval - 2024

The *Machine Learning Application Approval* (MLEAP) project, initiated by EASA under the Horizon Europe programme, primary focused on developing compliance guidance for ML-based systems in safety-critical contexts. The project concentrated on three technical pillars:

- Completeness and representativeness of the dataset(s);
- Generalisation properties of the model;
- Verification in terms of robustness and stability;

This gigantic work deepened a very numerous quantity of machine learning techniques, some yet theoretical, and tried to adapt them on their toy use cases, making a step forward towards the creation of a universal and scalable framework [5].

2.3 EASA Concept Paper: Guidance for Level 1 & 2 machine learning applications - 2024

The *EASA Concept Paper* (CP) aims to extend the initial guideline for AI assurance in aviation by moving beyond Level 1 basic assistance functions to include more advanced human–AI teaming, labelled Level 2. Compared to the first guide, it refines the objectives, introduces clearer classification criteria and, where possible, provides tentative Means of Compliance (MOC) for verification and validation of ML-based systems.

This document benefits from and incorporates all previous listed works; until the publication of the *Implementing Rules* (IR) and *Acceptable Means of Compliance* (AMC), it serves as a reference tool to support the preparatory phase of certifying and approving AI/ML-related products.

Challenges As the CP mentions, the main challenges that are intended to be addressed by this first set of EASA guidelines are:

- Modifying assurance frameworks to incorporate the specifics of identified AI techniques and addressed development errors in AI-based systems and their components;
- Addressing the unique sources of uncertainty related to AI/ML technology usage;
- Developing a data management framework to ensure the bias mitigation and completeness/representativeness of datasets used;
- Tackling the trade-off between model bias and variance at various stages of ML processes;
- Ensuring the robustness and prevention of unintended behaviour in ML/DL applications;
- Dealing with the limitations of human understanding of ML applications behaviour, given their stochastic nature and ML model complexity;
- Handling shared operational authority in human-AI collaboration (HAT);
- Building trust among end-users.

Machine Learning Constituent To properly describe these novel architectures, the CP highlights the need for an intermediate entity level between the system and the item one, called "AI/ML constituent"[1] (MLC). Figure 2.3 illustrates the suggested decomposition of an AI-based system, which was also used as a reference for this work.

[1]This subdivision is specifically aimed at providing terminology to complement the W-shaped learning assurance process (see Section 3.3)

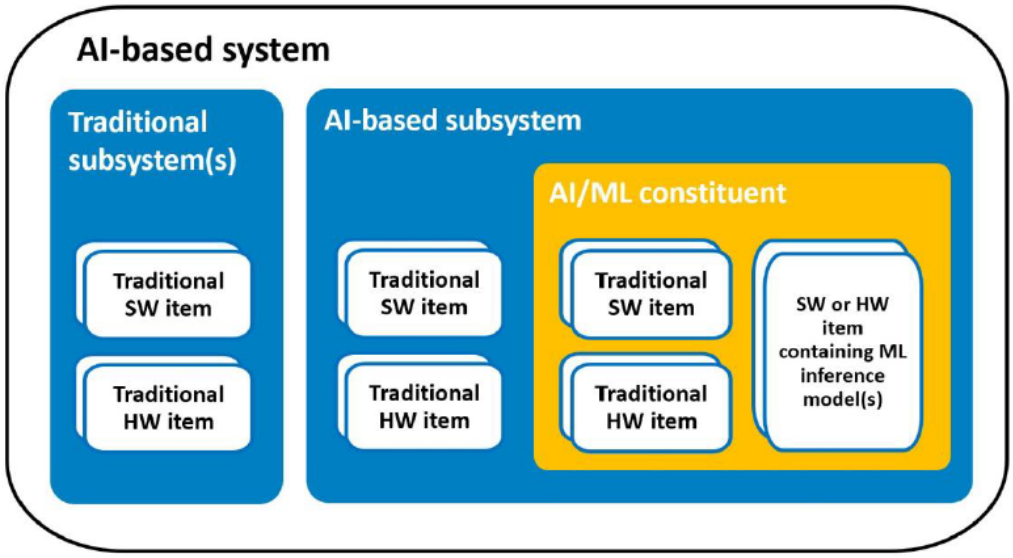

Figure 2.3: Decomposition of an AI-based system.
(Source: EASA CP [13])

An AI-based system encompasses several subsystems, and at least one of them is AI-based; an AI-based subsystem, in turn, embeds at least one AI/ML constituent.

The AI/ML constituent is a *defined and bounded collection of hardware and/or software item(s) which are grouped for integration purpose to support one AI-based subsystem function, including:*

- *at least one specialised hardware or software item containing one (or several) ML model(s), further referred to as "AI/ML item";*
- *the necessary pre- and post-processing traditional items.*[2]

Traditional (sub)systems and items have to follow the current existing guidance.

Classification of AI applications The EASA CP outlines three general AI application levels, to help industrial stakeholders to plan and evaluate the maturity of their products in the aviation scenario.

The subdivision starts with the assisting functions (Level 1 AI), then goes to human-AI teaming (Level 2 AI), and finally considers more or full autonomy of the machine (Level 3 AI). The core discriminating criterion lies in the end-user

[2]Definition from: EASA Concept Paper [13]

authority, i.e., the capability of the human to have decision-making power on the function delivered by the system.
For Level 1 AI, the end-user takes the decisions based on the AI-based system advisory; Level 2 AI-based systems, instead, can perform automatic implementation of actions and the end-user maintains the override capability at any time. Lastly, at Level 3 the AI-based system is given full authority to make decisions, even without end-user involvement.
Additional conceptual shades, A or B, have been introduced in the document to refine the aforementioned AI application levels. Table 2.1 shows the resulting elaboration of the three scenarios:

AI Level	Sub-level	Description
Level 1: Assistance to Human	1A	Human augmentation
	1B	Human cognitive assistance in decision and action selection
Level 2: Human-AI Teaming	2A	Human and AI-based system cooperation
	2B	Human and AI-based system collaboration
Level 3: Advanced Automation	3A	AI-based system makes decisions and performs actions, safeguarded by the human
	3B	AI-based system makes non-supervised decisions and performs non-supervised actions

Table 2.1: Classification of AI applications.

2.3.1 Safety Assessment objectives

Overview

The Safety Assessment is part of the AI trustworthiness analysis building block (see Figure 2.2) and represents, as for traditional systems, the process to assure compliance with acceptable levels of safety as described in the applicable regulations (further details are provided in Chapter 3, linking the Safety Assessment to the system development process). A logical and acceptable inverse relationship must exist between the occurrence probability of a failure condition and the severity of its consequences.

For non-AI systems, Safety Assessment methods differ across aviation domains, but only hardware is treated as subject to random failures, generally. Software reliability is not quantified directly; instead, adherence to recognised development assurance methodologies is assumed to reduce the likelihood of errors to an acceptable level. The probabilistic risk assessment mainly considers the reliability of digital inputs and the hardware platform executing the code.

Machine learning introduces new challenges due to its statistical nature and model complexity, such as predictability issues and sources of uncertainties. The core principle of the EASA CP on safety is to *demonstrate that AI/ML-based systems can ensure at least the same level of safety as traditional systems*, while remaining as consistent as possible with existing aviation Safety Assessment processes to minimise disruption.

Safety Assessment is further divided in two categories, depending on the life cycle phase of the product:

- *Initial Safety Assessment*, performed during the design phase, considering all different types of system components failures;
- *Continuous Safety Assessment*, an in-service analysis with the scope of monitoring that the guarantees stated during the initial phase remain valid.

AI Safety objectives

Related to the initial Safety Assessment, the document sets the following important objective:

> **Objective SA-01:** The applicant should perform a safety (support) assessment for all AI-based (sub)systems, identifying and addressing specificities introduced by AI/ML usage.

In order to support the certification process towards the compliance to this objective, given the fact that actual expertise has not reached the necessary maturity yet, the CP advances nine *anticipated Means of Compliance* (aMOC).

The initial phase of the safety process determines whether the preliminary system architecture can meet the identified safety criteria and establishes safety requirements that are allocated to all the items (even MLCs): those include requirements on the MLC development process, as well as performance requirements.
Subsequently, lower-level development processes need to confirm that their design solution fulfils those requirements.

Requirements on the item (or MLC) development process are associated with ALs and SWALs in the ATM/ANS domain and Item Development Assurance Levels

(IDALs) in the airborne systems domain.
The software (or MLC)[3] *assurance level assignment is determined based on the severity of the credible worst-case failure hazard effect* [1] and defines the level of rigor of the downstream development process.

> **Anticipated MOC SA-01:** DAL/SWAL allocation and verification:
> The following standards and implementing rules with adaptation may be used to perform DAL/SWAL allocation
>
> - For embedded systems:
>   - ED-79B/ARP4754B and ARP4761
> - For ATS providers in the ATM/ANS domain, the following implementing rule requirements (and the associated AMC and GM) are applicable:
>   - ATS.OR.205 Safety assessment and assurance of changes to the functional system
>   - ATS.OR.210 Safety criteria
> - For non-ATS providers in the ATM/ANS domain, the following implementing rule requirements (and the associated AMC and GM) are applicable:
>   - ATM/ANS.OR.C.005 Safety support assessment and assurance of changes to the functional system.
>
> Starting from the AI-based system and functional analysis, the DAL/SWAL allocation should be done down to the AI/ML constituent level.

The second aMOC relates system safety objectives to machine learning performance.

> **Anticipated MOC-SA-01-2:** Metrics
> The applicant should define metrics to evaluate the AI/ML constituent performance.
> Depending on the application under consideration, a large variety of metrics may be selected to evaluate and optimise the performance of AI/ML constituents. The selected metrics should also provide relevant information with

[3]Editor's note.

> regard to the actual AI/ML constituent reliability so as to substantiate the safety assessment (or impact on services performance in the case of safety support assessment).
>
> Performance evaluation is performed as part of the learning assurance per *Objectives LM-09* (for the trained model) and *IMP-06* (for the inference model).

The output of a ML model, due to its statistical nature, is typically an approximation of the functional intent. Moreover, the acceptability of the approximation that underlies performance metrics is application-dependent and is essential for evaluating whether the model satisfies the requirements. A complete definition of ML model performance requirements may not only involve metrics, but also quantitative criteria (with correlated tolerances, limits, and thresholds) and/or the corresponding confidence.

To better frame the third aMOC a brief description of the operating environment terminology is introduced below.

Firstly, it is necessary to define the *Concepts of Operations* (ConOps), which is a human-centric document that describes operational scenarios at the level of the product or of the AI-based system, from the user's viewpoint[4]. In addition to a list of potential end-users, goals, and tasks allocation scheme, it mainly includes the definition of the *Operational Domain* (OD).
The Operational Domain is the description of the operating conditions, at the system level, under which the AI-based system is specifically designed to deliver the intended behaviour, comprising, but not limited to, environmental, geographical, and/or time-of-day restrictions[4].

Figure 2.4 shows the interrelationship between the operational scenarios for the ConOps and the operating parameters for the OD.

Descending from the system level towards the item level there is the MLC *Operational Design Domain* (ODD), which is the set of operating parameters, together with the range and distribution within which the MLC is designed to operate nominally [4].
An even more refinement could be needed if the MLC ODD cannot be fully managed by the ML model design limitations, that is the ML model ODD.

[4]Definition adapted from: EASA Concept Paper - Issue 02 [13]

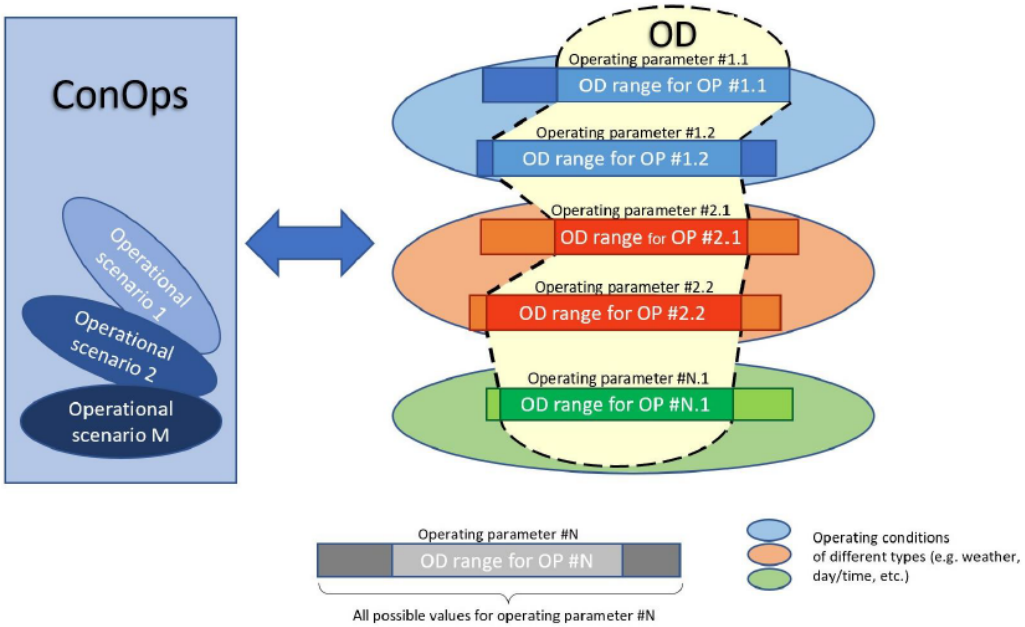

Figure 2.4: Interrelationship between ConOps and OD. (Source: EASA CP [13])

Anticipated MOC-SA-01-3: Exposure to data outside the OD or ODD

To mitigate the exposure to data outside the OD or ODD, these means or a combination of them are expected to be necessary to deliver the intended behaviour:

- Establish the monitoring capabilities to detect that the input data is outside the AI/ML constituent ODD, or the AI-based (sub)system OD;
- Put in place functions for the AI/ML constituent to continue to deliver the intended behaviour when input data is outside the ODD;
- Put in place functions for the AI-based (sub)system to ensure safe operation when input data is outside the OD.

For low-dimensional input space (e.g. sensors producing categorical data, tabular data, etc.), monitoring the boundaries of the ODD or OD could be a relatively simple task. However, monitoring the limits of the ODD or OD could be much more complicated for high-dimensional input spaces (such as in computer vision with images or videos, or in NLP). In such use cases, techniques such as the out of distribution (OoD) discriminator [23] could be envisaged.

When input data is outside the OD, the intended function cannot be fulfilled. In such a situation, it is expected that monitoring combined with alerting functions and procedures are implemented to ensure safe operation.

The aMOC-SA-01- 4,5,7 and 8 intend to address the intrinsic uncertain nature of the AI/ML constituent. To do so, the document proposes procedures to deal with the two crucial topics of uncertainty (see Section 4.5) and predictability (generalisation gap: see Section 4.1.3).

Anticipated MOC-SA-01-4: Identification and classification of uncertainties

Sources of uncertainties affecting the AI/ML constituent should be listed. Each should be classified to determine whether it is an aleatory or an epistemic source of uncertainties.

Anticipated MOC-SA-01-5: Assessment and mitigation of uncertainties

Aleatory uncertainties should be minimised to the practical extent. Effects of aleatory uncertainties should be assessed at system level. In particular, when a quantitative assessment is required, the aleatory uncertainties should be accounted for in a way that does not compromise safety.

Epistemic uncertainty is addressed through the learning assurance objectives.

For the sixth aMOC it is important to highlight that the concept of failure mode, i.e., *the manner in which a component, subsystem, or system fails to perform its intended function* [16], is extended to AI/ML systems and components.

Anticipated MOC-SA-01-6: Establishment of AI/ML constituent failure modes:

- Establish a taxonomy of AI/ML constituent failures;
- Evaluate possible failure modes and associated detection means.

The failure conditions allow to identify safety objectives that provide a starting point to review preliminary system architectures in the initial phases of the safety process.

Anticipated MOC-SA-01-7: Link between performance metrics and safety assessment

When a quantitative safety (support) assessment is required to demonstrate that the safety requirements are met, performance metrics should provide a conservative estimation of the probability of occurrence of the AI/ML constituent failures modes.
Performance evaluation performed as part of the learning assurance per *Objectives LM-09* (for the trained model) and *IMP-06* (for the inference model) is fed back to the safety assessment (support) process.

Anticipated MOC-SA-01-8: Link between generalisation bounds and safety assessment

Based on the generalisation gap evaluated per *Objective LM-04*, the applicant should assess the impact on the safety (support) assessment. This should be supported by specifying margins on performance requirements as part of the safety (support) assessment.

When a quantitative safety (support) assessment is required to demonstrate that the safety requirements are met, the probability of occurrence of the AI/ML constituent failure modes may be evaluated from the 'out-of-sample error' (E_{out}). One possible approach is to define the 'in-sample error' (E_{in}) using a metric that reflects application-specific quantities commensurate with the safety hazard. Then, provided that E_{in} is defined in a meaningful and practical way, E_{out}, that reflects the safety performance in operations, can be estimated from the E_{in} and the generalisation gap. Such errors are however quantities on average, and this should be taken into account.

The CP also provides a suggested example approach to *establish safety requirements associated with AI/ML constituent failure modes and the associated probability* [13]:

1. Describe precisely the desired inputs and outputs of the ML item and the pre-/post-processing steps executed by a traditional SW/HW item.
2. Establish AI/ML constituent failure modes (aMOC-SA-01-6).
3. Identify appropriate metrics to evaluate the model performance and initiate an early specification of the thresholds necessary to meet the safety objectives (aMOC-SA-01-2).
4. Identify how performance metrics translate into a probability of occurrence of the ML model failure mode (aMOC-SA-01-7).

5. Assess and quantify, when applicable, generalisation bounds either through the model complexity approach or through the validation/evaluation approach. This leads to bounds for almost all data sets on average over all inputs. Based on those bounds, specify margins on performance metrics.

   NOTE: The computation of the generalisation gap is part of the *Learning Management* objectives (*Objective LM-04*) and not of the initial safety assessment. However, the output of this objective may then be used to specify margins on performance metrics.

6. Identify how performance metrics with associated margins translate into a probability of occurrence of the ML model failure mode (aMOC-SA-01-8).

7. Analyse the post-processing system to show how it modifies the latter failure probabilities. Usually, the post-processing results in improved performance (with respect to the chosen metrics) and/or reduction of the impact of the ML model failures on the AI/ML constituent performance metrics.

8. Study the elevated values of the error metrics for the model on the training/-validation (eventually testing) data sets, and develop adequate mitigations.

9. Based on all the previous steps, derive the necessary safety requirements.

Last aMOC implies the verification of the implemented architecture against the associated requirements with adequate proofs.

