EC2

Methodological Guideline for Trustworthy AI Assessment

L2.2.2, L2.2.3, L2.3.3

contact@confiance.ai | www.confiance.ai

Document reference: 26A

Contributors

	Name	Organisation	Role
Responsible for the deliverable	Henri SOHIER	IRT SystemX	EC2.15 leader
Scientific responsible	Juliette MATTIOLI	Thales	EC2.15 co-leader
Co-authors	Faouzi ADJED	IRT SystemX	
	Kahina AMOKRANE-FERKA	IRT SystemX	
	Afef AWADID	IRT SystemX	
	Yacine BOUAOUNI	Valeo	
	Agnès DELABORDE	LNE	
	Martin GONZALEZ	IRT SystemX	
	Souhaiel KHALFAOUI	Valeo	
	Zakaria CHIHANI	CEA	
	Benoît LANGLOIS	Thales	
Reviewers	Bernard BOTELLA	CEA LIST	
	Bertrand BRAUNSCHWEIG	IRT SystemX	
	Benoit LANGLOIS	Thales	
	Xavier LEROUX	Thales	

Document Control

Revision	Date	Commentary	Author
v0.0	12/11/2023	1st version send to reviewers	J. Mattioli
v1.0	03/01/2024	1st release, using reviewers feedbacks	H. Sohier

Contents

Acronyms

Acronym	Signification
AI	Artificial Intelligence
ALTAI	Assessment List for Trustworthy Artificial Intelligence
ConOps	Concept of Operations
DL	Deep Learning
DM	Decision Makers
DPP	Determinantal Point Process
ETSI	European Telecommunications Standards Institute
EU	European Union
FAA	Federal Aviation Administration
FGSM	FAst Gradient Sign Method
FN	False Negative
FP	False Positive
HLEG	High Level Expert Group
IAA	Inter-Annotator Agreement
IAC	Intra-Annotator Consistency
ID	In Distribution
IT	Information Technology
KPI	Key Performance Indicator
MAE	Mean Absolute Error
MCDA	Multi-Criteria Decision Aiding
MDT	Mean Down Time
MHC	Meaningful Human Control
ML	Machine Learning
MTBF	Mean Time Between Failure
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair
MSE	Mean Squared Error
NIST	National Institute of Standards and Technology
ODD	Operational Design Domain
OOD	Out Of Distribution
PAT	Pumps As Turbines
PGD	Projected Gradient Descent
RAMS	Reliability, Availability, Maintainability, and Safety
RAMT	Reliability, Availability, Maintainability, and Testability
STEEPLE	Social, Technological, Economic, Environmental, Political, Legal, and Ethical

Acronym	Signification
OWL	Web Ontology Language
QA	Quality Assurance
RDF	Resource Description Framework
RMSE	Root Mean Squared Error
SE	Software Engineering
TN	True Negative
TP	True Positive
XAI	Explainable Artificial Intelligence

A. Introduction and abstract

A.1. General introduction to trustworthy AI challenges

Trustworthiness in Artificial Intelligence (AI) within critical systems (systems that can directly or indirectly affect human life and moral entities) is essential for its widespread adoption (by the industry, the decision makers, the general public, etc.) and poses the following significant challenges.

- First, how to design AI models, so that, by construction, they satisfy trustworthy properties (accuracy, robustness. . .).
- Secondly, how to characterize these AI models, for example to understand and explain their behavior and their adequacy to the operational domain.
- Then, how to implement and embed those AI models on hardware, by making them fit for the target without losing their trustworthy properties.
- Another question is, what methods of data engineering to apply in order to, among other topics, manage important volumes of data and adapt to the evolution of the operational domain.
- At system level, what verification and certification processes to consider specifically for AI-based systems.
- Finally, a federation of all these matters is necessary to build an end-to-end methodological approach, supported by a consistent engineering environment compatible with industrial practices.

These are the challenges, among others, that the Confiance.ai program addresses. The target audience of this document includes AI specifiers (product owners, system engineers, ...), AI developers (programmers, data scientists, ...) and AI validators (testers, auditors, certifiers, ...). It is also of interest for AI policy makers or AI users.

The trustworthiness characteristics can be assessed only if the Operational Design Domain (ODD)[1] is clearly defined. The ODD defines the conditions on the environment where the AI sub-system shall operate. Many AI prototypes neglect to describe their ODD or leave it vaguely defined as the domain covered by the distribution of data used during training. However, how to predict the performance or the risks of using this system in an operational context without analyzing and understanding if the real environment will match this training distribution? Therefore, it is necessary to abstract from the data used for the training of the models a characterization of the situations in which this model will be applied; and at the same time to characterize the situations in which we want the model to be applied for the construction of representative training data sets by covering these situations.

Trustworthy AI is based on three components High-Level Expert Group on Artificial Intelligence (2019), which should be met throughout the system's entire life cycle: (1) it should be

[1] An ODD is concept created initially for automated driving system (ADS) used to restrict where the ADS is valid Gyllenhammar et al. (2020). In the current work, ODD is a restriction of the domain where an AI-based system acts safely.

lawful, complying with all applicable laws and regulations (2) it should be ethical, ensuring adherence to ethical principles and values and (3) it should be robust, both from a technical and social perspective since, even with good intentions, AI systems can cause unintentional harm. But, in the following document *trustworthiness is the ability to meet stakeholders' expectations in a verifiable way* ISO/IEC TR 24028 (2020) and also in line with the HLEG requirements. Thus:

- Depending on the context or sector and also on the specific product or service, data and technology used, different characteristics apply and need verification to ensure stakeholders expectations are met.
- Characteristics of trustworthiness include, for instance, reliability availability, resilience, security, privacy, safety , accountability, transparency, integrity, authenticity, quality, usability (see fig. A.3).
- Trustworthiness is an attribute or a quality that can be applied to services, products, technology, data and information as well as, in the context of governance, to organizations.

A.2. Why assessing AI Trustworthiness?

AI-based system refers to a software-based system that comprises AI components besides traditional software components. AI-based system adoption depends on its ability to deliver the expected service safely (*i.e., conformance to requirements*), to meet user expectations (*i.e., fitness for use*) and to maintain service continuity. Moreover, trustworthiness is closely related to accountability which can be considered as a factor of trust or as an alternative to trust O'Neill (2014). Thus, such AI-based critical systems need to be valid/correct, accountable, explainable, resilient, safe, secure, compliant with regulations and standards (including ethics and sustainability).

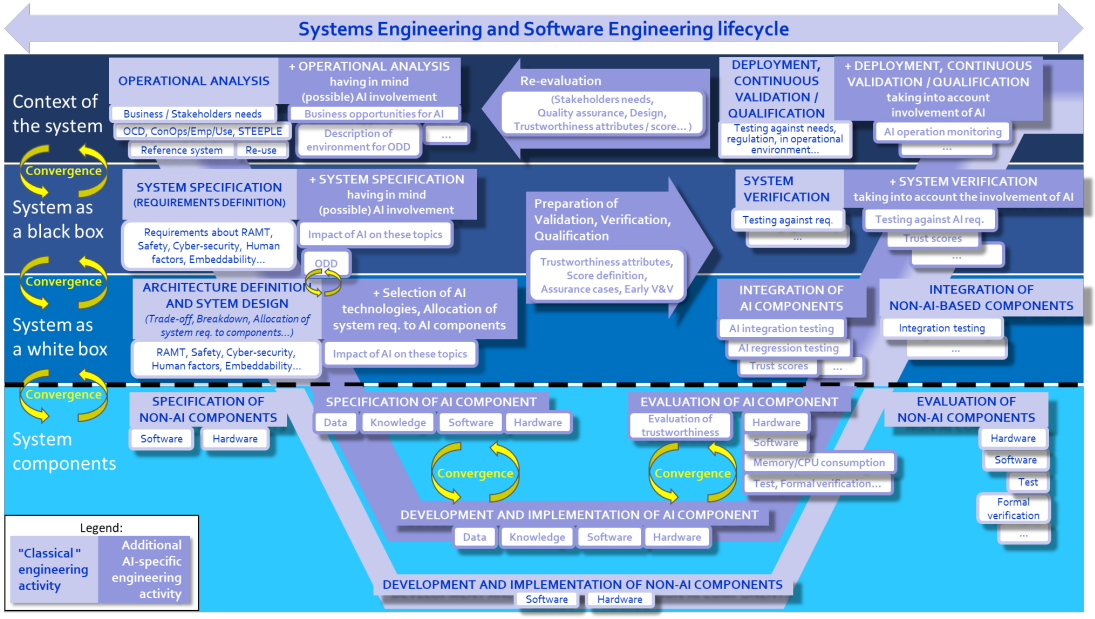


Figure A.1: Confiance.ai end-to-end approach presented as Systems Engineering and Software Engineering cycles augmented with AI perspectives

Most active academic research on AI trustworthiness has focused on the algorithmic properties of AI-models. However, advances in algorithmic research alone are not sufficient for building trustworthy AI products. This includes data preparation, algorithm design, development and deployment, and operations, monitoring and management. Efforts at multiple stages of this lifecycle, robust algorithms, anomaly monitoring, and risk auditing are required to improve the trustworthiness of a single aspect (e.g. robustness). Conversely, a breach in any single link or aspect can undermine the trustworthiness of the whole system. Therefore, throughout the lifecycle of an AI system, its trustworthiness should be systematically established and evaluated.

The value chain for AI/ML systems is very different from that for traditional software development, with responsibility for quality spread across the value chain. Conventional methods for testing and validating algorithms fall short due to the multi-dimensional nature of trustworthiness (accountability, accuracy, controllability, correctness, data quality, reliability, resilience, robustness, safety, security, transparency, explainability, fairness, privacy etc.). AI-based system design shines a light on quality requirements ("-ilities", or non-functional requirements) which appear particularly challenging. Beyond quality requirements, this can also encompass social-technical system risk and process considerations.

The expected attributes and the expected values for these attributes depend on contextual elements such as the level of criticality of the application, the application domain of the AI-based system, the expected use, the nature of the stakeholders involved, etc. This means that in some contexts, certain attributes will prevail, and other attributes may be added to the list.

All along the methodological steps, a risks/opportunities-driven approach has to be applied to ensure that the engineering orientations, decisions and technological choices, specific to the involvement of AI, are identified, analyzed and mitigated as early as possible (e.g. datasets evolution, occurrence of unwanted emerging behaviors, *etc*). Potential opportunities, typically enabled by some technologies, must be considered, analyzed and valorized. This risks/opportunities-driven approach may require additional iterations of the qualification and certifications phases. This process (see fig. A.2) aims at defining the subset of a target operational domain where systems/features of interest can be automated with an acceptable level of confidence. These systems/features have to be considered as functional chains, potentially involving several AI-based and non-AI-based components. This process relies on the reliability assessment process that aims at identifying and characterizing expectations on trust. To achieve this objective, the ODD analysis (see fig. A.1) processes hereafter from an initial legacy system/feature has to be identified from business domains and/or technological domains. This legacy system/feature is considered as a reference.

From this reference, system/feature automation objectives, expected level of automation and design intentions can be defined (for instance, human activities or behaviors that could be automated at a particular level, with regard to their operational environments, conditions and trust expectations). Based on the trust characteristics analyzed by the reliability Assessment Process, the ODD analysis process defines the observable/measurable conditions and properties that need to be supervised and monitored. It also defines nominal and edge/corner cases scenarios. All this characterizes and describes the ODD of the AI systems/features, related to the automation objectives, their associated level and design intentions and the target operational environment.

Just remind that in system engineering, a **corner case** (or pathological case) involves a problem or situation that occurs only outside of normal operating parameters—specifically one that manifests itself when multiple environmental variables or conditions are simultaneously at extreme levels, even though each parameter is within the specified range for that parameter. An **edge**

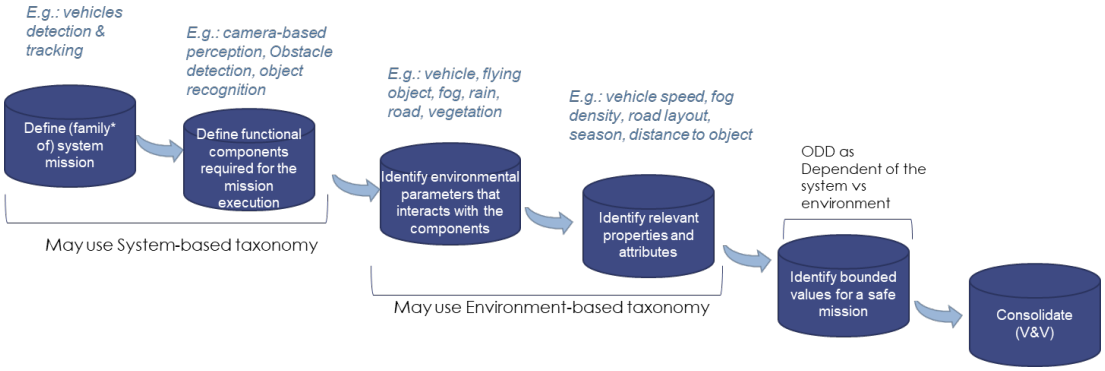


Figure A.2: The specification process of the ODD is grounded on two existing approaches: a taxonomy-based approach and an analytical based approach.

case is a problem or situation that occurs only at an extreme (maximum or minimum) operating parameter. For example, a stereo speaker might noticeably distort audio when played at its maximum rated volume, even in the absence of other extreme settings or conditions.

Trustworthiness characteristics (see fig. A.3) can be assessed only if the Operational Design Domain (ODD) is clearly defined Confiance.ai EC6 (2023), where ODD specifies the conditions on the physical and cybernetic environment where the AI sub-system shall operate. Many AI prototypes neglect to describe their ODD or leave it vaguely defined as the domain covered by the distribution of data used during training.

Definition 1 *(ODD) Operational Design Domain (ODD) is the set of operating conditions under which a given AI- system is specifically designed to function as intended, in line with its intended purpose Confiance.ai EC6 (2023).*

Operating conditions includes but is not limited to environmental, geographical, and/or time-of-day restrictions, age, knowledge and health conditions of users.

We need to abstract from the data/information used to train/build the AI models a characterization of the situations in which this model is running, and at the same time characterize the situations in which the model should run to construct representative training datasets by covering these situations or the knowledge bases to capture business knowledge and application constraints.

Assessments and audits may also be included in mandatory authorization and regulatory procedures. The European Commission's draft regulation indicates that such authorization procedures for AI will be introduced for the European market in the near future. As well as banning certain applications of AI, the draft directive requires high-risk systems to undergo a conformity assessment procedure. Last but not least, full trustworthiness in AI systems can only be established if all technical activities to establish trustworthiness are clearly defined for example by regulations, norms and standards to support the governance, processes of organizations and/or end-to-end methodology that use, develop and deploy AI.

Systematically implementing quality in the development of their own AI applications, as well as assessing the quality of third-party systems, is essential for AI engineers. Assessing the trustworthiness of AI becomes the cornerstone of successful improvement in the design and operation of critical systems Mattioli et al. (2023b). But it should be noted that the specific requirements for an AI application to be trustworthy are highly dependent on the technology used and the application context. KPIs for measuring the quality of AI applications are important.

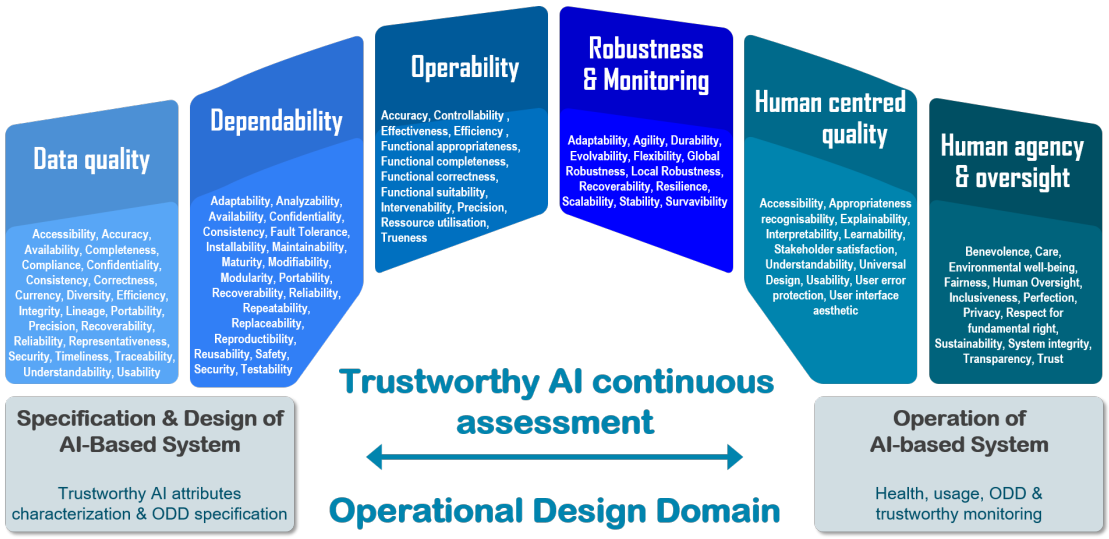


Figure A.3: Some trustworthy AI characteristics Mattioli et al. (2023a).

However, obtaining trustworthiness measures remains a challenging task. On the one hand, measuring trust can help identify problems with the system before they become critical and allow for mitigation action to be taken before a failure occurs. On the other hand, measuring trust can help to improve the design of critical systems. Indeed, by understanding the factors that contribute to user trust in AI systems, designers can create ones that are more reliable, safe and secure.

Another challenge in defining specific quality requirements for AI/ML applications is that different dimensions of trustworthiness cannot be assessed completely independently of each other. Instead, trade-offs must be made. For example, increasing performance, such as the recognition performance of deep learning on image data, may come at the expense of traceability, or increasing transparency (for example, by revealing all hyperparameters of a model) may lead to new attack vectors related to IT security.

As mentionned before, the assessment of many trustworthiness characteristics requires the definition of the ODD and depends on its quality. From the operational analysis level, trustworthiness properties have to be analyzed and refined into system/product detailed specifications, from the automation perspective in the framework of the ODD.

A.3. Purpose of this Guideline

The purpose of this document is to support two main engineering activities (see Fig. A.4):

1. **The ODD specification** by providing a list of trustworthy AI characteristics. For each characteristics, we illustrate the motivation for definition of each requirement.
2. **The trustworthiness attributes analysis**: We also survey approaches to assess them by defining their associated KPIs (key performance indicators). It should also be noted that the selected characteristics are not orthogonal, and some of them are closely correlated.

This methodology focuses o the trustworthiness assessment a system which embeds connectionist and statistical artificial intelligence techniques such as machine learning. However, the

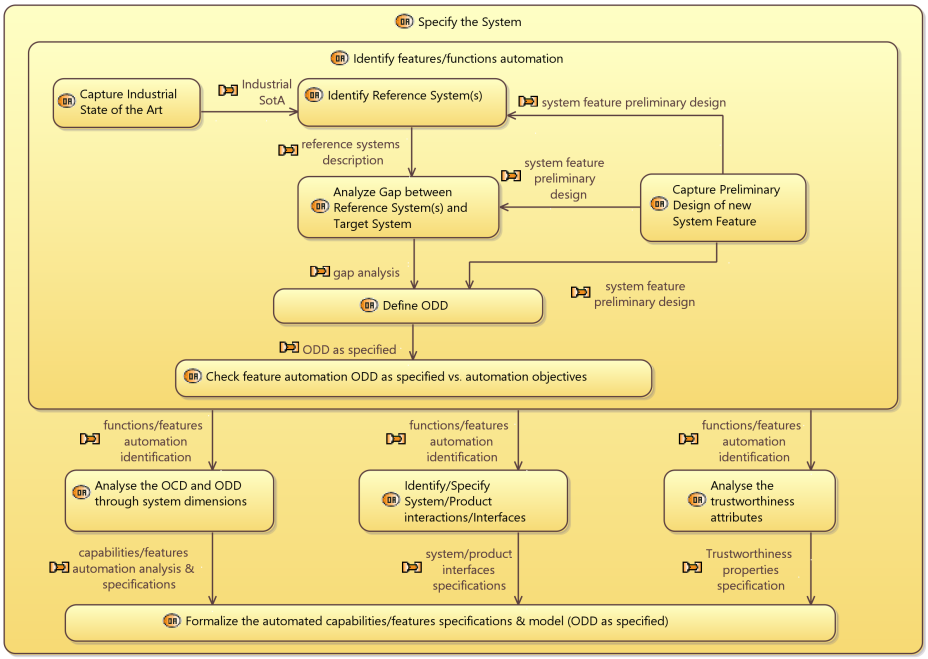


Figure A.4: Engineering Activities for “System Specification” from AI perspective

definitions, KPIs and metrics may cover other AI approaches in some cases where relevant.

B. Description of the method

B.1. Foreword

Trust is defined ISO/IEC 25010 (2011a) as "*the degree to which a user or other stakeholders has confidence that a product or system will behave as intended*". But, the trust literature distinguishes trustworthiness (the ability, benevolence, and integrity of a trustee) from trust (the intention to accept vulnerability to a trustee based on positive expectations of his or her actions) Colquitt et al. (2007). Trustworthiness is represented as an objective aspect of trust estimated based on evidences or observations; whereas trust includes subjective aspects of a cognitive entity's opinion such as a human.

Moreover, ML-based systems are data-intensive, continuously evolving, self-adapting, and their behavior has a degree of (commonly accepted) uncertainty due to inherent nondeterminism.

Trustworthy characteristics require adapted and new constructive and analytical quality assurance (QA) approaches from the field of software engineering (SE) in order to guarantee the quality during development and operation in live environments. However, so far the concept of "trustworthiness" in AI-based systems is not well-defined. Thus, in this guideline, we consider the following definitions:

Definition 2 (trustworthiness) *Trustworthiness ability to meet stakeholders' expectations in a verifiable way ISO/IEC TR 24028 (2020).*

Definition 3 (trust) *Trust is the degree to which a user or other stakeholder has confidence that a product or system will behave as intended ISO/IEC TR 24028 (2020).*

Risk assessment is a critical stage in the objective evaluation of trust. This can be based on three dimensions: uncertainty/ambiguity, vulnerability and impact. Uncertainty is the lack of evidence for the assessment of trust, while ambiguity is the conflict of evidence from different sources for the same proposition. Outcome is the result after the decision has been made. Depending on whether the decision turns out to be right or wrong, trust is adjusted accordingly. This is called trustworthiness, which represents objective trust based on the observed outcome, while overall trust can be formed based on the combination of subjective trust and objective trust. Trustworthiness is the trust that is validated by the actual evidence. The result is fed back into the trust assessment cycle in repeated situations. That is, validated trust can reinforce the prior belief, while betrayed trust can fix the prior belief and update the current belief based on the new evidence. In this sense, the mental process of trust assessment is recursive in nature, because a prior belief generates an outcome that is fed back as input to update trust.

System Trustworthiness is a measure of the extent to which a system can be trusted to meet desired critical requirements, often related to security, reliability, guarantees of real-time performance and resource availability, survivability, and more - all in the face of a wide range of adversities (known and some unknown). Trustworthiness depends on hardware, software, communications media, power supplies, physical environments, and ultimately on people in many roles - requirements specifiers, designers, users, operators, maintenance personnel..., and (unfortunately) the ability of attackers to exploit weaknesses or limitations in all of the above. Thus, trustworthiness concerns the product itself, but also the processes (how the product was made),

the tools and infrastructure (with what), the people (by whom), and the governance (who decides). Assessing trustworthiness also combines different approaches, such as risk management and quality management.

In 1977, a FAA (Federal Aviation Administration) panel dedicated to how to certify aircraft as airworthy explicitly linked the notion of trustworthiness to accounting, which can be considered as a factor of trust or as an alternative to trust O'Neill (2014).

In 2019, both SAE and EUROCAE have established standardization committees and working groups in preparation for the use of AI in aviation EASA (2020). The Aerospace Vehicles Systems Institute (AVSI) and its AFE-87 working group have been working on machine learning in recent years. Following this research Jenn et al. (2020), they are currently setting up another working group called "Machine Learning Certification". In addition, the DEEL project is working on reliable, certifiable and explainable artificial intelligence for critical systems Mamalet et al. (2021). This project involves participants from both academia and industry from several disciplines, including aeronautics. Certification authorities are also seriously addressing the issue of enabling AI in aviation.

Then, in Avizienis et al. (2004), dependability is used to represent the overall quality measure of a system based on four sub-attributes including security, safety, reliability, and maintainability. Thereafter, security and dependability became key attributes for computer-based system trust Cho et al. (2019). Then, security and dependability became key system attributes Cho et al. (2019) to assess the trustworthiness of a computer-based system: Avizienis et al. Avizienis et al. (2004) used dependability to represent the overall quality measure of a system based on four sub-attributes including security, safety, reliability, and maintainability. To depict dynamic system security metrics, Pendleton et al. Pendleton et al. (2016) presented a cyber-security metric framework based on the interactions between attackers and defenders. By taking into account a variety of probabilistic and/or analytical models to undertake a quantitative evaluation, Ramos et al. Ramos et al. (2017) primarily examined model-based network security indicators. Gol Mohammadi et al. Gol Mohammadi et al. (2013) proposed the first list of software trustworthiness attributes.

In 2019, the U.S. National Artificial Intelligence Research and Development Strategic Plan Science and on Artificial Intelligence (2019) emphasized that: *"standard metrics are needed to define quantifiable measures in order to characterize AI technologies"*. More recently, Schmidt et al. (2021) noted that "*significant work is needed to establish what appropriate metrics should be to assess system performance across attributes for responsible AI and across profiles for particular applications/contexts.*". Wing (2021) defines a trustworthy computing system using a combination of overlapping properties, namely reliability, safety, security, privacy, availability, and usability. For an AI-based system, this translates and extends to accuracy, robustness, fairness, accountability, transparency, explainability, and ethics. Delseny et al. (2021) also considers auditability for certifiable AI systems. These properties can be separated into two main categories: design process and final use. A transparent and ethical system must respect specific design requirements that can be proven using proper tracking and reporting. These properties are hard to formalize and hence cannot be formally nor numerically verified. Furthermore, Brundage et al. (2020) and Floridi (2019) provide industrial and institutional guidelines for a trusted / trustable / trustworthy AI design process.

The Assessment List for Trustworthy AI High-Level Expert Group on Artificial Intelligence (2019) considers 7 pillars of trustworthiness: 1) human agency and autonomy, 2) technical robustness and safety, 3) privacy and data governance, 4) transparency, 5) diversity, non dis-

crimination and fairness, 6) societal and environmental well-being, and 7) accountability. The European Commission has proposed a set of rules for AI, the AI Act European Commission (2021), regulating the technology.

In the aeronautic domain, EASA (2021) proposes a model of Machine Learning trustworthiness based on: the characterization of the Machine Learning (ML) application (high-level function/-task, concept of operations, functional analysis, classification of the ML application), safety assessment, information security management, and ethics-based assessment (which includes the 7 pillars of the ALTAI).

The Fraunhofer Mock et al. (2021) offered an analysis of the standard (ISO/IEC DIS 42001, 2022, Under development) on management system for AI, stating compliance to the standard can contribute to ensuring AI trustworthiness since it encompasses the pillars of the ALTAI, provided that a third-party verification has been performed and along with an adapted quality management system.

In the same period, the NIST (National Institute of Standards and Technology) produced an analysis of the components of trust Stanton et al. (2021) and highlighted several top level aspects for the design of a trustworthiness model, that should encompass the user experience, the perceived technical trustworthiness, the pertinence of each trustworthiness characteristic in the user's specific context of use, *etc*

Moreover, ETSI (European Telecommunications Standards Institute) set-up in 2019 an Industry Specification Group on Securing AI (ISG SAI) from attack to resilience ETSI (2021) providing existing and potential mitigation against threats for AI-based systems.

In an assessment context, trustworthiness means valid and reliable assessments that really do assess what they claim to, and assessment procedures that produce consistent results, when administered in similar circumstances, at different times, involving different raters and well-grounded in evidence. However, as a property of a ML-based system, such trustworthy concept is complex and determined by considering many characteristics as well as its application in particular contexts (see fig.B.1). Trustworthiness is not an independent quality characteristic in its own right, it is a super-set of measurable quality characteristics.