> **Anticipated MOC-SA-01-9:** Verification
> Verify that the implementation satisfies the safety (support) assessment requirements including the independence requirements.
>
> When classical architectural mitigations such as duplicating a function in independent items to improve reliability (i.e. 'spatial redundancy') are put in place, then particular care should be taken to ensure that the expected improvements are achieved (e.g. by checking that items required to be independent have uncorrelated errors).

Currently, it is highly recommended for the verification process to follow the traditional standards and implementing rules with appropriate adaptations, which are for embedded systems ED-79B/ARP4754B [15] and ED-135/ARP4761A [16], as previously stated.

3 | Safety Assessment and development process

As anticipated in Section 2.3.1, the Safety Assessment process in aviation aims at showing compliance with EASA Certification Specifications (CS 25 [7], CS 27 [8], CS 29 [9]). This process is strongly linked with the development process, which is performed in parallel.

The Safety Assessment process and the development process are regulated, respectively, by ED-135/ARP4761A [16] and ED-79B/ARP4754B [15] EUROCAE, and its SAE counterpart, recommended practices.

The aforementioned standards give applicants flexibility to pursue different approaches to comply with the regulations. When a new program starts, the first safety oriented consists in conceiving a *Safety Program Plan*, a document in which all standards, methods, and processes that support the design of the whole Aircraft/Helicopter (H/C) are defined.

3.1 The V-shaped Development Process

Traditional system development process starts from taking the aircraft requirements and refining those until reached the item-level in a top-down approach. Afterwards, an opposite bottom-up method has to be performed to provide correct system integration, verification, and validation. The result is a structured *V-shaped* process, as shown in Figure 3.1.

The ED-79B/ARP4754B [15] standard emphasises the capture, allocation, and traceability of requirements, ensuring that safety and operational objectives are demonstrably satisfied. It also clarifies the relationship between system-level requirements and the more detailed standards governing software (DO-178C/ED-12C [1]), airborne electronic hardware (DO-254/ED-80 [29]) and Safety Assessment (ED-135/ARP4761A [16]), as shown in Figure 3.2.

The items considered by the document are software (SW) and hardware (HW); since MLCs are a combination of HW and SW components and entail new requirements,

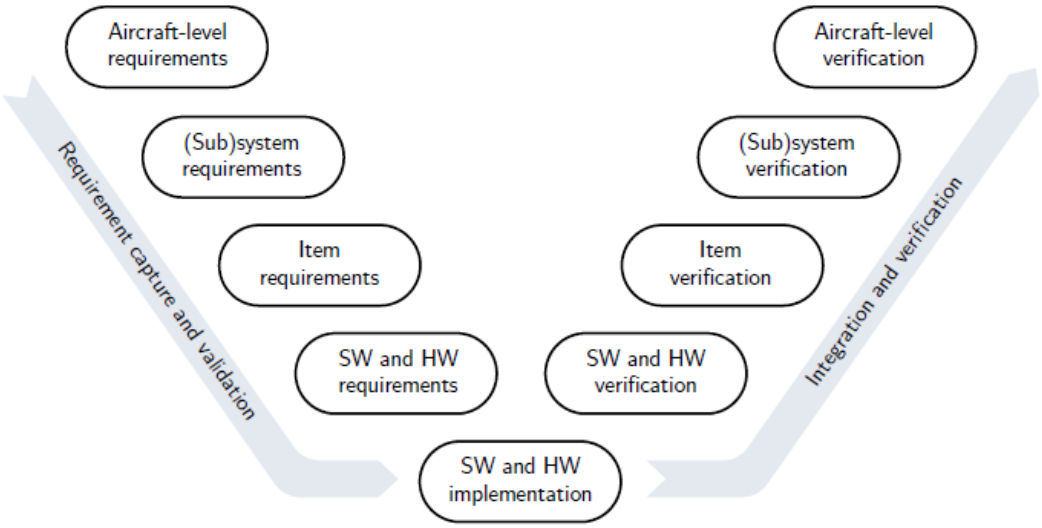

Figure 3.1: V-shaped model for system development.
(Source: CoDANN 2 [26])

it is clear that this development process cannot remain unchanged when these new technologies are present (see Chapter 3.3).

3.2 The Safety Assessment Process

3.2.1 Processes and Interactions

As previously noted, given that safety is an inherent goal of the development life cycle, the related processes and activities are organised to follow the V-model framework.

The guidelines to conduct the Safety Assessment are provided by the already mentioned ED-135/ARP4761A. The processes described in it are *usually applicable to new designs or to existing designs that are affected by changes to design or functions* [16]. The document introduces a multitude of sub-processes which are intended to be performed simultaneously to the system development process, with the aim of identifying system functions and their associated failure conditions, deriving quantitative and qualitative safety objectives (i.e., attributes necessary to achieve the required level of safety), and verifying the system compliance[1]. Once the failure conditions have been classified, the objective consists in determining

[1]The ED-135/ARP4761A does not include information on security threat considerations.

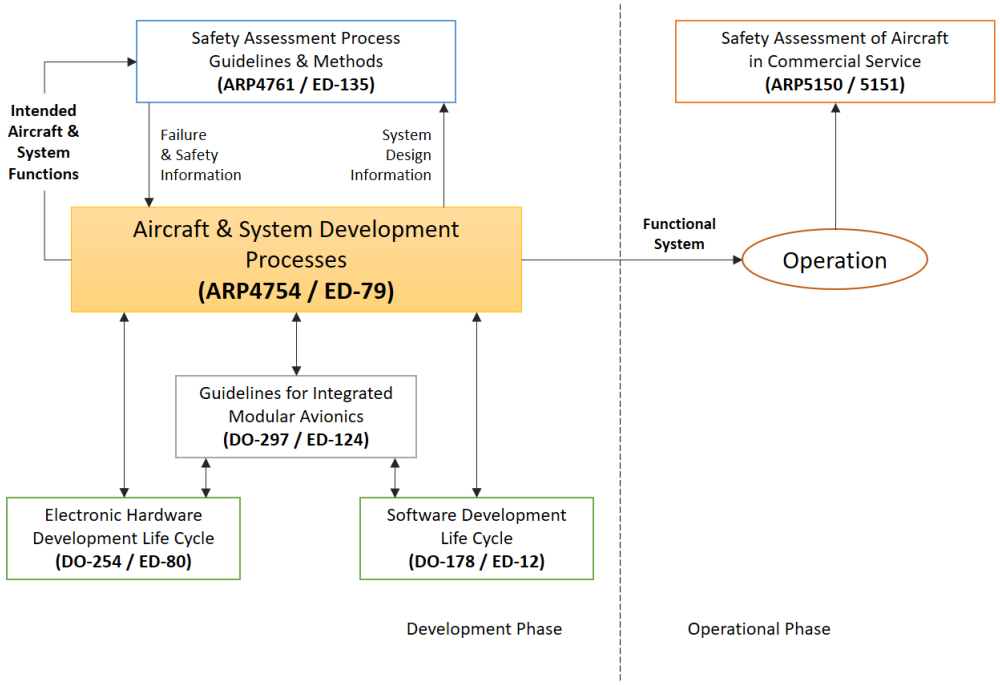

Figure 3.2: Guideline documents covering development and in-service/operational phases.
(Source: ED-79B/ARP4754B [15])

the minimum level of rigor to be applied to the related development assurance activities.

These processes, starting from the aircraft level, descending to the item-level and going up again to the aircraft, are[2]:

1. *Aircraft Functional Hazard Assessment* (**AFHA**): a systematic, comprehensive evaluation of aircraft functions to identify and classify failure conditions of those functions according to their severity, and provide a basis for aircraft-level safety objectives.

2. *Preliminary Aircraft Safety Assessment* (**PASA**): a systematic, comprehensive evaluation of a proposed aircraft architecture to provide confidence that the safety objectives resulting from the failure condition classifications can be met by developing safety requirements pertaining to those failure conditions.

3. *System Functional Hazard Assessment* (**SFHA**): a systematic, comprehensive

[2]Definitions from: ED-135/ARP4761A [16]

evaluation of system functions to identify and classify failure conditions of those functions according to their severity and provide a basis for system-level safety objectives.

4. *Preliminary System Safety Assessment* (**PSSA**): a systematic, comprehensive evaluation of a proposed system architecture and equipment/item designs to provide confidence that the safety objectives resulting from the failure condition classifications and system-level safety requirements from the PASA can be met by developing associated system-level, equipment-level, and item-level safety requirements pertaining to those failure conditions.

5. *System Safety Assessment* (**SSA**): a systematic, comprehensive evaluation of the implemented system to verify that applicable safety objectives and requirements are met.

6. *Aircraft Safety Assessment* (**ASA**): a systematic, comprehensive evaluation of the aircraft to verify that applicable safety objectives and requirements are met.

The Safety Assessment process begins with the AFHA process, which reviews aircraft-level functions, determines the respective failure conditions and allows to derive safety objectives to be used in the PASA process. In particular, the PASA process has to evaluate if the proposed aircraft architecture can match these derived safety objectives. Once defined, the safety requirements related to safety objectives are passed on to the PSSA processes, along with assigned Function Development Assurance Levels (FDALs), and then integrated into the development process, i.e., passed to other systems, subsystems, and items for implementation.

Similar processes occur at system level with SFHA and PSSA, establishing safety objectives and requirements for implementation.

The SSA process verifies that the implemented system design meets the qualitative and quantitative safety objectives and requirements from the SFHA, PSSA, and PASA. The same logic applies to the ASA, which verifies the overall aircraft design meets the safety expectations.

It is important to emphasise that these processes are intended to be **iterative**, as their nature requires continuous interrelationships, verifications, and return to previous stages for any change to the design.
Figure 3.3 depicts the typical sequence of development steps and their interactions with the Safety Assessment process.

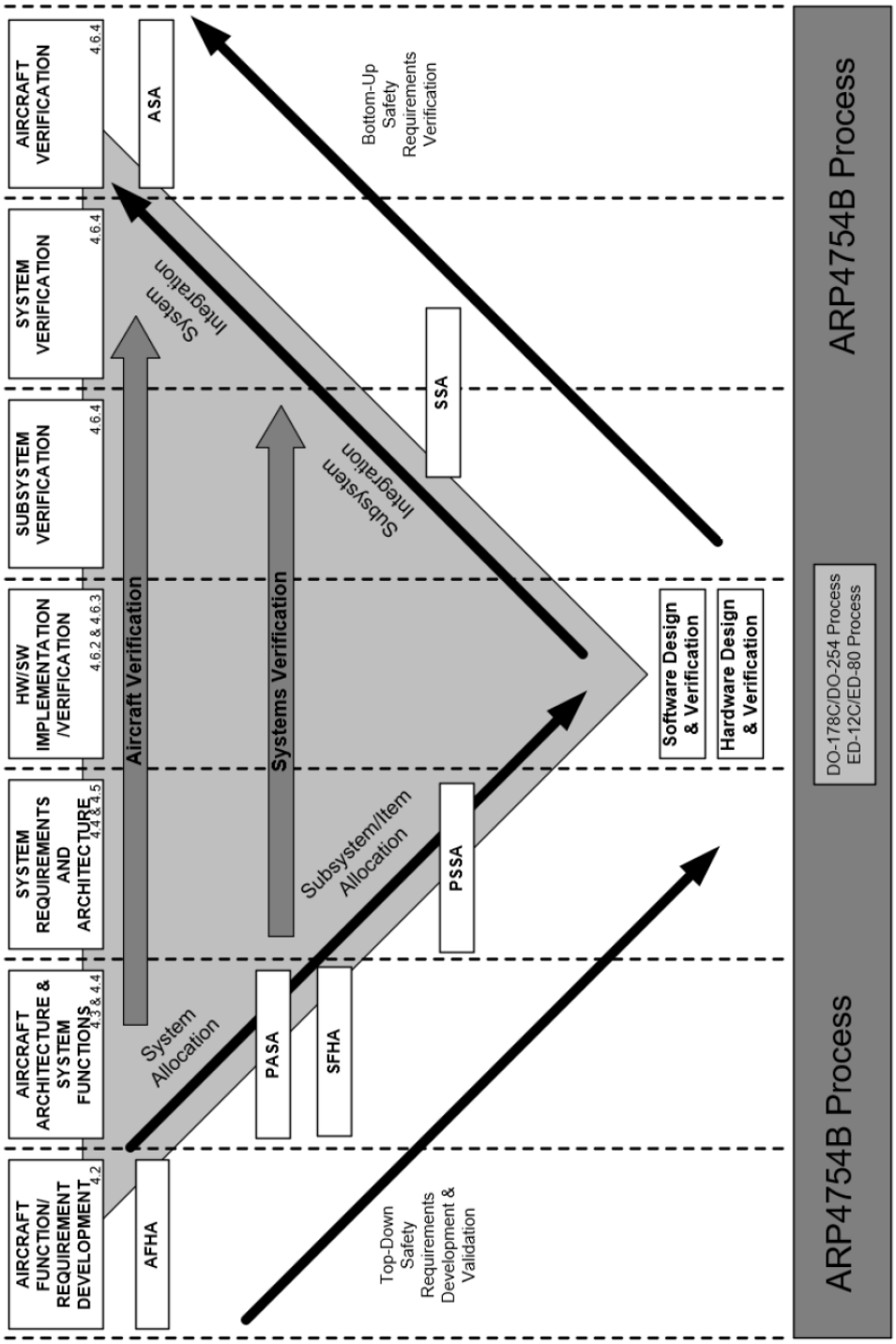

Figure 3.3: Interaction between Safety Assessment and development process. (Source: ED-79B / ARP4754B [15])

3.2.2 Activities

Functional Hazard Assessment

As anticipated in Subsection 3.2.1, the FHA, whether performed at aircraft or system-level, is based on:

1. The identification and description of the failure conditions;
2. A list of all associated failure effects, usually evaluated in the Most Critical Condition (MCC) scenario;
3. The allocation of the severity classification to each failure condition, according to Table 3.1 criteria.

The most explanatory report element of the FHA is the FHA Table, in which the following information is present[3]:

- **ID** - Unique and traceable numbering system assigned to each Function or Functional Failure (FF).
- **Function** - The function being analysed.
- **Functional Failure** - Description of the failed state of the function.
- **Flight Phase or MCC** - List of aircraft operational phases or the most critical one, i.e., the operational phase/condition and/or environmental/coincidental condition that would lead to the worst rotorcraft level effects, should the relevant functional failure occur.
- **Effect of Failure Condition** - Description of the failure condition effects on the aircraft, crew and occupants.
    1. "Effect on aircraft" refers to the ability of the aircraft to perform its functions and to the aircraft's structural integrity;
    2. "Effect on flight crew" refers to flight crew awareness and reaction to the failure conditions, as well as any physiological effects;
    3. "Effect on occupants excluding flight crew" refers to cabin crew or passengers discomfort, injury or fatalities.
- **Severity Classification** - Functional failure's severity classification.

[3]Definitions adapted from: ED-135 / ARP4761A [16]. Table entries may differ depending on the applicant.

- **Assumptions, Comments, Reference and Remarks** - Data supporting the determination of effects and classification of the failure condition.
- **H/C Level**: Only if the analysis is performed at system level (SFHA), in this cell it is reported the link to the identifier of the H/C level failure condition. One last letter, C or D, signals how the analysed failure contributes to higher-levels (i.e. H/C) one:
  - **(D)** if the system level failure condition is a direct contributor to the helicopter level failure condition;
  - **(C)** if the system level failure condition is only a contributor to the identified helicopter level failure condition when combined with another independent failure.

Failure Condition Severity	Effect on Aircraft	Effect of Flight Crew	Effect on Occupants
Catastrophic (I)	Loss of aircraft	Crew unable to accomplish required tasks; Required crew strength or skill in excess of crew capability; Crew incapacitation; Crew fatalities	Multiple occupant fatalities
Hazardous (II)	Large reduction in aircraft functional capability or safety margin	Excessive crew workload increase; Crew unable to fully accomplish required tasks; Crew physical distress	Small number of occupant fatalities or severe injuries
Major (III)	Significantly reduced aircraft functional capability or safety margin	Significant crew workload increase; Conditions impairing crew efficiency; Crew physical discomfort	Occupant physical distress or non-fatal injuries
Minor (IV)	Slightly reduced aircraft functional capability or safety margin	Slight crew workload increase	Occupant physical discomfort
No Safety Effect (-)	No effect or aircraft functional capability or safety margin	No effect on crew workload or physiology	No effect on occupant physiology

Table 3.1: Severity Classification.
(Adapted from: ED-135/ARP4761A [16])

Quantitative Allocation

The identified failure conditions with their respective severity classification are then supposed to be linked to the reference *Certification Specification* (CS), depending

on the domain. For large rotorcraft (> 3175 kg) the applicant has to refer to the CS 29, which imposes the quantitative requirements, illustrated in Table 3.2, for HW components (since, at the moment, they are the only ones considered to be subject to random failures).

Failure Condition Severity	**Safety Quantitative Requirement**[3]
Catastrophic (CAT or I)	$p < 10^{-9}$ (*extremely improbable*)
Hazardous (HAZ or II)	$p < 10^{-7}$ (*extremely remote*)
Major (MAJ or III)	$p < 10^{-5}$ (*remote*)
Minor (MIN or IV)	$p < 10^{-3}$ (*probable*)
No Safety Effect (NSE)	- (reasonably probable)

Table 3.2: Quantitative requirements.
(Adapted from: EASA CS 29.1309 [30])

Current DAL Allocation

As introduced in Section 2.3.1, a Development Assurance Level (DAL) has to be allocated depending on the severity of the credible worst-case failure hazard effect, and determines the minimum level of development rigor to guarantee for the entity (e.g., function, software, complex hardware, item) to which it is assigned.

According to DO-254/ED-80 guidance [29], complex hardware is defined as items for which functional correctness cannot be shown by tests and analyses alone; typical examples are custom microcoded devices like Field-Programmable Gate Arrays (FPGA) or Programmable Logic Devices (PLD). In this case, the hardware requires systematic design assurance planning and verification at the level imposed by its DAL.

Software DAL is not related to the probability of failure, but to the risk of systematic errors when the software takes logical decisions, generates control commands or processes data crucial for safety-critical functions. In such cases, software receives a Software DAL per DO-178C/ED-12C [1] commensurate with its failure impact. For instance, software items with DAL A require *independence of verification*: verification activities must be performed by individuals or groups that are independent of those who developed the software.

[3]Probability is measured *per flight hours.*

One important thing about DAL requirements associated to functions, SW, complex HW or items part of traditional systems is that they can be subject to reduction if certain factors occur: when architectural or design mitigations demonstrated through the safety assessment process ensure that no single failure or development error can directly lead to the associated aircraft-level failure condition. In such cases, where independence, redundancy or monitoring mechanisms are provided, a lower DAL may be assigned consistent with the reclassified residual safety impact. As previously mentioned, DAL reduction is not allowed on AI-based systems yet.

As per the Hazard Assessments, DALs are allocated starting from the function-level (FDAL) and descending to the item-level (IDAL).

Table 3.3 shows the connection between DAL and severity classification of the failure condition, according to ED-79B/ARP4754B [15] guidelines.

Failure Condition Severity	**DAL Requirement**
Catastrophic (CAT or I)	**A** (*highest rigor*)
Hazardous (HAZ or II)	**B**
Major (MAJ or III)	**C**
Minor (MIN or IV)	**D**
No Safety Effect (NSE)	**E** (lowest rigor)

Table 3.3: DAL requirements.

(System / Aircraft) Safety Assessment

Once defined the functions, identified their failure conditions, established the associated severity classification and assigned the safety requirements, it is necessary to guarantee and verify the compliance with those requirements.
The SSA and ASA conduct the Safety Assessment at system and aircraft-level, respectively.

The Safety Assessment process includes a variety of *safety analysis methods* among which applicants can choose to provide qualitatively and/or quantitative assurance on the safety of a design. These include propagation error methods, such as:

- Fault Tree Analysis (FTA);
- Dependence Diagrams (DD);
- Markov Analysis (MA);
- Model-Based Safety Analysis (MBSA);

- Failure Modes and Effects Analysis/Summary (FMEA/FMES);
- Cascading Effects Analysis (CEA);

and methods for common cause considerations, essential to guarantee the independence between systems, like:

- Zonal Safety Analysis (ZSA);
- Particular Risk Analysis (PRA);
- Common Mode Analysis (CMA).

3.3 Learning Assurance

The EASA W-shaped learning process (Figure 3.4) for learning assurance adapts the traditional V-shape process (see Figure 3.1) to the new challenges posed by ML-based systems.

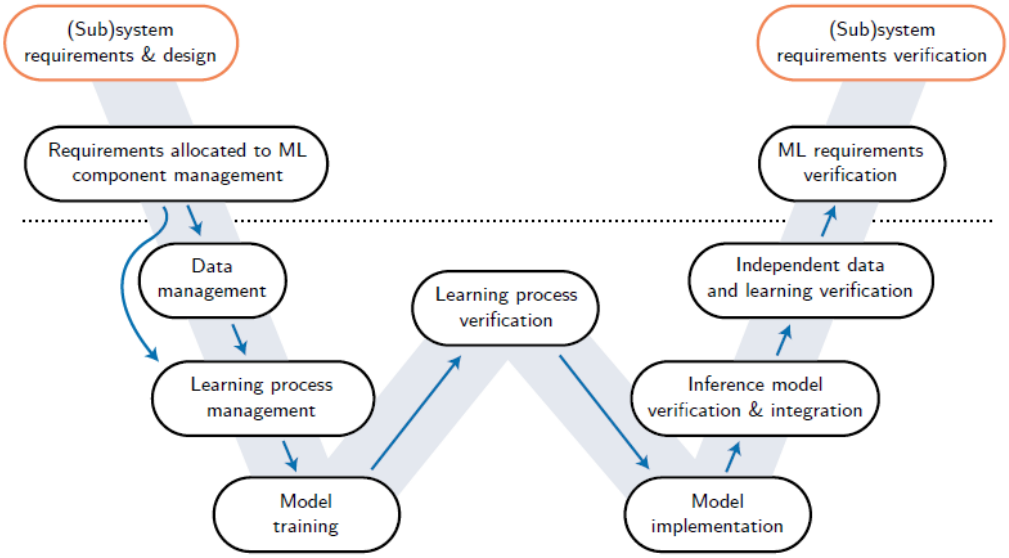

Figure 3.4: The W-shaped process.
(Source: CoDANN 1-2 [23][26])

The need to introduce a new assurance process comes from the unique nature of ML-based systems with respect to traditional ones. In fact, while traditional software development is based on human-written code, has a deterministic behaviour and follows human logic, the ML-based system, instead, mainly depends on the

statistical relationships between data the model was trained on. This results in a *computational graph* that cannot be inspected individually a posteriori, unlike human-written source code [27].
Therefore, the purpose of the W-shaped development process is to maintain control over ML system development so that performance guarantees on system outputs and behaviour can be achieved in a way comparable to traditional software.

3.3.1 W-shaped process steps

The path starts identically as for traditional systems from aircraft-level requirements until item-level, considered equivalent to ML component-level. From here, the process diversifies as explained in the following paragraphs[4].

Data management

This step is aimed at creating independent training, validation, and testing datasets that correctly describe the operational environment of the model. Since ML models are *data-driven*, data management represents the most important phase underlying the quality of the model.
Data has to be sufficiently numerous (*completeness*), representative and accurate (not presenting sampling or annotation errors).
To properly capture the real input probability space, i.e., data statistical distribution, there are two main techniques, which can also be combined:

1. Explicit operating parameters, suboptimal for higher-dimensional spaces;
2. Advanced data manipulation, e.g., dimensionality reduction (Principal Component Analysis - PCA [31]), suitable also for images and unsupervised learning.

The EASA Concept Paper addresses all the essential data management requirements in the LM-Objectives (**not under safety responsibility**).

Learning process management

This step concerns the setup of the model training, considering the training process and the training environment. The training process encompasses all the information and choices on the subject of:

- Model architectures, loss function, training parameters and hyperparameters (e.g., learning rate, batch size, number of epochs, etc.)

[4]Information regarding the W-process steps from [27] and [24].

- Performance metrics and their target values, derived from the high-level requirements of the previous step, "Requirements allocated to ML component management";

- Data augmentation.

The training environment, instead, concerns hardware and software to use for the training phase, which are completely different from the operational ones. Training requires high computational power and the capability to compute backward passes. Thus, requirements on the learning environment (*tool qualification*) are necessary. Dmitriev K. et al. [32] conducted an analysis on qualification aspects of ML-specific tools in the whole ML constituent life cycle.

Model training

In this step a family of ML models is firstly trained on the training dataset, then the most suitable one is chosen as the result of an optimisation on the validation dataset. While performing this process, it is necessary to provide traceable information and artifacts for reproducibility purpose.

Learning process verification

This is one of the most important steps for the safety assessment, as the selected model is tested to check whether it is able to return the intended behaviour or presents systematic errors or unintended properties. The desired model should meet the target on the metric(s) and guarantee robustness, stability and the absence of any biases or overfitting patterns.

This mid-process verification is what gives the entire process its characteristic W-shape.

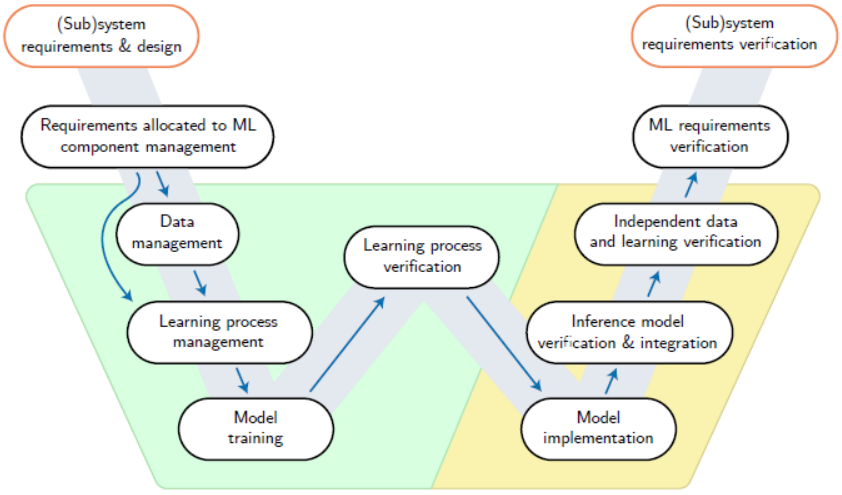

Figure 3.5: The learning (green) and inference (yellow) environments in the W-shaped process.
(Source: FAA and Daedalean [27])

All the mentioned steps until this point belong to the learning environment steps. Once this stage has been completed, the inference environments steps begins, as can be observed in Figure 3.5.

The activities in the right bracket of the W-shaped process involve the **model implementation** in the operational platform (i.e., the real-time safety-critical hardware and software in which the ML model is embedded), the **integration** with other system components and the **verification** that the MLC still delivers the expected behaviour. The passage to the inference environment usually requires performing optimisations such as operator fusion, pruning or quantization [26][5], while the inference model verification is based on *a posteriori* evaluation of the generalisation gap and test dataset considerations.

Finally, the **independent data and learning verification** is meant to close the data management life cycle, ensuring that data was correctly used throughout.

4 | The ML Safety Assessment process

This chapter, after a brief overview on machine learning to introduce the notations that will be adopted (Section 4.1), includes the proposed safety framework tailored for ML-based systems, in order to satisfy the EASA anticipated Means of Compliance, aMOC (see Section 2.3.1). After that, it is presented the ML Safety Assessment process for regression tasks.

4.1 What is Machine Learning?

4.1.1 Overview

AI is a field of computer science that includes a wide range of techniques and applications. According to the definition from the "Proposal for a regulation of the European Parliament and of the Council laying down harmonised rules on artificial intelligence" AI is a *technology that can, for a given set of human-defined objectives, generate outputs such as content, predictions, recommendations, or decisions influencing the environments they interact with* [33].

Figure 4.1 illustrates a taxonomy of all AI techniques available at the current state of the art. These days, one of the most popular sub-category of AI is machine learning (ML), in particular deep learning (DL).
ML *studies mathematical models that aim at achieving artificial intelligence through learning from data* [23]. This data can consist either of samples with labels (*supervised learning*), or without (*unsupervised learning*). Some modern models keep learning even during their operational phase (*continual/online learning*), comparing their response with real-world feedbacks, while others update their parameters to maximise a reward. This latter learning technique is called *reinforcement learning*. As anticipated in Section 1.4, this report will only take into account supervised learning and some aspects of the unsupervised one.

Given a complex function $f : X \to Y$, supervised learning aims at approximating f

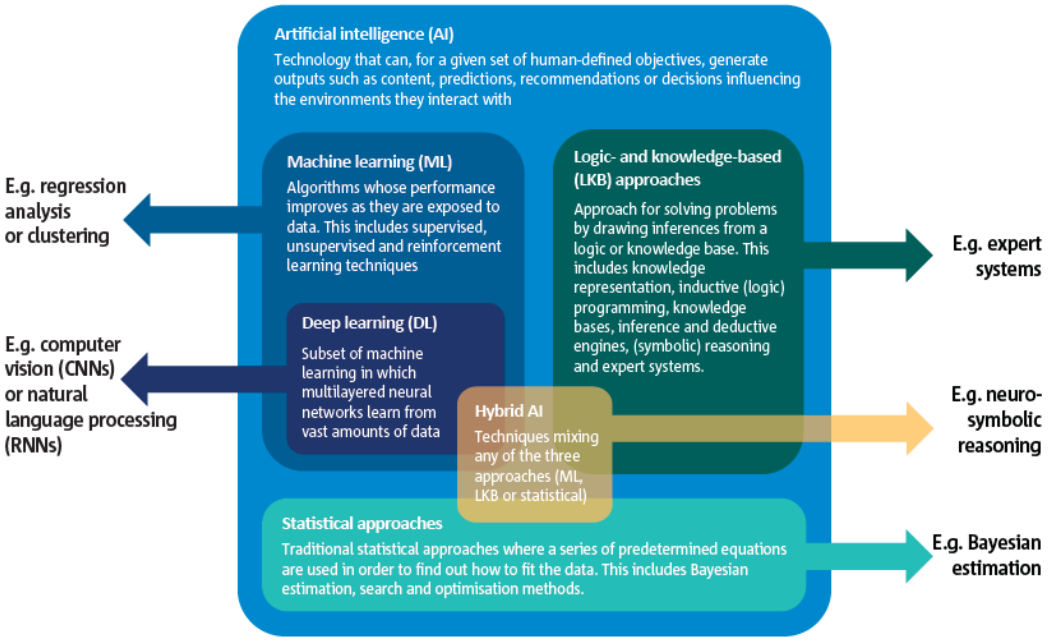

Figure 4.1: AI taxonomy.
(Source: AI Roadmap 2.0 [21])

with a model $\hat{f}$ derived from sample pairs $(x, f(x) + \epsilon_x)$: x is usually called *feature*, while $f(x)$, or y, *ground truth*. ϵ_x stands for the capture noise in collected data.

4.1.2 Different tasks and metrics

The pointwise quality of the approximation of the true function f by the model $\hat{f}$ is evaluated by performance **metrics** $m : Y \times Y \to \mathbb{R}$ demanding

$$m\left(\hat{f}(x)\,,\ f(x)\right)$$

to be closest to the target value (typically 0 or 1).

In ML there are two main types of task:

- **Classification**: when Y is a discrete set and the model assigns to each input $x \in X$ a "class" $y \in Y$ (to be more precise, a *soft score*: the likelihood for each category [34]).
- **Regression**: when Y is an infinite/continuous set, such as an interval $[a, b] \in \mathbb{R}$, and the model $\hat{f}$ estimates the continuous responses $f(x) \in Y$ to the inputs $x \in X$ that the function f represents. Regression was initially

born to understand the relationships between variables [35], but then in advanced modern applications it has been used to *predict* future values, to *spot anomalies* [36], or to perform complex computer vision tasks (e.g., bounding box regression) [37].

ML methods for regression include Linear Regression, Decision Tree Regression, Random Forest Models, Support Vector Regression Machines, Neural Network Regression, and others. Linear regression models use only input and output nodes to make predictions, while NNs also use hidden layers to make more accurate predictions [38].

Regression, by its very nature, is usually treated as a **supervised learning problem**; in fact, it requires a continuous target variable to predict. Without known values of the dependent variable Y it is not possible to "teach" the model which output to associate with which input data.

Depending on the application and the intended result, there are many different regression metrics but, according to Botchkarev [39], they all share three key components (dimensions) that determine the respective properties:

1. Method of determining point distance, $\mathbb{D}$;
2. Method of normalisation, $\mathbb{N}$;
3. Method of aggregation of point distances over a dataset, $\mathbb{G}$.

A generic formula can be written as follows:

$$m = \mathbb{G}^i\{\mathbb{N}^i[\mathbb{D}^i(y, \hat{y})]\} ,$$

where i refers to one of all available methods.

Table 4.1 shows a list of commonly used performance metrics for ML regression models. Green rows contain the top three metrics according to Botchkarev's survey (together with Mean Absolute Percentage Error - MAPE) [39]. "E" is the classical error value, computed by $(\hat{y} - y)$ for every sample in the chosen dataset, where $\hat{y}$ is the predicted output value by the model and y is the target output.

Metric	Formula	Description
Mean Error (ME)	$\mathrm{ME} = \frac{1}{n}\sum_{i=1}^{n} E_i$	It is the average of the simple amount of differences between a distribution and its true values. It is easy to apply and works with numeric data.

Metric	Formula	Description
Max Error	$\max\left(\lvert E_i\rvert,\ i = 1, \ldots, n\right)$	Max Error is either an absolute or a relative metric calculating the difference between a value in the input data and the corresponding value in the prediction from the AI system. The absolute Max Error is the maximal signed difference between a value in the input data and the corresponding value in the prediction from the AI system. The relative Max Error is the percentage of the width of the variation domain on which the AI system operates.
Mean Absolute Error (MAE)	$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\lvert E_i\rvert$	It measures the difference between two continuous variables. Uses a similar scale to input data and can be used to compare a series of different scales too.
Relative Absolute Error (RAE)	$\mathrm{RAE} = \sum_{i=1}^{n}\frac{\lvert E_i\rvert}{\lvert y_i - \hat{y}_{\mathrm{mean}}\rvert}$, $\hat{y}_{\mathrm{mean}}$ is the average value of the true labels.	Based on errors produced by a trivial model and works with numeric data. Need to handle carefully, since divisions by zero may occur (if true labels contain zeros).
Mean Relative Absolute Error (MRAE)	$\mathrm{MRAE} = \frac{1}{n}\sum_{i=1}^{n}\frac{\lvert E_i\rvert}{\lvert \hat{y}_i - \hat{y}_{\mathrm{mean}}\rvert}$	Based on absolute errors, it is more sensitive to outliers (especially of low values). Need to handle carefully, since divisions by zero may occur (if the true labels contain zeros).
Mean Squared Error (MSE)	$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}E_i^2$	Both MSE and RMSE are scale dependent. Models whose values are closer to zero present an adequate state. They are highly dependent on the fraction of data that is used (low reliability).
Root Mean Squared Error (RMSE)	$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}E_i^2}$	See MSE (scale dependent as well).

Metric	Formula	Description
Geometric Root Mean Squared Error (GRMSE)	$\text{GRMSE} = \sqrt[2n]{\prod_{i=1}^{n} E_i^2}$	GRMSE is also scale dependent. However, differently than MSE and RMSE, it is less sensitive to outliers.
Pearson Correlation coefficient (R)	$R = \frac{\sum_{i=1}^{n}(\hat{y}_i - \hat{y}_{\text{mean}})(y_i - y_{\text{mean}})}{\sqrt{\sum_{i=1}^{n}(\hat{y}_i - \hat{y}_{\text{mean}})^2 \sum_{i=1}^{n}(y_i - y_{\text{mean}})^2}}$	It measures the strength of association between variables. Values higher than 0.8 imply stronger correlations. As for R^2, it is the square of this one; values closer to 1 indicate stronger correlations too. Both R^2 and R work with numeric data.
Coefficient of Determination (R^2)	$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(\hat{y}_i - \hat{y}_{\text{mean}})^2}$	See R.
Scatter index (SI)	$\text{SI} = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_{\max(y)} - \hat{y}_{\max(\hat{y})}\right)^2}}{\hat{y}_{\max(\hat{y})}}$	Applied to examine whether RMSE is good or no, if its value is less than 1, then estimations are acceptable. It shows "excellent performance" when SI ≤ 0.1 and "poor performance" when SI > 0.3.
Performance index (PI)	$\text{PI} = \sqrt{\frac{\frac{1}{n}\sum(\hat{y} - y)^2}{1 + R}}$	It is an indicator for the evaluation of predictivity of a model. Lower PI values result in more accurate model predictions.

Table 4.1: List of commonly used performance evaluation measures for ML regression models.
(Source: MLEAP [5])

Even though these are the most common metrics, a considerable number of them has been criticised, or even rejected, during ML evolution. For example, Armstrong and Collopy stated that RMSE was not reliable, especially with time series [40]. Subsequently, Willmot et al. extended the previous conclusion to all squared error measures and strongly advised in favour of using MAE [41][42]. Further articles found that even RAE, MAPE, R and R^2 were *unreliable selection criteria* [39].

Most of researchers commonly agree on the fact that there is no need to try to find a single best metric, since *no single measure is universally best* [43]. In fact, the modern approach involves using a **multi-criteria process**, incorporating

traditional metrics (i.e., the commonly used ones) as primary indicators and designed objective functions tailored for the specific application [5].