Addressing AI trustworthiness characteristics individually or at component level will not ensure AI trustworthiness at system level; tradeoffs are usually involved. Moreover, it is rare that all of the characteristics apply in every situation, some will be more or less relevant in a given situation. For example, in certain scenarios tradeoffs may emerge between predictive accuracy and robustness. This implies that the relative importance of each attribute can fluctuate depending on the circumstances wherein such system is operating Braunschweig et al. (2022).

While most active academic research on trustworthy ML has focused on the algorithm properties, its holistic modeling has received very little attention given the lack of literature. Moreover, the life cycle of an ML application offers various approaches to mitigating risks. The measures of trustworthiness characteristics are divided into the following three categories according to the assets:

- data
- ML component
- ML system

From a strict metrological point of view, trustworthiness may not be measured as it is not a physical property that can be compared to a reference quantity of the same kind. Trustworthi-

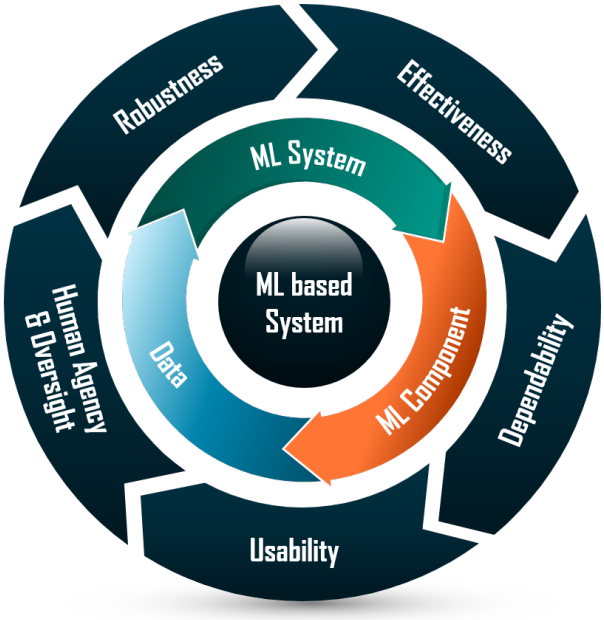


Figure B.1: Focussing on main trustworthy ML characteristics

ness is an aggregation of trustworthy characteristics and then does not have a global unit. More generally speaking, "to measure" refers to assign an element of a scale to an object in order to quantify an attribute of it. Thus, a metric related to a specific characteristic is defined as Adam et al. (2022) an "*objective, mathematical measure of a ML-based component/system that is sensitive to differences in safety-critical characteristics. It provides a quantitative measure of an attribute which the body of solution exhibits*".

B.2. Terminology

According to Ackoff (1989), the content of the human mind can be classified into five categories: Data, Information, Knowledge, Insight, and Wisdom. Data, Information and knowledge become decision resources that must be managed smartly. But, data, facts and information are often used interchangeably with knowledge. Nevertheless, data represents the properties of objects and events.

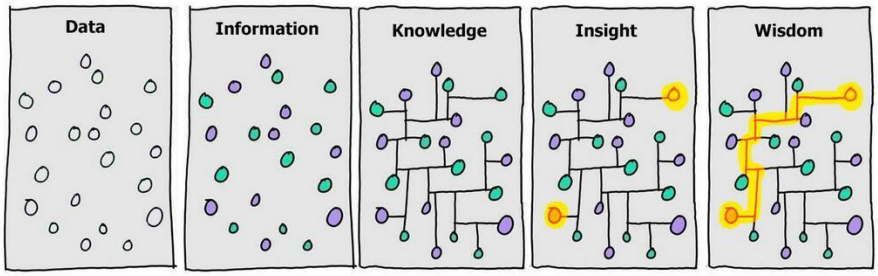


Figure B.2: Data, Information, Knowledge, Insight, Wisdom

In figure B.2, **Data** is represented by a series of random dots that could mean something – or nothing. Next comes **Information** where meaning or relationship are extracted from the raw

data (indicated by colors associated to the dots). **Knowledge** is obtained when we are able to memorize the information, for example: standard multiplication tables or sunrise and sunset times in a given month. As we gain knowledge we begin to make sense of the data by drawing connections between different pieces of information. However, it is at the **Insight** level where data becomes effectively useful. Insight is the ability to synthesize knowledge in order obtain a deep understanding of a problem. With insight comes the prospect of **Wisdom** – the ability to use insight to facilitate informed decision making.

Definition 4 (data item) *Data item is defined as re-interpretable representation of information in a formalized manner suitable for communication, interpretation or processing ISO/IEC 2382 (2015).*

Data can be processed by human or automatic means to the intelligent reasoning engines Ackoff (1989) in order to give them a representation of the world. Once the data samples are acquired by a system, its becomes information. This information is then interpreted by intelligent system, according to the context related to the acquisition of the data items, and the domain knowledge, among others. The domain knowledge is used to interpret such set of data items within a specific context.

Definition 5 (dataset) *A dataset is an aggregation of data, typically spawning more than one physical file,that are processed together and serve collectively as input or output of a computation or data acquisition process Branco et al. (2008).*

Definition 6 (information) *Information represents the properties of objects and events, but in a more compact and useful form. The difference between data and information is functional, not structural. Information can be represented as descriptions, answers to questions that begin with such words as who, what, when, where, and how many Ackoff (1989).*

Definition 7 (knowledge) *Knowledge is conveyed by instructions, answers to how-to questions. Understanding is represented by explanations, answers to why questions. No understanding is possible without knowing the context in which the process of perception of a situation occurs Ackoff (1989).*

Definition 8 (artificial intelligence) *Artificial intelligence (AI) is a theoretical and practical interdisciplinary field, with the objective of understanding the cognitive and thinking mechanisms, and their imitation by a material and software device, for assistance or substitution purposes of human activities. JORF (2018)*

Definition 9 (artificial intelligence system) *AI systems are software (and possibly also hardware) systems designed by humans that, given a complex goal, act in the physical or digital dimension by perceiving their environment through data acquisition, interpreting the collected structured or unstructured data, reasoning on the knowledge, or processing the information, derived from this data and deciding the best action(s) to take to achieve the given goal. AI systems can either use symbolic rules or learn a numeric model, and they can also adapt their behavior by analyzing how the environment is affected by their previous actions. High-Level Expert Group on Artificial Intelligence (2019)*

Definition 10 (machine learning) *Machine learning is a branch of artificial intelligence (AI) [...] that refers to the automated detection of meaningful patterns in data. [It covers] a set of techniques that can learn from experience (input data). Mamalet et al. (2021)*

Definition 11 (machine learning model) *A machine learning (ML) model is a mathematical construct that generates inferences, or predictions, and that a machine learning model is the result of training a machine learning algorithm ISO/IEC DIS 22989 (2021a).*

B.3. A brief overview of the method

Trustworthiness as a property of an AI system is complex, in the sense that it is made up of parts, and contextual usage, implying that the relative amount each component matters (and, potentially, how a given component is measured) can change based on the context in which the AI system is operating. The "trustworthiness" concept can be broken down into different attributes Liu et al. (2022).

B.3.1 A new ML trustworthiness meta-model

A trustworthy software is defined Wing (2021) by a combination of overlapping properties: reliability, safety, security, privacy, availability and usability. For a ML-based system, this translates and extends to accuracy, robustness, fairness, accountability, transparency, explainability and ethics. Delseny et al. (2021) also considers auditability.

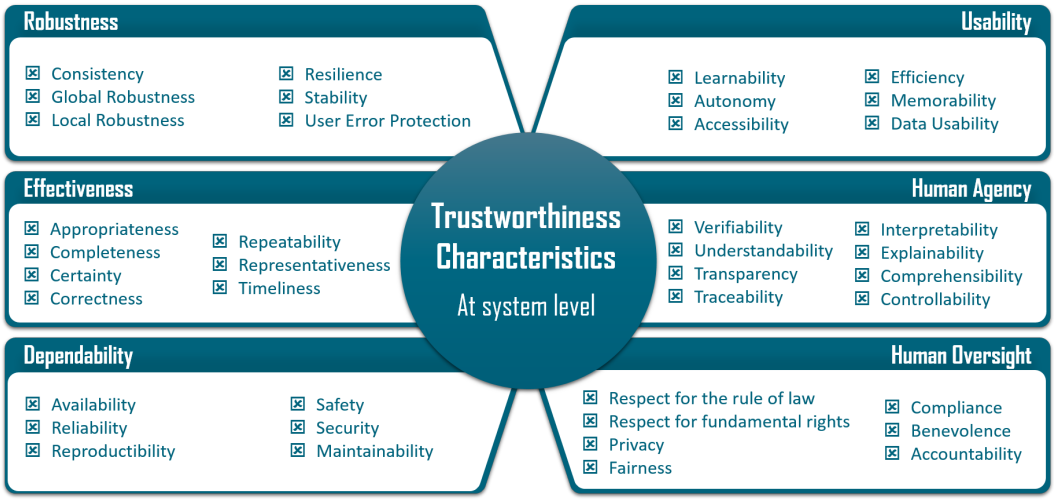


Figure B.3: The AI Trustworthiness characteristics and sub-characteristics

In this guideline, such attributes are currently grouped according to the capabilities they characterize (see fig. B.3): Robustness, Effectiveness, Dependability, Usability, Human agency and Human Oversight.

The first three characteristics (**Robustness, Effectiveness, Dependability**) are system and ML model centric. They refer to the ability to verify that the AI-based component has validity, performance and robust intrinsic properties such as accuracy, robustness, safety and security. Thus, AI-based systems should generate accurate output as consistent as possible with the ground truth. Additionally, AI systems should be robust to changes, specifically in complex, dynamic and uncertain real environments. Moreover, AI programs or systems must not harm any human being under any circumstances that prioritize user safety. In addition, the autonomy of trustworthy AI should always be under user's control. In other words, it has always been the human right to give the AI system decision-making authority or to revoke that authority at any time.

From **usability**'s perspective, trustworthy AI should possess the properties of efficiency, accessibility, usability and autonomy. Specifically, AI-based systems should not cease operation at inappropriate times (e.g. at times when the lack of output could lead to safety risks), and these programs or systems should be easy to use for people with different backgrounds.

Human agency and oversight, strongly linked in Europe to the notion of fundamental rights, are notably put forward in the work of the AI HLEG (High-Level Expert Group on Artificial Intelligence) of the European Commission High-Level Expert Group on Artificial Intelligence (2019). A system must, for example, be law-abiding, fair, accountable, environmentally friendly and compliant with the user privacy. Specifically, AI systems should operate in accordance with all relevant laws and regulations, as well as with the ethical principles of human society. Trustworthy AI solutions should also allow explanation and analysis by humans to reduce potential risks and harms and empower human users. In addition, trustworthy AI should be transparent so people can better understand its mechanism.

As some notions (e.g. explainability) concern several dimensions (ethics vs. interaction), Confiance.ai program made an arbitrary choice to be consistent with the methodology. Finally, the attributes for trusted ecosystem intermediaries focus on the relationships to third-parties, in particular quality assurance, audit and certification activities. All these properties apply to the AI-based component and the system that embeds AI, but they also apply to the quality of the data used for training connectionist AI and/or to the quality of knowledge representation and reasoning used in symbolic AI.

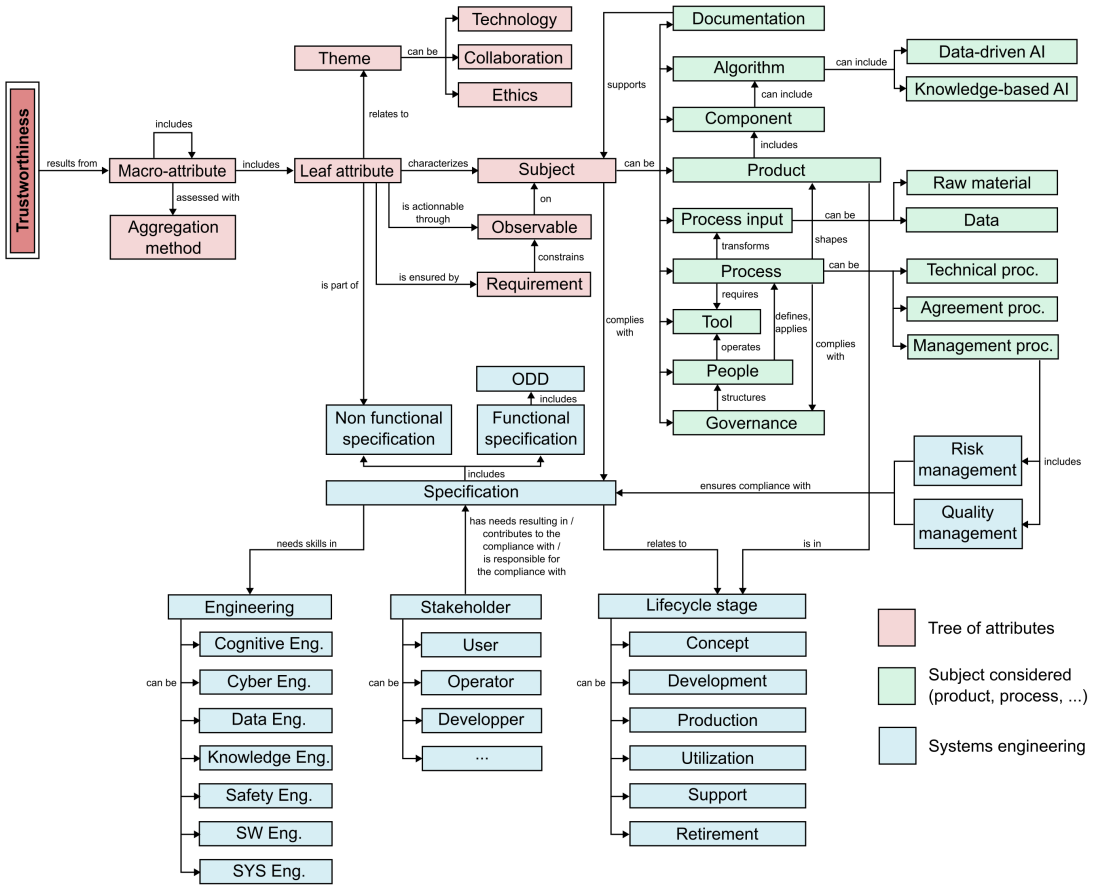


Figure B.4: A new ML trustworthiness meta-model Mattioli et al. (2023c)

To capture the type of considered information and the different inter-relations needed to assess ML trustworthiness, we proposed a meta-model with concepts in different abstraction levels (see fig. B.4). The red part describes the way the tree of attributes is built. It highlights the abstract

concepts central to trustworthiness assessment. An attribute which aggregates other attributes is called a macro-attribute (e.g. robustness, dependability, *etc*). It is assessed with an aggregation method. An atomic attribute (leaf attribute) is assessed with a clear and actionable observable which can take different forms (metric, "expected proof").

The green part of fig. B.4 is the meta-model fragment with concrete concepts. These concepts represent the different possible subjects and relations between them. For example, the product is developed following processes as technical processes (through which the product must go: design definition, implementation, operation...), agreement processes (with external organizations: acquisition, supply), and management processes (supporting the development of the product: quality management, risk management, *etc*). Risk and quality management ensures the compliance with the specification which includes the different expected trustworthiness attributes. Processes are applied with tools by people respecting a certain governance.

The blue part summarizes systems engineering key concepts more precisely part of the non-functional specification: they do not define what the system "does" or how the system works, but what the system "is". The attributes are also commonly referred to as "*-ilities*" as they often have this suffix. They can also be referred to as quality requirements. Whether a specification is functional or non-functional, it is influenced by stakeholders such as the user, the operator, the developer, *etc*

As opposed to non-functional requirements which define what the system is, functional requirements define what the system does: should it move? roll? roll fast? under what conditions? From this point of view, the Operational Design Domain (ODD), which characterizes the conditions of operation of the system, can be considered part of the functional specification relating to trustworthiness attributes in different ways: 1) Transparency on the ODD permits to understand the system's capabilities and limits (which is part of the AI Act's requirements); 2) The ODD is the domain to consider for the different operational trustworthiness attributes; 3) The ODD has its own attributes (it should be complete, free of inconsistencies, human readable, *etc*).

Definition 12 (operational design domain) *The operational design domain refers to operating conditions under which a given AI- system is specifically designed to function as intended, in line with its intended purpose.*

Thus, the trustworthiness attributes can be assessed only if the ODD is clearly defined. The ODD defines the conditions on the environment where the AI sub-system shall operate. Many AI prototypes neglect to describe their ODD or leave it vaguely defined as the domain covered by the distribution of data used during training. However, how to predict the performance or the risks of using this system in an operational context without analysing and understanding if the real environment will match this training distribution? Therefore, it is necessary: either to abstract from the data used for training the models a characterization of the situations where this model is performed ; and, at the same time, to characterize the situations where we want the model to be performed and construct representative training datasets by covering these situations. The assessment of many trustworthiness attributes requires the definition of the ODD and depends on its quality.

B.4. Unified approach to support trustworthiness assessment

Multi-Criteria Decision Aiding (MCDA) is a generic term for a collection of systematic approaches developed specifically to help one or several Decision Makers (DM) to assess or com-

pare some alternatives on the basis of several criteria Labreuche (2011). The difficulty is that the decision criteria are frequently numerous, dependent and sometime conflicting. For example, effectiveness may be conflicting with robustness, explainability, or affordability. The viewpoints are quantified through attributes (see §B.4.1).

First, to assess AI trustworthiness, the choice of the relevant attributes is not easy, since the selection pertains to the context of application, which is modeled according to several elements (Operational Design Domain, intended domain of use, nature and roles of the stakeholders, *etc.*) The attributes can be quantitative (typically numerical values either derived from a measure or providing a comprehensive and statistical overview of a phenomenon) or qualitative (based on the detailed analysis and interpretation of a limited number of samples). Then once the list of relevant attributes has been defined, the aggregation of several attributes remains complex due to commensurability issues: indeed, this is equivalent with combining "oranges and apples", none of the attributes having the same unit. In addition, one aims at making trade-offs and arbitration between the attributes. This means that the value of each attribute should be transformed into a scale common to all attributes and representing the preferences of a stakeholder, and that the values of the scales for the different criteria should be aggregated. These elements constitute the main steps for solving the problem using an MCDA approach.

Aggregation functions are often used to compare alternatives evaluated on multiple conflicting criteria by synthesizing their performances into overall utility values Grabisch and Labreuche (2010). Such functions must be sufficiently expressive to fit the DM's preferences, allowing for instance the determination of the preferred alternative or to make compromises among the criteria - improving a criterion implies that one shall deteriorate on another one.

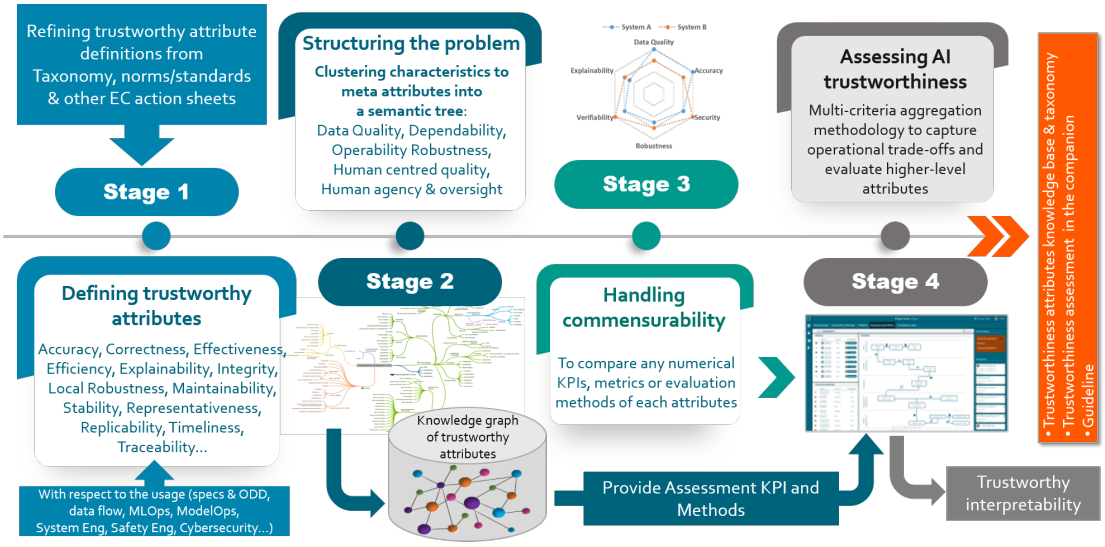


Figure B.5: The unified approach based on MCDA

MCDA provides a tool to specify the good compromises Labreuche (2011). Our approach is based on the following steps (see fig.B.5):

1. Step 1: Structuring attributes in a semantic tree;
2. Step 2: Identification of numerical evaluations;
3. Step 3: Adapting attributes for commensurability;
4. Step 4: Definition of an aggregation methodology to capture operational trade-offs and

evaluate higher-level attributes.

B.4.1 Step 1: Semantic tree

Based on different sources (norms, standards, scientific communications, industrial and institutional reports, Confiance.ai reports, *etc*.), the characterization and evaluation of trust attributes focus on defining and structuring the attributes that constitute trust in the context of AI-based safety critical systems Pons and Ozkaya (2019) going beyond a risk analysis as proposed in High-Level Expert Group on Artificial Intelligence (2019); Piorkowski et al. (2022).

Our problem of assessing Trustworthiness is decomposed in several sub-problems by introducing a hierarchy of an important number of specific criteria. This structuring phase aims to construct a tree representing a hierarchy of points of view in which the root represents the overall evaluation, and the leaves are the elementary attributes. In order to produce such a hierarchy, one shall succeed in grouping the criteria according to a classification that makes sense for the stakeholders. At the end of this step, one shall obtain the relevant criteria together with their organization in a tree.

B.4.2 Step 2: Numerical evaluations

All atomic attributes return a numerical evaluation. Specific Key Performance Indicators (KPI), metrics or evaluation methods are used to qualify the leaves of the tree according to the use cases. For example, data quality is a problem that has been studied for several decades now Mattioli et al. (2022b). However, primarily the focus has been on the data in operational databases and data warehouses. Now, Data-driven AI is generating renewed interest in data quality, but there is yet no consensus on what comprises the data quality characteristics. Thus, Wang and Strong (1996) were among the first arguing that limiting quality to the level of accuracy is not enough, highlighting that the level of quality for given data can depend on its purpose. Its principles require an assessment of the various quality attributes as presented in mainly in fig. B.3. Standards are currently being developed to define data quality attributes for ML (Machine Learning): ISO/IEC CD 5259-1 (terminology and principles) and ISO/IEC CD 5259-2 (data quality measures).

B.4.3 Step 3: Commensurability

Aggregating different attributes for a global assessment requires that they are commensurate. This implies that one shall be able to compare any numerical evaluation of an attribute with any numerical evaluation of any other attribute. To make the assessment 'comparable', sound methods of normalization (making comparisons between variables comparable) must be applied to individual variables to first make them comparable, that is, to transform different scales of variables into a single scale. The numerical evaluation of the attributes is thus encoded in the $[0, 1]$ interval where the value 0 corresponds to the total absence of the property beneath a trustworthiness criterion, and value 1 corresponds to the complete satisfaction of the criterion. The normalized indicators could be aggregated using specific formulas (e.g. min/max, arithmetic mean, weighted sum, *etc*.). If one attribute is more "important" than another with respect to stakeholder preference, the former is assigned a stronger weight than the latter within the aggregation procedure.

B.4.4 Step 4: Aggregation and trade-off

The global assessment would be made on the basis of several trustworthiness attributes denoted by $N = \{1,...,n\}$. The proposed approach provides a tool to identify best compromises from the stakeholder point of view. Each attribute $i \in N$ is quantified by a KPI – also called metric or Figure of Merit – represented by the set of its possible values X_i. The alternatives are characterized by a value on each attribute and can be fully described by elements of $X = X_1 \times ... \times X_n$. An alternative x can thus be represented by a vector $(x_1,...,x_n) \in X$.

Radar chart is a visual method for comprehensive evaluation, particularly useful for holistic and overall assessment through multivariate data. However, this representation does not allow understanding the interactions and dependencies between attributes.

The goal of MCDA is to define a numerical representation of the preferences of the stakeholders, expressed as a function $u : X \to [0,1]$. The function will be used to compare each alternative or assess each alternative's level of satisfaction in order to provide the overall level of satisfaction for each. As mentioned before, the scale $[0,1]$ can be interpreted as a degree of satisfaction. It is classical to write u in a decomposed way : $u(x) = F(u_1(x_1),...,u_n(x_n))$, for all $x \in X$, where $u_i : X_i \to [0,1]$ is a utility function (also called value functions) and $F : [0,1]^n \to [0,1]$ is called the aggregation function where $F(0,...,0) = 0$, $F(1,...,1) = 1$, and $F(x_1,...,x_n) \leq F(y_1,...,y_n)$ if $x_i \leq y_i$, $\forall i$. Moreover, the utility function normalizes the metrics and provides a measure of satisfaction for a single metric. The aggregation function takes a normalized score as input and returns an aggregate score. We recommend using the MACBETH approach Bana e Costa and Vansnick (1994); Bana e Costa and Oliveira (2012) to develop utility functions for resolving past difficulties related to interval scale construction and compatibility issues in a way that fully satisfies stakeholders. and mathematically significant.

The most widely used aggregation function is the weighted sum. It assumes the independence among the criteria. This is a major limitation as criteria often interact. We need to use other type of aggregation function such as the Choquet integral Choquet (1954); Sun et al. (2018), which is an extension of the weighted sum that is capable of measuring the influence of the importance of the individual criteria and the importance of the interrelationships among criteria.

Note that KPIs and metrics are often used interchangeably. However, there's a difference between the two. KPIs are numbers connecting performance to progress. Metrics are the building blocks of KPIs. They connect actions to performance.

B.5. Formalization as a knowledge graph

The different key information introduced above can be represented as a graph formalizing the different kinds of information and their interconnections. For example, the sections above included attributes, definitions, metrics, sources or engineering items. An attribute can have one or more definitions, and one or more metrics. Each definition and each metric has a source. Furthermore, some attributes or metrics can characterize a specific engineering item (data, process, ODD...).

In a graph, nodes are connected with edges. Graphs can be built with triples where a subject (a first node) is associated to an object (a second node) by a predicate (an edge). This notion of triples is the basis of semantic web languages like RDF and OWL (Allemang and Hendler (2011)). Semantic languages include the notion of classes and instances. For example, "at-

tribute" can be formalized as a class, and "robustness" as an instance of the class "attribute".

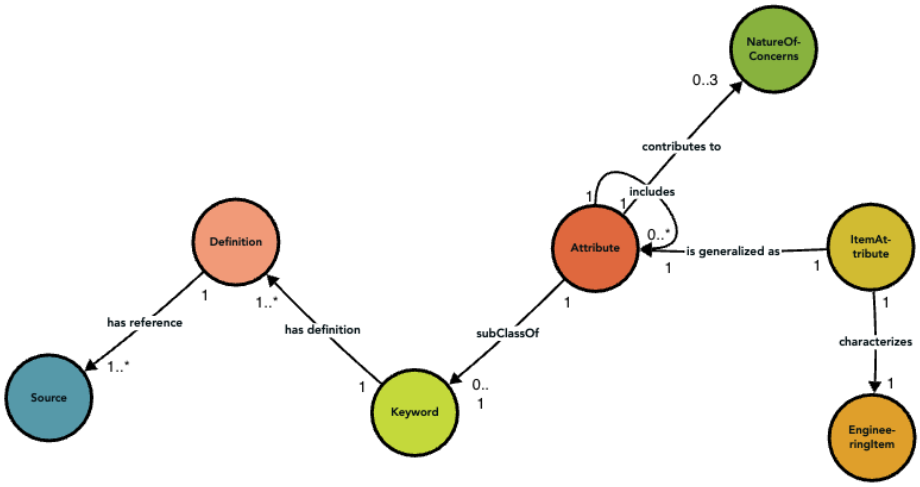


Figure B.6: Example of classes for a graph on AI Trustworthiness

Fig. B.6 is a graphical representation of classes generated by the tool Stardog Union (2020). These classes are not complete as they could also typically include a class for metrics.