4.1.3 Generalisability

Machine learning models are usually learned on so complex data that it is unrealistic to expect the model to be able to perfectly capture the inner relationships among data.
The architecture of the model $\hat{f}$ could be not complex enough to properly deliver the intended behaviour (*underfitting*) or, on the opposite way, could be so high to allow the model to just memorise by heart the dataset it has been trained on (*overfitting*). This is called the "Bias-Variance Trade-off" and it is a crucial factor in the generalisation capability - or *generalisability* - of the ML model, which is the ability to perform well on unseen data.

Literature usually refers to the **generalisation gap** of a model $\hat{f}$, with respect to an error metric m and a dataset D. The gap is illustrated in Figure 4.2 and defined as the difference:

$$\mathrm{GAP}(\hat{\mathrm{f}}, \mathrm{D}) = \left|\mathbb{E}_{\mathrm{out}}(\hat{\mathrm{f}}, \mathrm{m}) - \mathbb{E}_{\mathrm{in}}(\hat{\mathrm{f}}, \mathrm{D}, \mathrm{m})\right|,$$

where $\mathbb{E}_{in}$ is the empirical loss calculated on the known datasets (i.e., learning, validation, test) and $\mathbb{E}_{out}$ is the expected loss on unseen data that can only be estimated. One of most researched fields of ML is the one about deriving accurate

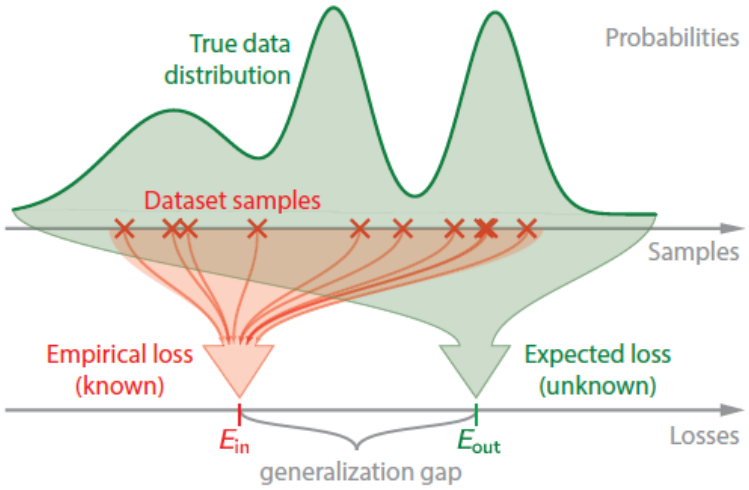

Figure 4.2: In-sample error $\mathbb{E}_{in}$ (empirical loss), out-of-sample error $\mathbb{E}_{out}$ (expected loss), and the generalization gap between them.
(Source: CoDANN I [23])

generalisation gap, given a model. Literature provides an enormous quantity of different techniques, that can be data- or algorithm-dependent, but the almost totality of them are either narrowly-applicable, or present scalability issues, or provide too much conservative bounds [5].

From a practical perspective, there are two main methods to provide generalisation bounds:

- Training/model complexity approach, based on theoretical assumptions, like Rademacher complexity [44][45] or PAC-Bayes bounds [46][47];
- Validation evaluation-based approach, grounded on empirical computations on the validation dataset [48][49].

Although this is a crucial topic for ML, this section does not offer further related information because **estimating generalisability is not under Safety responsibility**. However, Safety requires using generalisation evaluations to verify that there is sufficient margin on the model performance, as will be shown in Section 4.7.

4.1.4 Parametric algorithms

Many applications, especially neural networks, are founded on parametric machine learning algorithms. The general methodology of parametric supervised learning, while still following the W-shaped development process (see Section 3.3), is to:

1. Collect representative datasets. Considering the training dataset:

$$\mathrm{D}_{\text{train}} := \{(x_i, f(x_i) + \epsilon_i) : i = 1 \ldots, n\}$$

   The presence of ϵ_i denotes that it is not often possible to collect the actual ground truth and samples contain small capture or labelling errors.

2. Choose one model family $\hat{f}_\theta$, parametrized by θ.
3. Apply a supervised learning algorithm to find θ so that f well approximates $\hat{f}$, i.e., E_{in} on $\mathrm{D}_{\text{train}}$ is small.
4. The evaluation is repeated on the validation dataset to get a more realistic measure.
5. If results are not acceptable, point 2 and 3 have to be repeated in a iterative process until satisfactory outcomes.
6. Choose the best candidate model $\hat{f}$ from the previous steps. The decision is made on performance and generalisation considerations.

7. Evaluate how the model performs on unseen data using the test dataset.

In Figure 4.3, it is possible to visualise the general problem of model selection. There are three model families, indicated by the coloured lines, and each point along these curves corresponds to a particular model obtained from estimating the respective parameters on the training dataset [50]. The green dot, instead, represents the true model f and its distance by the machine-learned model highlights the bias.

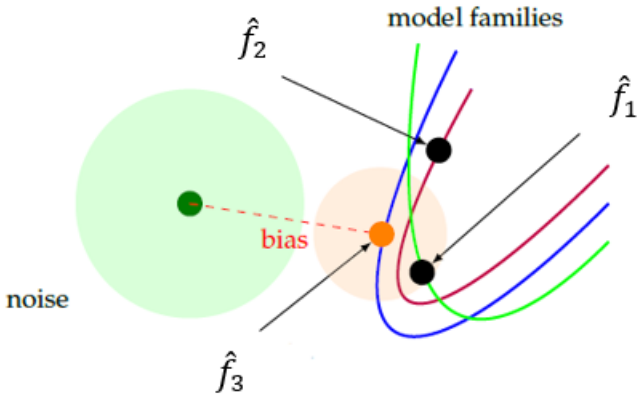

Figure 4.3: Idealized visualization of the model selection process. (Adapted from: Emmert-Streib and Dehmer [50])

4.2 DAL allocation

The aMOC-SA-01-1 related to DAL allocation does not introduce new activities; therefore, the allocating process follows the existing DO-178C [1] standard, in accordance with what was explained in Section 3.2.2; the only exception is that **no reduction** is permitted.

It is interesting to consider that the development assurance process also includes testing activities or specific requirements on traceability and explainability, which may not be extendable to machine learning algorithms. Dmitriev et al. examined for a DAL C [51] and a DAL D [52] use case the applicability of DO-178C objectives to their NN-based system and provided for each objective a compliance analysis; when they encountered no correspondence between traditional and AI requirements, new considerations were presented.

4.3 Exceedance Rate

The metrics illustrated in Section 4.1.2 are **aggregated** performance indices. This means that they are only able to grasp the general trend and provide guarantees of the average model behaviour. For example, a regression model with a very promising value of MAE might have good values on average and critical ones locally, which is unacceptable for safety purposes. Therefore, there is a strong need for a metric consistent with safety requests.

Inspired by current aircraft monitoring methods, that depend on exceedance detection with predefined thresholds to identify anomalous behaviour [36], it is proposed the **Exceedance Rate (ExR)** metric. The definition of the ExR is:

$$\mathbf{ExR}(\tau) = \frac{\sum_{i=1}^{|D|} \mathbf{1}_{\{|E_i|>\tau\}}}{|D|}, \quad with \quad \mathbf{1}_{\{E_i>\tau\}} = \begin{cases} 1, & \text{if } |E_i| > \tau, \\ 0, & \text{otherwise.} \end{cases}$$

where the involved variables have the following meaning:

- **E** is the classical error value;
- $|\mathbf{D}|$ is the size (i.e., total number of samples) of the dataset under consideration;
- $\boldsymbol{\tau}$ is the *safety threshold*;
- $\mathbf{1}_{\{|E_i|>\tau\}}$ is the indicator function.

Essentially, this metric counts the number of times that the error, defined in a suitable way for the considered variables, exceeds a certain safety threshold, and then everything is divided by the dataset size. The result, converted into percentage, is an **empirical measurement of the failure probability** of the model under examination, based on the following assumptions divided by macro-themes:

- Data and Operative Design Domain (ODD)
    1. The datasets used are complete and representative (under data management objectives, in particular aMOC-DM-07-1 and DM-07-2 [13]);
    2. The ground truth is accurate and presents minimal labelling errors (under data management objectives, in particular aMOC-DM-07-3 [13]);
    3. The domain of the model is stationary in time and there is no concept drift, i.e., the relationship between dependent and independent variables does not change over time (under Continuous Safety Assessment objectives, in particular Objective SA-03 [13]).

- Failure Definition

    4. The safety threshold τ is well-calibrated, so that exceeding this value actually results in a failure. In this way, the estimated probability coincides with the model failure.

So, the term "failure" refers to the exceedance of the safety threshold τ by the regression model. The ExR is a value on average as well, but the outcome is more

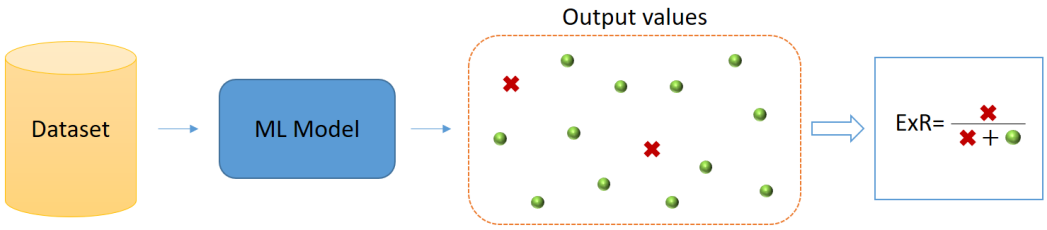

Figure 4.4: A simplified visualisation of how Exceedance Rate works: green dots represents acceptable outputs, red crosses values higher than the threshold.

controllable since it derives by safety considerations and it allows to detect critical zones (i.e., subgroups in the dataset characterised by higher exceedance rates).

Although this is a single metric, it is able to adapt to different tasks and applications in a flexible way. In addition, it does not limit the model training process, since it is possible to use whichever loss function preferred in the optimisation stage.

The definition of the Exceedance Rate satisfies the aMOC-SA-01-2 (see Section 2.3.1) related to metrics definition and aMOC-SA-01-7 for the probability estimation based on performance metrics.

The following step will be to impose a quantitative requirement on this metric, so that model safety can be partially guaranteed already during the learning process verification phase, allowing the model developer to be decoupled from the future implementer. In this way, applicants can already request the necessary specifications from their suppliers without necessarily having to wait to integrate the model into their operating environment.

4.4 Exposure to Data outside the Operational Design Domain (OOD)

According to aMOC-SA-01-3 the system should be able to recognise when input data is outside the OD or ODD; when the system is in this condition, it should put in place mechanisms to allow the delivery of the intended function, such as the

fallback to a traditional equivalent system.

The Concepts of Operations definition (ConOps) plays an essential role in the identification of OODs, since it describes the operational scenario with its respective limits and corner cases. This document is not part of the safety activities; instead, it is addressed to the Characterisation Objective CO-04 of the EASA CP [13] and it is, in fact, an input of the safety process.

Based on the information deriving from the ConOps, it is necessary to detect in real-time when the input is OOD. Methods to perform the OOD detection may include, but are not limited to, the following two methodologies:

1. **Explicit operating parameters** - If the application allows to define an explicit operating space (OS), based on a finite number of operating parameters $\varphi_i \in P_i$, where P_i is a compact interval, it is possible to outline the area in which the MLC is designed to work.

   $$OS = P_1 \times P_2 \times \cdots P_n$$

   Once the OS has been defined, a traditional software can be developed to monitor if input data is nominal. Unluckily, this approach suffers from human bias, because the OS is determined based on an expert judgement that could not be enough accurate to catch all the parameters taken into account by the MLC logic in its *black box* behaviour.

2. **Distribution Discriminator** - A high-fidelity AI component, named *input distribution discriminator*, accompanies the main model $\hat{f}$ and provides a value in the [0,1] interval: it returns values closer to 0 if the processed data is OOD, and vice versa. The discriminator needs to be trained on another dataset, named "out-of-distribution dataset" D_{ood}, that includes data outside the ConOps.

   This method has the criticality to introduce another AI component in the process, but the advantage is that is not subject to human bias.

Regardless of the strategy selected, it is then necessary to derive qualitative and/or quantitative requirements, to justify the assumptions and choices, and to verify the related accuracy.

4.5 Uncertainties

Models must not only be accurate, but one would also like to get the associated uncertainties under control.

The aMOC-SA-01-4 and aMOC-SA-01-5 guide the identification, listing and assessment of the uncertainties. Throughout the EASA CP [13], the following taxonomy is considered as follows:

- *Epistemic uncertainty* refers to the deficiencies due to lack of knowledge or information. In the context of ML, epistemic uncertainty corresponds to the situation where the model has not been exposed to data adequately covering the whole ODD or where the ODD definition needs to be refined or completed.
- *Aleatory uncertainty* refers to the intrinsic randomness in the data. This can derive from data collection errors, sensor noise, or noisy labels. In this case, the model has learnt based on data containing uncertainties.

Epistemic uncertainty can be reduced adopting more meticulousness in the model development, therefore it is addressed trough the learning assurance objectives, and is not under safety responsibility. The aleatory uncertainty instead, cannot be deleted, but should be minimised to the extent practical; to achieve this goal, a series of mitigations can be introduced.

According to the already proposed method by [12], the assessment of the uncertainties has to be performed with a Table including the elements shown in Table 4.2.

AI-CONSTITUENT	The AI/ML constituent affected by the uncertainty.
SOURCE	Source of uncertainty.
DESCRIPTION	Brief description of how the source of uncertainty affects the system.
TYPE	Epistemic (E) or aleatory (A).
MITIGATION	Description of the selected mitigation, only if the uncertainty is aleatory.

Table 4.2: Table entries for the uncertainties assessment.

A large number of theoretical methods, namely *uncertainty quantification* (UQ), allow to get even an estimation of the uncertainty associated to the model output; in this way, the uncertainty of each prediction is quantified, overcoming the limitations imposed by the aggregated accuracy metrics. However, due to this variety, **it is not possible to derive general quantitative requirements on uncertainties**; the Safety Assessment depends on the selected technique and the application. The general underlying idea is to identify a *safety threshold* which, once exceeded, establishes that the model's response is too uncertain and cannot be accepted.

For instance, some valid methods can be:

- Generation of *confidence intervals* (CIs) or *prediction intervals* (PIs), which are techniques that provide a lower and upper bound for each prediction and

assurance that, with some high probability (e.g., 95% or 99%), the realised data point will fall between these bounds [53].

PIs are usually less narrow than the CIs, because they also capture the aleatory uncertainty and, some more advanced methods, even the *heteroskedasticity* of the noise. The three primary methods to quantify uncertainties with bounds, especially in regression with NNs, are:

  - The Delta method;
  - Mean variance estimation (MVE);
  - The Bootstrap;

- Based on the variance of the output of an *ensemble* of models, trained with different initialisations, it is possible to estimate the aleatory uncertainty (for example, with the Mahalanobis distance) and the epistemic one [27].
- Statistical methods, such as *dynamic Bayesian networks* (DBNs) [4].
- Indicators to assess the reliability of ML models with respect to the provided prediction on the input x. The indication is derived from another high-fidelity model with this unique purpose [54].
- Use of a Kalman filter that provides the estimated uncertainty quantification via the state-error covariance matrix.
- Monte Carlo simulations.

4.6 Failure Modes Evaluation

The aMOC-SA-01-6 concerns the establishment of AI/ML constituent failure modes and the evaluation of associated detection means.

Considering regression tasks, the identified failures plausible for a ML model are:

1. **Loss of function**: the model does not respond;
2. **Misleading function**: the model returns a value that is unacceptably distant from the target.

   In addition to the aspects listed, the presence of OOD input data contributes to this functional failure and can be considered a part of the development process as well, since it derives by the ODD definition.

   In some scenarios, it is sure worth analysing the *frozen output*, a subcase of misleading in which the model continuously returns the same value without updating, characterised by different triggering factors and detection means.

The most appropriate approach to establish the functional failures of AI/ML-based systems is the already existing Functional Hazard Assessment (FHA), regulated by the before mentioned ED-135/ARP4761A [16].

This activity, conceived before the consolidation of AI technologies, is also well suited to AI-based systems, as shown in [27][3][55]. In detail, the safety activity of interest is the System Functional Hazard Assessment (SFHA), because MLC spans from the system level to the item level. When performing the SFHA, the procedure follows exactly the traditional one; there is only one element which is added to the table entries: a column reporting "Function provided by AI/ML constituent?", that can be filled with "YES" or "NO", as shown in Table 4.3.

Function Provided by AI/ML constituent?	YES or NO depending on whether there is an AI-based item contributing or not.

Table 4.3: New SFHA Table entry.

To distinguish if a hardware or software component belongs to the MLC, the applicant should validate his choice based on the MLC definition offered by the EASA CP [13]. If in this last column the answer reported is "NO", the process continues along its standard path; otherwise, it is necessary to follow the Safety process specially tailored for AI-systems proposed in this thesis.

In case it is important to highlight some aspects at item-level, whether it is the type of the components (HW or SW) involved or the effect of the duration of the failure on its severity classification or to provide some subcases of the identified system failure, it can be useful to perform a table that reaches a lower level in the aircraft architecture, named AI Table. As mentioned, this table can contain every aspect that can be useful to further characterise the model and its impact on the overall safety.

4.7 Generalisation bound

According to aMOC-SA-01-8 "link between generalisation bound and safety assessment", when assigning quantitative requirements to the model metrics, it necessary to account the performance degradation of the model in the inference scenario by adding a margin to the metrics: the *generalisation bound*. This margin is the upper limit of the generalisation gap (see Section 4.1.3) and it quantifies its impact on

the Safety Assessment. By definition this bound $\boldsymbol{\varepsilon}$ is:

$$|\mathbb{E}_{out} - \mathbb{E}_{in}| \leq \boldsymbol{\varepsilon}$$

For the proposed framework, the in-sample-error $\mathbb{E}_{in}$ is the Exceedance Rate ExR, while the out-of-sample-error is the real failure probability p_{fail} and can only be estimated.

As mentioned in Section 4.1.3, the generalisation bound can be obtained with a variety of techniques. Instead of directly computing $\mathbb{E}_{out}$, it is usually derived ε by calculating the probability that the gap value is smaller than the higher bound, with a **confidence tolerance $1-\boldsymbol{\delta}$**.

$$\mathrm{P}(|p_{fail} - ExR| \leq \varepsilon) \geq 1 - \delta$$

The generalisation bound is evaluated per Objective LM-04 and is not under safety responsibility; however, the safety assessment determines the value of the significance level δ to be guaranteed.

The value obtained of ε has to be converted as percentage to fit with the proposed metric.

Table 4.4 illustrates the safety objective of this value, based on [12].

Parameter	**Safety Objective**
Significance level δ	0.0001
$p_{fail} \leq ExR + \varepsilon$	

Table 4.4: Generalisation requirements.

When performing the Safety Assessment, it is required to the p_{fail} to be less probable than a specific percentage p_{req}. In order to account the generalisability problem it is necessary to respect the following inequality in the learning verification step:

$$\mathbf{ExR \leq p_{req} - \boldsymbol{\varepsilon}}$$

It is worth noting that the safety objective on the significance level δ also indirectly constitutes a requirement on the size of the datasets.

Section 4.8 will show the quantitative requirements on the metric with the generalisation bound mentioned.

4.8 Proposed quantitative requirements

This section presents three quantitative requirements that the candidate model has to satisfy to be considered acceptably safe, covering the **misleading functional failure**; there are two requirements on the Exceedance Rate ExR, one global and one local, and the last one is on the safety threshold τ. The requirements are adapted according to the most severe risk associated to the misleading functional failure of the MLC. At the moment, the EASA Concept Paper does not allow to implement AI systems whose failure would result in Hazardous or Catastrophic failure effects. Therefore, the proposed framework focuses on the same applicability, but, at the same time, envisages the riskier scenarios to address future guidelines.

4.8.1 First requirement: global metric

Starting from the global requirement, this is the one that evaluates the model as a whole, because it is computed on all the samples in the chosen dataset. In detail, the compliance with this requirement must be proven on the **test dataset**. Table 4.5 illustrates the proposed *upper bounds* for the Exceedance Rate depending on the severity of the application.

Global Requirement on ExR					
Failure Severity	NSE	MIN	MAJ	HAZ	CAT
ExR upper bound	-	$10\% - \varepsilon$	$5\% - \varepsilon$	$3\% - \varepsilon$	$1\% - \varepsilon$

Table 4.5: Global requirement. Green cells are the ones considered by the EASA Concept Paper.

The ε that appears in Table 4.5 is the generalisation bound, which accounts the degradation of performance of the model in the inference scenario (see Section 4.7).

The chosen values are quite ambitious, considering the actual state of the art, yet still credibly demonstrable and reachable with a well executed learning process. At the same time, this value does not represent a stand-alone safety objective; rather, it is a constituent-level constraint that is combined with architectural mitigations (monitoring, fallback functions, human oversight, etc.), other requirements, and operational limitations. Hazardous and catastrophic classes may require further investigation.

Guidelines for error function definition

[ERR-01] Error definition The error E is evaluated comparing the model output $\hat{y}$ with respect to the ground truth y; an accurate definition of this function is essential to capture risk in the right way.

$$\mathbf{E} = e\left(\hat{y}(x)\,,\, y(x)\right)$$

The classical error function for symmetrical scalars, which can be used as reference for the majority of the problems, is defined as:

$$e = |y(x) - \hat{y}(x)|$$

[ERR-02] Error transformations Monotonic transformations $g(\cdot)$ on the reference error e should only be used for training/diagnostics. The ExR metric is always computed on the base error e or on $g(e)$ with threshold $g(\tau)$, demonstrating the equivalence.

4.8.2 Second requirement: local metric

As previously introduced, aggregated metrics fail to capture local dangerous trends, since their evaluation is on average. Based on this concept, it is necessary to provide quantitative assurance at the local level on those subgroups of the dataset where the MLC struggles to provide an acceptable behaviour, namely *critical bins*. Considering that, inevitably[1], in these subgroups the MLC will return borderline values, the aim of this requirement is to accept the unavoidable decline in performance, but, at the same time, impose safety margins so that the error does not "explode". These relaxed values are justified by the limited exposure time.

Table 4.6 shows the proposed quantitative requirement for the Exceedance Rate at the local level, depending on the severity of the application.

Local Requirement on ExR					
Failure Severity	NSE	MIN	MAJ	HAZ	CAT
ExR upper bound	-	15%	7.5%	4.5%	1.5%

Table 4.6: Local requirement. Green cells are the ones considered by the EASA Concept Paper.

[1]The ML constituent is less likely to work in these corner cases (e.g., strong gust), thus it has been trained on less samples covering those scenarios; the datasets follow the expected distribution of the operational domain (representativeness objective).

Hazardous and catastrophic classes may require further investigation.
If this local requirement is not met in the critical bins, it is mandatory to narrow the selected ODD by excluding the areas where model behaviour is not acceptable, **or to return to the learning/development process** and add further constraints.
The generalisation gap is not considered in this requirement, because the aim of this constraint is to spot areas in which the model does not perform well, not assuring the global behaviour of the model; furthermore, there is the risk that, to obtain a small generalisation gap, the samples in the bins will have to be increased, thus changing the actual representativeness of the dataset.

Guidelines for critical bins identification

Before the training phase, the ODD has already been defined according to Objective CO-04 [13].

[BIN-01] ODD bin taxonomy Before the model selection, the bin taxonomy must be defined based on Objective CO-04 output. The subdivision should be done starting from the explicit parameters or, if not possible otherwise, using more complex criteria (e.g., part of the day, visibility, image metrics such as brightness, contrast, etc.), or a combination of them.

[BIN-02] Identification of critical bins The rule must be established a priori. Valid methods:

- Choice 1 - *Error percentiles*: a bin is considered critical if its Exceedance Rate is equal or grater than the **80th percentile**.
- Choice 2 - *Feature space characterization* models, such as equivalence partitioning, centroid positioning, boundary positioning, pair-wise boundary conditioning [5]: a bin is considered critical if its coverage is lower than **5%**.

[BIN-03] Temporality The computations related to critical bins can be performed iteratively during the development phase for exploratory purposes. Once the model has been chosen, bins subdivision must be frozen and documented. At last, the quantitive requirement allocated must be verified on the test dataset.

> This box illustrates the steps of the first method usage with an example of an AI model having images as input.
>
> 1. Bin taxonomy: bins subdivision is made in this example using complex variables, such as the part of the day and the visibility conditions;

2. Model training;
3. Application of the model on the validation dataset $|D|_{val}$;
4. ExR computation for every bin identified;
5. Bin aggregation and distribution determination;
6. Model choice and bin subdivision freezing;
7. Verification on test dataset.

Assuming the following Table encompasses the ExR evaluation for each bin after having selected the model, the intended procedure is shown below. The "$|D|_{val}$ *Samples*" column represents the number of samples of the validation dataset contained in each bin.

Bin	**Conditions (light / visibility)**	**$\|D\|_{val}$ Samples**	**EXR**	**Percentile**	**Label**
B1	day / >5km	3333	0.3%	10	Nominal
B2	night / >5km	2500	0.8%	30	Nominal
B3	dusk / 1-5km	1000	1.4%	50	Nominal
B4	night / 1-5km	200	2.5%	70	Nominal
B5	dusk / <1km	155	3.1%	80	Critical
B6	night / <1km	96	4.8%	100	Critical

The local quantitative requirement must be verified on B5 and B6 critical bins.

4.8.3 Third requirement: operational tolerance

The third quantitative requirement concerns the choice of an adequate safety threshold, which is essential in the definition of the failure, intended as the exceedance of that limit. The safety threshold cannot be greater than the operational tolerance L:

$$\boldsymbol{\tau} \leq L$$

Guidelines for the operational tolerance definition

Given that machine learning is still a new technology, the operational tolerance must be defined as follows:

- **Case 1** - Any applicable *regulations* must be observed, where present: for example, Performance-based Navigation Manual (PBN)[56], or Minimum Operational Performance Standards (MOPS). If not present, consider Case 2.

- **Case 2** - The tolerances of a *traditional equivalent system* should be used as a reference, if available. If not available, consider Case 3.

- **Case 3** - If none of the above cases applies, the operational tolerance must be defined based on *engineering judgement* and agreed with the aviation safety Agency.

  Two methods could be taken into account for this purpose:

  1. Deriving the operative tolerance by the system margin, i.e., maximum permissible amplitude of the variable of interest Δ_{sys}^{max}:

     $$L \leq L_{sys} = \frac{\Delta_{sys}^{max}}{k},$$

     where k is how much the model error is amplified (i.e., maximum sensitivity). k can be obtained from a sensitivity analysis; if the model output is processed by a function $A(\cdot)$, k can be computed as the Jacobian of $A(\cdot)$ [27].

  2. Considering that the slightest error that can be made is the irreducible aleatory error σ (i.e., the variance of the intrinsic noise in the data, representing the minimum achievable prediction error even for the optimal model [57]), a $k-th$ multiple of this latter variable could be considered as upper limit:

     $$\sigma \leq L \leq k\sigma$$

4.8.4 Closing Remarks

This subsection summarises and groups together the quantitative requirements proposed throughout the Section (4.8), considering only safety critical scenarios (i.e., ML models whose worst misleading failure does not lead to "No Safety Effect") and excluding the ones at the moment not considered by the EASA CP, since too risky for the actual state of the art, namely Hazardous and Catastrophic scenarios.

The proposed quantitative requirements have to be respected following the guidelines provided.

These quantitative requirements allow to assure that the development process was carried out thoroughly and the model, **with regard exclusively to the learning process**, can be considered acceptably safe.

Worst Failure Severity of the MLC	MINOR	MAJOR
Global Requirement	$ExR^{global} \leq 10\% - \varepsilon$	$ExR^{global} \leq 5\% - \varepsilon$
Local Requirement	$ExR^{bin} \leq 15\%$	$ExR^{bin} \leq 7.5\%$
Threshold Requirement	$\tau \leq L$	

Table 4.7: Summary of proposed quantitative requirements.

To ensure the complete safety of the model, it is necessary to implement mitigations and architectural strategies to also address failures deriving from hardware components (e.g., redundancy) or uncertainty in the model output (e.g., UQ).

Further crucial key properties to guarantee the model delivers the intended behaviour are the stability, robustness, and timeliness requirements, which are part of the learning assurance objectives (not under safety responsibility).

4.9 Verification

The aMOC-SA-01-9 concerns the Safety Assessment of the MLC.

According to Goodloe (NASA) [58], there are five main approaches for the assurance of ML-based safety-critical systems:

1. *Testing* remains the most established method.

2. *Formal methods* aim to mathematically verify ML models by encoding their behaviour as constraint-solving problems, or by applying abstract interpretation techniques; however, these methods are currently valid only for small networks.

3. *Runtime verification* monitors system behaviour during operation to detect violations of predefined safety properties; though, it still requires well-defined specifications to be effective.

4. *Explainability* encompasses post-hoc analysis, developer-focused debugging tools, and user-level explanations; its main problem relies on the lack of concrete proofs.

5. The fifth one proposes to *license* ML components as if they were humans, but at the moment it is just a speculative approach.

For the proposed framework, the main activity planned is testing combined with a runtime monitor. The applicant is asked to verify that everything that has been implemented complies with safety requirements. In detail, the list of necessary verifications regarding ML-based systems is provided below:

- Requirements related to the allocated DAL are met;
- Performance requirements, evaluated at item level during learning process verification, both global and local, are met;
- It is reached the required significance level when calculating the generalisation bound;
- Independence requirements are met (especially if an *ensemble* of models is implemented);
- All sources of uncertainties are identified and all the aleatory ones are opportunely mitigated;
- The architecture to detect OODs and the associated strategies adopted to assure the continuity of the desired functions are presented.
- Random hardware failures are analysed with an opportune propagation error method and related safety objectives are met according to the reference CS;
- All the assumptions made during the development process are validated.
- When integrating the ML-item in the operational environment, the MLC performance needs to not deteriorate any more than already foreseen in the generalisability step. The assessment should be based on the generalisation gap and the evaluation of the integrated model on the test dataset;

4.10 Updated Safety Process

In this Section, the activities previously presented are allocated throughout the Safety Assessment Process and integrated with traditional ones.

4.10.1 Aircraft Functional Hazard Assessment (AFHA)

The H/C FHA is performed as per ARP4761A [16]: the introduction of AI-based systems has no impact on the process at H/C level.

4.10.2 Aircraft Preliminary Safety Assessment (PASA)

The PASA is performed as per ARP4761A [16]: the introduction of AI-based systems has no impact on the process at H/C level.

4.10.3 System Functional Hazard Assessment (SFHA)

The activity follows the structure of the traditional process as per ARP4761A [16].

The updated SFHA Table is characterised by an additional column that indicates whether the function is provided by an AI/ML constituent. The remaining part of the Table preserves the traditional layout. If the answer to the new AI column of a specific functional failure is "NO", no further action is required; otherwise, an additional step exploring AI/ML-item failure modes and associated detection means has to be performed.

4.10.4 Preliminary System Safety Assessment (PSSA)

The PSSA examines the proposed system architecture and determines the following requirements for AI/ML items:

- IDALs allocation;
- Requirements on performance, especially those related to performance metrics;
- Hardware failure rate targets to meet the quantitative safety objectives;
- Specific assumptions and criteria for the detection of out of distribution inputs;
- Qualitative and/or quantitative requirements for the identification, quantification, and mitigation of uncertainties.
- Independence requirements.

4.10.5 System Safety Assessment (SSA)

The updated SSA has to show compliance to the new AI/ML requirements previously outlined in the PSSA.

In addition, proofs demonstrating that the AI/ML-item still delivers the intended behaviour even when integrated in the operational environment have to be provided. The assessment of this step should be based on generalisation gap calculation, and model performance evaluation on the test dataset.

4.10.6 Aircraft Safety Assessment (ASA)

The ASA is performed as per ARP4761A [16]: the introduction of AI-based systems has no impact on the process at H/C level.

5 | Use case

This chapter puts into practice what has been proposed throughout the thesis, demonstrating its applicability with an example: the DL-based system called *Runway Alignment System* (**RAS**), which embeds the Trajectory Change Predictor (**TCP**) machine learning constituent[1]. After a brief description of the system, **for the sole purpose of providing context for the safety analysis**, the core part will be addressing the aMOC of the Objective SA-01 [13], following the proposed framework.

5.1 Introduction

A very consistent percentage of accidents involving airplanes and helicopters occur during landing [59], the majority of which surprisingly happen in Visual Meteorological Conditions (VMC) [27]. This is the reason why the proposed use case aims to help pilots aligning with the runway, to reduce human errors during the landing phase, and improve safety.

During daytime VMC flight under Visual Flight Rules (VFR), another AI-based system, the VLS [12], detects hard-surface runways in the field of view, and enables the operator to choose the runway intended for landing or to rely on a pre-configured selection derived from the flight plan [27].

Recognising the runway acts as a switch that turns on the Runway Alignment System (RAS), and within it the TCP machine learning constituent; once a runway has been selected and once the aircraft begins its final descent towards it, the TCP provides **latitude and longitude variations** over a time horizon of 10 seconds as the model output and, after post-processing, a correction vector towards the runway centreline.

Table 5.1 summarises the expected RAS functions.

[1]Although the model is based on deep learning (DL), in order to remain consistent with literature, the more inclusive term "Machine Learning Constituent" (MLC) will be used.

The MLC interfaces with a traditional out of distribution data (outliers) detection software, which ensures that input data satisfy the conditions where system performance guarantees hold. Further improvements of the TCP could concern the use of a combination of classical filtering and tracking software components.

Flight Phase	Function
Cruise	Receive confirmation of runway detection.
Descent	Provide landing guidance with latitude and longitude variations.
Approach	Stop when radar altitude with respect to runway = 50 ft.

Table 5.1: RAS functions w.r.t. flight phases.

5.2 Characterisation of the ML application

According to the EASA CP, among the first steps to perform when developing an ML item there is the characterisation of the AI/ML application. To do so, there is a series of Characterisation Objectives CO [13] to comply with. These objectives are not under safety responsibility, but they are important to perform a correct Safety Assessment.

Objective CO-01: Identifying the list of end-users intended to interact with the AI-based system, the roles and expected skills.

- Primary end-users: pilots (1^{st} and/or 2^{nd}).
- Teaming: cooperation (*advisory guidance*).

Objective CO-02: Goals and high-level tasks identification.

- Pilots' goal:
  - Follow a stable trajectory towards the runway;
  - Anticipate short-term trajectory changes.
- Pilots' tasks:
  - Monitor aircraft's latitude and longitude variation: Δlat , Δlon;

  - Cross-check the variations with Flight Management System (FMS);
  - Apply minor corrections.

Objective CO-03: AI-based system definition.

Inter-related items that constitute RAS:

- An MLC consisting of pre-processing, DL-model and post-processing. The MLC provides aircraft's latitude and longitude variations;
- A traditional software component performing OOD detection;
- The system interfaces with the Runway Detector [12](part of the VLS) and takes as input if it has found the runway (0 or 1);
- A database containing runway information;
- Displaying and indicating subsystem.

Objective CO-04: Concepts of operations definition and documentation.

The Concepts of Operations (ConOps) is reported in the following Table.

	Operational Concept
Application	ML-based landing advisor
Aircraft Category	Large Rotorcraft (CS-29)
Flight Rules	Visual Flight Rules (VFR) in daytime Visual Meteorological Conditions (VMC)
Level of Automation	Pilot assistance (1A)
System interface	In-cockpit display
Configuration	Stable Approach

Table 5.2: Concepts of Operations.

The objective also asks to define the Operational Domain (system level); details are provided in Table 5.3.

	Operational Domain
Activation Range	RA $3000 \div 50$ ft AGL Disabled if out of ODD
Flight Phase	Descent and Approach
Aircraft Equipment	GNSS ($RNP = 0.3\,NM$) , AHRS, RadAlt, ADU, Radar Weather, NAV-computer
Weather	Absence of wind shear
Special considerations	no ILS, marked concrete runways, Compatibility with VLS
Time of Day	Daytime, Sun higher than 5° above horizon
Time of year	Every season

Table 5.3: Operational Domain.

In order to be more detailed and to define the limits for the OOD monitor, it is provided in Table 5.4 the Operational Design Domain (ODD) as a series of explicit parameters.

Parameter	**Description**	**Unit**	**Source**	**Range**
MLC ODD				
Runway QFU	Magnetic Orientation	deg	Runway DB	-
Runway Lat, Long	Runway Coordinates	deg	Runway DB	-
Model input data (included in MLC ODD)				
Radar-Altitude (RA)	Altitude above ground level	ft	RadAlt	$50 \div 3000$
Distance to Threshold	Horizontal distance from runway	m	FMS + Runway DB	$0 \div 10000$
Ground Speed (GS)	Speed with respect to ground	kn	NAV-computer or ADU	$120 \div 160$
Vertical Speed (VS)	Descent Rate	ft/min	NAV-computer	$-2000 \div -200$

Parameter	Description	Unit	Source	Range
H/C Lat, Long	H/C Position Status	deg	GNSS	lat_{rwy} $\pm$ 0.05° $long_{rwy}$ $\pm$ 0.05°
Track, Heading	Horizontal Kinematics	deg	AHRS or NAV-computer	NOTE[1]
Other Characteristics				
Capturing frequency	Sensor sampling rate	Hz	-	5

Table 5.4: Operational Design Domain.