A graph permits to centralize information and make it accessible by any tool. Thus, a graph represents the basis of knowledge-based AI. Information formalized in RDF or OWL can be used through requests made in languages such as SPARQL. It is for example possible to select the whole tree-like structure of attributes, or to more specifically select the definitions of a given attribute and the related sources.

RDF and OWL can be serialized in different file formats. For example, Turtle is more human-readable, RDF/XML relies on the common XML standard, and HDT is a binary format which permits to compress data. There are also other formats, such as N-Triples or JSON-LD.

These files can be managed in databases, called triplestores, which handle SPARQL requests. The graph database then allows the information to be processed and presented in diverse ways, tailored to the user's needs.

For example, in an AI engineering tool, attributes could be extracted from the graph and be displayed as a interactive tree to select appropriate system requirements. As the attribute definitions are available in the graph, they can for example be shown when hovering the cursor over an attribute. Trees can be represented in various languages (Python, Javascript...), in various forms (nodes and arrows, sunburst...) with various libraries (Plotly, D3js...). Fig. B.7 represents

(a) (b) (c)

Figure B.7: Different ways to show the data in the graph

three easy-to-achieve displays of the data contained in the graph. These displays were prepared during a workshop on the Confiance.ai graph.

B.6. Assessment theory

B.6.1 What to assess

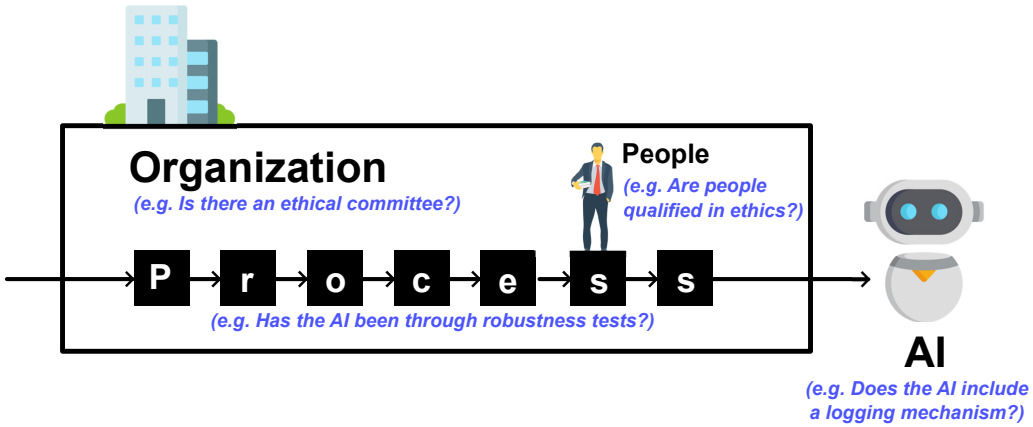


Figure B.8: Illustration of assessments associated to an AI system

In the context of the development of an AI system, an assessment can focus on:

1. Numerical performance: A numerical value represents how-good the AI system (or engineering item) is or how-well it behaves. The value can be measured with an instrument and provided with a given unit (e.g. the system reached the target 2cm from its center) or represent an objective observation (e.g. the system reaches the target 50% of the time). The value can potentially be associated to a requirement (e.g. the system should reach the target less than 5cm from its center). The measure can be at a particular moment in time, over a period of time, or across a population of systems of the same kind. Also metrics can be standardized, the choice of a metric is generally context dependent.

2. Functionality: Whether the AI system possesses certain functionalities, such as a logging mechanism to record actions or a stop button to enable emergency shutdowns. A functionality is often less context-dependent than a numerical performance.

3. Process: Whether the processes followed to develop (or support) the AI system respected some constraints. For example, whether the development processes included thorough robustness tests, or whether the specification of a robustness objective was documented. While the object of assessment was the AI system itself in the two previous bullet points ("numerical performance" and "functionality"), the object of assessment is in this case the organizational enabling system (i.e. the organization or company developing the system). Processes are often much less context-dependent than numerical performances, but also functionalities. AI Trustworthiness frameworks often focus on processes, both for in terms of quality (i.e. to aim for the best) and risk (i.e. to avoid the worst).

4. Organization: Beyond development processes, whether the organization responsible of an AI system (e.g. during its deployment, or during its operation) respects some constraints. For example, whether the organization implements robust security measures, maintains an ethics

committee, or engages with associations to enhance the handling of customer feedback.

5. Skill: Whether the people involved in the organizations responsible of an AI system respect some constraints. For example, the development of an AI system requires skills in ethics.

B.6.2 How to assess

It is important to distinguish **what** to assess, and **how** to assess it. If the process must include thorough robustness tests (what to assess), there are different ways to assess whether this is true: 1) checking that the development processes of the organization include robustness tests; 2) checking the results of the robustness tests; 3) checking an ad-hoc summary of the robustness-tests; 4) having an auditor on-site during robustness tests, etc.

The field of "conformity assessment" specifically focuses on **how** to assess whether a requirement is satisfied. In standardization, requirements and conformity assessments are in separate documents. Conformity assessment is often associated to the notion of "evidence" or "observable".

B.6.3 Multi-level assessments

G2 - Clarity of concept and operation The aims, desired outcomes, and methodological approach of the system should be clearly specified to generate a reference and highlight implicit assumptions.	The organization shall: a) Explicitly specify the purpose and application domain of the AIS system b) Define the intended user base of the AIS system c) Define an accepted performance threshold on nominal tasks d) Do due diligence in identifying potential systemic biases (positive and negative) during nominal system operation	N	HI	D, I, O, M	The organization shall provide in clear and concise terms: a) Documentation detailing the intended purpose and application domain of the AIS system. b) Documents and diagrams detailing the general methodology and pipeline followed by the system. c) Documents, test results, and audit reports supporting the acceptability and attainment of the performance threshold. d) Documents identifying potential systemic biases, where they occurred in the system, and what could help mitigate them.	Multilevel measurement on 1-5 scale: 5- Excels baseline requirements 4- Sustains baseline requirements 3- Meets baseline requirements (typical pass mark) 2- Needs improvement 1- Does not meet requirements

Figure B.9: Multilevel assessment in the IEEE labelling scheme

The assessment of the existence of a functionality in an AI system (bullet 2 above) can be considered, at first, as Boolean. For example, the AI system either includes a logging mechanism or it does not include it. The assessment of the requirements defined for the processes (bullet 3), the organizations (bullet 4) and the skills (bullet 5) can also generally be considered, at first, as Boolean. Such assessment results in a binary value (0 or 1) contrary to numerical performances (bullet 1).

However, there are two popular ways to define multi-level metrics based on this kind of assessment. The first way is to use generic levels which offer some nuance. For example, the IEEE certification scheme IEEE (2022) evaluates whether an AI system "Does not meet requirements", "Needs improvement", "Meets baseline requirements", "Sustains baseline requirements", or "Exceeds baseline requirements".

The second way is to explicitly define, for any requirement, different levels of satisfaction with reined sub-requirements. For example, the VDE labeling scheme VDE (2022) defines five levels of satisfaction for the requirement to have a clearly defined and documented Operational Design

R1.1		A	B	C	D	E
Is the operational design domain of the AI system / application clearly defined and documented?	*The Operational Design Domain (ODD) describes the conditions and environment an AI system/application is intended to operate within, and reasonably expected to encounter. This ODD should be described accurately and in enough detail such that the environment and boundaries of operation are clear. The user and stakeholders should be able to easily deduce from this whether the planned/intended use of an AI system is within the scope of the ODD.*	A **onthologically complete, structured and detailed** description of the: ■ operational design domain ■ and the intended use cases These are published and well understood by: ■ the users of the AI system ■ auditors ■ regulatory bodies ■ all additional stakeholders	A **onthologically complete, structured and detailed** description of the: ■ operational design domain ■ and the intended use cases These are published and well understood by: ■ the users of the AI system ■ auditors ■ regulatory bodies	A description of the: ■ operational design domain ■ and the intended use cases These are published and well understood by: ■ the users of the AI system, ■ auditors ■ regulatory bodies	A description of the: ■ operational design domain ■ and the intended use cases These are published and well understood by: ■ the users of the AI system.	A description of the: ■ the intended use cases These are published and well understood by: ■ the users of the AI system

Figure B.10: Multilevel assessment in the VDE labelling scheme

Domain (ODD). At the lowest level, the evidence is to deliver relatively simple : it is a description of intended use cases which should be clearly understood the users. At the highest level, the evidence is to deliver is much more complex : it is an ontologically complete, structured and detailed description of the intended uses cases and ODD, and it must be clearly understood by the users, the auditors, the regulatory bodies and the other stakeholders.

The second way is less subjective but it raises different questions such as: 1) Are the different levels always equally spaced? (for a given requirement, but also across the AI trustworthiness framework); 2) Are the evidences cumulative? (do the evidences of a given level include the evidences of the lower levels?)

B.7. Trustworthiness characteristics

The success of AI/ML technology in the past few decades has largely benefited from the accuracy based performance measurements. By assessing task performance based on quantitative accuracy or loss, training AI models becomes tractable in the sense of optimization. Meanwhile, predictive accuracy is widely adopted to indicate the superiority of an AI product over others. However, with the recent widespread applications of AI, the limitation of an accuracy-only measurement has been exposed to a number of new challenges (see. fig. B.3), ranging from robustness, reliability, dependability, safety and security, *etc*.

Trustworthy AI is composed of three pillars and six requirements (cf. fig; B.3: the legal, ethical, and technical pillars; and the following requirements: robustness; effectiveness; dependability including safety and security, usability, human agency including transparency, interpretability and explainability and human oversight including ethical issues. Moreover, trustworthy does not concern only the system itself, but also other actors and processes that take their part during the AI life cycle. This requires a holistic and systemic analysis of the characteristics defined through the ODD specification and the system risk analysis for contributing to the generation of trust in the user of an AI-based system.

Moreover, the characteristic relationship addresses all interactions between the stakeholders (engineers, operators, certification authorities, insurance companies...) and the system. Thus, it is important to understand the big picture of different aspects of AI trustworthiness, in addition to taking a holistic view of the trustworthiness of AI systems across all stages of their lifecycle. From the engineering and design phase, engineers must be able to build trust against the System they will deliver to operators. Indeed also, obviously, operators must be confident in the System features they will use. Trustworthiness relationships depend also on the autonomy objectives to be fulfilled vs. the environmental and human conditions in which they can operate trustfully. Consequently, trustworthiness relationships have to be analyzed according to each stakeholder's viewpoint involved in the autonomy objectives, and to be defined and refined so that they can be supported by the System. Rather than pursuing AI trustworthiness by setting requirements for each specific aspect, we draw attention to their combination and interaction, which are important and under-explored issues for trustworthy real-world AI systems. For these reasons, trustworthiness relationships must be established and maintained at each phase of the System lifecycle, from engineering and design, until operation in a target environment. Indeed, we tackle the entire lifecycle of the design and deployment of a AI-based systems to provide the practitioners and stakeholders an operational guidebook for how to assess trustworthy.

Moreover, the importance of data quality for the quality of AI components is emphasized by Felderer and Ramler (2021). Such data quality characteristics can be divided into inherent and system-dependent data characteristics, according to ISO/IEC 25012 (2008). Inherent data quality refers to the data itself, in particular to the values of the data domain and any constraints, the relationships between the data values and the metadata. The system-dependent data quality refers to the degree to which the quality of the data is achieved and maintained within a system when the data is used under a given set of conditions. For system/software product quality, the classical software quality characteristics based on ISO/IEC 25010 (2011b) are applicable: effective, efficient, satisfying, free from risk and contextual, for usability quality, and functional, performance, efficient, compatible, usable, reliable, secure, maintainable and portable for system/software product quality. Any activity aimed at detecting differences between existing and required behavior of AI components or AI-based systems is considered to be testing of AI components or AI-based systems.

Definitions (what does the trustworthiness characteristics stand for?) adopted in Confiance.ai, motivations (why is the requirement relevant for trustworthiness?) and a short glimpse at assessment methods (how can we assess the level of satisfaction of the requirement?) will be given for each of these trustworthy characteristics in its respective section. For easy identification, these definitions are also highlighted. They are based on states of the art, on international standards and also on specific Confiance.ai studies such as on confidence scores or on data quality metrics. These various characteristics are described in terms of their artefact type dimension: dataset, ML model and AI-based system. Finally, AI-based characteristics assessment approach described in this document is used to perform and structure the analysis of characteristics relationships.

The following macro-trustworthy AI characteristics will be considered in the following chapters:

- **Robustness** the system's outcome sensitivity to a change in the input - Chapter C;
- **Effectiveness** is a measure of its ability to perform the functions necessary to achieve goals or objectives - Chapter D;
- **Dependability** defined as the ability of a system to deliver a service that can be justifiably trusted - Chapter E;

- **Usability** describes the degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use - Chapter F
- **Human Agency and Oversight** means that AI systems shall be developed and used as a tool that serves people, respects human dignity and personal autonomy, and being under human control and oversight - Chapter G.

All these characteristics (see fig. B.3) must be mapped onto the AI processes and life cycle, keeping track of their relations with the different stakeholders. Indeed, the definition of trustworthiness is up to individual interpretation and preference. For instance, to feel confident in the handling of safety-critical data, end-users can be concerned with usability.

The quantification of AI-based system trustworthiness Braunschweig et al. (2022) has become a hot topic. Usually, it is determined by the quantification of elementary scores (*e.g.*, for reliability: Fleiss Kappa score, goodness-of-fit tests, or for accuracy: precision, recall, F-score, *etc.*). From a strict metrological point of view, trustworthiness may not be measured as it is not a physical property that can be compared to a reference quantity of the same kind. Trustworthiness does not have units. However, more generally speaking, "to measure" refers to assign an element of a scale (*e.g.*, number scale, nominal scale, ordinals scale) to an object in order to quantify an attribute of this object Adam et al. (2022). In our context,

Definition 13 (trustworthiness metric) *A trustworthiness metric is defined as an objective, mathematical measure of the AI-based component/system that is sensitive to differences in safety-critical characteristics. It provides a quantitative measure of an attribute which the body of solution exhibits.*

A metric is a "*system or standard of measurement*" represented in units that can be utilized to describe more than one attribute. Measurements can be used to assist in estimating quality. One should note that a metric can also be computed based on qualitative/subjective observations (i.e. the opinions of human experts). However, the assessment should then follow strict rules for ensuring that the final result is obtained through appropriate scientific methodology (reaching consensus, uncertainty assessment, etc.).

In this document, for some properties we have gone so far as to give formulas for measuring them, while for others we have only provided definitions to clarify the subject.

B.8. Core process

Trustworthiness is a core value at the systems engineering, data, and machine learning levels, each one with its own meaning of trustworthiness. The next sections of this document detail the trustworthiness characteristics with their specificities. This section explains the core process that uses those characteristics; it is generic to those three levels.

The systems engineering addresses the end-to-end process. It starts with the collect of requirements, expressed by the stakeholders, from which trustworthiness characteristics and associated KPIs (Key Performance Indicators) are extracted. The system characteristics are refined at the component level, that is dedicated here to data and machine learning. In the lines below, for a genericity reason, system, data and machine learning are named *item*.

The core process is based on a **methodological pattern** composed of the following activities:

1. *Specify engineering item.* In the case of component (i.e., data and machine learning), the

system requirements are refined. For instance, machine learning requirements are isolated or deduced from a long list of system requirements. A specification document (e.g., data specification document) is produced from the system and refined requirements.

2. *Characterize trustworthiness evaluation*. This activity consists in exhaustively structuring and describing all the characteristics from the requirements of the considered item (e.g. robustness, effectiveness for machine learning).
3. *Implement metrics*. Next, from this description of evaluation, metrics for each characteristic are implemented.
4. *Evaluate and report engineering item trustworthiness*. During development, or later at deployment time, the implemented metrics are applied on a specific item. For instance, the robustness characteristic evaluates the robustness of a ML model, or representativeness on a dataset verifies that all the dataset items are correctly represented. The result, available in a report, is an objective result from the implemented metrics. From this result, the context, and the past actions of quality improvement on the given item, decisions are made by comparison with the KPIs to reached. When KPIs are not met, an improvement feedback is given to the development, or even cascades up to the specification if necessary, in order to converge at the best to the KPIs which represent the quality target to reach.

The second activity mentioned above of **characterization of the trustworthiness evaluation** is itself broken down in several activities, according to the MCDA method introduced in Section B.4. Those are:

1. *Define trustworthiness characteristics*. All the characteristics of the considered item are identified and described (i.e., their name, properties).
2. *Structure attributes in a semantic tree*. The characteristics (i.e., quality attributes) are organized in a tree, from the most general down to the leaf characteristics.
3. *Identify numerical evaluations*. Each characteristic is typed by a numerical value domain.
4. *Adapt attribute for commensurability*. The characteristics can follow different forms of distribution, with different value domains. The purpose is to make them compatible in order to compare and operate them together.
5. *Define aggregation methodology*. MCDA enables to explore several solutions, compare them and to keep the best one. This step is optional here, when there is a need of trade-off, not between characteristics, but between sets of characteristic values (e.g., to explore and compare sets with different levels of KPIs, more or less restrictive).

To help its understanding, the proposed pattern has been illustrated on data. Refer to chapter "Evaluate Data Trustworthiness" in the methodological guideline of EC5N22 Confiance.ai EC5 (2023).

This core process is modeled using the Capella tool. It is depicted in the following figure.

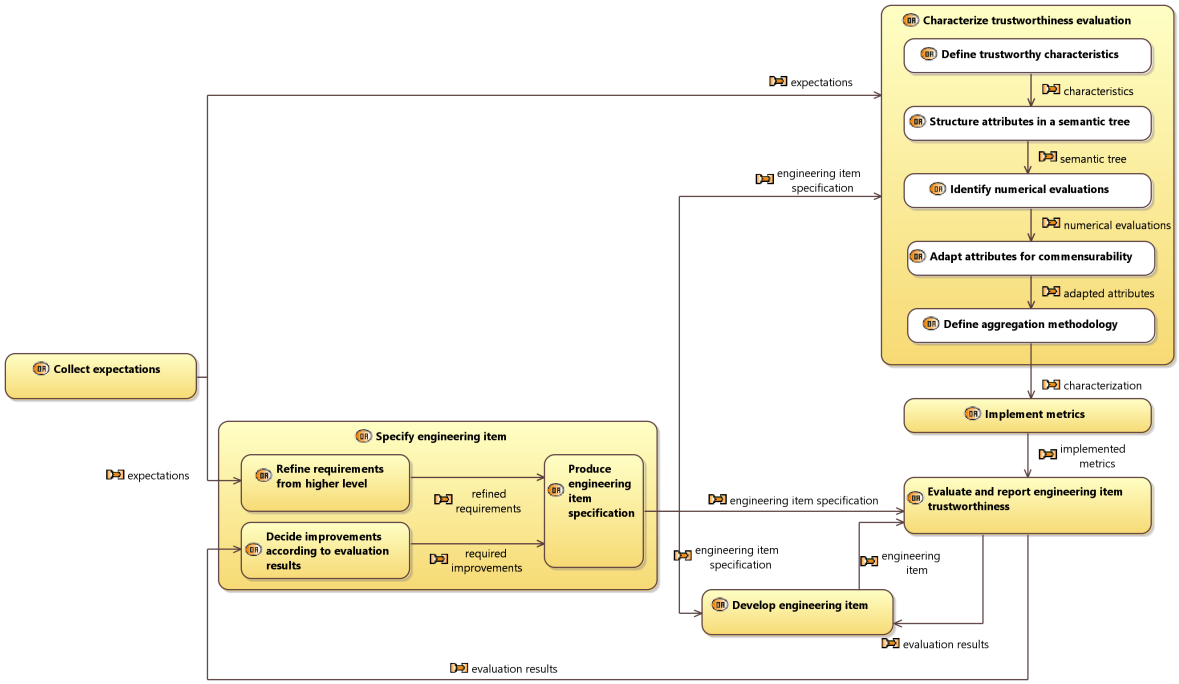


Figure B.11: Pattern for trustworthiness evaluation of any engineering item

C. Robustness

"*AI is starting to play a crucial role in systems for decision-making and autonomous processes, with potential consequences for our lives. A major concern comes from the various serious vulnerabilities affecting artificial intelligence techniques. These vulnerabilities could severely impact the robustness of current systems, leading them to uncontrolled behavior and allowing potential adversaries to trick the algorithms to their advantage.*" - Robustness and Explainability of Artificial Intelligence - Hamon et al. (2020)

C.1. Motivation

Robustness is one of those concepts that are relatively easy to understand via examples but very difficult to define. Robustness has been recognized as a desirable property in systems where the consequences were deemed unacceptable relative to the initiating damage. In the light of the recent advances in AI, the serious negative consequences of its use for EU citizens and organizations have led to multiple initiatives from the European Commission to set up the principles of a trustworthy and secure AI. Among the identified requirements, the concept of robustness Hamon et al. (2020) of AI systems has emerged as key elements for a future regulation of this technology.

The IEEE glossary of software engineering ANSI/ IEEE Std 729-1983 (1983) defines **robustness** as *the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions*. Moreover in ISO/IEC TR 24029-1 (2021), it is *the ability of an AI system to maintain its level of performance under any circumstances*.

These various definitions may be interpreted as the deliverance of correct service despite possibly adverse situations, and then classified into four types of "situation tolerance":

1. tolerance to any situation: deliverance of correct service in both adverse and nominal situations,
2. tolerance to adverse situations: deliverance of correct service in non-nominal situations,
3. tolerance to explicitly-specified adverse situations: deliverance of correct service in adverse situations mentioned in the system's specifications,
4. extra-tolerance, or tolerance to unexpected adverse situations: deliverance of correct service in adverse situations over and above those mentioned in the system's specifications.

Thus, the induced risks of the robustness characteristic arise when **input data is corrupted or manipulated, but for which accurate processing by the AI component is intended. Both qualitative and quantitative input data perturbations are considered, such as noise or adversarial examples**.

The robustness of a system relates to the question whether or not the system can be trusted to do well its intended purpose on the envisioned ODD. Mainly this characteristic ensures that the system will keep its performance properties (like accuracy and functional suitability). The ODD is key as defining the robustness of an AI system requires defining the AI system's function and the environment in which it will operate.

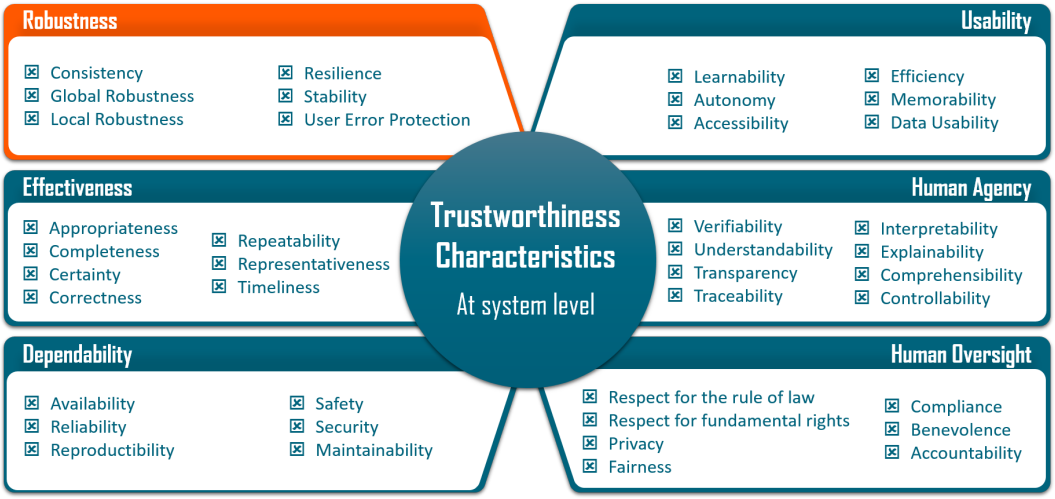


Figure C.1: The six critical characteristics of robustness at system level

The six critical characteristics of robustness at system level are:

- **Consistency** is the degree of uniformity, standardization and freedom from contradiction among the documents or parts of a system or component.
- **Local Robustness** is concerned with the response of the system *w.r.t.* deviations (*i.e.,* within a small "neighborhood") from a given input value.
- **Global Robustness** takes into account the combination of multiple "input" deviations with regards to the behavior of the system.
- **Resilience** is the ability to bounce back and withstand problems.
- **Stability** is the extent to which the system provides equivalent responses for similar inputs.
- **User Error Protection** is the capability of a system to protect users against making operation errors.

These critical characteristics induce specific ML model robustness attributes as:

- **ML model consistency** is defined as the ability to make consistent predictions across successive model generations for the same input.
- **Global Robustness** is defined as the stability of the model to perturbations in all its possible inputs within a certain neighborhood.
- **Local Robustness** is defined as the stability of the model to perturbations within a neighborhood of a selected (presumably relevant and representative) set of inputs.
- **Resilience** means that a model behavior is not easily manipulated through exploitation of vulnerabilities (either within the code or the training data).
- **Stability** means that the model performs well both generally and under stress conditions, such as in edge cases. It is neither overly sensitive to naturally occurring nor to intentional, targeted noise.
- **User Error Protection**:

Assessing the robustness of an AI system serves as an important means of avoiding vulnerabilities and controlling risks.

C.2. Consistency

A robust AI-based system is one that is also able to maintain the integrity and consistency of data, even in the face of unexpected events, failures or external stresses, over a long period of time. Moreover, consistency in software development entails uniformity, conformity, or a singular way of doing things.

Definition 14 (Consistency) *Consistency is the degree of uniformity, standardization and freedom from contradiction among the documents or parts of a system or component ISO/IEC 21827 (2008).*

Inconsistency bleeds your engineering organization in other ways, creating differences between teams and making it harder to move engineers between product opportunities. Ultimately, consistency benefits software developers by making it easier to develop clean code and improve robustness. Additionally, the final product's end users benefit by getting a product that is easy to use.

Consistency in software development is categorized into four main types:

- **Visual consistency** entails making all similar elements appear alike.
- **Functional consistency** entails related applications working similarly.
- **External consistency** entails related applications and software products looking and working similarly.
- **Internal consistency** in software development entails this key factor: introducing additional features that are consistent with the existing ones.

Visual consistency contributes to usability. Internal consistency ensures that different software development artifacts are used in different parts of an AI-based system, without causing contradictions between the different parts of the system.

C.2.1 ML model Consistency

Definition 15 (ML model consistency) *Consistency of a ML model is defined as the ability to make consistent predictions across successive model generations for the same input Wang et al. (2020).*

This definition is different from the replicability of model performance at an aggregate level - with a stable training pipeline of a classifier, aggregate metrics can be relatively consistent across successive generations, but changes to the training data or even retraining on the same data often causes changes in the individual predictions. Consistency is applicable to both correct and incorrect outputs, however, the more desirable case is producing consistently correct outputs for the same inputs. We define ability to make consistent correct predictions across successive model generations for the same input as correct-consistency.

Let us consider a safety scenario with two model generations $Model_i$ and $Model_j$, and an input X to further understand the effect of consistency and correct-consistency on user trust.

- If the outputs from both models are correct, then the issue of inconsistency does not arise and does not affect the user. This is a case of correct-consistency.
- If the output from $Model_i$ is incorrect while from $Model_j$ is correct, it won't adversely affect users' trust but in-fact can be considered as an improvement in the system.

- If the output from $Model_i$ is correct while from $Model_j$ is incorrect, it is a very severe case because this can adversely affect users trust in the system as well as its usability. In this case correct-consistency is desired.
- If the outputs from both models are incorrect, although it is an undesirable scenario and can affect users but it is still less severe from consistency point of view.

C.2.2 Data Consistency

Definition 16 (Data item consistency) *Data consistency is defined by the degree to which data has attributes that are free from contradiction in a specific context of use. ISO/IEC 25012 (2008).*

Thus **data consistency** is the degree to which several data items of a given dataset are coherent with each other in a specific context of use. For example, multiple instances of the same dataset should contain the same data which ensures the coherence of the data, and annotations of two annotators should be the same to avoid contradictions.

There are several steps you can take to verify the quality, reliability and integrity of your training data when ensuring data consistency in data-driven AI.

- **Data collection**: To avoid any bias or inconsistency inherent in a particular dataset, data should be collected from reliable and diverse sources.
- **Data preprocessing**: The cleaning and pre-processing of data to remove missing values, outliers or irrelevant information that could introduce inconsistencies.
- **Data validation**: To ensure the accuracy and integrity of the data collected, extensive data validation checks are performed. This can include cross-referencing with other sources, checking for duplicates and checking against pre-defined rules.
- **feature engineering**: Carefully engineer features to extract valuable information from the data. Make sure that the features chosen are relevant and reliable for the machine learning task.
- **Randomisation and shuffling**: To reduce any biases or patterns that may affect the learning algorithm, randomise and shuffle the data. This helps to create a more representative and unbiased training data set.
- **Continuous Model Evaluation**: Continuous evaluation of the machine learning model's performance on both training and validation data sets. This process is useful for the identification of any inconsistencies or anomalies that may arise during the training process.
- **Cross-validation**: Employ cross-validation techniques, such as k-fold cross-validation[1], to ensure the model's performance generalizes well across different subsets of the data. This helps in detecting overfitting and ensures consistency in performance.
- **Error analysis**: To identify and understand any inconsistencies or trends in the model's predictions, perform a thorough error analysis. This analysis can help to identify areas in which the model may be experiencing problems due to inconsistencies in the data.
- **Monitoring and updating**: Monitor the performance of the model in real-world scenarios on a regular basis and update the training data set accordingly. New and relevant data can be added in order to maintain consistency and adapt to changes in patterns.

[1] K-fold cross-validation is a technique for evaluating predictive models. The dataset is divided into k subsets or folds. The model is trained and evaluated k times, using a different fold as the validation set each time. Performance metrics from each fold are averaged to estimate the model's generalization performance

- **Documentation and version control**: Maintain detailed documentation of the data sources, pre-processing steps, and model versions that are used in machine learning projects. Version control can help track and manage changes to data and models over time. This ensures consistency and reproducibility.

C.2.3 Data Consistency for supervised ML

More specifically, in supervised ML, the quality of training data relies on consistency, which is the degree at which the labeler's annotations (human or machine) agree with one another. Accuracy and consistency reinforce each other: accuracy isn't a useful measure of quality if the labels aren't applied consistently, and consistency isn't a useful measure of quality if the labels aren't accurate. Together, they make up a gold standard for data quality.

In general, the consistency of a dataset for binary or multi-class classifiers is measured through a consensus value that may be trivially calculated by dividing the sum of agreeing on labels by the total number of labels per asset.

Two kinds of consistency are important for model training: consistency over time, measured by intra-annotator consistency, and consistency between annotators, measured by inter-annotator agreement.

- **Intra-Annotator Consistency** (IAC) measures the annotator's reliability over time. To track IAC, periodically send the same example(s) to the same annotator(s) and measure the rate at which the new labels agree with the earlier labels for the same task.
- **Inter-Annotator Agreement** (IAA) measures consensus between labelers. To track IAA, have two or more annotators label the same example and measure the rate at which they agree.

C.3. Global and Local Robustness

In the present context, the robustness of an AI-enabled system essentially depends and is focused on the ML or DL (Deep Learning) components and the phase in the development cycle where the training model is designed and tested (*i.e.* the phase involving ML/DL algorithms, training and testing datasets). As it has been highlighted in Szegedy et al. (2013), ML/DL models exhibit counter-intuitive properties like (1) the misclassification of minor (adversarial) perturbations that are statistically indistinguishable from the ones in the training dataset and (2) the misclassification of data subsets representing a semantic unit/object in the presence of negligible perturbations w.r.t. the overall extent of the data subset. Overall, a definition allowing to test the robustness of a model M is the measure of the impact of the minimum adversarial perturbation across many samples x.

C.3.1 Inherent AI Robustness

Let S be an AI system. Let $E(S)$ be the correctness of S. Let $\delta(S)$ be the AI system with perturbations on any machine learning components such as the data, the learning program, or the framework making E be *less correct*. The robustness of S is a measurement of how distant $E(\delta(S))$ is from $E(S)$: the larger the distance, the less robust S is with respect to δ.

C.3.2 Local and Global Robustness

Input stability measures the amplitude of the change on a system resulting from perturbing its inputs. Alternatively, it informs on how likely the system's behavior is going to change when perturbing its inputs. Consider a network $F = (F_1, \ldots, F_m) : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ that categorizes inputs into m different classes, denote $f := \text{argmax}_i\{F_i\}$ the function representing the predictions of F, and denote d a choice of distance between inputs. For $\delta > 0$, we will say that f is

1. δ-locally stable at x w.r.t. d iff

$$\forall x' : d(x,x') \leqslant \delta \Longrightarrow f(x) = f(x'). \tag{C.1}$$

   Most work on robustness verification has focused on this local robustness property; but the global robustnessvcaptures the operational properties of on-line local robustness certification. Clearly, local robustness cannot be simultaneously satisfied at every point—unless the model is entirely degenerate, there will always exist points that are arbitrarily close to a decision boundary.

2. δ-globally stable over D w.r.t. d iff

$$\forall x, x' \in D : d(x,x') \leqslant \delta \Longrightarrow f(x') \longrightarrow f(x). \tag{C.2}$$

   In particular, if the output space has a well-defined notion of distance d_2, this can be further specialized into

$$\forall x, x' \in D : d(x,x') \leqslant \delta \Longrightarrow d_2(f(x'), f(x)) < \varepsilon.$$

In practice, stability training involves formulating a loss that aims to flatten the model output in a small neighborhood of any input data, forcing the output to be similar between the original and perturbed copy.

Remark 1 *In the literature, one usually sees the term "local robustness" for the above properties and will interpret robustness as a synonym for stability in this context.*

Remark 2 *Notice that the prediction $f(x)$ might not correspond to the ground-truth value y of x. As such, stability here is regarded as an accuracy-agnostic property. This might seem confusing since the notion of robustness we defined (as the difference $E(\delta(S)) - E(S)$) should be interpreted as the measuring of how robust are true predictions of the model i.e. how robust is the model when functioning according to plan. Indeed, we could have defined δ-local robustness for a model f at a couple (x,y), where y is the ground-truth value of x and where $f(x) = y$, via the property*

$$\forall x' : d(x,x') \leqslant \delta \Longrightarrow f(x) = y = f(x'). \tag{C.3}$$

We choose not to distinguish stability and robustness in this way to isolate the property here to be verified to that of the model's accuracy. Indeed, one could obtain (C.3) *as the formal intersection of property* (C.1) *with the generic property* $[f(x) = y]$.

C.3.3 Adversarial Robustness evaluation & verification

To invalidate property (C.1) formally amounts to validate the following:

$$\exists x' : d(x,x') \leqslant \delta \quad \text{and} \quad f(x) \neq f(x'). \tag{C.4}$$

As such, the minimal distance δ^* such that (C.4) holds is exactly given as the distance between $f(x)$ and the decision boundary of f. Computing the value δ^* has been shown to be NP-complete even for a simple ReLU network. Therefore, in practice one can only propose efficient upper and lower bounds of the value δ^*

$$\underline{\delta} \leqslant \delta^* \leqslant \bar{\delta}.$$

Computing $\underline{\delta}$ and $\bar{\delta}$ do not answer to the same problems: while finding $\underline{\delta}$ implies finding a neighborhood of x where we can *formally* verify/demonstrate property (C.1), finding $\bar{\delta}$ implies finding a single sample - which we will call adversarial attack - representing the *empirical* evaluation of our model's (lack of) robustness.

C.4. Empirical Adversarial Robustness

C.4.1 Evaluation

Validating property (C.4) means crafting adversarial examples. This is usually formalized as a problem of finding

$$\underset{x' \in B(x)}{\operatorname{argmax}}\ g(x') \tag{C.5}$$

where $g(x')$ measures the successfulness of the attack, $B(x)$ is a chosen neighborhood of the reference input x indicating the feasible perturbation range. Creating such a counter-example does not directly give information on overall behavior of the model. Evaluating adversarial robustness empirically means quantifying this behavior. The usual way of producing such *empirical local characterization* is to have an algorithmic way of translating a clean input into an adversarial one so that one can produce an adversarial copy of a test set, and then measuring the model's performance drop when tested on such adversarial test set.

C.4.2 Norm-based white-box threat models

Defining g can be done in multiple ways depending on the amount of available information on the model and the intention for crafting such attacks. The point of view which we take here is that of a ML model engineer whose aim is to provide a *reliable* worst-case performance evaluation of a system, thus having access to any information likely to improve the attack's strength. Under these circumstances, many of today's best available attacks express g in terms of the model's loss function and, as a consequence, one can use the latter to modify the learning objective of our model hoping to empirically mitigate its brittleness against such attacks.

Let $g(x) = \mathscr{L}_f(\theta, x)$ be the loss function used to train the model f with learning parameters $\theta \in \Phi$ over training inputs $x \in \mathscr{X}$. While the learning procedure of f involves the computation of the gradient $\nabla_\theta \mathscr{L}_f$, the first efficient method to craft adversarial examples involves computing the positional gradient $\nabla_x \mathscr{L}_f$. The idea is to take a clean input x and formally translate it in the input space by adding to it a noise vector ε in the direction of $\nabla_x \mathscr{L}_f$ whose magnitude is small enough to satisfy the condition $d(x, x+\varepsilon) \leqslant \delta$ but big enough so that $f(x) \neq f(x+\varepsilon)$. Concretely, let us constrain our perturbations to be inside the ℓ_∞-ball

$$B_\infty(x, \delta) := \{x' : \|x' - x\|_\infty \leqslant \delta\},$$

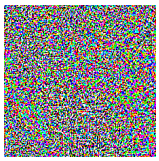
and hypothesize that we are modeling translation attacks of the form $x \mapsto x' := x + \varepsilon$. Then one can conduct the first-order *positional* Taylor expansion of $\mathscr{L}_f(\theta, x')$ at x:

$$\mathscr{L}_f(\theta, x') = \mathscr{L}_f(\theta, x) + \nabla_x \mathscr{L}_f(\theta, x) \cdot \varepsilon + \mathscr{O}(\|\varepsilon\|_\infty^2).$$

Then one can produce a linear approximation of (C.5) by means of the following simplifications:

$$\begin{aligned} \underset{x' \in B_\infty(x,\delta)}{\operatorname{argmax}} \mathscr{L}_f(\theta, x') \approx \underset{x' \in B_\infty(x,\delta)}{\operatorname{argmax}} [\mathscr{L}_f(\theta, x) + \nabla_x \mathscr{L}_f(\theta, x) \cdot \varepsilon] &= \underset{x' \in B_\infty(x,\delta)}{\operatorname{argmax}} \nabla_x \mathscr{L}_f(\theta, x) \cdot \varepsilon \\ &= x + \underset{\varepsilon \in B_\infty(0,\delta)}{\operatorname{argmax}} \nabla_x \mathscr{L}_f(\theta, x) \cdot \varepsilon \\ &= x + \operatorname{sign}(\nabla_x \mathscr{L}_f(\theta, x)) \cdot \delta \end{aligned}$$

where $\operatorname{sign}(\cdot) \in \{\pm 1\}$ denotes element-wise sign values. The above method is known as the fast gradient sign method (FGSM) and the crafted attack $x + \operatorname{sign}(\nabla_x \mathscr{L}_f(\theta, x)) \cdot \delta$ is the one-step steepest ascent update under the ℓ_∞ norm. The archetypal example of such an attack is [Goodfellow-et-al]:

Visualization		$+0.007\times$		$=$ 
Formalization	x		$\operatorname{sign}(\nabla_x \mathscr{L}_f(\theta, x))$	$x + \delta \operatorname{sign}(\nabla_x \mathscr{L}_f(\theta, x))$
Prediction	"panda"		"nematode"	"gibbon"
Confidence	57.7%		8.2%	99.3%

One can readily improve and generalize this procedure to the first-order multi-step case, in which one will choose a smaller radius $\alpha < \varepsilon$ and a number of steps T and decompose the translation $x \mapsto x + \varepsilon$ into T intermediate translations by iteratively updating the direction of the noise to the steepest ascent w.r.t. α. As we are conducting a search within $B_\infty(x, \delta) = [-\delta, \delta]^n$, we project back to this regions any iteration that may fall out of it. This can be formalized as an iteration rule

$$x_0 = x, \qquad x_{t+1} = \Pi_{[-\delta,\delta]^n}[x_t + \alpha \operatorname{sign}(\nabla_x \mathscr{L}_f(\theta, x_t))], \quad t = 0, \ldots, T-1.$$

This method of producing adversarial attacks is called iterative FGSM or (basic) PGD method and are among the most effective methods both for attacking and defending a model.

C.4.3 Robust Learning

A striking fact encountered when evaluating adversarial robustness on practically all ML models is that they are extremely brittle to them so a natural response from the ML community is to design methods that empirically improve the adversarial characterization of their models. The idea will be to leverage the maximization problem

$$\underset{\varepsilon \in \Delta(x)}{\operatorname{argmax}} \mathscr{L}_f(\theta, x + \varepsilon),$$

where the perturbation region $\Delta(x)$ may depend on x as seen in the PGD method, for which we presented practical methods to find sub-optimal solutions, to craft new learning objectives as key

components of what is widely known as adversarial training. The most basic example is given in the context of supervised learning, by considering the traditional risk

$$\mathbb{E}_{(x,y)\sim\mathscr{D}}[\mathscr{L}_f(\theta,x,y)]$$

but, instead of suffering the loss on each couple $(f_\theta(x),y)$ point-wise, we suffer the worst-case loss over the region $B(x)$. This incurs into a min-max optimization objective

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathscr{D}}\left[\max_{\varepsilon\in\Delta(x)} \mathscr{L}_f(\theta,x+\varepsilon,y)\right]. \tag{C.6}$$

The reason why one can approach this via gradient descent

$$\theta \leftarrow \theta - \nabla_\theta\left[\max_{\varepsilon\in\Delta(x)} \mathscr{L}_f(\theta,x+\varepsilon,y)\right]$$

is that the optimal argument ε^* of the inner optimization problem can be treated as independent of θ when taking the gradient ∇_θ. Indeed, Danskin's Theorem tells us that

$$\nabla_\theta\left[\max_{\varepsilon\in\Delta(x)} \mathscr{L}_f(\theta,x+\varepsilon,y)\right] = \nabla_\theta[\mathscr{L}_f(\theta,x+\varepsilon^*,y)], \quad \text{where } \varepsilon^* = \operatorname*{argmax}_{\varepsilon\in\Delta(x)} \mathscr{L}_f(\theta,x+\varepsilon).$$

One can then use the same techniques we used to attack the model to approach the inner optimization problem and use classic descent techniques for the outer one:

$$\begin{aligned}
\text{Standard ERM} &\longrightarrow \text{Robust ERM}\\
\text{single optimization} &\longrightarrow \text{double optimization}\\
\text{objective function} &\longrightarrow \text{Robust objective function}\\
\min_\theta \mathbb{E}_{(x,y)}[\text{Loss}(f_\theta(x),y)] &\longrightarrow \underset{\text{SGD}}{\min_\theta \mathbb{E}_{(x,y)}}\left[\underset{\text{PGD}}{\max_{\delta\in\Delta}} \text{Loss}(f_\theta(x+\delta),y)\right]
\end{aligned}$$

On the one hand, this results into a framework that creates an interplay between the challenge of finding adversarial examples (using FSGM, PGD or any other method) and the process of training a robust model and has shown to produce the most empirically robust models to this day.

On the other hand, there are several practical downsides to this. First, reaching the theoretical optimum value δ^* in the inner problem, which is needed to apply Danskin's Theorem, has proven to be out of reach so adversarial training heavily depends on how good the inner approximation really is. This has made possible to create attacks that by-pass adversarial training precisely by creating attacks that out-optimize the above inner approximation. Second, the learning curve of adversarial training shows to be more unstable and need further techniques to mitigate this. Third, it has been widely shown that training such models is much more computationally expensive and creates an irreducible trade-off between accuracy and robustness. It is interesting to see that this trade-off was first detected when conducting empirical robustness characterization and was later proven theoretically.

Remark 3 *It might be tempting to think that finding the optimal value δ^* would resolve most of the problems underlying ML safety. After all, the value $\underline{\delta}$ ensures the behavior of the model regardless of the kind of threat that the model might be subject to and the value*

$\bar{\delta}$ tells you that it is possible to craft a sufficiently good adversarial attack method to fool the model. Although the above robustness attributes are necessary to ensure a trustworthy AI, they hardly cover real-world scenarios that were the original concerns. Now, solving this seemingly "toy problem" has shown to be out of reach with the current approaches, creating an implicit bias about how much we can expect from it - if any. Adversarial robustness, from it's very inception, was always meant to be tractable first stepping stones towards a wider notion of robustness that includes scenarios such as distribution shifts, common corruptions and even ordinary generalization. Within this larger scope, where the expectation is that the AI system will perform well in the scope of its ODD, adversarial robustness serves as worst-case in-distribution analysis for model's robustness, consistently informing on if a model's general robustness properties are being overestimated or underestimated.

C.5. Formal Adversarial Robustness

Neural network verification may be possible not only for robustness but any other safety specification that can be formalized as a locally affine function. Indeed, formal verification techniques and tools can theoretically be used as soon as a formal specification has been established. Two problems usually arise for this discipline. The first is that a desired or undesired behavior is sometimes hard to formally specify in a logical and mathematical language (*e.g.,* "avoid pedestrians" needs to formally define what a pedestrian is). The second is that, even if the theoretical methods are available, and the proof problem is decidable, there can be significant scaling problems, especially with large unstructured software such a modern neural network.

Formally proving adversarial robustness does not suffer from the first problem (previous sections have rigorously mathematically defined this notion, which is ample formal specification). The scaling problem, however, remains an hurdle today, despite a considerable amount of research on the matter. That being said, the current state of the art still makes formal verification a worthwhile stronger version of empirical adversarial robustness evaluation. In a nutshell, instead of selecting a set of inputs I, then generating (an obviously limited number of) adversarial examples I' within a given neighborhood (a distance ε) of these inputs, and using I' set to evaluate the model, formal verification allows to directly evaluate the entire neighborhood of the inputs in I within ε. It is still not covering all possible inputs (only I), but it is strictly more exhaustive than the generation of a limited number of adversarial inputs.

In both approaches, a special care should be taken in the selection of I, such that this set is as representative of the ODD as possible. Then, the two approaches should be evaluated in a cost-benefit analysis: the cost of the adversarial generation approach is obviously the optimization runs to attain an adversarial example plus the cost of running the network and checking the results, while the cost of formal verification will be the static analysis cost of propagating the neighborhood ε around each point in I.

C.6. Resilience

Resilience does not mean that problems will never occur. Basically, a system is resilient if it continues to carry out its mission in the face of adversity (i.e., if it provides required capabilities despite excessive stresses that can cause disruptions). Being resilient is important because no matter how well a system is engineered, reality will sooner or later conspire to disrupt the system.

Residual defects in the software or hardware will eventually cause the system to fail to correctly perform a required function or cause it to fail to meet one or more of its quality requirements (e.g., availability, capacity, interoperability, performance, reliability, robustness, safety, security, and usability). The lack or failure of a safeguard will enable an accident to occur. An unknown or uncorrected security vulnerability will enable an attacker to compromise the system. An external environmental condition (e.g., loss of electrical supply or excessive temperature) will disrupt service. Resilience is defined in Mamalet et al. (2021) as follow:

Definition 17 (Resilience) *Resilience is the ability for a system to continue to operate. while an error or a fault has occurred.*

The National Institute of Standards and Technology (NIST) explains that resilience is the ability of an information system to reduce the magnitude, impact and/or duration of disruptive events or, more generally, any known or unknown changes in the operating environment (including deliberate attacks, accidents and naturally occurring threats or incidents) by a) anticipating and preparing for such events (through e. g. risk management, contingency and continuity planning), b) being able to withstand and adapt to attacks, adverse conditions or other stress and potential disruptions, and continuing (or rapidly recovering the ability) to operate (even if in a degraded or debilitated state) while maintaining essential and required operational capabilities, and c) recovering full operational capabilities after such a disruption in a time frame consistent with mission needs. In consequence, “resilient AI” has links to questions of acceptance and trust, and consequently connections to fairness, accountability and transparency. "*No AI system can be considered resilient if its use is fraught with fundamental mistrust, responsibility vacuums, (nearly always justified) accusations of injustice or critique of its black-box-ness, to mention only the most prominent aspects*" Eigner et al. (2021).

ML systems will be expected to operate in contested and adversarial environments. Their resilience, or ability to adapt to risks, will be critical to each mission that they support.

C.7. Stability

As datasets get bigger and data-driven AI methods become more complex, the need for reproducibility has increased significantly. In fact, a minimal requirement is that one can reach the same conclusion by applying the described analyses to the same data, a notion some refer to as replicability. There are many ways to define and quantify the stability of a ML algorithm. The natural way of making such a definition is to start from the goal: we want to get bounds on the generalization error of specific learning algorithm and we want these bounds to be tight when the algorithm satisfies the stability criterion.

Definition 18 (Stability) *Stability is defined as the extent to which the system provides equivalent responses for similar inputs Mamalet et al. (2021).*

A more general requirement is that one can reach a similar result based on independently generated datasets. In the ML model, the randomness comes from the sampling of the training set. We will thus consider stability with respect to changes in the training set. Moreover, we need an easy to check criterion so that we will consider only restricted changes such as the removal or the replacement of one single example in the training set. The first such notion was used by Devroye and Wagner (1979) in order to get bounds on the variance of the error of local learning algorithms. Later, Kearns and Ron (1997) stated it as a definition and gave it a name. Bousquet and Elisseeff (2002) slightly modified the previous definition which suits our needs.

Definition 19 (Hypothesis Stability) *An algorithm A has* ***hypothesis stability*** β *with respect to the loss function* ℓ *if the following holds*

$$\forall i \in \{1, \cdots, m\}, \ \mathbb{E}_{S,z}\left(|\ell(A_S, z) - \ell(A_{S^{\setminus i}}, z)|\right) \leq \beta$$

Note that this is the L_1 norm with respect to the distribution D, so that we can rewrite the above as

$$\mathbb{E}_S[\| \ \ell(A_S, .) - \ell(A_{S^{\setminus i}}, .) \ \|_1] \leq \beta$$

Bousquet and Elisseeff (2002) also uses a variant of the above definition in which instead of measuring the average change, we measure the change at one of the training points.

Definition 20 (Pointwise Hypothesis Stability) *An algorithm A has pointwise hypothesis stability* β *with respect to the loss function* ℓ *if the following holds*

$$\forall i \in \{1, \cdots, m\}, \ \mathbb{E}_S\left(|\ell(A_S, z_i) - \ell(A_{S^{\setminus i}}, z_i)|\right) \leq \beta$$

C.8. User Error Protection

"*Error is human*", and errors happen when people engage with user interfaces. The term "**user error**" implies that it's the user's fault when they do something wrong. But in the vast majority of cases, the fault actually rests with the designer for having created an interface that is confusing or makes it too easy for the user to make a mistake.

Definition 21 (Use error) *Use error is an act or omission of an act that results in a different system response than intended by the manufacturer or expected by the user. ISO 9241-210 (2019)*

The solution to user errors is not to blame the user or try to train the mistakes out of them. The solution is to redesign the product in such a way that it prevents errors from occurring in the first place.

Definition 22 (User error protection) *User error protection is the degree to which a product or system protects users against making errors ISO/IEC 25010 (2011a).*

According to Reason (1990), there are two categories of user errors: slips and mistakes. Mistakes are errors in choosing an objective or specifying a method of achieving it whereas slips are errors in carrying out an intended method for reaching an objective "*The division occurs at the level of the intention: A Person establishes an intention to act. If the intention is not appropriate, this is a mistake. If the action is not what was intended, this is a slip.*"

Good design should help prevent mismatches between the user's expectations and the interface. Instead of ignoring these errors by blaming humans, a "root cause analysis" explore an error, continually investigating until to found a foundational cause for the error. But this method is often misused and the root cause is often found to be a person (still blaming humans!) instead of a process or product flaw.

Guardrails are safety measures that restrict users from doing things that could result in error messages. If the AI-based system is well designed the user shouldn't even be aware the restrictions exist.

C.9. Conclusion

Every AI system must have a fall-back plan in case a problem occurs. It must be ensured that the AI acts according to the proposed regulations towards its goal without harming any human being or the environment. The fall-back may include moving from a statistical approach to a rule based approach. The system may even take permission from the human operator before performing further tasks.

D. Effectiveness

In ML discipline, most of the research is focusing on model performance improvement more than on datasets Mazumder et al. (2022). In the recent decade, ML techniques have advanced significantly and achieved a high maturity level Adam et al. (2022).

Classical ML practices consist typically in using the existing datasets and in leveraging performances challenges through techniques complexity enhancement. In the other hand, data-driven AI takes a broader approach by placing a greater emphasis on the data itself Jakubik et al. (2022); Jarrahi et al. (2022). Instead of simply looking for patterns and relationships within the input features, data-driven AI involves collecting, processing, and analyzing large amounts of data to create more accurate and robust models Mattioli et al. (2022a).

Moreover, a real challenge today is to associate datasets to the ODD from the operational level of the system definition. Indeed, these datasets include several factors such as user needs Chapman et al. (2020) and related metadata. Moreover, Mountrakis and Xi (2013) highlights that dataset quality may have a more significant impact on performance than any model design choice. Many industrialization crisis often result from the data used to train the models instead of the model designs and architectures. Without a systematic assessment of their quality, data-driven AI risks losing control of the various steps of data engineering such as collection, annotation and feature engineering. Doing without data quality assessment would result in assuming that data engineering can not be further improved and that problems will always be detected without systematic analysis.

Thus, in a given end-to-end AI-based system process, the data quality assessment brings an evaluation of some ODD description aspects. These evaluation goes through a set of metrics, illustrated in Fig.A.3, such as data accuracy, data representativeness and data diversity. Furthermore, to ensure conformity to the ODD specifications, well-founded metrics assess the reached data quality level. Both research and industrial practices have developed relevant data quality metrics in the AI-based system, such as accuracy and completeness. However, many of them still lack a sound foundation Heinrich et al. (2018). Thereafter, a definition and a brief technical description of five metrics for data quality assessment are given.

D.1. Motivation

Let us begin with the difference between effectiveness and efficiency ISO 9000 (2015a):

- **Efficiency** is defined as the relationship between the result achieved and the resources used.
- **Effectiveness** relates to the extent to which planned activities are realized and planned results are achieved.

In line with this definition, effectiveness was defined by Clements (1991) as how well the process/product meets end users demands. It focuses on whether the goal was accomplished and whether the outcomes align with the objectives. It emphasizes the quality and the relevance of the outcomes. Even if a task or process is resource-consuming, if it results in achieving the desired goals, it can be considered effective. However, efficiency is about achieving those goals

in the most optimized and resource-saving manner. According to O; and M (2013), system effectiveness, ranging from 0 to 1 "accounts for the influence of the different performance criteria of the system". The corresponding effectiveness equation can be expressed as follows:

$$E = \prod_{i=1}^{n} A_i \tag{D.1}$$

where A_1, ..., A_n are the n performance indicators of the system determining/influencing its overall effectiveness. For instance, in the design of Pumps as turbines (PAT) (referring to a pump that can be used as a turbine to generate electricity), three indicators have been considered in evaluating E (n = 3): capability (ηp), flexibility (ϕp), and reliability (μp) These indicators represent, respectively, how the system:

- performs its intended production activity according to expectation;
- performs for a Hd variation around the design value;
- operates for a given time without failure.