[1]The check does not directly concern the track and heading variables, but their value in relation to the magnetic orientation of the runway (QFU):

- $|track - QFU| \leq 15°$
- $|heading - QFU| \leq 20°$

Objective CO-06: Functional analysis, decomposition and allocation.

- **High-level function**: "To provide vectorial advisory guidance in order to assist pilots in aligning with the runway during descent/approach in Visual Meteorological Conditions, based on the computed prediction of latitude and longitude variations over a time horizon of 10 seconds."
- **Sub-functions and respective allocation to the subsystem(s)**:
  1. Data acquisition and pre-processing (normalisation): traditional SW;
  2. Monitoring of OODs (advisory deactivation if input data are OOD) and engagement of Runway Detector (activation check): traditional SW;
  3. Data elaboration and feature engineering (time vectors construction): traditional SW;
  4. Trajectory change prediction (latitude and longitude variations): regressor - DL (neural network);
  5. Post-processing (denormalisation, transformation in vector guidance; for future developments, also Kalman filter): traditional SW;
  6. Cockpit display advisory (advisory visualisation, loss of function indication): traditional SW.

5.3 System design

5.3.1 Overview

As already mentioned, the key component of RAS is the TCP neural network, which provides a vector estimate of the short-term geodetic displacement:

$$\Delta lat\,,\ \Delta long\ [^\circ]\ \ for\ t = t_0 + 10s$$

starting from a historical telemetry window of $12s$ at $5Hz$ capture frequency ($T = 60$ samples). The NN works as advisory (no authority on the actuators); the output is used to estimate the future position and, through post-processing, a correction vector towards the runway centreline.

The entire system is engaged only if a dedicated traditional software communicates that the runway has been detected and the input data is inside the Operational Design Domain.

Thanks to a combination of information gathered from the Flight Management System (FMS) and a database (DB) on board the aircraft, the system has a series of variables that are important for the final guidance, such as the magnetic orientation of the runway QFU, or latitude and longitude coordinates of the runway threshold.

Based on the RAS overview described, the selected architecture of the complete system is illustrated in Figure 5.1.

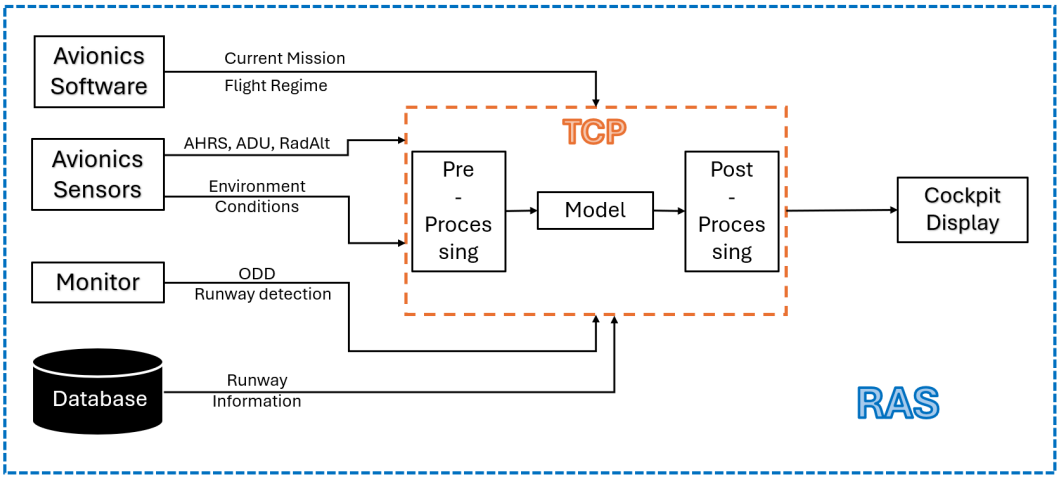

Figure 5.1: Overview of RAS architecture.

5.3.2 Neural network architecture

Model

The regressor model relative to the TCP constituent is obtained by optimising the parameters (weights) of a family of Recurrent Neural Networks (RNN) using the gradient method. In particular, Gated Recurrent Units (GRU) were chosen because they are efficient (less parameters than other solutions) and suitable for capturing short-range temporal dependencies. Furthermore, these units have update/reset gates that regulate what to store/forget along the sequence, ideal for the prediction of trajectories.

The resulting model architecture is presented in Figure 5.2 and consists of 2 GRU layers with 64 cells, followed by 1 dense layer of 64 neurons and 2 output neurons, with full connectivity.

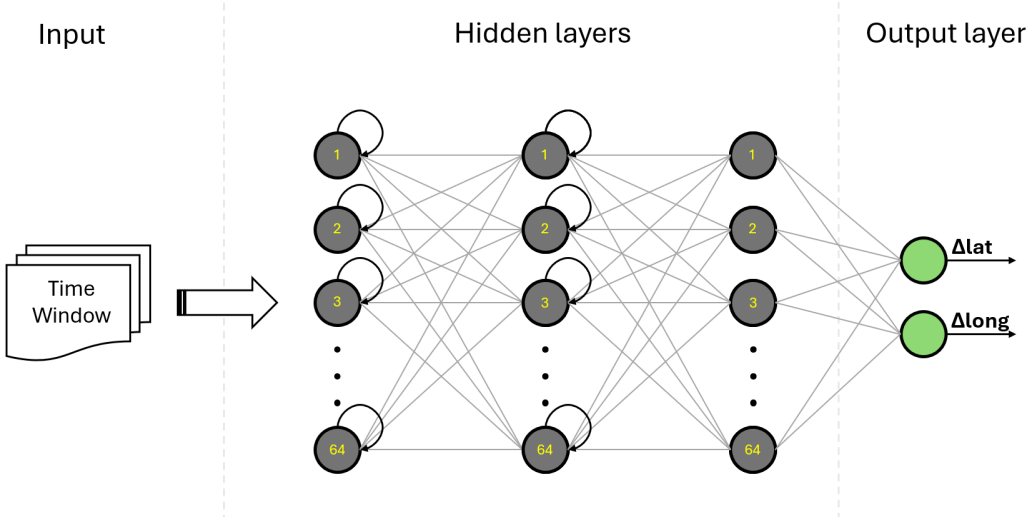

Figure 5.2: Model architecture.

Input

Model input data are listed in Table 5.4, the only precaution is that, to avoid problems with angles, tracking and heading are decomposed into their respective sine and cosine components. The distance from the runway threshold is computed from latitude and longitude values both of the helicopter and the threshold, using the Haversine formula [60]. Thus, the total number of features is 10; below is a summary:

1. H/C latitude and longitude [°];

3. Ground Speed (GS)[kn];

4. Vertical Speed (VS)[ft/min];
5. Radar-altitude (RA)[ft];
6. Distance to runway threshold [m];
8. $sin(track)$ and $cos(track)$;
10. $sin(heading)$ and $cos(heading)$;

As already mentioned, the model receives as input a time window of 60 samples ($12s$ history at $5Hz$ sampling frequency), each consisting of **10 features**: the resulting time window is a 60×10 matrix.

Output

The model output is a 2 components vector $(\Delta lat\,,\, \Delta long)[^\circ]$ at $t = t_0 + 10s$, where t_0 is the time at which the model receives the input data, ignoring latency.

5.3.3 Pre- and Post-procesing

Pre-processing

TCP Pre-processing activities concern:

- Receiving input data and monitor's output;
- Heading and track decomposition in $sin()$ and $cos()$ components.
- Computation of the distance to runway threshold using the Haversine formula;
- normalisation (standardisation);
- Construction of historical time window.

Post-processing

TCP Post-processing activities concern:

- Prediction of future position:

$$lat_t = lat_{t_0} + \Delta lat$$
$$long_t = long_{t_0} + \Delta long$$

- Graphical advisory with correction vectors towards runway centreline: projection of future position on centreline.

No further details are provided on the system design, since, as anticipated, the system has the sole purpose of providing context for the safety analysis.

5.4 System Safety Assessment

5.4.1 FHA

H/C FHA

As illustrated in Chapter 3, the process begins outlining the aircraft-level functional failures.

The RAS introduces a new H/C level function that can be listed under the already existing first level function "*provide navigation facilities* (ID 8)" and the second level function "*provide flight management computing* (ID 8.5)". The associated functional failures are:

1. **Loss of Runway Alignment System guidance** - ID 8.5.3;
2. **Erroneous RAS guidance** - ID 8.5.4.

The results are shown in Table 5.5.

Avionics FHA

The analysis is in-depth at system level; the respective failure effects are described in the avionics FHA, as depicted in Table 5.6.

In this context, functional failures are classified as those with the task of "*providing navigation assistance* (ID 00.05)" and in the new sub-category of "provide VLS + RAS Assistance (ID 00.05.4)"; these functions are linked to the H/C level ones, labelled with ID 8.5.

The avionics FHA Table includes the following functional failures (FFs):

- Total loss of function, both in case the loss is indicated to the crew or not (ID FF 00.05.4.2 & 00.05.4.3);
- Misleading function with input data accepted in ODD (ID FF 00.05.4.5);
- Erroneous behaviour of the monitoring function, either blocking nominal inputs, equivalent to loss of function, or allowing OOD inputs, equivalent to misleading (ID FF 00.05.4.6 & 00.05.4.7).

As mentioned in Section 4.6, this table encompasses the new column to distinguish whether the function in subject is provided by a MLC or not. When the answer is "YES" the AI Safety process is performed.

Symbolic Fault Tree Analysis

Once the FHA has been completed from H/C to component level, it is performed a symbolic fault tree analysis (FTA) to obtain a complete insight on the high-level failure and its related sources.

Figure 5.3 illustrates the symbolic FTA for one of the functional failures identified in Table 5.6: "Total loss of RAS function, not indicated to the crew" ID 00.05.4.2.

H/C level Functional Hazard Assessment									
ID	First H/C level function	ID	Second and Third H/C level function	ID	Function Failures at H/C level	MCC	Effect of Functional Failure on H/C or Crew	Class.	Remarks
8	Provide Navigation facilities	8.5	Provide Flight management computing	8.5.3	Loss of Runway Alignment System guidance	Single Pilot VFR Approach	Slight increase in workload	MIN	The system is intended to work in VFR & VMC
				8.5.4	Erroneous RAS guidance	Single Pilot VFR Approach	Significant reduction in safety margins	MAJ	The system is intended to work in VFR & VMC

Table 5.5: H/C FHA.

Runway Alignment System (RAS) Functional Hazard Assessment Table										
ID Function	**Function**	**ID Functional Failure**	**Functional Failure**	**MCC**	**Effect of FF on Other Systems**	**Aircraft Level Effect in MCC**	**FF Classificiation in MCC**	**Remarks**	**H/C level**	**Function provided by AI/ML Constituent?**
00.05	Provide Navigation Assistance	**00.05.4**	**Provide VLS + RAS Assistance**							
		00.05.4.2 Advisory	**Total loss of RAS function, not indicated to the crew**	Single Pilot VFR Approach	None	System does not alert to the loss of function, but crew will easily detect the loss. Slight reduction of safety margins.	**MIN**	The crew follows the standard path indicated by the FMS	8.5.3 (D)	YES
		00.05.4.3 Advisory	**Total loss of RAS function, indicated to the crew**	Single Pilot VFR Approach	None	System provides indication of the loss of function. Slight increase in workload.	**MIN**	The crew follows the standard path indicated by the FMS	8.5.3 (D)	YES
		00.05.4.5 Advisory	**Misleading RAS function (accepted in ODD)**	Single Pilot VFR Approach	None	System provides erroneous indication that seems correct to the crew. Significant decrease of safety margins.	**MAJ**	Flight crew will detect the erratic behaviour by visual confirmation or crosscheck with other aircraft systems, correct the path and follow the standard path indicated by the FMS	8.5.4 (D)	YES
		00.05.4.6 Monitor OOD	**Monitor software does not block OOD inputs**	Single Pilot VFR Approach	None	Significant decrease of safety margins.	**MAJ**	Flight crew will detect the erratic behaviour by visual confirmation or crosscheck with other aircraft systems, correct the path and follow the standard path indicated by the FMS	-	NO
		00.05.4.7 Monitor OOD	**Monitor software blocks nominal inputs**	Single Pilot VFR Approach	None	Slight increase in pilot's workload.	**MIN**	The monitor software deactivates the RAS function, even though input data is nominal. Flight crew will rely on the FMS	-	NO

Table 5.6: RAS Avionics FHA.

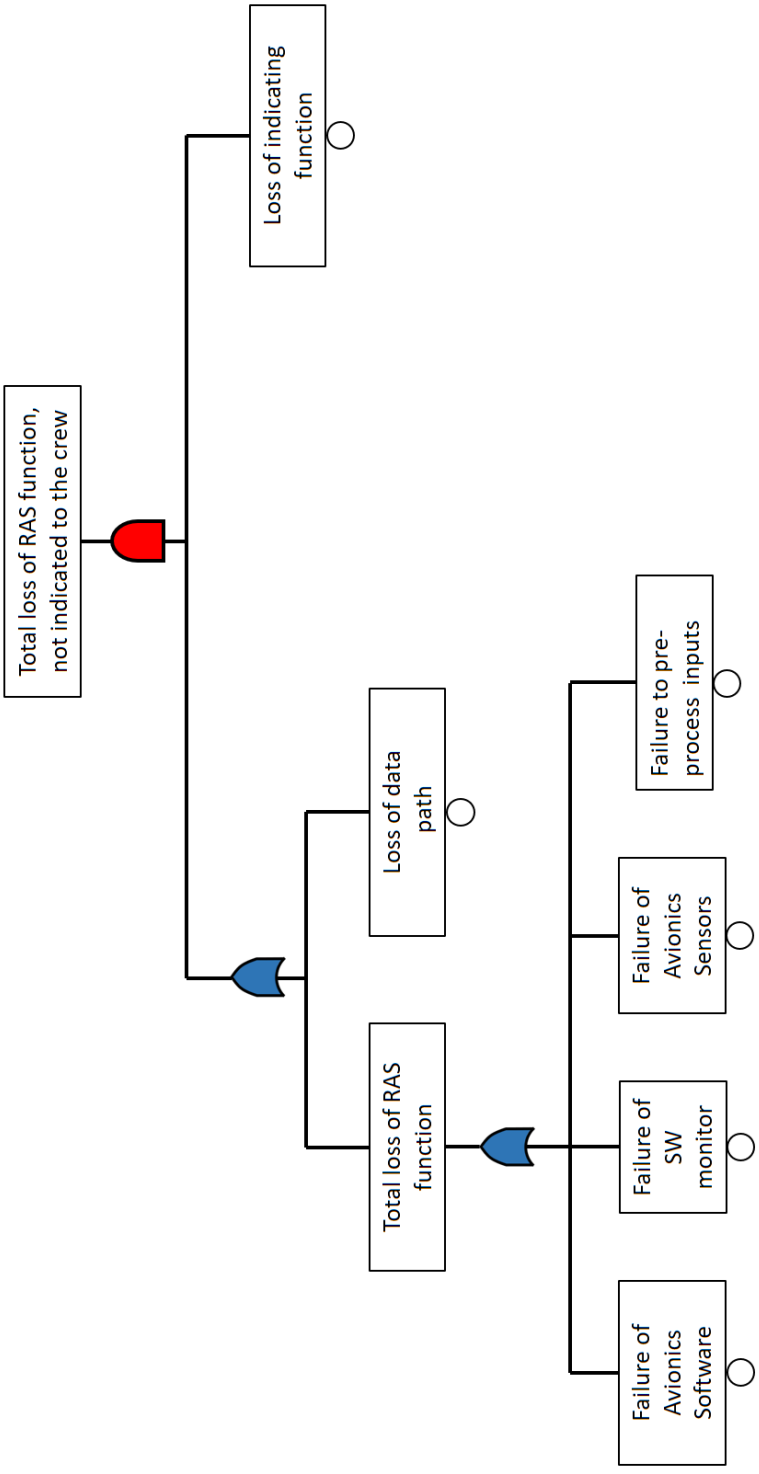

Figure 5.3: FF 00.05.4.2 Symbolic FTA.

5.4.2 PSSA

DAL and Hardware Safety Objectives allocation

Based on the outcomes of the FHA and the classification described in Table 3.3, the FDAL allocation is provided in Table 5.7.

Use case	Functional Failure	Allocated **FDAL**
Advisory	FF 00.05.4.2	**D**
	FF 00.05.4.3	**D**
	FF 00.05.4.5	**C**
	FF 00.05.4.6	**C**
	FF 00.05.4.7	**D**

Table 5.7: FDAL allocation: level is assigned considering the highest severity that involves the function.

Since no reduction is allowed, the considerations at function-level are directly reflected at item-level in IDAL allocation.

Table 5.8 reports the assignment of probability requirements for random hardware failures, as traditionally required by ED-135/ARP4761 [16]. These requirements are intended to be satisfied by the MLC hardware as a whole and to be shown compliance with them choosing an adequate propagation error method.

Hardware failure is nevertheless developed considering each functional failure and not as a single generic event for greater formal completeness. This avoids, in certain situations, excluding from the analysis other hypothetical events that could lead to other functional failures.

System FF ID	FF Severity	Probability of occurrence
00.05.4.2	MIN	$\mathbf{p < 10^{-3}}$
00.05.4.3	MIN	$\mathbf{p < 10^{-3}}$
00.05.4.5	MAJ	$\mathbf{p < 10^{-5}}$
00.05.4.6	MAJ	$\mathbf{p < 10^{-5}}$
00.05.4.7	MIN	$\mathbf{p < 10^{-3}}$

Table 5.8: Hardware Safety Objective: target is assigned considering the functional failure with highest severity.

Performance Requirements

Considering the highest failure severity with regard to the misleading of the function, which is *Major* related to FF 00.05.4.5 (in this case the only one present), the associated quantitative requirements, according to Section 4.8, are shown in Table 5.9.

Misleading Failure Mode		Requirements		
FF ID	**FF Severity**	**Global**	**Local**	**Threshold**
00.05.4.5	MAJ	$ExR^{global} \leq 5\% - \varepsilon$	$ExR^{bin} \leq 7.5\%$	$\tau \leq L$

Table 5.9: Quantitative Requirements.

The global quantitative requirement is intended to be satisfied also considering the generalisation bound ε; it is not relevant which generalisation model is chosen, but, for every case, the significance level δ must be lower than 0.0001.