Thus, equation (D.1) specifies:

$$E = A_1.A_2.A_3 = (\eta p).(\phi p).(\mu p) \tag{D.2}$$

Effectiveness can also be calculated as the percentage of users successfully achieving their goals vs. the total number of users: normally, as a result of coming through user scenarios, users either achieve their goals or fail to achieve them.

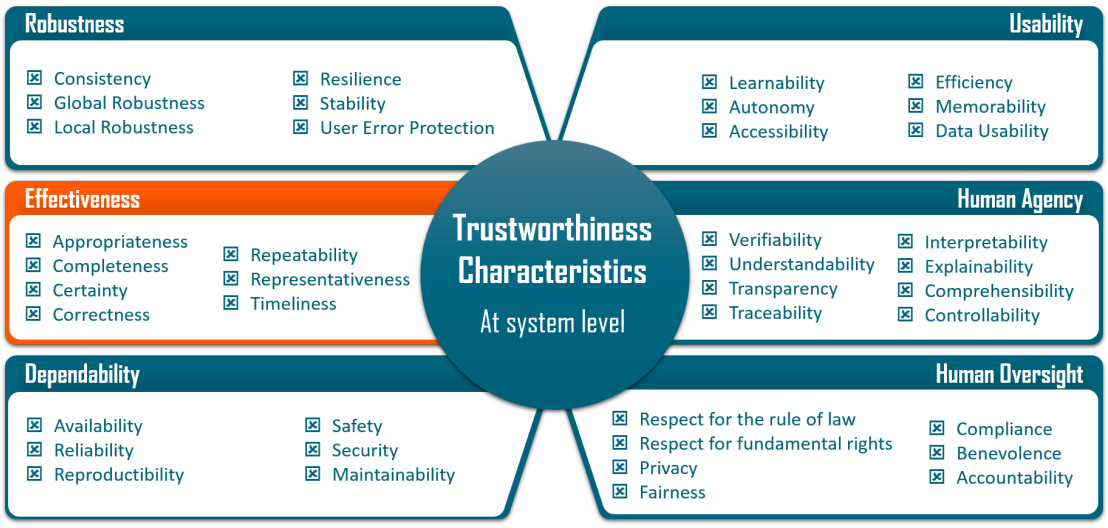


Figure D.1: Major characteristics of Effectiveness

D.2. ML model Effectiveness

The following list provides an overview of key performance metrics for different (also unsupervised) ML tasks.

- Regression

  - (Mean) squared error
  - (Mean) absolute error
- Classification
  - (Mean) accuracy
  - F1 score
  - Precision and recall
  - Sensitivity and specificity
  - AUC value
- Ranking
  - Mean reciprocal rank
  - Discounted cumulative gain
- Clustering
  - Silhouette value
  - Adjusted mutual information score
  - Completeness score
- Computer vision
  - Peak signal-to-noise ratio (SNR)
  - Adjusted mutual information score
  - Completeness score

D.3. Accuracy and Correctness

D.3.1 Motivation

In computer science, an algorithm is correct with respect to a specification if it behaves as specified. Best explored is functional correctness, which refers to the input-output behavior of the algorithm (i.e., for each input it produces an output satisfying the specification).

Thus correctness is freedom from error while accuracy is the state of being accurate; freedom from mistakes.

Definition 23 (Correctness) *Correctness is the degree to which a system or component is free from faults in its specification, design, and implementation ISO/IEC/IEEE 24765 (2017).*

Closely related is the accuracy problem, any software is expected to be deterministic and correct. Any deviation from the expected behavior and any difference in the output is considered a defect that is supposed to be fixed.

Definition 24 (Functional correctness) *Functional correctness is defined by as the degree to which a product or system provides the correct results with the needed degree of precision ISO/IEC 25010 (2011b).*

Thus, functional correctness measures what proportion of functions provides the correct results. An incorrect function is one that does not provide a reasonable and acceptable outcome to achieve the specific intended objective.

It is well known, that real-world software is not defect-free and there is no perfect system. However, the underlying principles and the level of correctness currently achieved in software engineering is different from what AI-based systems exhibit. AI-based systems, especially ML-based systems are accepted to be inherently "defective", as they usually operate in a defined accuracy range. Yet a system with 99% accuracy will "fail" in about one out of hundred runs. Applying conventional testing and quality assurance principles and approaches is incompatible with the underlying assumption that the system is considered correct although it exhibits a high number of contradicting ("failing") cases. The corresponding testing techniques and quality metrics developed for deterministic systems first need to be adapted before they can be used to assess systems with probabilistic behavior.

D.3.2 ML model Correctness

In machine learning, correctness measures the probability that the ML system under test "gets things right".

Definition 25 (ML model correctness) *Let D be the distribution of future unknown data. Let x be a data item belonging to D. Let h be the machine learning model that we are testing. $h(x)$ is the predicted label of x, $c(x)$ is the true label. Zhang et al. (2020) defined the ML model correctness $E(h)$ as the probability that $h(x)$ and $c(x)$ are identical.*

$$E(h) = Pr_{x\sim D}[h(x) = c(x)] \tag{D.3}$$

Achieving acceptable correctness is a basic requirement for an ML scheme. The real performance evaluation of an ML system should be on future data, as future data is often not available. Classic machine learning validation is the most well-established and widely-used technology for correctness testing. Typical machine learning validation approaches are cross-validation and bootstrap. The principle is to isolate test data via data sampling to check whether the trained model fits new cases. There are several approaches to perform cross-validation. In hold out cross-validation, the data are split into two parts: one part becomes the training data and the other part becomes test data.

Moreover, let $\mathscr{X} = (x_1, ..., x_m)$ be the set of unlabelled test data sampled from D. Let h be the machine learning model under test. Let $\mathscr{Y}' = (h(x_1), ..., h(x_m))$ be the set of predicted labels corresponding to each training item x_i. Let $\mathscr{Y} = (y1, ..., ym)$ be the true labels, where each $y_i \in \mathscr{Y}$ corresponds to the label of $x_i \in \mathscr{X}$. The **empirical correctness** Zhang et al. (2020) of model (denoted as $\hat{E}(h)$) is:

$$\hat{E}(h)) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{I}(h(x_i) = y_i) \tag{D.4}$$

where $\mathbb{I}$ is the indicator function; a predicate returns 1 if p is true, and returns 0 otherwise.

D.3.3 Data Correctness

The following definitions are the results of different Confiance.ai convergence workshops:

Definition 26 (Data item correctness) *Degree of conformity of a data item to a true or standard value (ground truth) in the context of use.*

Definition 27 (Dataset correctness) *Dataset correctness is the distributions characteristic of data items correctness.*

Thus, accuracy or precision could have a similar meaning but should be avoided to avoid confusion.

Dataset correctness could be defined as:

$$data\_correctness = \frac{1}{(1+d(\omega,\omega_m))} \tag{D.5}$$

where ω is the data value to be assessed, ω_m is the corresponding real value and d is a domain-specific distance measure such as the Euclidean or Hamming distance.

D.3.4 Accuracy

D.3.4.1 Motivation

The words accuracy, trueness and precision are important differentiated terms when referring to measurements in the scientific and technical context. Generally speaking, accuracy refers to how close a measured value is in relation to a known value or standard. However, the International Organization for Standardization (ISO) uses “trueness” for the above definition while keeping the word “accuracy” to refer to the combination of trueness and precision. On the other hand, precision is related to how close several measurements of the same quantity are to each other. In the field of statistics it is rather common to use the terms “bias” and “variability” to refer to the lack of “trueness” and the lack of “precision” respectively.

As a performance metric, accuracy should be central to trustworthy AI. However, you will often need to refine or change your measure of accuracy in order to establish an appropriate level of performance for your system.

Definition 28 (Accuracy) *Accuracy is defined as the degree of conformance between the estimated or measured value and its true value ED-76A (2015).*

For example, if some errors are more significant or costly than others, a total cost metric can be built into your model to weigh the cost of one class of error against another. Similarly, if the precision and sensitivity of the system are paramount for the detection of rare events, such as medical diagnosis of rare cases of illness, the precision and recall technique may be used. This method of dealing with unbalanced classification would allow you to weigh the proportion of correct detection by the system - of both common and rare outcomes - against the proportion of actual detection of the rare event (i.e. the ratio of the true detection of the rare outcome to the sum of the true detection of that outcome plus the missed detection or false negatives for that outcome). In general, the measurement of accuracy in the face of uncertainty is a challenge that should be the subject of considerable thought. The problems inherent in attempting to model a chaotic and changing reality will heavily influence the confidence level of your AI system. Accuracy concerns must address unavoidable noise in the data sample, architectural uncertainties caused by the possibility that a given model misses relevant features of the underlying distribution, and inevitable changes in input data over time.

ISO/DIS 5725-1 (2020) uses two terms "trueness" and "precision" to describe the accuracy of a measurement method. Trueness refers to the closeness of agreement between the arithmetic mean of a large number of test results and the true or accepted reference value. Precision refers to the closeness of agreement between test results obtained under stipulated conditions.

Definition 29 (Precision) *Precision is a performance metric used to evaluate a classifier, which measures the proportion of predicted positives that were correct ISO/IEC TR 29119-11 (2020).*

In ISO/DIS 5725-1 (2020), precision is defined by the closeness of agreement between independent test results obtained under stipulated conditions.

- Note 1: Precision depends only on the distribution of random errors and does not relate to the true value or the specified value.
- Note 2: The measure of precision is usually expressed in terms of imprecision and computed as a standard deviation of the test results. Less precision is reflected by a larger standard deviation.
- Note : Quantitative measures of precision depend critically on the stipulated conditions. Repeatability and reproducibility conditions are particular sets of extreme conditions.

D.3.4.2 ML Model Accuracy

In ML, the accuracy of a model is the proportion of examples for which it generates a correct output. This performance measure is also sometimes characterized conversely as an error rate or the fraction of cases for which the model produces an incorrect output. Keep in mind that, in some instances, the choice of an acceptable error rate or accuracy level can be adjusted in accordance with the use case specific needs of the application. In other instances, it may be largely set by a domain established benchmark. Thus

Definition 30 (ML model accuracy) *Accuracy of a ML model refers to the model ability to correctly predict the outcomes by generating less false positives and false negatives.*

D.3.5 Classification performance Assessment

The goal of a machine learning model is to learn patterns that generalize well on unseen data instead of just memorizing the data that it was trained on. When our model is ready, we would use it to predict the answer on the evaluation or test dataset and then compare the predicted target to the actual answer (ground truth). This is a typical approach that is followed to evaluate model performance. However, this comparison between predicted and actual values is performed based on a number of different metrics. The choice of the actual metric depends on the ML problem in hand.

		Target		
		Positive	Negative	
Model	Positive	TP	FP	$Precision = \frac{TP}{TP + FP}$
	Negative	FN	TN	
		$Recall = \frac{TP}{TP + FN}$	$Specificity = \frac{TN}{FP + TN}$	$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$
		$F1 = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$		

Figure D.2: Performance Metrics for Classification Problems

Classification is a prediction type used to give the output variable in the form of categories with similar attributes. Some of the popular classification metrics are Accuracy, Precision, Recall, F1

Score where:

- **Precision** is a measure of exactness in model operation.
- **Recall** is a measure of completeness or quantity.

"High precision" means that a model returns substantially more relevant results than irrelevant ones. "High recall" means that a model returns most of the relevant results that are available. There is often a trade-off between precision and recall during model training, and practitioners must determine what balance is best for any given case. Getting the balance wrong may impact the robustness of the model.

Confusion Matrix is a core element that can be used to measure the performance of the ML classification model but it's not considered a metric. By nature, it is a table with two dimensions showing actual values and predicted values. Each row of the confusion matrix represents the instances in a predicted class and each column represents the instances in an actual class. Confusion Matrix is not exactly a performance metric but sort of a basis on which other metrics evaluate the results. Each cell in the confusion matrix represents an evaluation factor. For example, for a binary classification as "positive" and "negative" :

- True Positive (TP) signifies how many positive class samples our model predicted correctly.
- True Negative (TN) signifies how many negative class samples our model predicted correctly.
- False Positive (FP) signifies how many negative class samples our model predicted incorrectly.
- False Negative (FN) signifies how many positive class samples our model predicted incorrectly.

The accuracy of the ODD relies on the assumption of ODD completeness and deals with the accuracy of the limitation constraints expressed on the concepts defined from the ontology. If we call IODD (Ideal ODD) the unknown exact domain (set of situations) where the system is able to operate and ODD the domain that we have characterized. The precision and coverage of this ODD are obtained by the following metrics. Then:

- TP represents the set of situations of IODD included in the ODD
- FP, the set of situations outside of IODD included in the ODD
- FN, the set of situations of IODD that are not included in ODD.

In practice : Precision can be evaluated by sampling situations in the ODD and evaluating the terms TP and FP to obtain the percentage of situations where the system correctly operates. Coverage may be evaluated by sampling situations in an envelope around the ODD, excluding the ODD, and evaluating the capacity of the system in these situations to obtain an approximation of the FN term.

Other classifier performance metrics could be used. Let N be the total number of instances in the dataset, N_{c1} be the total number of positive instances and N_{c0} be the total number of negative instances in the dataset. The following equations define the performance metrics related to accuracy and predictive values:

$$ACR = \frac{TP + TN}{N}$$

$$MCR = \frac{FP+FN}{N}$$

$$TPR = \frac{TP}{N_{c1}}$$

$$TNR = \frac{TN}{N_{c0}}$$

$$FPR = \frac{FP}{N_{c0}}$$

$$PPV = \frac{TP}{TP+FP}$$

$$NPV = \frac{TN}{TN+FN}$$

where ACR and MCR represent the classification accuracy and misclassification, respectively; TPR, TNR, FPR, and FNR represent the true positive rate, true negative rate, false positive rate, and false negative rate respectively; PPV and NPV represent the positive predictive value (also known as Precision) and negative predictive value, respectively.

When thresholds are used comparisons can be made across all thresholds using some area under the curve measures, such as ROC. For regression tasks, there is a zoo of accuracy measures that quantify the typical distance between predictions and expected values, such as Mean Absolute Percentage Error or Mean Squared Error (MSE). Mean Absolute Error (MAE) like MSE, is a metric for regression problems. It measures the average absolute difference between the predicted and actual values, providing a measure of the average prediction error. For ranking problems yet other accuracy measures are introduced, such as MAP@K or Mean Reciprocal Rank.

MSE is a commonly used metric for regression problems. It measures the average squared difference between the predicted and actual values, providing an overall measure of the model's accuracy.

Root Mean Squared Error ($RMSE$) is another metric for regression problems and is calculated by taking the square root of the mean squared error. It provides a measure of the average magnitude of the prediction error.

A **precision score** (see Fig. D.2) towards 1 will signify that our model didn't miss any true positives, and is able to classify well between correct and incorrect labeling of data.

The **F-score** is a single value a measure of the model accuracy. It is calculated from the precision and recall, where the precision is the number of true positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of true positive results divided by the number of all samples that should have been identified as positive.

$$Recall = \frac{TP}{TP+FN}$$

Precision is also known as positive predictive value, and recall is also known as sensitivity in diagnostic binary classification.

$$Precision = \frac{TP}{TP+FP}$$

Specificity is the proportion of actual negatives that the model has correctly identified as such out of all negatives.

A high **F1 score** symbolizes a high precision as well as high recall. It presents a good balance between precision and recall and gives good results on imbalanced classification problems. The $F1$ score is the harmonic mean of the precision and recall.

$$F1 = 2\frac{precision \cdot recall}{precision + recall}$$

F1-score is the harmonic mean of precision and recall and is used to balance these two metrics. It provides a single overall performance measure by considering both precision and recall. The more generic F_β score applies additional weights, valuing one of precision or recall more than the other.

$$F_\beta = (1+\beta^2)\frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

The Receiver Operating Characteristic curve (**ROC Curve**) is a plot which shows the performance of a binary classifier as function of its cut-off threshold. It essentially shows the true positive against the false positive rate for various threshold values. The area under the ROC curve is a single-value measurement, with its value ranging from 0 to 1. A classifier that provides a large area under the curve is generally preferable over a classifier with a smaller area under the curve. This is used for binary classification problems and assesses the model's ability to discriminate between positive and negative instances. It measures the area under the receiver operating characteristic curve, where the curve represents the true positive rate against the false positive rate.

The area under the precision-recall curve (*PRC*) is a single-value measure, with values ranging from 0 to 1. The *PRC* diagram depicts the trade off between *Recall* and *Precision*. A perfect classifier results in an area under the *PRC* of 1.

The **Fowlkes–Mallows index** FM is an external evaluation method that is used to determine the similarity between two clusterings (obtained after a clustering algorithm), and also a metric to measure confusion matrices. This measure of similarity could be either between two hierarchical clusterings or a clustering and a benchmark classification. A higher value for FM index indicates a greater similarity between the clusters and the benchmark classifications. The FM index, when results of two clustering algorithms are used to evaluate the results, is defined as

$$FM = \sqrt{\frac{TP}{TP+FP} \cdot \frac{TP}{TP+FN}}$$

where TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives. The minimum possible value of the FM index is 0, which corresponds to the worst binary classification possible, where all the elements have been misclassified. And the maximum possible value of the FM index is 1, which corresponds to the best binary classification possible, where all the elements have been perfectly classified.

Introduced by Youden (1950), the **J index** is a balanced accuracy metric defined as

$$J = sensitivity + specificity - 1$$

The intuition behind the J index is to express how much a classifier is able to appropriately discriminate positive and negative examples within, respectively, a control and a test group. The J-index produces a value ranging from 0 to 1, where the former indicates that all predicted instances were misclassified, and the latter that neither false positives nor false negatives were produced, i.e. a perfect classifier. The J-index has been proposed as a way to improve the standardisation of ML processes. This is due to the difficulties in using sensitivity and specificity for cross-study comparisons. The J-index can also be related to ROC analysis. It is usually understood as a criterion for selecting a threshold in the ROC space.

The **ABC metric**, or **attribution-based confidence metric**, is a metric used for image classification to calculate the confidence a model has in its prediction.

It is based on the principle of neighborhood sampling : the confidence of a model on a given input can be measured by sampling the neighborhood of the input and observing whether the model's output changes or conforms to the original output. The main difficulty being that the sampling becoming exponentially more difficult with an increase in the dimension of the input, the ABC metric uses attributions to focus the sampling around features that contribute the most to the output of the model.

The project EC3 of Confiance.ai implemented the method and tested it with success on various datasets. Our main findings are the following:

1. the cost of calculating attributions offsets the increase in the number of required samples in the datasets we tested. Therefore, we hypothesize that using vanilla neighborhood sampling is sufficient is most classification use-cases.
2. we adapted the neighborhood sampling algorithm to image detection, with fair success.

D.4. Completeness

D.4.1 Motivation

Completeness measures what proportion of the specified functions has been implemented. A missing function is detected when the system or software product does not have the ability to perform a specified function. It is the fraction of 1) Number of functions missing and 2) Number of functions specified.

For example, being evaluated for ML-based systems, we interpret that "missing" functions are the functions that were not successfully trained, even though developers specified to train them. In other words, such systems under evaluation fail on these missing functions, even though developers specified the training or test datasets for the ML components used in the system so that the datasets include such functions. On the other hand, functions "specified" are the functions that are included in such datasets, as well. Then, in order to measure functional completeness, we must check the induced training dataset of such ML models.

D.4.2 Data completeness

It has been agreed, following different Confiance.ai convergence workshops, that the dataset completeness can be defined following ISO/IEC 25012 (2008) (although this standard refers to the completeness of data, rather than the completeness of a dataset):

Definition 31 (Dataset completeness) *The degree to which subject data associated with an entity has values for all expected attributes and related entity instances in a specific context of use.*

The dataset has "dataset structural completeness" if all elements of the dataset have all attributes previously defined. Thus, data completeness for ML datasets refers to the degree to which it contains the necessary information required to accurately model the underlying patterns by the learning algorithm. Measuring dataset completeness includes evaluation of the amount of missing items, outliers and errors.

Dataset completeness metric could be based on the Ge and Helfert (2006) ratio defined as:

$$data\_completeness = \sum_{i=1}^{N} \frac{\gamma(d_i)}{N}$$

where $\gamma(d_i)$ is 0 if d_i is a missing data, and 1 otherwise.

D.4.3 Data Diversity

The following definition is the result of different Confiance.ai convergence workshops:

Definition 32 (Dataset diversity) *The presence in the dataset of all required information defined in the specification (requirements, ODD...).*

Data diversity is assessed by the evaluation of the presence of all required information and quantifies how the dataset fits the environment and application domains described in the specifications. During ML model design, training and testing, the level of diversity should be equally distributed for the different data subsets being selected. This should ensure that the ML model is enough diversified so as to cover its domain of possible stimuli. According to Gong et al. (2019), the only diversity metric used for supervised ML is the Determinantal Point Process (DPP) introduced by Kulesza et al. (2012). Then, Dereziński (2019) regularizes the DPP (R-DPP) to accelerate the training process. Moreover, other diversity indexes, used in biology and ecology, could be adapted for ML models such as Shannon entropy and mean proportional species abundance Tuomisto (2010).

D.4.4 Coverage

Unlike metrics that only depend only on the information contained in the dataset, coverage is a metric which characterizes both the ML system and the dataset. In other words, the coverage metric has data and model as input variables. In computer science, code coverage has been used since the 1960s and gradually made its way from research to the industry Ivanković et al. (2019). It is based on executing a set of unit or functional tests and aims at ensuring all lines of code are executed. In parallel, for machine learning and deep learning, coverage testing of the learned model is aimed at making sure that the output space is properly covered by the test dataset. Thus, the coverage is becoming a *sine qua non* condition for model validation Mani et al. (2019).

The following definition is the result of different Confiance.ai convergence workshops:

Definition 33 (Dataset and ML Model coverage of an expected space) *The coverage of a couple "Dataset + ML Model" is the ability of the execution of the ML Model on this dataset to generate outputs for each item of the expected output space.*

In practice, *Precision* can be evaluated by sampling situations in the ODD and evaluating the terms TP and FP to obtain the percentage of situations where the system correctly operates. Coverage may be evaluated by sampling situations in an envelope around the ODD, excluding the ODD, and evaluating the capacity of the system in these situations to obtain an approximation of the FN term, where

- $Precision = Area(ODD \cap IODD)/Area(ODD)$
- $Coverage = Area(ODD \cap IODD)/Area(IODD)$

D.5. Representativeness

D.5.1 Data Representativeness

Data representativeness refers in statistics to the notion of sample and population Mamalet et al. (2021). Transposed to AI, the sample corresponds to the data-set available for the development of the model (training, validation, testing), and the population corresponds to all possible observations in the field of application. Moreover, a dataset is representative when it describes the environment observations, and the distribution of its key characteristics is conform to the specifications need, requirements and the ODD of the targeted application.

The following definition is the result of different Confiance.ai convergence workshops:

Definition 34 (Dataset representativeness) *The conformity of the distribution of the key characteristics of the dataset according to a specification (requirements, ODD...)*

There are multiple existing methods to quantify the representativeness of datasets, stemming from statistics and ML fields. Indeed, Student, Chi-square and Kolmogorov-Smirnov tests may be applied to assess the goodness of fit of specified distributions. Furthermore, in case of large datasets, the confidence interval combined with the maximum entropy probability could be used to determine, in terms of dataset size and acceptance thresholds, the suitable dataset for ML need Blatchford et al. (2021).

D.6. Currency and Timeliness

Currency rules may be defined to assert limits to the lifetime of a data value, indicating that it needs to be checked and possibly refreshed. On the other hand timeliness refers to the time expectation for the accessibility of data.

D.6.1 Data Currency

The following definition is the result of different Confiance.ai convergence workshops:

Definition 35 (Data item currency) *The degree to which data has attributes that are of the right age in a specific context of use.*

Thus, data currency refers to the degree to which information is current with the world that it models. It can measure how "up-to-date" information is, and whether it is correct despite possible time related changes Redman (1996). In Data-driven AI, the real-life dataset could become obsolete rapidly. Data Currency refers to the degree to which data has attributes that

are of the right age in a specific context of use. Thus, it is the degree to which a datum is up to date. A datum value is up-to-date if it is correct in spite of possible discrepancies caused by time-related changes to the correct value.

D.6.2 Data Timeliness

The following definition is the result of different Confiance.ai convergence workshops:

Definition 36 (Data item timeliness) *The time delay from data production and acquisition to utilization.*

This is not specific to AI-based systems, any IT-based system has this problem. Nevertheless, it should not be forgotten.

E. Dependability

E.1. Motivation

In cases where system failure may have extremely severe consequences, overall dependability is the main trustworthiness requirement over and above basic functionality. This is the case for the following reasons: system failures affect a large number of people; users often reject systems that are unreliable, unsafe, or insecure; system failure costs may be enormous; and undependable systems may cause information loss. AI-based system and software dependability include such characteristics as availability, reliability, safety, and security. When developing such critical systems, tools and techniques can be applied to reduce the risk of injecting faults. Verification, validation, and testing processes, techniques, methods, and tools identify faults that impact dependability as early as possible in the life cycle. Additionally, mechanisms may need to be in place to guard against external attacks and to tolerate faults.

Thus, safety and security analyses need, as input, the definition of the system and in particular its application domain provided by the ODD. To ensure maintenability of the system in case of changing environment, the characteristics of the expected nominal environment conditions are necessary to evaluate the impact of its evolution.

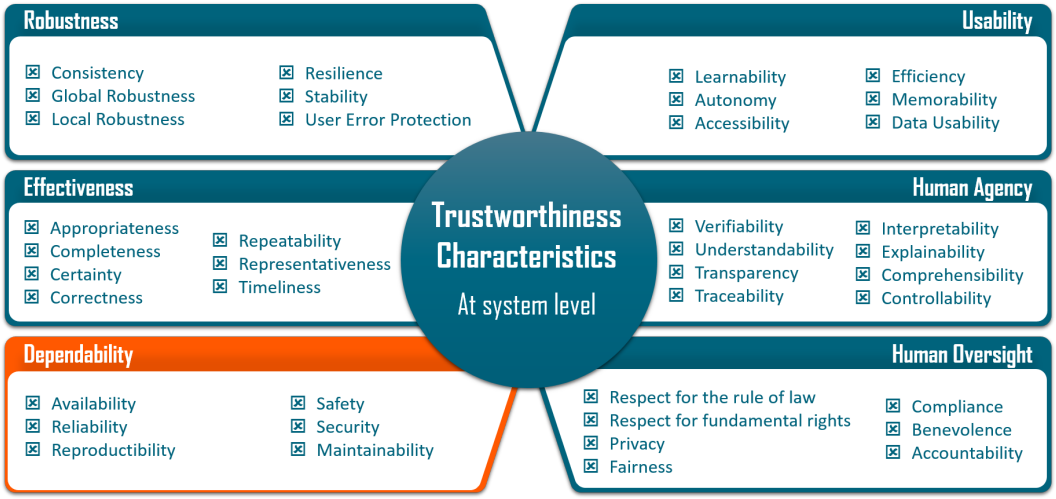


Figure E.1: The five critical characteristics of dependability at system level

In systems engineering, dependability can be defined as "*the ability of a system to deliver a service that can be justifiably trusted*" Avizienis et al. (2004). As said previously, over the years, the dependability concept has evolved to integrate other qualitative attributes, that we recall in the following ISO/IEC/IEEE 15026-1 (2019):

- **availability** (§ E.2): readiness for correct service,
- **reliability** (§ E.3): continuity of correct service,
- **safety** (§ E.5): absence of catastrophic consequences on the user(s) and the environment,

- **security** (§ E.6): ability to protect against unauthorized access to its features, data, or hardware to avoid their theft or misuse;
- **maintainability** (§ E.7): ability to undergo repairs and modifications.

Depending on the application(s) intended for the system, different emphasis may be put on different attributes. The description of the required goals of the dependability attributes in terms of the acceptable frequency and severity of the failure modes, and of the corresponding acceptable outage duration (when relevant), for a stated set of faults, in a stated environment, is the dependability requirement of the system. Several other dependability attributes have been defined that are either combinations or specializations of the six basic attributes listed above.

Security is the concurrent existence of a) availability for authorized users only, b) confidentiality, and c) integrity with 'improper' taken as meaning 'unauthorized'. Characterizing a system reaction to faults, is of special interest, via ,e.g., robustness (see chapter C), i.e. dependability with respect to erroneous inputs.

There are three main terms that must be clearly understood:

- **Failure**: A failure occurs when the observed behavior differs from the expected one. A failure is an instance in time when a system displays behavior that is contrary to its specification. An error may not necessarily cause a failure, for instance an exception may be thrown by a system but this may be caught and handled using fault tolerance techniques so the overall operation of the system will conform to the specification.
- **Error**: An error is the part of the system state which may lead to a failure Avižienis et al. (2004). An error is a discrepancy between the intended behavior of a system and its actual behavior inside the system boundary. Errors occur at runtime when some part of the system enters an unexpected state due to the activation of a fault. Since errors are generated from invalid states they are hard to observe without special mechanisms, such as debuggers or debug output to logs.
- **Fault**: A fault (which is usually referred to as a bug for historic reasons) is a defect in a system. The presence of a fault in a system may or may not lead to a failure. A fault is the cause of an error. A software fault lies in software, a hardware fault lies in hardware. For instance, although a system may contain a fault, its input and state conditions may never cause this fault to be executed so that an error occurs; and thus that particular fault never exhibits as a failure.

A correct system is delivered when the system implements the system function. A failure is an event that occurs when the delivered system deviates from correct system. Failure is a transition from correct system to incorrect system. Restoration is the transition from system service to correct system.

It is important to note that failures are recorded at the system boundary. They are basically errors that have propagated to the system boundary and have become observable. Faults, errors and failures operate according to a mechanism. This mechanism is sometimes known as a Fault-Error-Failure chain.

For dependable AI, standards, best practices, and established methods are still missing. Some guidelines have just been published (e.g., the EU's Ethics Guidelines for Trustworthy AI) or must still be adapted with respect to AI (such as the new version of ISO 26262-1 (2011) or the extension of the ISO/IEC 25000 series with respect to AI).

E.2. Availability

Availability: Is the system is able to fulfill its intended function at a point in time?

E.2.1 Motivation and definition

Availability and reliability are often used interchangeably but they actually refer to different things. Availability of a system is affected by planned and unplanned downtimes. However, asset reliability refers to the probability of an asset performing without failure under normal operating conditions over a given period of time. This does not include unplanned downtimes.

System availability (also known as equipment availability or asset availability) is used to gauge if an asset's production potential is being maximized, which has a direct impact on the financial health of a business. It measures the probability that a system is not failed or undergoing a repair action when it needs to be used. There are three qualifications that need to be met for a system to be available:

- Functioning equipment: Not out of service for repairs or inspections;
- Functioning under normal conditions: Operates in an ideal setting at an expected rate;
- Functioning when needed: Operational at any time production is scheduled

The objective of this section is to extend the concept of "availability" at AI-based system level but also at ML Model level.

Definition 37 (Availability) *Availability is defined as the degree to which a system, product or component is operational and accessible when required for use ISO/IEC 25010 (2011b).*

Thus, availability is a property of being accessible and usable on demand by an authorized entityISO/IEC 27000 (2018).

System availability analyzes the percentage of time a system is available for use, when it is down for a maintenance issue or when the system is actively undergoing maintenance. An availability of 0.995 means that in every 1000 time units, the system is likely to be available for 995 of these. There are several classifications of availability which include:

- **Instantaneous availability**: This availability measures the probability a system or piece of equipment is ready for operation at a specific point in time.
- **Average uptime availability**: This metric represents the time a system is available for use over a specific period.
- **Steady-state availability**: Often referred to as long-term availability, this represents the availability of a system as a company analyzes an infinite amount of operating time.
- **Inherent availability**: This metric only measures the availability of a system by analyzing operating time and corrective maintenance and doesn't account for system standby or preventative maintenance measures.
- **Achieved availability**: This metric accounts for corrective and preventative maintenance when calculating system availability.
- **Operational availability**: This represents the proportion of time a system is operating or capable of operating.

E.2.2 System availability

System availability metrics are crucial for evaluating system performance, identifying areas for improvement, and ensuring that systems are running efficiently. They help in minimizing system downtime, maintaining customer satisfaction, and optimizing resources for better system maintenance and management.

The system availability measure provides an indication of the percentage of the time that the system is actually available over the scheduled operational time, translated by the probability that the system is functioning when needed, under normal operating conditions. The Mean Down Time (MDT) measure is the average time that the system stays non-operational (unavailable). The downtime appears after the occurrence of a failure and includes all downtime associated with repair, corrective and preventive maintenance, and logistic delays to correct the system in order to be operational again. A lower MDT means better availability and consequently better reliability of the system.

The first step in calculating availability is deciding the period we want to analyze. Then, it is calculated by dividing $Uptime$ by the total sum of $Uptime$ and $Downtime$.

$$Availability = \frac{Uptime}{(Uptime + Downtime)}$$

$Downtime$ has the biggest impact on availability and is one of key KPIs for maintenance and in service support activities. Downtime can be broken down into planned vs. unplanned and frequency vs. length. Each component can be further broken down until an anomaly is identified. Once issues are pinpointed, they can be addressed and can improve availability. Moreover, $MTBF$ values are estimated for each component. Estimating AI-based software $MTBF$ is a tricky task, because it is really the time between subsequent reboots of the software. This interval may be estimated from the defect rate of the system. The estimate can also be based on previous experience with similar systems. The $MTTR$ is the time taken to reboot the failed processor. Then

$$Availability = \frac{MTBF}{(MTBF + MTTR)}$$

E.2.3 Data availability

The following definition is the result of different Confiance.ai convergence workshops:

Definition 38 (Data item availability) *Degree to which data has attributes that enable it to be retrieved by users and/or applications in a specific context of use.*

At the dataset level, the data storage cost is an unusual data quality metric. The storage cost of data does not depend on the data itself but the provider's quoted rate. The cost of storing data is tied closely to completeness, consistency, timeliness, and duplication. If our data storage costs keep rising, it is a sign that we might be collecting too much unusable data. Duplicate or incomplete data, for example, are not usable but still consume storage space. To reduce data storage costs, we need to make an inventory of our data storage needs. If our applications use data only for a specific time frame, it might be time to delete older records. We might also need to remove duplicate or incomplete records before storing them.

E.3. Reliability

E.3.1 Motivation and definition

Validation confirms, by providing objective evidence, that requirements for a specific intended use or application are met ISO 9000 (2015b). Deployment of AI systems which are inaccurate, unreliable, or poorly generalized to data and settings beyond their training creates and increases negative AI risks and reduces trustworthiness. For any AI-system, reliability is often of interest because failure events can lead to safety concerns. Failures of AI systems can lead to economic loss and even, in some extreme cases, lead to loss of life.

Reliability is closely related to robustness (see chapter C) and resilience, but the focus is on the time dimension. There is little work that formally defines the reliability of AI systems. ISO/IEC TS 5723 (2022) defined the reliability as the "*ability of an item to perform as required, without failure, for a given time interval, under given conditions*". This means that an AI system should be able to operate correctly under all conditions of its intended usage and throughout its lifetime. In the following, the program Confiance.ai adopts the following definition:

Definition 39 (Reliability) *Reliability is the ability of the AI system to perform correctly (i.e. the AI system fulfills stated requirements and demonstrates consistent intended behavior and results and demonstrates consistent intended behavior and results) ISO/IEC DIS 22989 (2021b).*

From a technical perspective, the reliability of an AI system is an umbrella term that involves different aspects of the quality of its AI component: the correctness of the outputs, the assessment of the AI model uncertainty, robustness to corrupted or manipulated inputs, and unexpected situations, and of course, intercepting errors. It is ensured by borrowing three principles: 1) Failure Prevention, 2) Failure Identification and Reliability Monitoring, and 3) Maintenance (see §E.7).

The key element in AI reliability is "failure". To prevent failures in AI systems, current state-of-the-art methods attempt to proactively identify likely sources of error resulting from 1) bad or inadequate data, 2) differences or changes in the environment, 3) model-related errors, or 4) poor reporting, and develop methods to correct for them in advance. After deployment, reliability mechanisms assess the model output for each new input and reject the unreliable output based on the auditing criteria of the density principle and the local fit principle.

The description of reliability requirements with quantitative measures and goals requires both mathematical expertise and application domain knowledge to determine under which conditions the AI application can be considered reliable. Thus, the probability that an AI system will perform a required function under specified conditions, without failure, over a given period of time Aerospace (2010) is an indicator of the reliability.

Corruptions in the input data, which can occur during normal operation and are part of the application domain, are usually represented in the database as well. At the same time, for some application contexts, it may be useful to extract and target such corruption to strengthen both data coverage and AI application performance. The robustness characteristic (see Chapter C) is designed to ensure that the AI component performs as consistently as possible, even at the boundary of the application domain. To do this, it deals with corrupted or manipulated input data such as sensor noise. It also deals with adversarial examples.

The factors that can affect AI reliability can fall into three levels: system, data, and model (i.e., algorithm). Fig. E.2 shows the Venn diagram for the three factors that affect AI reliability which will be further described.

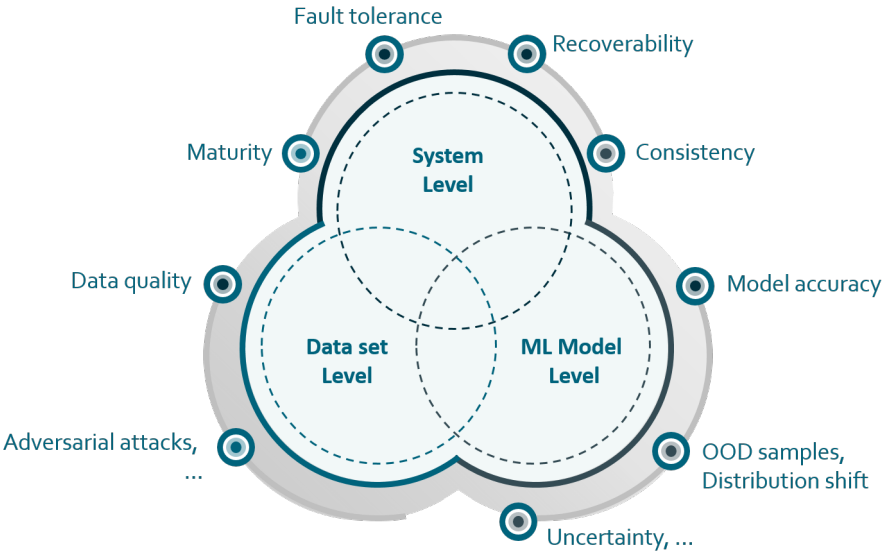


Figure E.2: Illustration of ML reliability affecting factors

E.3.2 System reliability

Metrics are needed to characterize reliability for AI systems such as failure rate, event rate, error rate, etc. Common assessments at system level are $MTBF$ and mean time to failure ($MTTF$). The former measures the average time between two consecutive failures, while the latter accounts for the time elapsing from the beginning of operation to the detection of the first failure. A higher $MTBF$ indicates that a system works longer before failing, increasing its maturity and consequently its reliability. The failure rate provides the frequency with which the system fails, expressed in failures per unit of time. An empirical method to assess reliability is proposed in engineering by: let N_t be the number of items tested over a period t, and suppose N_f items have failed. Assuming the sample is sufficiently large, then the probability of a failure, P_f, is $P_f = N_f/N_t$. The reliability, R, is defined as $R = 1 - P_f$, namely, the probability of success. When one attempts to apply these concepts to data-driven AI based system, a number of questions immediately arise. A data item that has been recorded can be either correct or erroneous, so the concept of $MTTF$ is not very adequate. Consequently, the criteria by which data reliability is assessed are to be defined. They do depend on the particular users who set them, and they may have to adapt to the dynamics of the perception of reality.

Two other measures are proposed by ISO/IEC 25023 (2015), but are more related to the design, coding and testing phases. The fault correction measure indicates the proportion of detected faults that have been corrected during the design/coding/testing phase, and the test coverage measure indicates the percentage of the system capabilities or functions that are executed when a particular test suite runs. A system with higher test coverage has more of its functions executed during the testing phase, which suggests a lower possibility of presenting undetected bugs when compared with a practice with a lower test coverage. However, when aiming to assess the operation phase of an system, these last two measures are of low relevance.

Reliability is composed of the following sub-characteristics:

- **Maturity** ISO/IEC 25010 (2011b) - *Degree to which a system, product or component meets needs for reliability under normal operation.*

- **Fault tolerance** ISO/IEC 25010 (2011b) - *Degree to which a system, product or component operates as intended despite the presence of hardware or software faults.*
- **Recoverability** ISO/IEC 25010 (2011b) - *Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.*
- **Consistency** ISO/IEC 21827 (2008) - *Degree of uniformity, standardization and freedom from contradiction among the documents or parts of a system or component*

Similar attributes can be used for data. For example, it has been agreed, following different Confiance.ai convergence workshops, that the data item recoverability can be defined following ISO/IEC 25012 (2008) (although this standard refers to the recoverability of data, rather than the recoverability of a data item):

Definition 40 (Data item recoverability) *Degree to which data has attributes that enable it to maintain and preserve a specified level of operations and quality, even in the event of failure, in a specific context of use.*

E.3.3 ML model reliability

A distinction is made here between two signs indicating that a learned machine learning model is not reliable:

- Poor performances: the model cannot perform well in the task in conditions that are considered as normal for humans;
- Vulnerabilities: the model performs well but has vulnerabilities that may lead to malfunctions in specific conditions. These malfunctions may appear either naturally in the course of the execution of the program, or be intentionally provoked by an adversary with malicious intentions.

Assessing the reliability of a system requires then to consider these two aspects;

ML models face a common challenge when dealing with out-of-distribution (OOD) inputs. These are essentially data samples from a different distribution than what the model has encountered during its training phase. Consequently, a dependable classifier should not only excel at accurately categorizing known in-distribution (ID) samples but also be able to label any OOD input as "unknown." This highlights the critical importance of OOD detection, a process that discerns whether an input belongs to the ID or OOD category and equips the model to take necessary precautions.

For ML component, one common contribution to failure events is that the inference are different from the training, which is referred to as OOD samples. For example, a new object appears and the ML algorithm can not recognize it. If the algorithm fails to detect the OOD samples in making a prediction, an incorrect decision is likely to be made, and potentially leads to errors. Thus, OOD detection, and being able to make an appropriate adaptation for OOD samples are important in improving AI reliability.

The aim of an OOD detectorConfiance.ai et al. (2023) is to decide whether a sample $x \in \mathscr{X}$ is from the training distribution $\mathscr{P}_{in}$ or not. The detection can thus be formulated as a binary classification problem, where positives correspond to out-of-distribution (OOD) data, and negatives to in-distribution (ID) data. Then, the decision is made in relation to a detection threshold λ,

chosen such that:

$$G_\lambda(x) = \begin{cases} \text{OOD} & S(x) \geq \lambda \\ \text{ID} & S(x) < \lambda \end{cases}$$

where $S(x)$ is the score returned by the OOD detector.

Determining the ideal OOD detection method can be challenging due to the variability in results across different datasets and model architectures but the Confiance.ai has defined a Methodological Guidelines to support that activity Confiance.ai et al. (2023). Moreover, to assess the performance of an OOD detector, we evaluate its ability to distinguish between in-distribution and out-of-distribution (OOD) data. Typically, the test set of the classifier represents in-distribution (ID) data, while other datasets with alternative distributions serve as OOD data. These following metrics can assess the performance of an OOD detectors: AUROC, FPR95TPR, and TPR5FPR.

- **AUROC**: AUROC (Area Under the Receiver Operating Characteristic curve) is a widely utilized metric for assessing binary classifier performance. It quantifies the detector's ability to differentiate between positive and negative samples across all possible decision thresholds (λ). To construct the AUROC, we follow these steps:
  1. Calculate the true positive rate (TPR) and false positive rate (FPR) for various threshold values of λ.
  2. Plot a receiver operating characteristic (ROC) curve with TPR on the y-axis and FPR on the x-axis. This curve represents the classifier's performance at different decision thresholds.
  3. The AUROC is determined by computing the area under the ROC curve. It ranges from 0 to 1, where 0.5 indicates random performance, and a value of 1 signifies perfect discrimination.
- **FPR95TPR**: This metric calculates the false positive rate when the true positive rate is held constant at 95%. In simpler terms, it answers the question: "If I want to be really sure I catch 95% of OOD data, how often will the In-Distribution (ID) data be wrongly classified as OOD?"
- **TPR5FPR**: This metric calculates the true positive rate when the false positive rate is held constant at 5%. Put differently, it addresses the question: "If I can tolerate a 5% rate of mistakenly identifying ID data as OOD, how many of the actual OOD data can I successfully identify?" This metric is less commonly used in research but could be particularly relevant for manufacturers or systems with strict constraints on false alarms.

However, we need to distinguish between general performance metrics and the loss function, the latter being an important part of the training of many ML models, especially supervised ML, where learning algorithms such as gradient descent are typically used to adjust the model at each training step to gradually improve the loss function of the training data set. The value of the loss function (also called the loss) on the training data, in comparison to the loss on the test data, is an indication of the quality of the training and can indicate issues such as potential overfitting.

E.3.4 Data reliability

Many forecasting errors are caused by distribution shifts where the ML-inference differs from the ML-training. The following definition of a distribution shift is the result of different Confiance.ai convergence workshops:

Definition 41 (Distribution shift between datasets) *Distance between datasets, according to defined properties of these datasets.*

In an adversarial attack, a small permutation is applied to the data to make the prediction of the model inaccurate, leading to a reliability event. Data quality can also lead to software failures. If the data that is fed into an ML algorithm is noisy, if it is contaminated, it can also lead to failures. Thus, data quality affects the reliability of an ML-based system Kavzoglu (2009). To illustrate, taking a classification task, imbalance and noise in the training set can cause the accuracy to decrease. The quality of the training data is also highly related to the performance of the ML algorithm. Data with errors can lead to prediction errors. Another aspect of data quality Mattioli et al. (2022b) is related to the issue of data bias/imbalance. This is more relevant for many classification algorithms used in ML-based systems. Data quality can affect their performance. Thus, it is of interest to study how data and algorithms affect model accuracy. Adversarial attacks can be seen as a special kind of "data quality" problem, where problematic inputs to the algorithm are deliberately used so that the AI system fails to produce the correct output. A key to the reliability of an AI system is the robustness of the algorithm against adversarial attacks.

Herein, imbalance means the imbalance in the shares of observations between different class labels. Furthermore, the deviation of the distribution of the labels in the test set from the training set also affects the robustness of AI algorithms. It is important to investigate How data quality affects model accuracy.

Moreover, EUROCAE WG114 – SAE G34 (2021) defines the concept of **data reliability** by the confidence level in the goodness of the data (e.g. because it's provided by a trusted source, or by a high-fidelity model).

The following definitions are the results of different Confiance.ai convergence workshops:

Definition 42 (data item source reliability) *Confidence level in the source (physical sensor, human, etc.) of the data item.*

Definition 43 (dataset source reliability) *In the dataset, ratio of data items having a source reliability above a defined threshold in the dataset.*

A data item that has been recorded can be either correct or erroneous, so the concept of $MTTF$ is not very adequate. Consequently, the criteria by which data reliability is assessed are to be defined. They do depend on the particular users who set them, and they may have to adapt to the dynamics of the perception of reality. It is proposed, therefore, to assign three measurements to data reliability: internal reliability, relative reliability, and absolute reliability.

Let N be the number of data items in a database, and let N_r be the number of correct items in the database. Then the reliability R is defined as Nr/N. The term "reliability" here holds for any of the above three measurements. The distinction between them is based on a different definition of the notion "correct"; in other words, the value of N_r may vary. Note that the definition is similar to the one presented above, except that here we do not account for the time dimension. Note also that the definition is based on an implicit assumption that the data items possess the same value to the user; that is, the "worth" of each correct datum is the same (otherwise, a weighting function should be introduced; it is omitted here for simplicity).

E.3.5 Data Drift

Concept drift refers to situations where the underlying patterns or relationships in the data change over time. A robust AI system should be able to adapt and update its model or algo-

rithms to account for concept drift and maintain its accuracy.

In the context of machine learning, the performances and reliability of predictive models depend heavily on the quality and consistency of the underlying input data. In machine learning, it's often assumed when training models that the process responsible for generating data is stationary. However, real world application are far from static, and data often evolves over time, presenting a challenge known as data drift. Data drift refers to the phenomenon where the statistical properties and distribution of the incoming data change, whether it be the input or targets. This can potentially alter the performances of the previously trained models.

Detecting data drift is of utmost importance to maintain the performance of machine learning models. It's a component that must be included in any machine learning operations lifecycle. Failure to address data drift can lead to degraded predictions, reduced efficiency, and potential risks in decision-making processes. Therefore, understanding and effectively detecting data drift have become essential tasks in machine learning.

Data drift can arise from various sources and manifest in different types, making it a multifaceted challenge in machine learning. The sources of data drift can include evolving user behavior, changes in data sources, modifications in underlying processes, or shifts in external factors. Moreover, data drift can be categorized into different types depending on weither the drift alters the statistical properties of the feature, the labels or both. The appropriate solution for addressing data drift often depends on the specific type of drift encountered. By leveraging drift detection techniques, it's possible to raise alarms and identify problems with the data collection process, enabling them to take corrective measures and maintain the reliability and effectiveness of their models.

Definition 44 (Data drift) *Data drift is the process that occurs when the data distribution (inputs/outputs) at test time is different from the distribution used to learn the machine learning model.*

Mathematically, it's a change in the joint distribution:

$$\mathbf{P}(X_{ref}, Y_{ref}) \neq \mathbf{P}(X_{target}, Y_{target})$$

Where $\mathbf{P}(X_{ref}, Y_{ref})$ is the reference joint distribution and $\mathbf{P}(X_{target}, Y_{target})$ is the target joint distribution. In practice, these two distributions can refer to the training and the testing distributions respectively. Note that X represents the features and Y are their corresponding targets.

This equation suggests that the data drift can occur either because of a change in the distribution of the features, the targets or their conditional distribution. Therefore, there are various types of data drift that can be classified under these three main categories:

1. **Concept Drift**: Concept drift occurs when the relationship between the input features and the target variable also called the concept changes. This means that the function that maps the feature space to the target space changed. For example, in a spam email classification model, if the characteristics of spam emails change over time, the model trained on historical data may not perform as well on new incoming emails.

$$\mathbf{P}(Y_{ref}|X_{ref}) \neq \mathbf{P}(Y_{target}|X_{target})$$

2. **Covariate Drift**: Also known as covariate shift or feature drift, occurs when the input features' distribution changes over time while the relationship between the features and the target variable remains similar. This means that the statistical properties of the input

data shift. Since the models are trained on a certain input distribution, changing the statistical properties of this distribution can affect the model's performance. For instance, if a weather prediction model is trained on data collected from a certain geographical area and is then applied to a different region with different climate patterns, the model may experience covariate drift.

$$\mathbf{P}(X_{ref}) \neq \mathbf{P}(X_{target})$$

3. **Prior Probability Drift**: This type of drift refers to changes in the class distribution or the balance of different classes in the target variable. When the relative frequencies of classes change significantly over time, models that were trained on historical data may not accurately predict the current class probabilities. For example, if a model is trained to detect fraud in financial transactions and the prevalence of fraudulent activities decreases or increases over time, the model's performance may be affected due to prior probability drift.

$$\mathbf{P}(Y_{ref}) \neq \mathbf{P}(Y_{target}) \quad \text{and} \quad \mathbf{P}(X_{ref}|Y_{ref})) = \mathbf{P}(X_{target}|Y_{target})$$

It's important to note that these types of drift are not mutually exclusive. It's possible to have multiple types of drift simultaneously in real-world scenarios.

E.3.5.1 Drift and Temporality

Data drift is inherently connected to temporality, since it refers to the changes in the data distributions over time. Temporality plays a crucial role in understanding and managing data drift, as different types of drift have different patterns and might require a different detection solution. The four predominant data drift patterns are widely recognized as follows:

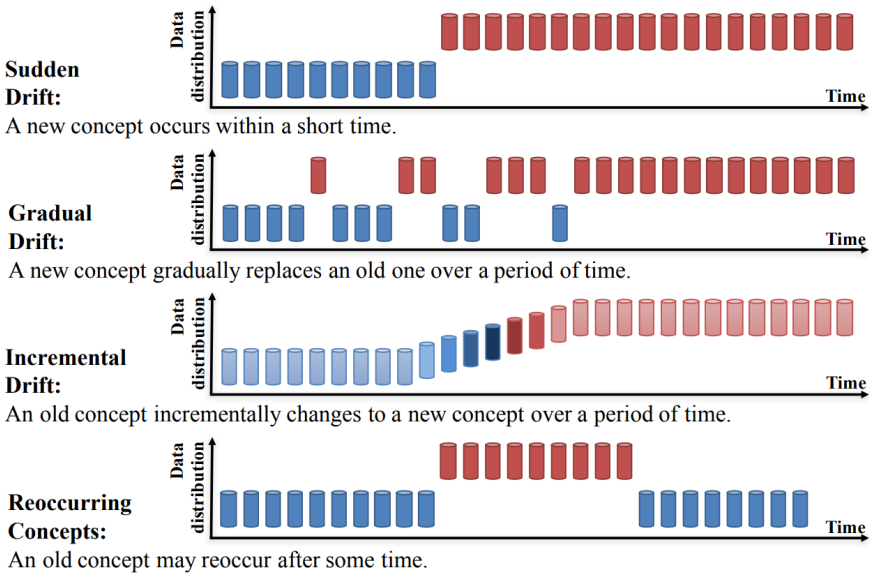


Figure E.3: Data drift patterns as described in Lu et al. (2020)

- Sudden data drift happen when there is an abrupt and significant change in the data distribution. This drift pattern can occur due to unexpected events, abrupt system failures, or sudden shifts in user behavior. For example, a change in the parameters of a camera or

its position in a visual inspection task can provoque a sudden change in the geometric or photometric properties of the images.

- Gradual data drift refers to a slow and continuous change in the data distribution over time. This type of drift often occurs due to gradual shifts in the underlying processes or external factors influencing the data. For instance, the change of preferences in a fashion industry, where trends evolve slowly over time, leading to a gradual shift in demand of different styles of clothing.
- Incremental data drift occurs when small, incremental changes accumulate over time. These changes are small and may require some time to become noticeable but they can have a substantial impact when they accumulate over an extended period. The stock market is a great example of incremental drift. The daily fluctuations in stock prices might not be big but their contribution creates a significant shift in the market dynamics.
- Recurring data drift refers to a cyclical or periodic pattern of drift that repeats at regular intervals. This can be due to seasonal variations, economic cycles or any other periodic influence. For instance, sales of summer products (e.g., sunscreen) may consistently increase during summer months and decrease during winter months, leading to recurring drift in the purchasing patterns.

Considering the different temporal drift patterns is crucial is essential for implementing effective monitoring and adaptation strategies to ensure the continued accuracy and reliability of machine learning models. Each pattern brings its own challenges and requires an adapted strategy to handle them effectively.

E.4. Reproductibility

E.4.1 Motivation

In general, scientists perform the same experiment several times in order to confirm their findings. These findings may show variation. In the context of an experiment, repeatability measures the variation in measurements taken by a single instrument or person under the same conditions, while reproducibility measures whether an entire study or experiment can be reproduced in its entirety.

Such issues serve as an essential step to verify a critical system. In terms of AI trustworthiness, this verification facilitates the detection, analysis, and mitigation of potential risks in an AI system, such as a vulnerability on specific inputs or unintended bias. Thus AI reproducibility is emerging as a concern among AI Engineers. Moreover, an AI-system must work with a variety of input in order to obtain different outputs, hence it must be reliable. Also, an AI task must produce the same output when repeatedly performed under the same conditions.

Reproducibility can be defined as the ability of AI experiments to behave similarly when repeated under identical conditions. Reproducibility is related to replicability, which refers to the ability to independently arrive at non-identical conclusions that are at least similar, even though there may be differences in sampling, research procedures, and data analysis.