For binomial variables like the one concerning the model output to be higher than the safety threshold ($X_j = 1$: $|e_j| > \tau$, $X_j = 0$: $|e_j| < \tau$) an adequate generalisation bound could be the Hoeffding inequality [61]:

$$P\left(|\mathbb{E}_{out} - \mathbb{E}_{in}| \geq \varepsilon\right) \leq 2\exp(-2n\varepsilon^2) \leq \delta$$

where n represents the number of independent observations.

The operational tolerance L has been established following the proposed guidelines in

Section 4.8.3; the definition is based on the GPS Required Navigation Performance, which is, for the approach phase, $RNP = 0.3\,\text{NM}$. [56][62]. Since the system in question could present errors both latitudinally and longitudinally, the operational tolerance is calculated considering the worst-case in which the error is made along the diagonal of a square:

$$L' = \frac{0.3}{\sqrt{2}} \approx 0.21\,\text{NM} \Rightarrow \boxed{0.2\,\text{NM}}$$

Starting from L', the tolerance is converted into the same unit of measurement of the output (degrees) and it is calculated considering the different nature of parallels and meridians; then, the more conservative one is chosen for both the outputs.

Latitude

$$1^\circ \; of \; latitude \approx 60\,\text{NM}$$

$$1NM_{lat} \approx \frac{1^\circ}{60} \approx 0.0167^\circ \;\rightarrow\; L_{lat} \equiv 0.2\,NM_{lat} \approx 0.0033^\circ$$

Longitude

The calculation depends on latitude because the meridians converge: a representative value of $45^\circ N$ has been chosen.

$$1^\circ \; of \; longitude \approx 60\,\text{NM} \cdot \cos(lat)$$

$$1NM_{long} \approx \frac{1^\circ}{60} / \cos(lat) \approx 0.0167^\circ / \cos(45^\circ) \;\rightarrow\; L_{long} \equiv 0.2\,NM_{long} \approx 0.0047^\circ$$

After this calculations, the most stringent operative tolerance is taken:

$$\boxed{\varepsilon \leq L_{lat} = 0.0033^\circ}$$

Runway Detector Switch and Out of Domain Detection

The same software component evaluates both whether a runway has been detected and if input data can be delivered to the MLC.

The former function only acts as a switch: turning on the MLC when a runway is successfully detected and turning it off when the approach is aborted. The latter, instead, performs the OOD monitoring through a traditional software that uses the operating space, defined in the ConOps, to provide the MLC only with nominal data, ensuring that the model can work into the design domain. Since the operational volume only involves explicit parameters with a predefined range, the software just checks with logical functions if input data is within the acceptable domain.

Uncertainties Management

Six sources of aleatory uncertainties are identified with respect to the system:

1. Sensor noise
   When measuring a variable, the sensors induce an additive noise in the output of the model.

   △ Mitigations: typically, sensor variances are known and can be exploited within an appropriate mathematical model. In addition, a series of dedicated filters, such as median or EMA/EWMA for time series, are designed to reduce this kind of noise.

2. Wind presence
   The presence of wind may cause uncertainties in the model output, as it was not explicitly considered during the model training (wind is just a check for the OOD detector and is implicitly present in heading and track variations).

   △ Mitigation: add explicit wind components in the learning algorithm and compute the wind triangle ($\overrightarrow{wind}$, $\overrightarrow{track}$, $\overrightarrow{heading}$).

3. Label noise
   Even the ground truth can be subject to inaccuracies when labelling dataset samples, leading to uncertainty in the model output.

   △ Mitigation: let labelling be carried out by a group of experts, as envisaged in supervised learning.

4. Database errors
   The runway database errors are a source of uncertainty for the model output.

   △ Mitigation: the database has to be certified and subject to periodical update.

5. Time correlations
   It is not necessarily true that consecutive errors are independent; different error correlations have a significant influence on the model output.

   △ Mitigation: perform error correlation analysis.

The results of this analysis are illustrated in Table 5.10, following Section 4.5 guidelines. Since there is only one AI/ML constituent that comprises the system, the "AI-constituent" table entry is replaced by the uncertainty name.

Uncertainty	Source	Description	Type	Mitigation
Sensor noise	Sensors	Noise induced by sensors during acquisition	A	Mathematical models and dedicated filters
Wind presence	Wind	Wind components are not present explicitly during training	A	Add explicit wind components
Data quality	Training data	Dataset not complete or not representative enough	E	Not under Safety responsibility
Label noise		Inaccurate labels	A	Group of experts
Model incapacity	Development inaccuracies	Inadequate architecture/-complexity	E	Not under Safety responsibility
Model instability		Inadequate stability	E	
Time correlations	Failure properties	Consecutive errors not independent	A	Correlation analysis

Table 5.10: Uncertainties identification and mitigation.

5.4.3 Support Safety Assessment

Verification

At the end of the Runway Alignment System (RAS) safety analysis, the verification of the identified objectives is required.

The objectives to be verified are:

- DAL allocation: the safety objectives related to DAL C have been achieved; No reduction is allowed.
- Performance metrics requirements for the training of the algorithm, both global and local, have been achieved with the required generalisation bound

and associated significance level.

- Random hardware failure quantitative objectives are achieved and analysed with an adequate error propagation model (e.g., FTA).
- Specific assumptions and criteria for the detection of out of distribution (OOD) inputs are verified.

6 | Future Developments and Conclusion

6.1 In-depth Analysis of the Framework

This section presents a possible continuation for the proposed quantitative requirements on the model performance. However, the following theory is based on strong assumptions, such as considering independent events or that engineering judgment is capable of determining specific MLC failure modes; for this reason, the following analysis has not been included in Chapter 4 and has to be considered as a possible reference for future developments.

6.1.1 Towards a Probability of Failure per Flight Hours

The global exceedance rate evaluated on the test dataset, as mentioned in Section 4.3, is assumed to be the empirical evaluation of the probability of (misleading) failure p **per input** related to the MLC.

$$ExR^{global} = \frac{\#\{|E_i| > \tau\}}{|D_{test}|} \;\rightarrow\; p = ExR^{global}$$

Starting from this empirical probability per input it is possible to make a conversion into a probability per flight hours F_{fh}: it is only necessary the number of independent events N during one flight hour and the nature of the failure, i.e., what circumstance causes the failure.

$$N = f \cdot 3600\,, \quad \text{where } f[\text{Hz}] \text{ is the model output frequence}$$

Based on the nature of the failure, three cases are outlined:

1. Single exceedance causes the failure

   $F_{fh} = 1 - (1-p)^N$, probability to have at least one exceedance.

2. k consecutive exceedances cause the failure

   $k = t \cdot f$, consecutive exceedances.

   $F_{fh} = \frac{N}{k} p^k$, probability of k consecutive exceedances occurring.

3. At least m exceedances in a time window Δ cause the failure

   $n_\Delta = f \cdot \Delta$, number of samples per window. The number of exceedances X_Δ in the time window follows a binomial distribution $X_\Delta \simeq Binomial(n_\Delta, p)$.

   $P_{\Delta,m} = P(X_\Delta \geq m) = \sum_{j=m}^{n_\Delta} \binom{n_\Delta}{j} p^j (1-p)^{n_\Delta - j}$, probability that in n_Δ samples at least m of them are not acceptable.

   $F_{fh} = 1 - (1 - P_{\Delta,m})^N \approx P_{\Delta,m} \cdot N$, extension of the probability in flight hours.

The underlying idea is that, based on engineering judgment, it is possible to attribute the correct nature of the failure to the MLC and to obtain a probability of failure in flight hours. In this way it is possible to compare this probability with the classical ones involved in random *hardware* failure requirements (see Section 3.2.2). However, **it is not necessarily true that those hardware requirements are still adequate** to a completely different reality like AI and new quantitative objectives may be required.

Finally, it only remains to ensure that:

$$F_{fh} + P_{GAP} \leq P_{req}$$

where P_{GAP} is the pejorative addition that takes into account the performance reduction of the model during inference (generalisation) and P_{req} is a generic probability target.

6.1.2 New ways to satisfy the requirements

Considering this new framework in which it is used the (misleading) failure probability in flight hours, new different mathematical models can be implemented to also include safety measures that do not directly influence the exceedance rate metric, such as using a model *ensemble*.

Considering an ensemble of M models, given one single input, X_j is the exceedance indicator (indicator function) for each model:

$$X_j = 1_{\{|E_j| > \tau\}} \in \{0,1\}$$

The overall counting of models with exceedance for each input is:

$$S = \sum_{j=1}^{M} X_j$$

for each input, the model ensemble fails if at least k models exceed the safety threshold τ:

$$F_{ens} = \{S \geq k\}, \quad k = \begin{cases} M/2 \;\; if \; M \; is \; even \\ \frac{M+1}{2} \;\; if \; M \; is \; odd \end{cases}$$

Assuming the models to be independent and identically distributed (i.i.d.), the probability of failure $\overline{p}$ associated to each model, obtained from the Exceedance Rate, is the same. Once $\overline{p}$ is known, the probability of failure of the ensemble per input is:

$$p_{ens} = p = P(S \geq k \,|\, X) = \sum_{r=k}^{M} \binom{M}{r} \overline{p}^{r} (1-\overline{p})^{M-r} \approx \binom{M}{k} \overline{p}^{k}$$

This probability per input can then be converted into flight hours as previously shown (Section 6.1.1), provided that there is no time aggregation, e.g., average over multiple instants, voting over a time window. If the models are highly correlated, this estimate may be optimistic.

Similar mathematical methods could be derived to take into account the positive effect of post-processing or of an output acceptability monitor on the probability of failure.

6.2 A Dedicated Method for Deep Regression Models: PAGER

Thiagarajan et al. [63] published in 2024 a promising article that claimed to be able to detect potential failure modes of **deep regression models**.

Their framework, named PAGER (Principled Analysis of Generalization Errors in Regressors) is not limited to relying solely on epistemic uncertainty, which is considered insufficient due to feature heterogeneity in the training data, but also involves novel non-conformity scores that measure adherence to the training data manifold. The aim of their analysis is to *identify groups of varying expected risk*; in particular, the samples from test dataset are organised into *ID* (i.e., in distribution, were the model is expected to generalise well), *Low Risk*, *Moderate Risk* and *High Risk*, thus enabling a comprehensive analysis of model errors. Their approach is illustrated in Figure 6.1.

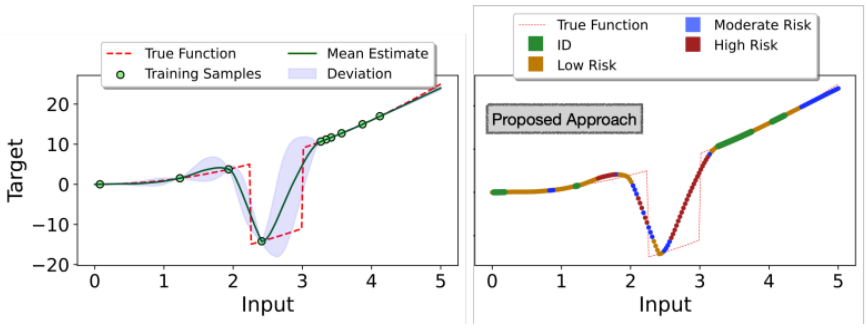

Figure 6.1: PAGER approach and groups of expected risk.
(Source: PAGER [63])

The division is based on the combination of conditional quantile ranges of two scores, the former deriving from epistemic uncertainty via forward anchoring, and the latter from a non-conformity score via reverse-anchoring. The resulting groups are differentiated as shown in Figure 6.2.

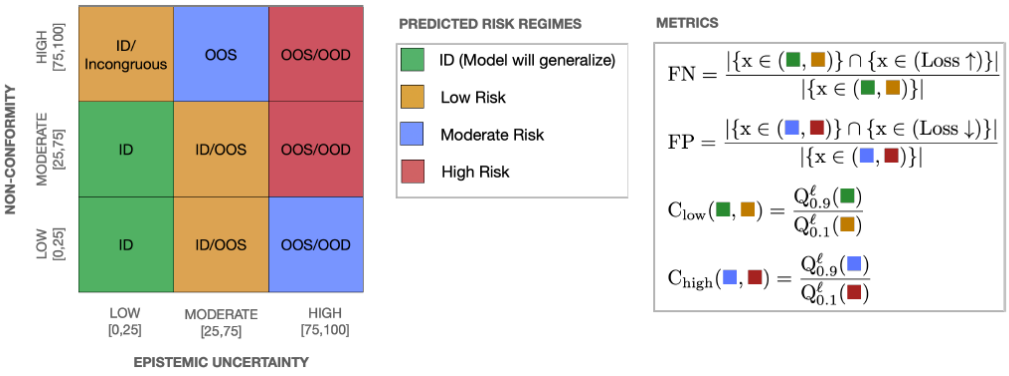

Figure 6.2: Overview of the framework. With such a categorisation, PAGER associates samples into 4 levels of expected risk. A suite of metrics that enables a holistic assessment of failure detectors are also provided.
(Source: PAGER [63])

In the Safety Assessment of deep regression models context, PAGER could be used to verify that a model provides the intended behaviour, especially in situations in which it is very difficult to derive acceptable generalisation bounds: for example, it could be required to ensure that the majority of outputs, evaluated on the test dataset, are *ID* or *Low Risk*, to adopt adequate mitigations for *Moderate Risk* groups (Corner Cases) and to exclude *High Risk* samples (OOD) from the model ODD; otherwise, the proposed metrics could be used directly.

Alternatively, PAGER could be adapted to evaluate whether the identified bins

are critical: for example, it could be verified that the majority of the samples in a given bin are not part of *High Risk* groups.

6.3 Open Challenges

Even though this proposed framework is supposed to be as much general as possible, there are some unaddressed scenarios.

6.3.1 Complex Tasks

There are some advanced computer vision tasks, formulated as a regression problem, in which the Exceedance Rate metric may not be a valid safety assessment tool. Some examples of these tasks are:

- Object localisation (bounding-box regression);
- Pixel-level prediction tasks like super-resolution, denoising, and colourisation.

For bounding-box regression, the Jaccard distance, i.e., the complementary area of the *Intersection over Union* (IoU), is a core metric [23][37]. Figure 6.3 illustrates the computation of the metric with examples.

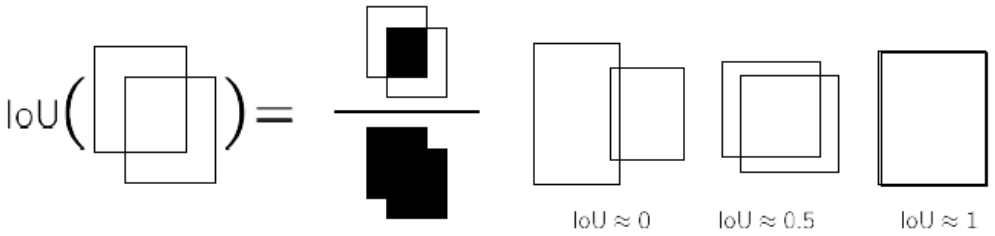

Figure 6.3: Computation of the intersection over union (IoU) of two masks and examples. The Jaccard distance is respectively 1, 0.5 and 0.
(Source: CoDANN I [23])

6.3.2 DAL Safety objectives

Certification bodies strongly believe in DAL allocation even for AI-items, viewing it as a key procedure to ensure reliability over the development process, as already conducted for traditional items. However, it is worth considering that DAL safety objectives related to software development (set by DO-178C [1]) also include testing activities or specific requirements on traceability and explainability, which are not

necessarily extendable to machine learning algorithms. At the same time, even the hardware platforms currently on the market (GPUs) would not be able to meet the safety requirements related to the DAL associated; great expectations are indeed placed in the Field Programmable Gate Arrays (FPGA), that consist in prefabricated chips whose logic can be configured to the user specification at the price of more computational cost.

There will certainly be a need for supporting theory to address the explainability issue and new DAL safety objectives suited to AI nature.

6.3.3 Beyond supervised learning

In most cases, regression is inherently a supervised learning task because it requires known target values to train a model that maps inputs to continuous outputs. However, unsupervised techniques can still play an important supporting role. Methods such as auto-encoder pre-training or clustering-driven data structuring can be applied before the supervised phase to improve representation quality or reduce dimensionality. Thus, while regression itself remains supervised, unsupervised learning can significantly enhance the overall pipeline.

Recently, new hybrid approaches, named *self-supervised* and *semi-supervised learning* have emerged. In the former technique, models learn useful representations from unlabelled data by generating pseudo-labels, while the latter uses a combination of a small amount of labelled data and a large amount of unlabelled data to uncover structural information [2].

At the moment, it is not clear how the regulatory bodies (i.e., EASA, FAA, etc.) intend to deal with these new approaches; the EASA Concept Paper [13] currently only considers traditional basic methodologies, i.e., supervised and unsupervised learning. More time will be needed for new issues of the document to take into account such new techniques (in addition to reinforcement learning), as already envisaged by the AI Roadmap 2.0 [21].

6.4 Final Remarks

The main purpose of this thesis was to propose a new safety process that could be applied to aeronautical ML-based systems in airborne domain that perform regression tasks and to provide a reference basis for future more complex AI-based applications.

The whole framework was founded on addressing the first proposals of Means of Compliance (aMOC) reported in the EASA Concept Paper: Guidance for Level 1 & 2 machine learning applications [13]. Table 6.1 summarises the strategies adopted throughout the thesis to meet these aMOC.

To better understand the scenario, and to introduce Safety and machine learning realities with associated terminologies, a series of related chapters accompanied the discussion.

In particular, attention was initially focused on how the introduction of AI algorithms would revolutionise the development process and, with it, the safety process: the classical "V-shaped" process is replaced with a "W-shaped" process due to the need for an intermediate learning verification that draws the AI learning phase to a close. This new approach applies only to system that embed AI, so other systems that are instead composed of traditional HW and SW follow the approach required by ED-135/ARP4761A [16] and ED-79B/ARP4754B [15]. The whole approach, differentiated depending on the system type, is illustrated in Figure 6.4.

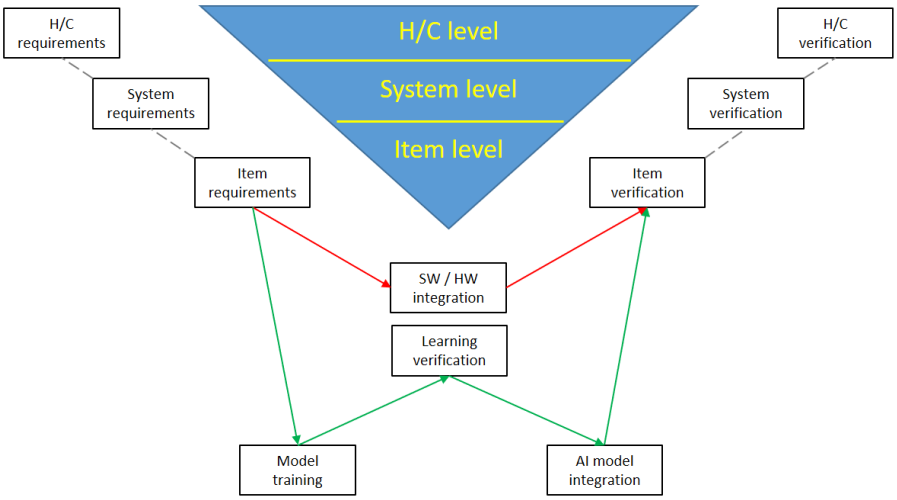

Figure 6.4: Overview of the new development process: the split begins at item-level, where AI-based systems follow the "W-shaped" process (green).