E.4.2 Definitions

Over several years and in several publications, Gundersen (2021) defined reproducibility as "*the ability of independent researchers to draw the same conclusions from an experiment by relying on the documentation shared by the original researchers when following the scientific method. The documentation relied on by the independent researchers specifies the type of reproducibility study, and the way the independent researchers reached their conclusion specifies to which degree the reproducibility study validated the conclusion*". As both concepts are essential, Confiance.ai proposed the following definitions Díaz-Rodríguez et al. (2023):

Definition 45 (Repeatability) *Repeatability (same team, same experimental setup), which means that an individual or a team of individuals can reliably repeat his / her / their own experiment.*

Definition 46 (Replicability) *Replicability (different team, same experimental setup): an independent group of individuals can obtain the same result using artifacts which they independently develop in their entirety.*

Definition 47 (Reproducibility) *Reproducibility (different team, different experimental setup with stated precision): a different independent group can obtain the same result using their own artifacts.*

E.4.3 Metrics

Hence, reproducibility requires that another, independent team of investigators have to conduct the same experiment. This is contrasted with repeatability, which is the ability of the same investigators to produce the same result when repeating an experiment. The key is that an independent research team should produce the same results as the original team based only on the documentation created by the original team. In AI research, the documentation has three components: the documentation of the AI method that the original research team has developed and wants to test; the experiment description, which is written both as text and as code; and the data that are used for evaluating the AI method.

Gundersen (2019) defined three metrics to quantify whether an experiment e is $R1$, $R2$, or $R3$-reproducible, and to what degree. The metrics $R1D(e)$, $R2D(e)$, and $R3D(e)$ measure how well the three factors, Method, Data, and Experiment, are documented for experiment e:

$$R1D(e) = \frac{\delta_1 Method(e) + \delta_2 Data(e) + \delta_3 Exp(e)}{\delta_1 + \delta_2 + \delta_3}$$

$$R2D(e) = \frac{\delta_1 Method(e) + \delta_2 Data(e)}{\delta_1 + \delta_2}$$

$$R3D(e) = Method(e)$$

where $Method(e)$, $Data(e)$, and $Exp(e)$ are the weighted sums of the truth values of the variables listed under the three factors $Method$, $Data$, and $Experiment$. The weights of the factors are δ_1, δ_2 and δ_3, respectively.

This means that the value for $Data(e)$ for experiment e is the summation of the truth values for whether the training, validation, and test data sets as well as the results are shared for e. It is of course also possible to give different weights to each variable of a factor.

E.5. Safety

Safety refers to the property of an AI system that is does not, under defined conditions, lead to a state in which human life, health, property or the environment is endangered. The use of AI systems can introduce new safety issues.

Avoiding unreasonable risk of hazards that could cause harm to the users of the system or its environment is the objective of system safety engineering. The definition of unreasonable risk, in accordance with regulation and legal product liability considerations, needs to be determined for the specific system context through a risk analysis study. This requires a careful evaluation of the causes and effects of system errors, an assessment of the probability of them occurring, and the definition of strategies to eliminate or reduce the residual risk associated with such errors. Moreover, the use of AI introduce significant new challenges to the classical safety assurance process.

Some of AI technologies are inherently statistical in nature. In general, they tend to be obtained through non-deterministic (non-repeatable) means and may give answers that are only correct to some probability – if a probability can be assigned at all. Validating such systems presents challenges not typically found in more deterministic, conventional software. The considerable size and dense interconnections of some AI models (typically artificial neural networks) present an additional challenge to traditional methods and tools applied to software safety. However, by far, the biggest obstacle depends specifically, not on the AI methods used, but on the function, they are used on: typically, unstructured data such as images and sound. Indeed, one source of difficulty is the inherent lack of structure that prevents most of the meaningful and desirable safety properties to even be formally specified, let alone measured and verified. While work is being carried out on the specification of AI applications, much is yet to be done.

However, and depending on the AI methods used, functional safety can still be measured and verified, insofar as such properties are formally defined. For example, the safety properties of neural networks applied to control and command have been entirely verified in the field of aviation. For more traditional AI (*e.g.,* symbolic AI such as constraint programming), it is all the more possible to use traditional ways of measuring and ensuring safety, albeit with necessary adjustments to the evolution of the field. Several reports, particularly from the Confiance.ai Project dedicated to Trustworthy Characterization, give an overview of the existing methods and propose ways to move forward in the future to achieve a wider range of safety measures and verification for the AI-enhanced systems to come.

In Machine learning approach, the Kendall rank correlation coefficient, commonly referred to as Kendall's τ coefficient, is a statistic used to measure for identifying monotone relationships between two data sequences. A τ test is a non-parametric hypothesis test for statistical dependence based on the τ coefficient. It is a measure of rank correlation: the similarity of the orderings of the data when ranked by each of the quantities.

Let $(x_1, y_1)\ldots,(x_n, y_n)$ be a set of observations of the joint random variables X and Y, such that all the values of (x_i) and (y_i) are unique. Any pair of observations (x_i, y_i) and (x_j, y_j), where $i < j$, are said to be concordant if the sort order of (x_i, x_j) and (y_i, y_j) agrees: that is, if either both $x_i > x_j$ and $y_i > y_j$ holds or both $x_i < x_j$ and $y_i < y_j$; otherwise they are said to be discordant. The **Kendall τ coefficient** is defined as:

$$\tau = \frac{\text{number of concordant pairs} - \text{number of discordant pairs}}{\text{number of pairs}}$$

$$= 1 - \frac{2.\text{number of discordant pairs}}{\binom{n}{2}}$$

where

$$\binom{n}{2} = \frac{n(n-1)}{2}$$

is the binomial coefficient for the number of ways to choose two items from n items.

E.6. Security

The security of a system is a property that reflects the system's ability to protect itself from accidental or deliberate external attack. Security is essential as most systems are networked so that external access to the system through the Internet is possible. Security is an essential pre-requisite for availability, reliability and safety.

Let us precise some terms.

- **Asset**: Something of value which has to be protected. The asset may be the software system itself or data used by that system.
- **Attack**: An exploitation of a system's vulnerability. Generally, this is from outside the system and is a deliberate attempt to cause some damage.
- **Control**: A protective measure that reduces a system's vulnerability. Encryption is an example of a control that reduces a vulnerability of a weak access control system.
- **Exposure**: Possible loss or harm to a computing system. This can be loss or damage to data, or can be a loss of time and effort if recovery is necessary after a security breach.
- **Threat**: Circumstances that have potential to cause loss or harm. You can think of these as a system vulnerability that is subjected to an attack.
- **Vulnerability**: A weakness in a computer-based system that may be exploited to cause loss or harm.

Security is fundamental to protect against antagonistic or cyber attacks. The use of a security management framework, such as ISO/IEC 27000 (2018), ensures controls are in place to enforce confidentiality, integrity and accountability in the management of every numerical assets. Thus, a secured system is one that is able to maintain the integrity of the information of which it is composed. This includes protecting its architecture from unauthorized modification or damage to any of its components. Even under hostile or adversarial conditions, such system will continue to function and be accessible to its authorized users, and will keep confidential and private information secured.

Just like any software, AI systems also have the vulnerability of getting attacked by adversaries (eg. hacking). In case an AI system is attacked by an adversary, there are chances that the AI system may respond differently and produce an unwanted output. It may even shut down. Hence, in order to mitigate this, the AI's security must be taken into account while designing and developing the AI system.

Security assurance strategies could be based on:

- **Vulnerability avoidance**: The system is designed so that vulnerabilities do not occur. For example, if there is no external connection then external attack is impossible.

- **Attack detection and elimination**: The system is designed so that attacks on vulnerabilities are detected and neutralized before they result in an exposure. For example, virus checkers find and remove viruses before they infect a system.
- **Exposure limitation and recovery**: The system is designed so that the adverse consequences of a successful attack are minimized. For example, a backup policy allows damaged information to be restored.

AI-based systems should follow such usual security best practices, such as hardening and applying patches. Additionally, AI systems should also be robust to adversarial attacks that attempt to misclassify data, e.g. by using a photo of a face in a facial recognition system.

Definition 48 (Security) *Security in systems and software engineering is the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization ISO/IEC 25010 (2011b).*

This characteristic is composed of the following sub-characteristics:

- **Confidentiality** ISO/IEC 25010 (2011b) - *Degree to which a product or system ensures that data are accessible only to those authorized to have access.*
- **Integrity** ISO/IEC 25010 (2011b) - *Degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.*
- **Non-repudiation** ISO/IEC 25010 (2011b) - *Degree to which actions or events can be proven to have taken place so that the events or actions cannot be repudiated later.*
- **Accountability** ISO/IEC 25010 (2011b) - *Degree to which the actions of an entity can be traced uniquely to the entity.*
- **Authenticity** ISO/IEC 25010 (2011b) - *Degree to which the identity of a subject or resource can be proved to be the one claimed.*

One of the major security risks to AI-based systems is the potential for adversaries to compromise the integrity of their decision-making processes so that they do not make choices in the manner that their designers would expect or desire. One way to achieve this would be for adversaries to directly take control of an AI system so that they can decide what outputs the system generates and what decisions it makes. Alternatively, an attacker might try to influence those decisions more subtly and indirectly by delivering malicious inputs or training data to an AI model.

E.6.1 Confidentiality

Definition 49 (Confidentiality) *A piece of data and/or the information within, denoted by I, is said to be confidential to a set of parties $\{P_i\}$ if any other party $At \notin \{P_i\}$ is unable to access such information, or even if accessing it, the non-intended party is unable to interpret and disclose I. Thus, confidentiality can be represented as a -strong- relationship $\triangleright_c$ defined by:*

$$P_j \triangleright_c I, \forall P_j \in \{P_i\} \wedge At \not\triangleright_c I, \forall At \notin \{P_i\} \tag{E.1}$$

The following definition is the result of different Confiance.ai convergence workshops:

Definition 50 (Dataset confidentiality) *Degree to which the dataset has attributes that ensure that it is only accessible and interpretable by authorized users in a specific context of use.*

Dataset confidentiality directly relates to "Dataset accessibility".

In practice, the assessment of confidentiality as defined in E.1, for a given system depends upon (1) the specific elements used to build and deploy the relationship between the set of parties $\{P_i\}$ and the information I, and (2) the measures (or barriers) used to impede that any non-intended party At breaks the relationship. The problem is quite complex if we consider the different levels at which information can be represented, stored, and exchanged in the system, the variety of channels used to do so, their extension and exposed surface, and the high number of potential non-intended parties. In addition, the means, motivations, and intelligence of hostile parties can easily overflow any attempt to ensure confidentiality without thorough foundations. Thus, formal and mathematical methods can provide certainty on the level of confidentiality achieved (limited to a given system and operational environment). In that respect, some representative families of confidentiality mechanisms that ensure suitable levels of security are listed below. All the families target the non-disclosure of information I assuming that certain pre-conditions are satisfied, in particular the secrecy of some information tokens k_{P_i} with high-entropy, named keys and only known by $\{P_i\}$.

Symmetric cryptography: Symmetric cryptography refers to a family of algorithms that are based upon the same key k for the exchanges between a set of parties $\{P_i\}$. The relationship E.1 is built based upon a ciphering algorithm $encrypt(I,k)$ that produces an encrypted code non-interpretable unless the $decrypt(.,.)$ algorithm and key k are known. Indeed, the latter retrieves the original information when applied to the encrypted code $I = decrypt(encrypt(I,k),k)$. From a computational standpoint, the robustness of the confidential scheme essentially depends upon the length of the key $n = |k|$. This is measured by the overall probability by an attacker At to guess the key:

$$P[At \rightarrow k \in K_{At}] = \frac{1}{2^n},\ n = |k|, k \in \{0,1\}^n \tag{E.2}$$

The complexity can vary depending on the specific symmetric algorithm. Representatives in this family are Data Encryption Standard (DES) and Advanced Encryption Standard (AES) Al-Sabaawi (2022).

Asymmetric cryptography: Whereas symmetric cryptography offers suitable performance and levels of confidentiality, it also imposes certain limits, in particular in public exchanges where new unknown parties need to be added to the set $\{P_i\}$. To overcome this issue, the Diffie-Hellman protocol Li (2010) allows to public exchange information tokens in order to build a couple of secrets (private) and public keys (k_{si}, k_{pi}) for each party P_i. The relationship E.1 is built based upon the set of keys $\{(k_{si}, k_{pi})\}$ and an a couple of asymmetric algorithms $encrypt(.,.)$, $decrypt(.,.)$ which satisfy $I = decrypt(encrypt(I,k_{si}),k_{pi})$. Whereas the public keys $\{k_{pi}\}$ can be freely distributed, confidentiality relies upon the secrecy of the keys $\{k_{si}\}$ only known by their respective parties. From a computational standpoint, the robustness of the scheme also depends on the length of the secret keys. However, since asymmetric cryptography relies upon a mathematical foundation (solution of complex algebraic equations in a large mathematical field), the length of the keys can be increased without significantly augmenting the complexity of the algorithms $encrypt(.,.)$ and $decrypt(.,.)$. This offers an advantage in protection against different categories of attacks while still preserving performance. Representatives in this family are RSA algorithms Alam et al. (2016), ElGamal Encryption Alam et al. (2016) and Eliptic Curves

Fang and Wu (2017).

Homomorphic cryptography: It is a particular family of cryptographic algorithms based upon the mathematical notion of homomorphism. It is indeed a function h mapping two spaces A, B (groups, rings, vector spaces) along with two (algebraic) operations $(.)$ and $(*)$ respectively defined over the spaces such that:

$$h : A \mapsto B,\ h(x.y) = h(x) * h(y),\ \forall x, y \in A \tag{E.3}$$

An homomorphism is said to preserve the algebraic structure of the space A over B and the operations/spaces are said to be homomorphic. This property is used in cryptography to handle and operate ciphered (non-interpretable) information without the need to share or know secret keys $\{k_{si}\}$. Indeed, if the $encrypt()$ function is an homomorphism, then

$$encrypt() : \{0,1\}^m \mapsto \{0,1\}^n,\ encrypt(I_1.I_2) = encrypt(I_1) * encrypt(I_2),\ \forall I_1, I_2 \in \{0,1\}^m \tag{E.4}$$

Homomorphic ciphering stands for preservation of secrecy whereas still allowing certain operations to be applied on ciphered information. This is particularly useful when I conveys personal information and the privacy of data subjects needs to be protected. Such is the case in image/face recognition where sensitive materials are sampled and used to train and feed the ML/DL model in operation mode. It is also the case in edge-cloud based architectures where sensitive data need to be shared, e.g. to train a federated ML/DL model. Representatives in this category are Lattice Cryptography Plantard et al. (2013) and Fully Homomorphic Encryption Chaudhary et al. (2019).

E.6.2 Integrity

Definition 51 (Integrity) *Integrity is the property of protecting the accuracy and completeness of assets.*

The following definitions are the results of different Confiance.ai convergence workshops:

Definition 52 (Data item integrity) *The degree of assurance that the data item has not been lost or altered (since the data item origination or authorized amendment).*

Definition 53 (Dataset integrity) *The degree of assurance that the content of the dataset has not been lost or altered (since the dataset origination or authorized amendment).*

Integrity is essentially related to a piece of information I and their preservation across time and across different (unitary or binary) operations $\{Op_i\}$ applicable to it. The operations represent actions a system can perform over I, like for instance, $read()$, $copy()$, $write()$, $move()$, $submit()$, $receive()$ or the usage of I as input of any function $f()$ of the system. The set of operations can be extended with a particular set of distorting operations $\{Op_{phy}\}$ due to the physical nature of informatics systems, components, storage, memories, and channels and the intrinsic electronics and electromagnetic phenomena that can influence and distort I. For instance, $noise()$, $signal\_decay()$, $bits\_flip()$, $bandwidth\_loss()$ belong to this category. A third set of operations $\{Op_{At}\}$ is considered due to the actions a threatening party At can play to intentionally $alter()$, $disrupt()$, $tamper\_with()$, or simply $discard()$ the information I. Thus, given a system $S = \{C_j\}$ where C_j is a SW/HW component or a channel, the integrity of the piece of information

I is preserved if the following relationship holds:

$$\forall Op \in \{Op_i\} \cup \{Op_{phy}\} \cup \{Op_{At}\}, \forall C_j, Op(C_j, I) = I$$

In practice, the assessment of the integrity property becomes highly complex given that the number of operations potentially impacting the information is enormous and moreover, for many of them the equivalence $Op(C_j, I) = I$ is unfeasible to be ensured in most cases, *i.e.* $Op(C_j, I) = I'$ and $I \neq I'$. For instance, the inequality appears when C_j is a lossy or faulty channel or when the channel is undermined by an attacker. In such cases, the assessment of integrity demands the implementation of algorithms, mechanisms and protocols that allow to identify the distorted information I', correct it if possible or simply discard it. A brief description of representative families to assess integrity are inline.

Error detecting codes: It is a family of algorithms based upon the information and communication theories, *i.e.* Shannon Theory Shannon (1948), which aims to detect and correct errors occurred during message communication due to noisy or misbehaving channels. In the case of Error Correction Codes (ECC), there is a one-way function (injective, hard to invert) $Ecc : \{0,1\}^m \mapsto \{0,1\}^n$, with $n << m$ that is used to generate a code $Ecc(I)$ which is submitted together with I thus adding redundancy, *i.e.* the message is composed as $m = (I, Ecc(I))$. Once m is received, it can be denoted by $m' = (I', Ecc(I))$ and any modification can be detected by re-calculating the correction code, *i.e.* if $Ecc(I') \neq Ecc(I)$ then I' is a distorted payload and $I \neq I'$: the integrity is not satisfied and m can be discarded. Certain correction algorithms also allow to reconstruct the original message I by iteration on I' and $Ecc(I)$, then the error can be corrected. Representatives in this category are Block Codes Lin and Chen (2005) and Convolutional Codes Bai et al. (2007).

Hash functions: A function $h : \{0,1\}^m \mapsto \{0,1\}^n$, $n << m$ is a hash function if it satisfies the following properties.

- **Uniqueness.** For each piece of information $I \in \{0,1\}^m$, the hash code $h(I) \in \{0,1\}^n$ is unique. A sufficient condition is that the function h is injective, i.e., $h(I_1) = h(I_2) \Rightarrow I_1 = I_2$ for all $I_1, I_2 \in \{0,1\}^m$.
- **Negligible collision probability.** In practice, uniqueness is rather unfeasible to achieve. Instead, the probability of code collision is constrained, *i.e.* the probability of the event $[h(I_1) = h(I_2) \Rightarrow I_1 \neq I_2]$ is requested to be negligible.
- **Hardly invert-able.** By applying exhaustive algorithms, like brute-force, for a given code $h(X)$, the complexity of finding the piece of information $I \in \{0,1\}^m$ such that $h(I) = h(X)$ is greater than polynomial.

The properties offered by Hash functions ensure suitable levels of protection against integrity attacks in particular by algorithms based upon secret keys. The application of hashing schemes is as follows. Any information I is submitted in a message including the code $h(I)$, *i.e.* $m = (I, h(I))$. As for the correction codes, the integrity of the received message $m' = (I', h(I))$ is verified by re-calculating the hash code of the payload I' and by comparing it to the original code: $h(I') = h(I) \Rightarrow I' = I$. If the codes differ $h(I') \neq h(I)$ then an integrity issue occurred and the payload is corrupted/attacked $I' \neq I$. Some representatives in this category are the SHA family (128, 256 and 512 bits) Bakhtiyor et al. (2020).

E.6.3 Non-repudiation

Definition 54 (Non-repudiation) *Given a component based system $S = \{C_j\}$ each component or channel C_j is allowed to perform or support a given set of (binary or unitary) operations $\{Op_i\}$. In a more detailed view, executing an operation Op_i over a piece of information I is determined and can be traced by (1) the structure of I and their contents (often called signature)* i.e. *$I = (I_1, \ldots, I_m)$, (2) the involved functions $f_i()$, $i = 1 \ldots m$ (if any) and the knowledge K owned by the component and used to build the signature, and (3) the meta-data related to the operation execution,* i.e. *initial time t_o, duration δ, final time t_f, etc. In the case the executed operation involves an exchange between components,* e.g. *via a channel, the meta-data can also include the channel identification C_{ch}, the emitter component C_o, the intended receiver C_d, etc. For a given component C_j, an event e is a tuple including an information signature I, the operation Op_i concerned by the event, the time set T stamping the operation execution, and the set of identifiers ID of the involved components.*

$$e := (C_j, I, Op_i, T, ID)$$

The non-repudation of an event e_f by the component C_j is ensured, if there exist a subset of components $C_l \in S$ with associated knowledge K_l, $l = 1 \ldots n$, and an ordered sequence of events $(e_1, \ldots, e_n)$ such that e_f is part of the sequence and once e_n occurred, the involved components $\{C_l\}$ own the same common knowledge $I_{nr} = (C_j, e_f)$ (or can compose it), *i.e.* $I_{nr} \in K_l$, $l = 1 \ldots n$, meaning that C_j can not reject involvement in e_j:

$$\forall e_f, C_j, \exists (e_1, \ldots, e_n) \Rightarrow I_{nr} = (C_j, e_f) \wedge I_{nr} \in K_l, l \in \{1, \ldots, n\}$$

The assessment of the non-repudation property mainly depends upon the system architecture, its configuration and the intended threats it should face. Overall, a non-repudiation scheme or protocol is often built upon other properties like integrity, confidentiality and authenticity of the information and components involved in the events $(e_1, \ldots, e_n)$. Assuming that referred properties are ensured, then the information conveyed in the events become legitimate. Then, the assessment of non-repudiation is reduced to ensure that the knowledge, sequentially acquired by the components, is consistent and sufficient to build the common agreement $I_{nr} = (C_j, e_f)$:

$$[e_1 \rightarrow I_1 \in K_1, \ldots, e_n \rightarrow I_n \in K_n] \Rightarrow I_{nr} = (C_j, e_f) \wedge [C_1, K_1] \rightarrow I_{nr} \wedge \ldots \wedge [C_n, K_n] \rightarrow I_{nr}$$

In other words, each event occurrence e_l implies that the piece of information I_l is added to its knowledge K_l which is represented by $e_l \rightarrow I_l \in K_l$. Once the sequence of events is finished, each component C_l is able to rely upon own knowledge to produce the common agreement $I_{nr} = (C_j, e_f)$, what is represented by $[C_l, K_l] \rightarrow I_{nr}$. Such agreement stands as proof of involvement of C_j in e_f.

E.6.4 Authenticity

Definition 55 (authenticity) *Authenticity is related to the occurrence of two ordered events, $e_o = (C_o, I_o, Op_o, T_o, ID_o)$ and $e_d = (C_d, I_d, Op_d, T_d, ID_d)$, where $Op_o = send_{C_o}(I_o)$, $T_o < T_d$, $Op_d = receive_{C_d}(I_d)$ and the meta-information are as follows:*

$$ID_o = \{sender : C_o,\ receiver : C_d,\ inf : I_o\},\ \wedge ID_d = \{sender : C_o,\ receiver : C_d,\ inf : I_d\}$$

If e_o and e_d are not consecutive, some intermediary events happen in-between what is denoted by $(e_o, \ldots, e_d)$.

Then, the authenticity of the events e_o, e_d is ensured if once the intended receiver C_d is reached, as an outcome of the sequence of operations $(Op_o, \ldots, Op_d)$, the (last) claimed sender truly corresponds with the origin C_o:

$$(e_o, \ldots, e_d) \Rightarrow send_{C_o}(C_d, I_o) \wedge \ldots \wedge Op(C_j, I_j) \wedge \ldots \wedge receive_{C_d}(C_o, I_d)$$

The authenticity property is often referred as authenticity of subjects (or parties). A second type of authenticity is related to the information I_o, I_d respectively involved in e_o, and e_d. Indeed, the authenticity of information exchanged is ensured if after the occurrence of the sequence of events, $I_d = I_o$ and C_o was legitimate to produce I_o, *i.e.*, $[C_o, K_o] \rightarrow I_o$. This is also known as data origin authenticity:

$$(e_o, \ldots, e_d) \Rightarrow [C_o, K_o] \rightarrow (C_d, I_o) \wedge \ldots \wedge [C_d, K_d] \rightarrow (C_d, I_d) \wedge I_d = I_o$$

As for previous security attributes, the assessment of authenticity depends upon the system architecture and configuration. In a hostile environment, the assessment of authenticity often requires ensuring the confidentiality and integrity of exchanges in the first place, which even if necessary, might be insufficient. Whereas a protocol or scheme ensuring authenticity involves both sender and receiver components, the assessment of authenticity is essentially conducted on the receiver's side. Some relevant mechanisms used to deploy and ensure authenticity are listed below.

Message Authentication Codes (MAC): These algorithms operate under a symmetric cryptography scheme what implies that a set of components $\{C_i\}$ should share the same secret symmetric key k and the algorithms $encrypt()$ and $decrypt()$ as introduced in Section E.6.1. A *MAC* algorithm exhibit the same properties of a hash function, as introduced in Section E.6.2. However, *MAC*s additionally rely upon the secret key k, *i.e.* for a piece of information I the message for authentication is composed as $m = (I, MAC(I,k))$. Upon reception, the message $m' = (I', MAC(I,k))$ is authenticated by re-computing the *MAC* of the payload I' using the secret key k. Only if $MAC(I',k) = MAC(I,k)$ the authenticity of the message is granted. Given the properties of the MAC algorithms, only the components $\{C_i\}$ owning the key k are able to compose and successfully verify the messages which ensure authenticity for $\{C_i\}$. Representatives in this category are HMAC Zhou et al. (2020).

Public Key Signatures (PKS): Signatures work under an asymmetric cryptography scheme what means that the set of components $\{C_i\}$ respectively have a set of secret and public keys $\{(k_{si}, k_{pi})\}$ and deploy the asymmetric encryption algorithms as introduced in Section E.6.1. The signature algorithms own strong properties as the hash functions introduced in Section E.6.2. Similar to their symmetric counterparts, an asymmetric signature S relies upon the secret keys $\{k_{si}\}$, *i.e.* for a piece of information I the message composed by a component C_i is given by $m = (I, S(I, k_{si}))$. The message can be validated by any other component or party owning the public key k_{pi} of C_i. Indeed, upon reception of the message $m' = (I', S(I, k_{si}))$, the authentication process requires to validate the signature with the respective public key k_{pi}, *i.e.* $S^{-1}(I', k_{pi}) = S(I, k_{si})$. If the validation fails, the payload is not authentic. Algorithms based upon the Public Key Infrastructure Gao et al. (2004) are representatives in this category.

E.6.5 ML Model security

By definition, information about training data is encoded in a model itself: a model can be thought of as an aggregated understanding of a scenario or task, derived from analyzing many examples of that scenario. Techniques exist by which an adversary can infer aspects of this information, for example the reconstruction of training data examples (so-called model inversion), which is a violation of confidentiality and potentially privacy if a model has been trained on personal data. Similarly, querying a model can reveal information about the model itself, which may constitute proprietary leakage. This can be particularly harmful where a business model is built around a well-trained model.

In an adversarial context, ML models open up new attack vectors compared to classical software: as shown in fig.E.4, which illustrates the paradigm shift brought about by AI components in the cybersecurity of digital systems, vulnerabilities can be exploited at the level of the different elements present in the AI processing chain, multiplying the potential threats and then the global risk of failure.