The real intent of this framework was to provide a basis that could be effectively applied, linking the indisputable need for security required by regulatory bodies with the desire for concrete measures and guidelines on the part of industry. For this purpose, a hypothetical use case was presented in Chapter 5; in fact, it was crucial to show how a real AI algorithm could respond to what authorities will ask for.

To conclude, it is well known that the path towards the certification of AI-based systems in aviation has only just begun. The keen interest shown by researchers around the world in this discipline and the incredible technological improvements associated with it suggest that this turning point is drawing ever closer. Starting with simple and safe applications, AI systems will be granted increasing authority

under the control of regulatory bodies, following the milestones outlined in the AI Roadmap 2.0 [21].

Anticipated-MOC	Proposed approach
① DAL allocation	The allocation is performed as for the traditional process following the ED-135/ARP4761A guidelines and DO-178C standard, with the exception that no reduction is permitted.
② Metrics	New preliminary safety objectives, based on the Exceedance Rate and the generalisation gap, are introduced to ensure the model learning process.
③ Exposure to OOD data	A traditional SW component performs the OOD data detection checking if the explicit parameters of input data belong to the ODD.
④ Identification and classification of uncertainties	Addressed through the introduction of the uncertainties assessment table.
⑤ Assessment and mitigation of uncertainties	Addressed through the introduction of the uncertainties assessment table.
⑥ Establishment of MLC failure modes	Addressed through the Functional Hazard Assessment (FHA).
⑦ Link between metrics and safety assessment	The metric introduced is already an empirical evaluation of the probability of misleading failure of the MLC.
⑧ Link between generalisation bounds and safety assessment	Any method to obtain a generalisation bound can be used, provided that it is achieved the required significance level δ.
⑨ Verification	The verification is performed as already required by existing guidelines, except for the verification of the performance requirements that must be verified at the end of the learning phase.

Table 6.1: Anticipated-MOC and proposed approaches for regression tasks.

Bibliography

[1] *ED-12C: Software Considerations in Airborne Systems and Equipment Certification*. Equivalent to RTCA DO-178C. 9-23 Rue Paul Lafargue, 93200 Saint-Denis, France: EUROCAE, 2011 (cit. on pp. 1, 7, 16, 22, 30, 43, 83).

[2] Romeo Valentin. «Towards a Framework for Deep Learning Certification in Safety-critical Applications using Inherently Safe Design». MA thesis. Zurich: Department of computer science of ETH Zurich, 2024 (cit. on pp. 1, 84).

[3] EASA and Collins Aerospace. *Formal Methods use for Learning Assurance (ForMuLA)*. Tech. rep. Apr. 2023 (cit. on pp. 2, 11, 49).

[4] Erfan Asaadi, E. Denney, and G. Pai. «Quantifying Assurance in Learning-enabled Systems». In: *Computer Safety, Reliability, and Security – SAFECOMP 2020*. Vol. 12234. Lecture Notes in Computer Science. Springer, 2020, pp. 270–286. DOI: 10.1007/978-3-030-54549-9_18 (cit. on pp. 2, 48).

[5] MLEAP Consortium. *EASA Research – Machine Learning Application Approval (MLEAP) final report*. Horizon Europe research and innovation programme report. European Union Aviation Safety Agency, May 2024 (cit. on pp. 2, 11, 35, 40–42, 53).

[6] Gabriel Laberge et al. Florian Tambon. «How to certify machine learning based safety-critical systems? A systematic literature review». In: *Autom Softw Eng* 29.38 (2022), pp. 17–21 (cit. on pp. 2, 4).

[7] European Union Aviation Safety Agency. *Certification Specifications and Acceptable Means of Compliance for Large Aeroplanes (CS-25)*. EASA, Dec. 2023. URL: https://www.easa.europa.eu/en/document-library/certification-specifications/group/cs-25-large-aeroplanes (cit. on pp. 3, 22).

[8] European Union Aviation Safety Agency. *Certification Specifications and Acceptable Means of Compliance for Small Rotorcraft (CS-27)*. EASA, Feb. 2023. URL: https://www.easa.europa.eu/en/document-library/certification-specifications/group/cs-27-small-rotorcraft (cit. on pp. 3, 22).

[9] European Union Aviation Safety Agency. *Certification Specifications and Acceptable Means of Compliance for Large Rotorcraft (CS-29)*. EASA, Dec. 2023. URL: https://www.easa.europa.eu/en/document-library/certification-specifications/group/cs-29-large-rotorcraft (cit. on pp. 3, 22).

[10] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. «Multilayer Feedforward Networks Are Universal Approximators». In: *Neural Networks* 2.5 (1989), pp. 359–366. DOI: 10.1016/0893-6080(89)90020-8 (cit. on p. 4).

[11] Ram Shankar Siva Kumar, David R. O'Brien, Kendra Albert, Salomé Viljóen, and Jeffrey Snover. «Failure Modes in Machine Learning Systems». In: *arXiv preprint arXiv:1911.11034* (2019) (cit. on p. 4).

[12] Claudia Ranieri. «Safety Process for Aeronautical AI-based Systems Certification». Master thesis in collaboration with Leonardo Helicopters. MA thesis. Milan: Politecnico di Milano, 2023 (cit. on pp. 5, 47, 50, 60, 62).

[13] European Union Aviation Safety Agency. *EASA Concept Paper: Guidance for Level 1 & 2 Machine Learning Applications*. Tech. rep. Concept Paper, Issue 2. EASA, Mar. 2024 (cit. on pp. 6, 9, 13, 17, 18, 20, 44, 46, 49, 53, 60, 61, 84).

[14] Wikipedia contributors. *ARP4754*. https://en.wikipedia.org/wiki/ARP4754. 2024 (cit. on p. 7).

[15] EUROCAE Working Group 63. *ED-79B: Guidelines for Development of Civil Aircraft and Systems*. EUROCAE Standard, jointly developed with SAE ARP4754B. EUROCAE, Dec. 2023 (cit. on pp. 7, 21, 22, 24, 26, 31, 85).

[16] EUROCAE Working Group 63 and SAE S-18 Committee. *ED-135: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment*. EUROCAE Standard, jointly developed with SAE ARP4761A. EUROCAE, Dec. 2023 (cit. on pp. 7, 19, 21–24, 27, 29, 49, 57–59, 73, 85).

[17] EUROCAE and RTCA. *ED-128/DO-331: Model-Based Development and Verification, Supplement to ED-12C/DO-178C and ED-109A/DO-278A*. Standard. 2011 (cit. on p. 7).

[18] EUROCAE and RTCA. *ED-217/DO-332: Object-Oriented Technology and Related Techniques, Supplement to ED-12C/DO-178C and ED-109A/DO-278A*. Standard. 2011 (cit. on p. 7).

[19] EUROCAE and RTCA. *ED-216/DO-333: Formal Methods, Supplement to DO-178C and DO-278A*. Standard. 2011 (cit. on p. 7).

[20] European Union Aviation Safety Agency. *Artificial Intelligence Roadmap: A Human-Centric Approach to AI in Aviation*. Tech. rep. EASA, 2020 (cit. on p. 8).

[21] European Union Aviation Safety Agency. *Artificial Intelligence Roadmap 2.0: A Human-Centric Approach to AI in Aviation*. Tech. rep. EASA, 2023. URL: https://www.easa.europa.eu/en/newsroom-and-events/news/easa-artificial-intelligence-roadmap-20-published (cit. on pp. 8, 37, 84, 86).

[22] High-Level Expert Group on Artificial Intelligence. *Ethics Guidelines for Trustworthy AI*. Tech. Report. European Commission, Apr. 2019. URL: https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai (cit. on p. 9).

[23] Daedalean AG. *Concepts of Design Assurance for Neural Networks (CoDANN)*. Tech. Report. EASA, Mar. 2020. URL: https://www.easa.europa.eu/en/document-library/general-publications/concepts-design-assurance-neural-networks-codann (cit. on pp. 9, 18, 32, 36, 41, 83).

[24] European Union Aviation Safety Agency. *First Usable Guidance for Level 1 Machine Learning Applications*. Concept Paper. EASA, Apr. 2021 (cit. on pp. 10, 33).

[25] *Regulation (EU) 2018/1139 of the European Parliament and of the Council of 4 July 2018*. Aug. 2018 (cit. on p. 10).

[26] Daedalean. *Concepts of Design Assurance for Neural Networks (CoDANN) II with appendix B*. Tech. Report. EASA, Jan. 2024 (cit. on pp. 10, 23, 32, 35).

[27] Daedalean. *Neural Network Based Runway Landing Guidance for General Aviation Autoland*. Tech. rep. Federal Aviation Administration, Nov. 2021 (cit. on pp. 10, 33, 35, 48, 49, 55, 60).

[28] Federal Aviation Administration. *14 C.F.R. Part 91 – General Operating and Flight Rules*. https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-91. 2023 (cit. on p. 10).

[29] RTCA, Inc. and EUROCAE. *DO-254 / ED-80: Design Assurance Guidance for Airborne Electronic Hardware*. Standard. 2000 (cit. on pp. 22, 30).

[30] European Union Aviation Safety Agency. *Certification Specifications and Acceptable Means of Compliance for Large Rotorcraft, Subpart F – Equipment; Equipment, Systems and Installations (CS-29.1309)*. 2022. URL: https://www.easa.europa.eu/en/document-library/easy-access-rules/online-publications/easy-access-rules-large-rotorcraft-cs-29 (cit. on p. 30).

[31] Michael Greenacre, Patrick J. F. Groenen, Trevor Hastie, Alfonso Iodice D'Enza, Angelos Markos, and Elena Tuzhilina. «Principal component analysis». In: *Nature Reviews Methods Primers* 2.100 (2022). DOI: 10.1038/s43586-022-00184-w (cit. on p. 33).

[32] Konstantin Dmitriev, Fateh Kaakai, Mohamad Ibrahim, Umut Durak, Bill Potter, and Florian Holzapfel. «Tool Qualification Aspects in ML-Based Airborne Systems Development». In: *Software Engineering 2023 Workshops*. Ed. by Iris Groher and Thomas Vogel. Gesellschaft für Informatik, 2023. DOI: 10.18420/se2023-ws-19 (cit. on p. 34).

[33] EU Commission. *EU Commission — Proposal for a Regulation of the European Parliament and of the Council laying down harmonised rules on artificial intelligence (Artificial Intelligence Act) and amending certain Union legislative acts, COM/2021/206 final*. 2021 (cit. on p. 36).

[34] Tomasz Szandała. «Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks». In: *Bio-inspired Neurocomputing*. Singapore: Springer Singapore, 2021, pp. 203–224. DOI: 10.1007/978-981-15-5495-7_11 (cit. on p. 37).

[35] Peter Bruce, Andrew Bruce, and Peter Gedeck. *Practical Statistics for Data Scientists: 50 + Essential Concepts Using R and Python*. 2nd ed. Boston, MA, USA: O'Reilly Media, 2020 (cit. on p. 38).

[36] H. Lee, J. Li, A. Rai, and A. Chattopadhyay. «Real-time anomaly detection framework using a support vector regression for the safety monitoring of commercial aircraft». In: *Advanced Engineering Informatics* 44 (2020). DOI: 10.1016/j.aei.2020.101071 (cit. on pp. 38, 44).

[37] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. «Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 658–666. URL: https://doi.org/10.48550/arXiv.1902.09630 (cit. on pp. 38, 83).

[38] Vagelis Plevris, German Solorzano, Nikolaos P. Bakas, and Mohamed El Amine Ben Seghier. «Investigation of Performance Metrics in Regression Analysis and Machine Learning-Based Prediction Models». In: *Proceedings of the 8th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS Congress 2022)* (Oslo, Norway). ECCOMAS, June 2022. DOI: 10.23967/eccomas.2022.155 (cit. on p. 38).

[39] Alexei Botchkarev. *Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology*. Tech. rep. 2018. URL: https://arxiv.org/abs/1809.03006 (cit. on pp. 38, 40).

[40] J. S. Armstrong and F. Collopy. «Error measures for generalizing about forecasting methods: Empirical comparisons». In: *International Journal of Forecasting* 8.1 (1992), pp. 69–80. DOI: 10.1016/0169-2070(92)90008-W (cit. on p. 40).

[41] C. J. Willmott and K. Matsuura. «Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance». In: *Climate Research* 30.1 (2005), pp. 79–82. DOI: 10.3354/cr030079 (cit. on p. 40).

[42] C. J. Willmott, K. Matsuura, and S. M. Robeson. «Ambiguities inherent in sums-of-squares-based error statistics». In: *Atmospheric Environment* 43.3 (2009), pp. 749–752. URL: https://doi.org/10.1016/j.atmosenv.2008.10.005 (cit. on p. 40).

[43] E. A. Silver, D. F. Pyke, and R. Peterson. *Inventory Management and Production Planning*. 3rd ed. New York: John Wiley & Sons, 1998 (cit. on p. 40).

[44] Constantinos Daskalakis and Maha et al. Shady. *Lecture 3: Generalization Theory and Rademacher Complexity*. Lecture Notes, 6.883 Science of Deep Learning. 2018 (cit. on p. 42).

[45] Jun Yang, Shengyang Sun, and Daniel M. Roy. «Fast-rate PAC-Bayes Generalization Bounds via Shifted Rademacher Processes». In: (2019). URL: https://doi.org/10.48550/arXiv.1908.07585 (cit. on p. 42).

[46] Sijia Zhou, Yunwen Lei, and Ata Kabán. «Toward Better PAC-Bayes Bounds for Uniformly Stable Algorithms». In: *Proceedings of the 37th Conference on Neural Information Processing Systems*. NeurIPS. 2023 (cit. on p. 42).

[47] Andres Potapczynski, Sanae Lotfi, Anthony Chen, and Chris Ick. «Understanding the Generalization of Deep Neural Networks through PAC-Bayes Bounds: A Survey». In: (2023). URL: https://sanaelotfi.github.io/files/project_reports/pac_bayes_bounds_survey.pdf (cit. on p. 42).

[48] Peter B. Harrington. «Multiple Versus Single Set Validation of Multivariate Models to Avoid Mistakes». In: *Critical Reviews in Analytical Chemistry* 48.1 (2018), pp. 33–46 (cit. on p. 42).

[49] Yue Xu and Royston Goodacre. «On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning». In: *Journal of Analysis and Testing* 2 (2018), pp. 249–262. URL: https://doi.org/10.1007/s41664-018-0068-2 (cit. on p. 42).

[50] Frank Emmert-Streib and Matthias Dehmer. «Evaluation of Regression Models: Model Assessment, Model Selection and Generalization Error». In: *Machine Learning and Knowledge* 1.1 (2019), pp. 521–551. URL: https://doi.org/10.3390/make1010032 (cit. on p. 43).

[51] Konstantin Dmitriev, Johann Schumann, Islam Bostanov, Mostafa Abdelhamid, and Florian Holzapfel. «Runway Sign Classifier: A DAL C Certifiable Machine Learning System». In: *2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC)*. IEEE, 2023. URL: https://arxiv.org/abs/2310.06506 (cit. on p. 43).

[52] Konstantin Dmitriev, Johann Schumann, and Florian Holzapfel. «Toward Design Assurance of Machine-Learning Airborne Systems». In: *AIAA/IEEE 40th Digital Avionics Systems Conference (DASC)*. IEEE, 2021. URL: https://ntrs.nasa.gov/api/citations/20210025705/downloads/main.pdf (cit. on p. 43).

[53] Tim Pearce, Mohammed Zaki, and Alexandra Brintrup. «High-Quality Prediction Intervals for Deep Learning: A Distribution-Free, Ensembled Approach». In: (2018). URL: https://proceedings.mlr.press/v80/pearce18a.html (cit. on p. 48).

[54] Xiaoge Zhang and Indranil Bose. «Reliability Estimation for Individual Predictions in Machine Learning Systems: A Model Reliability-Based Approach». In: *Decision Support Systems* 186 (2024). DOI: 10.1016/j.dss.2024.114305 (cit. on p. 48).

[55] Konstantin Dmitriev, Julian Rhein, Lukas Beller, Johannes Brocker, Evangelos Huber, Johann Schumann, and Florian Holzapfel. «Safety Assessment of a Machine Learning-Based Aircraft Emergency Braking System: A Case Study». In: (2024). URL: https://ntrs.nasa.gov/api/citations/20240008842/downloads/DASC_2024-ML_Cert_End_to_End.pdf (cit. on p. 49).

[56] *Doc 9613: Performance-Based Navigation (PBN) Manual*. International Civil Aviation Organization. 2023 (cit. on pp. 55, 75).

[57] David Dalpiaz. *R for Statistical Learning*. Chapter 8: Bias–Variance Tradeoff. Self-published, 2020. URL: https://daviddalpiaz.github.io/r4sl/ (cit. on p. 55).

[58] Alwyn E. Goodloe. «Assuring Safety-Critical Machine Learning Enabled Systems: Challenges and Promise». In: *Proceedings of the 2022 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. 2022 (cit. on p. 56).

[59] European Union Aviation Safety Agency (EASA). *Annual Safety Review 2024*. Technical Report. European Union Aviation Safety Agency, 2024. URL: https://www.easa.europa.eu/en/document-library/general-publications/annual-safety-review-2024 (cit. on p. 60).

[60] Wikipedia. *Haversine formula*. https://en.wikipedia.org/wiki/Haversine_formula. 2025 (cit. on p. 66).

[61] Wikipedia contributors. *Hoeffding's inequality*. 2025. URL: https://en.wikipedia.org/wiki/Hoeffding%27s_inequality (cit. on p. 74).

[62] Inc. RTCA. *Minimum Aviation System Performance Standards: Required Navigation Performance for Area Navigation*. Tech. rep. DO-236D. 2022, p. 176 (cit. on p. 75).

[63] Jayaraman J. Thiagarajan, Vivek Narayanaswamy, Puja Trivedi, and Rushil Anirudh. «PAGER: Accurate Failure Characterization in Deep Regression Models». In: *Proceedings of the 41st International Conference on Machine Learning*. Vol. 235. Proceedings of Machine Learning Research. PMLR, 2024, pp. 21069–21082. URL: https://proceedings.mlr.press/v235/j-thiagarajan24a.html (cit. on pp. 81, 82).