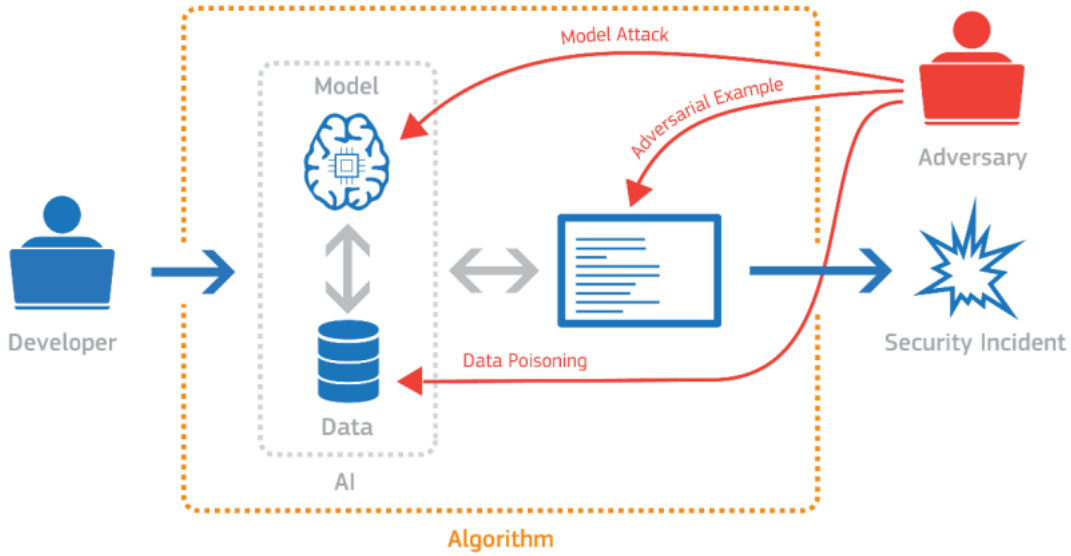


Figure E.4: Paradigm change in the ML model cybersecurity Hamon et al. (2020)

E.7. Maintenability

E.7.1 Motivation and definition

Assessing the maintainability of an AI-system involves evaluating its ability to be updated, enhanced, and fixed over time. AI-based maintenance activities can be classified as:

- **Corrective maintenance** – costs due to modifying AI-based system to correct issues discovered after initial deployment;
- **Adaptive maintenance** – costs due to modifying an AI-based solution to allow it to remain effective in a changing environment;

- **Preventive maintenance** – costs due to improving or enhancing an AI-based solution to improve overall performance ;
- **Enhancements** – costs due to continuing innovations.

Since maintenance costs eclipse other AI engineering activities by a large amount, it is imperative to answer the following question: How maintainable is our AI-based system, really? The answer to this question is non-trivial and requires further understanding of what does maintainability mean and induce.

Definition 56 (Maintainability) *Maintainability represents the ability of extending/improving a given system while maintaining its compliance with the unchanged requirements Mamalet et al. (2021).*

Thus, maintainability is the ease with which an AI system can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment. Some key factors have to be considered for AI system maintainability:

- **Modularity**: Consider whether the system is divided into modular components with clearly defined boundaries. Modularity makes the system easier to maintain, as updates or fixes can be focused on specific components without affecting the whole system.
- **Documentation**: Assess the completeness and quality of AI system documentation. Good documentation makes it easier to update or fix components by helping users understand the system's architecture, algorithms and data flows.
- **Version Control**: Evaluate whether the system uses version control to track changes. Version control enables developers to revert to previous versions, compare changes and work together efficiently.
- **Readability and code quality**: Check the clarity, readability and adherence to coding standards of the code. It is easier to maintain and debug well-written code with meaningful variable names, comments and proper structure.
- **Testing and Debugging Support**: Check that AI systems have extensive testing and debug tools. Test coverage ensures that changes or updates won't break existing features, and debug tools help identify and resolve issues effectively.
- **Dependency Management**: Assess how the AI system handles dependencies on external libraries, frameworks or APIs. Proper dependency management helps ensure that the system remains serviceable even as external components develop or become obsolete.
- **Error Handling and Logging**: Analyze system error handling mechanisms and logging. Effective error handling can help identify and resolve problems by providing detailed error messages. Detailed reports help diagnose problems during maintenance.

E.7.2 Maintenability assessment

To study AI system maintenance, few characteristics are analyzed:

- **Modularity** - *Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.*ISO/IEC 25010 (2011b)
- **Reusability** - *Degree to which an asset can be used in more than one system, or in building other assets.*ISO/IEC 25010 (2011b)

- **Analysability** - *Degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified.*ISO/IEC 25010 (2011b)
- **Modifiability** - *Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality.*ISO/IEC 25010 (2011b)
- **Testability** - *Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.*ISO/IEC 25010 (2011b)

Modularity measure ISO/IEC 25023 (2015) via coupling of components, is the ratio of completely independent components in a system to the number of components that must be independent. An independent component has zero impact on other parts of the system during maintenance. The standard also adds another software-specific measure, relating to the adequacy of cyclomatic complexity, computed as the proportion of all software modules that have an acceptable cyclomatic complexity. Acceptable cyclomatic complexity depends on the type of system.

There is no single metric that can accurately capture the notion of maintainability of an application. Oman and Hagemeister (1992) introduced a composite metric for quantifying software maintainability to help predict the maintainability of the application using the Halstead Volume (effort or volume), Cyclomatic Complexity, Total SLOC (source lines of code) and Comments Ratio. The metric originally was calculated as follows:

$$MI = 171 - 5.2 \ln V - 0.23G - 16.2 \ln L + 50 \sin \sqrt{2.4C}$$

Where:

- V is the average Halstead Volume per module
- G is the average Cyclomatic Complexity per module
- L is the average number of Source Lines of Code (SLOC) per module
- C is the average number of comment lines per module

Halstead Volume focuses on the readability of the code by providing us information about amounts of operators and operands needed for its execution. It's part of many other Halstead metrics introduced by Maurice Howard Halstead in 1977 Halstead (1977). It helps us understand how much information the reader of code has to absorb to understand its meaning. Halstead volume is calculated by using the Halstead Length (sum of all operators and operand occurrences) and Halstead Vocabulary (sum of distinct operator and operand occurrences). Because a lower score is ideal, we should target for concise and shorter pieces of code, thus avoiding noise Hariprasad et al. (2017).

Cyclomatic Complexity corresponds to a score calculated as the sum of linearly independent paths in a section of code. Thomas J. McCabe developed this metric in 1976 to represent the control flow of a program. Control flow depicts a program with a graph consisting of nodes and edges. Each node can be interpreted as a program command, while edges represent connections or paths between those commands. The resulting score identifies the parts of code that need to be covered by unit tests. It also helps testers determine independent path executions; testers can then target having the number of test cases equal to the cyclomatic complexity score. This score

defaults to 1, and we should aim to be as low as possible, avoiding too many branching and logic inside a single code piece.

Lines of Code measures the total number of executable code lines while ignoring comments and empty lines. Like the first two measurements, the lower score is better, thus preventing adding too much logic into one place.

The use of the previous formula meant that it ranged from 171 to an unbounded negative number. As code tended toward 0, it was clearly hard to maintain it and the difference between code at 0 and some negative value was not useful. As a result of the decreasing usefulness of the negative numbers and a desire to keep the metric as clear as possible, we decided to treat all 0 or less indexes as 0 and then re-base the 171 or less range to be from 0 to 100. For this reason, the used formula is:

$$MI = \max\left(0, 100.\frac{171 - 5.2\ln V - 0.23G - 16.2\ln L + 50\sin\sqrt{2.4C}}{171}\right) \tag{E.5}$$

The use of this metric is debatable but could be used in conjunction with the above metrics or our team could create a compound metric based on the above dimensions! As long as the metric makes sense to our team and our organization you're free to create our own, albeit meaningful, metrics.

More modular systems are easier to maintain as the addition of a component or an update to an existing one tends to have a limited impact on the rest of the system.

F. Usability

Usability: Can a human use it easily?

F.1. Motivation

Human-centered quality ISO 9241-210 (2019) concerns which requirements for usability, accessibility, user experience and avoidance of harm from use are met. These requirements, can only be managed to the extent that they can be controlled by designed aspects of the interactive system.

From such perspective, trustworthy AI should possess the properties of usability, and explainability. Specifically, AI-based systems should not cease operation at inappropriate times (e.g. at times when the lack of output could lead to safety risks), and these programs or systems should be easy to use for people with different backgrounds. Last, but not least, trustworthy AI must allow for explanation and analysis by humans, so that potential risks and harm can be minimized, and human users can remain empowered. In addition, trustworthy AI should be transparent so people can better understand its mechanism.

Usability and utility are equally important and together determine whether something is useful: It matters little that something is easy if it's not what you want. It's also no good if the system can hypothetically do what you want, but you can't make it happen because the user interface is too difficult. To study a design's utility, you can use the same user research methods that improve usability.

- Definition of Utility = whether it provides the features you need.
- Definition of Usability = how easy and pleasant these features are to use.
- Definition of Useful = usability + utility.

F.2. Usability definitions

Usability is a limited concern compared to the larger topics of system acceptability, which is essentially whether the system is good enough to satisfy all the needs and requirements of users and other potential stakeholders. Usability applies to all aspects of a system with which a human being can interact, including installation and maintenance procedures Nielsen (1994).

Definition 57 (Usability) *Usability is the degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. ISO/IEC 25010 (2011b)*

In ISO 9241-210 (2019) usability extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

Usability describes the quality of user experience across websites, software, products, and environments. Usability is a component of user experience design. Usability issues are critical in

many AI-based critical systems, where a human works with the system and apply results, and when the AI system serves as the user interface for the user (as with chatbot systems). AI is also applied in some systems to build a computer model of the user (e.g. digital twin), which is then used to help anticipate the user's needs and optimize the interface (as in computer-aided instruction systems and adaptive systems) Mattioli et al. (2023b).

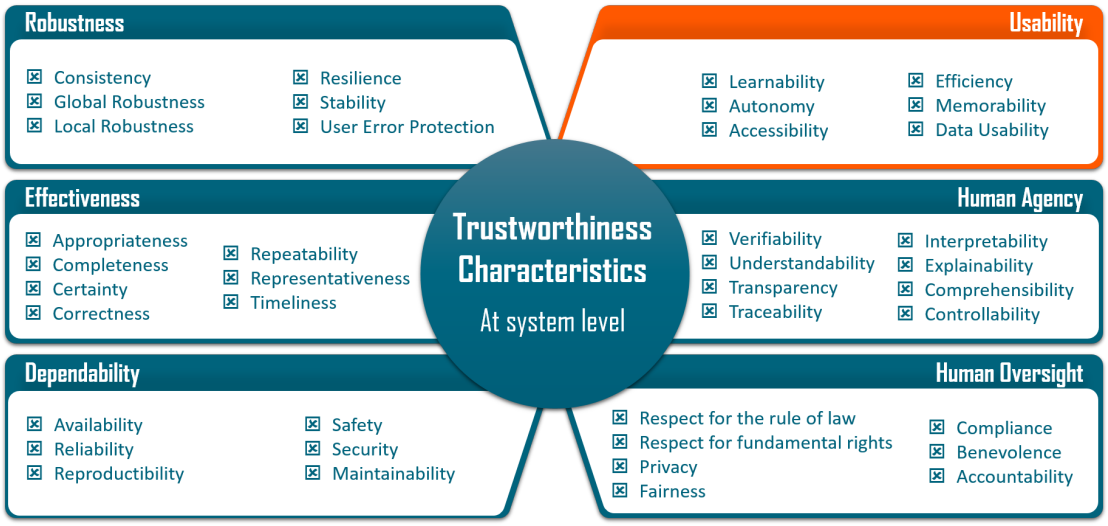


Figure F.1: Major characteristics of usability

To measure/evaluate usability, several survey and questionnaire were proposed in the literature, like for example SUMI (Software Usability Measurement Inventory) Kirakowski et al. (1993), SUS (System Usability Scale) Brooke (1996), USE (Usefulness, Satisfaction, and Ease of us) questionnaire Lund (2001), or recently Bot Usability Survey Borsci et al. (2022)

Usability is not a single, one-dimensional property of a user interface. It has multiple components and is at the beginning associated with these five usability attributes Nielsen (1994)

Usability is a quality attribute that assesses how easy user interfaces are to use. The word "usability" also refers to methods for improving ease-of-use during the design process.

- Learnability: How easy is it for users to accomplish basic tasks the first time they encounter the design?
- Autonomy: Is the system capable of performing all tasks in all contexts?
- Accessibility: How the system could be used by anyone in various contexts? goal in a specified context of use
- Efficiency (of use): Once users have learned the design, how quickly can they perform tasks?
- Memorability: When users return to the design after a period of not using it, how easily can they reestablish proficiency?
- Satisfaction: How pleasant is it to use the design?

F.2.1 Learnability

The system should be easy to learn, so that the user can rapidly start working with it Nielsen (1994).

Definition 58 (Learnability) *Learnability is defined as the degree to which a product or system enables the user to learn how to use it with effectiveness, efficiency in emergency situations. ISO/IEC 25010 (2011a)*

In software domain, learnability is capability of the software product to enable the user to learn its application. It is in some sense the most fundamental usability attribute, since most systems need to be easy to learn, and since the first experience most people have with a new system is that of learning to use it.

Learnability is described as covering the time to mastery and error avoidance or recovery. It is is a quality attribute that evaluates 'ease-of-learning' and assesses user efficiency and effectiveness in carrying out specific tasks. It is evaluated by counting errors and measuring the time that is needed to perform a task as well as the time that is needed for error recovery. In other vision, learnability could be operationalised as the time that a new user needs to reach a predefined level Mbelekani and Bengler (2023)

F.2.2 Autonomy

The term "autonomy" pertains to the task being performed. In other words, a system is said to be autonomous with respect to a specific task or set of tasks, not all tasks. In fact, there is no such thing as a fully autonomous system that is capable of performing all tasks in all contexts. Thus, this recognition is what makes autonomy tractable. As important, the different types of human-machine relations also need to be clarified, so that their implications are clear. Madni and Madni (2018) Measuring the autonomy of an AI-based system is a complex process. It primarily relies on approximations. The measurement involves quantifying the autonomy of algorithmic systems for their future behavior based on the retrospective effect of their past behavior or current situations. Nevertheless, quantifying autonomy enhances the performance of AI-based systems. It aids in tracking, analyzing, and predicting the performances of AI-based systems. Consequently, autonomy measurement is a fundamental operation for the distribution and adjustment of autonomy Mostafa et al. (2019).

Many autonomy measurements are proposed in the literature, we can cite the one of Hexmoor et al. (2003) who proposes a quantitative measurement methodology for assessing various aspects of autonomy exhibited by agents in a multi-agent environment. The aspects considered include autonomy in decision-making and action, autonomy of user agents, and the distribution of autonomy within a group and across different groups of agents, accompanied by the necessary metrics.

The autonomy of the agent a regarding the agent set S is: $\underset{S}{\overset{a}{A}} = \frac{\overset{a}{P}}{\frac{\underset{a}{S}}{\underset{a}{P}}}$

The autonomy of the agent set S is:

$$\dot{\lambda} = -\frac{\partial H}{\partial SOC} = 0$$

$$A^S = \sum_{a \in S} A^a_{S\{a\}}$$

The autonomy of the agent set S regarding an agent b is:

$$A_b^S = \frac{\sum_{a \in S} A^a_{(S \cup \{b\}) \setminus \{a\}}}{A^S}$$

A is a set of agents, $S \subset A$, $a \subset S$, $\underset{a}{\overset{a}{P}}$ represents an agent a performance according to a performance scale ratio measure when only a is present, $\underset{S}{\overset{a}{P}}$ represents the agent a performance when S agents are present.

In this case, agents acquire autonomy based on their performance. An agent that performs a wide range of activities is said to have higher autonomy and vice versa.

Roehr and Shi (2010) consider that autonomy distribution and adjustment depend on trust and self-confidence, gauged through human observation of the robots' performance. Trust calculation is restricted to the likelihood of previous task successes, where each task aligns with a particular level of autonomy, outlined as follows:

$$J(a) = \omega P(\frac{S}{\varepsilon}, a) + (1 - \omega) P(S)$$

Self-confidence is confined to the probability of successes

$$P(S)$$

given a task as follows:

$$P(s) = \frac{number\ of\ successful\ task\ execution}{number\ of\ execution}$$

where J is a trust function, a is an autonomy level, w is a weight as min $(1, \frac{number of samples}{minimum required samples})$, $P(S)$ is a prior probability of success and ε is a command and environment parameters.

F.2.3 Accessibility

Definition 59 (Accessibility) *Accessibility is the degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use ISO/IEC 25010 (2011a).*

Accessibility contains all the attributes that allow physically impaired users to use the system. This sub-characteristic is decomposed into the following attributes: Subtitle Support, which refers to the system's capability to provide adequate subtitles for hearing impaired players; and Magnifier Support, which concerns the applications's capability to provide adequate sized subtitles for visually impaired usersFernandez et al. (2012).

F.2.3.1 Data accessibility

The following definitions are the results of different Confiance.ai convergence workshops:

Definition 60 (Dataset accessibility) *Degree to which the dataset can be accessed in a specific context of use.*

F.2.4 Efficiency

Definition 61 (Efficiency) *Efficiency is the relationship between the result achieved and the resources used ISO 9000 (2015b).*

The principal requirement is that the system should be efficient to use, so that once the user has learned the system, a high level of productivity is possible . In Frøkjær et al. (2000), efficiency is defined as the relationship between users' goal attainment accuracy, completeness and the resources they invest to achieve those goals.

Efficiency can be measured by various indicators, such as task completion time, learning time, completion rate efficiency, *etc.,* in terms of task time. That is, the time (in seconds and/or minutes) the participant takes to successfully complete a task. The time taken to complete a task can then be calculated by simply subtracting the start time from the end time as shown in the equation below:

$$Task\_Time = End\_Time - Start\_Time$$

Efficiency can then be calculated in one of 2 ways:

- Time-Based Efficiency:

$$Time\_Based\_Efficiency = \frac{\sum_{j=1}^{R}\sum_{i=1}^{N}\frac{n_{ij}}{t_{ij}}}{NR}$$

  Where: N is the total number of tasks (goals); R is the number of users n_{ij} represents the result of task i by user j; if the user successfully completes the task, then $N_{ij} = 1$, if not, then $N_{ij} = 0$; t_{ij} represents the time spent by user j to complete task i. If the task is not successfully completed, then time is measured till the moment the user quits the task

- The overall relative efficiency uses the ratio of the time taken by the users who successfully completed the task in relation to the total time taken by all users. The equation can thus be represented as follows:

$$Overall\_Relative\_Efficiency = \frac{\sum_{j=1}^{R}\sum_{i=1}^{N} n_{ij}t_{ij}}{\sum_{j=1}^{R}\sum_{i=1}^{N} t_{ij}} \times 100\%$$

For example, in Frøkjær et al. (2000) the authors are focused on task completion time as the primary indicator of efficiency, in this study the task est to solve 20 information retrieval tasks concerning programming problems. In Teruel et al. (2014) the authors used task time and Completion rate efficiency (effective correct tasks per hour) and in Xu et al. (2023) the authors used task completion time to evaluate human AI collaboration efficiency.

F.2.5 User satisfaction

Definition 62 (User satisfaction) *User satisfaction refers to the level of comfort and positive attitudes users have towards a system Frøkjær et al. (2000).*

It can be assessed using attitude rating scales like SUMI Kirakowski et al. (1993), USE questionnaire Lund (2001), or recently Bot Usability Survey Borsci et al. (2022). In this study, preference is considered as the main indicator of user satisfaction Frøkjær et al. (2000).

In term of requirement, the system should be pleasant to use, so that users are subjectively satisfied when using it; they like it.

In addition to these attributes, others have been added to cover new requirements

F.2.6 Universal Design

Universal design, also known as "design for all", can be defined as the design of products and environments that can be used and experienced by people of all ages and abilities, to the greatest extent possible, without adaptation Mace (1990). It encompasses the deliberate endeavor to incorporate and addresses the broadest array of end user needs during the entire development process of products or services Akoumianakis and Stephanidis (1989). In architecture universal design is defined, rather than solely meeting the basic legal requirements that necessitate a few specific features for individuals with disabilities, there is an opportunity to design the majority of manufactured goods and building components in a manner that is accessible to a wider spectrum of individuals. This includes children, elderly individuals, people with disabilities, and people of varying sizes Akoumianakis and Stephanidis (1989). and recently, Universal Design is defined specifically for AI systems that should address the widest possible range of users and, in particular, be accessible to them. Therefore, universal design principles should be considered during the planning and development of an AI system, especially taking into account and, if possible, involving, potential end-users with special needs. It should be ensured that no group of people is disproportionately affected by the results of the AI system [Fraunhofer Institute for Intelligent Analysis and Information Systems IAIS 2021]. Universal design encompasses the concept of ensuring accessibility and usability for all individuals to the highest degree feasible. It entails considering and incorporating inclusive design principles in every aspect and area, aiming to make elements and spaces accessible and usable by everyone. This approach requires careful and deliberate planning and design implementation throughout all stages of a design project Mace (1990).

F.2.7 Appropriateness recognisability

Definition 63 (Appropriateness recognizability) *Appropriateness recognizability is described as the degree to which users can recognize whether a product or system is appropriate for their needs ISO/IEC 25010 (2011a)*

This covers the supplier's marketing literature. The availability of formal training and on-line help assist in several areas, but they are features of the supplier's solution, and it is the resulting usability of the product that is important. Moreover, appropriateness recognizability contains the attributes of the system that ease the understanding of the application. These attributes are: Visibility, which focuses on visual recognisability, and legibility by measuring the ease of perception of the application's graphic information, it can be measured by taking into account percentage of screen usage; Interface Simplicity and Control Simplicity, which evaluate the complexity of the graphical user interface and the application controls and it can measured by total number of GUI elements and total number of control Mappings, respectively; and Consistency, which focuses on the degree of similitude and coherence between the elements of the system and it can measured by the ratio of similitude between screens Fernandez et al. (2012).

F.2.8 User interface aesthetics

Communication between any system and a human being involves a user interface. Today, these interfaces are an integral part of everyday human life, and this trend is unlikely to diminish. Firstly, user interfaces are designed to attract users' attention and, secondly, to enable them to consult it easily and efficiently. While content is of great importance, form is just as essential.

Indeed, appreciating design is the first interaction people have with an interface when they first encounter it. According to some studies, it has been found that this interaction can be completed in less than half a second. Additionally, there is a compelling indication that the perceived quality of a User Interface (UI) positively affects users' perceptions of the system's usefulness and usability Zen and Vanderdonckt (2014). This concept is known also as User interface aesthetics witch defined in ISO/IEC 25010 (2011a) by the degree to which a user interface enables pleasing and satisfying interaction for the user. Aesthetics in Computer Science refers mainly to UI design, where requirements can typically concern the overall usability of the system as well as specific aesthetic considerations. Yet, a distinction can be drawn between aesthetics and usability Bevan (2008). As metrics or measures, 14 were brought up: balance, equilibrium, symmetry, sequence, cohesion, unity, proportion, simplicity, density, regularity, economy, homogeneity, rhythm, order and complexity. All those measures describe, how objects are placed in composition and the size of the objects Ngo et al. (2003).

G. Human Agency and Oversight

G.1. Introduction

While Artificial Intelligence is getting more and more efficient, and is in many ways surpassing human intelligence, it may be tempting to rely heavily on highly automated systems for making decisions and recommendations with high risks consequence. Still, Artificial Intelligence is not Human Intelligence, and AI systems need to be human-centric and base their behaviors on human needs, intentions and wills.

To ensure trustworthy AI, it is important to go beyond the AI model itself (inputs, features and outputs) and consider dynamics of the model interacting with the overall system, including end-users. Proper oversight mechanisms need to be ensured, which can be achieved through human-in-the-loop, human-on-the-loop, and human-in-command approaches. From such perspective, trustworthy AI should be both usable and explainable, meaning that it should not stop working at inappropriate times (which could create safety risks) and should be user-friendly for individuals with diverse backgrounds. Moreover, trustworthy AI must allow for human explanation and analysis to mitigate risks and empower users, as well as transparent to promote understanding of its workings mechanism.

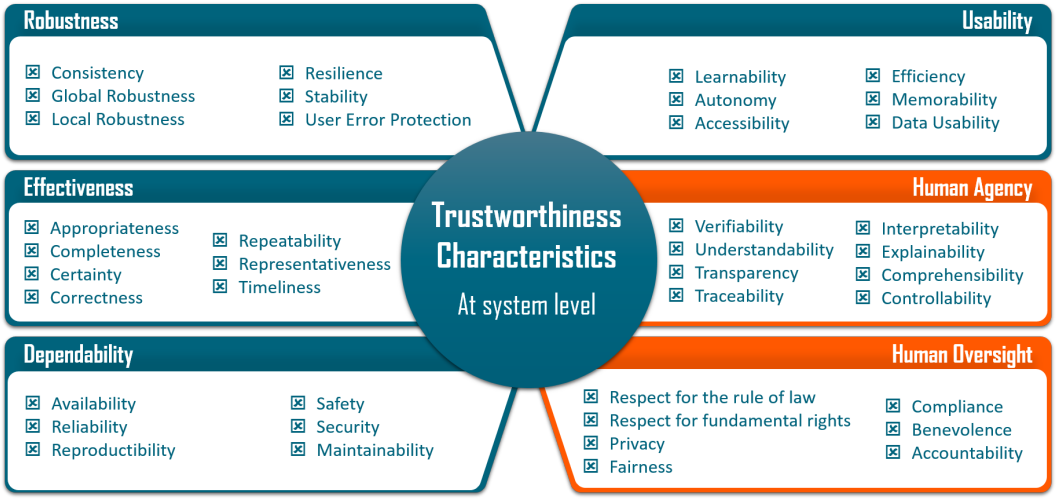


Figure G.1: Focus on human agency and oversight characteristics

The concept of human agency is based on the principle that it is up to human beings to make their own choices and to set these choices on their environment. Therefore, the level of automation/autonomy of AI systems should not be at the expense of individual autonomy, but should serve individuals, humanity and the common good. In this context, users should be able to make informed autonomous decisions regarding AI systems. They should be given the knowledge and tools to comprehend and interact with AI systems to a satisfactory degree and, where possible, be enabled to reasonably self-assess or challenge the system. AI systems should support individuals in making better, more informed choices in accordance with their goals. AI systems can

sometimes be deployed to shape and influence human behavior through mechanisms that may be difficult to detect, since they may harness sub-conscious processes, including various forms of unfair manipulation, deception, herding and conditioning, all of which may threaten individual autonomy. The overall principle of user autonomy must be central to the system's functionality. The crucial principle here is the right not to be subject to a decision based exclusively on automated processing where that decision has legal consequences or a significant impact on users in a similar way.

Human oversight in the context of machine learning refers to involving humans in controlling the decision-making process of AI systems. To ensure the ethical and responsible use of machine learning models, it involves human intervention, monitoring and accountability. Thus, it is at the heart of human agency, as oversight implies that individuals can comprehend what the system is doing and are able to intervene when necessary. Oversight could be fully human, but it is most likely that it will be hybrid, meaning a layer of automated oversight, with a possibility for individuals to bypass or takeover the system.

G.2. Explainability

Explainability is at the heart of trustworthy AI and must be guaranteed to develop AI systems that empower and engage people across scientific disciplines and industries such as AI engineers and data/AI scientists, end-users and auditors. A human-machine symbiosis, where humans retain responsibility for decisions but rely on machine aids, is needed in many practical decision-making scenarios.

Hamon et al. (2020) and Delseny et al. (2021) define explainability as *the ability of a model to provide elements to explain the results to a human in understandable terms*.

In Mamalet et al. (2021), it is defined as *the extent to which the behavior of a ML model can be made understandable to humans.*

One of the issues that hinders the establishment of common grounds is the interchangeable misuse of interpretability and explainability in the literature. There are notable differences among these concepts. In this section, we will clarify all.

G.2.1 Explainability

Explainable artificial intelligence (XAI) is a subfield of artificial intelligence (AI) that provides explanations for the predictions, recommendations, and decisions of intelligent systems.

Some XAI definitions are focused solely and narrowly on the technical concepts while others focus only on the broad social and political dimensions. Lacking "*a theory of explainable AI, with a formal and universally agreed definition of what explanations are*", the fundamentals of this field are still being explored, often from different disciplinary perspectives. As such, a definition of XAI must encompass not just the techniques, as important and necessary as they are, but also the context within which XAI operates.

The US Defense Advanced Research Projects Agency (DARPA) description of XAI captures the breadth and scope of the field. The purpose of XAI is for AI systems to have "the ability to explain their rationale, characterize their strengths and weaknesses, and convey an understanding of how they will behave in the future" and to "enable human users to understand, appropriately trust, and effectively manage the emerging generation of artificially intelligent partners." XAI is

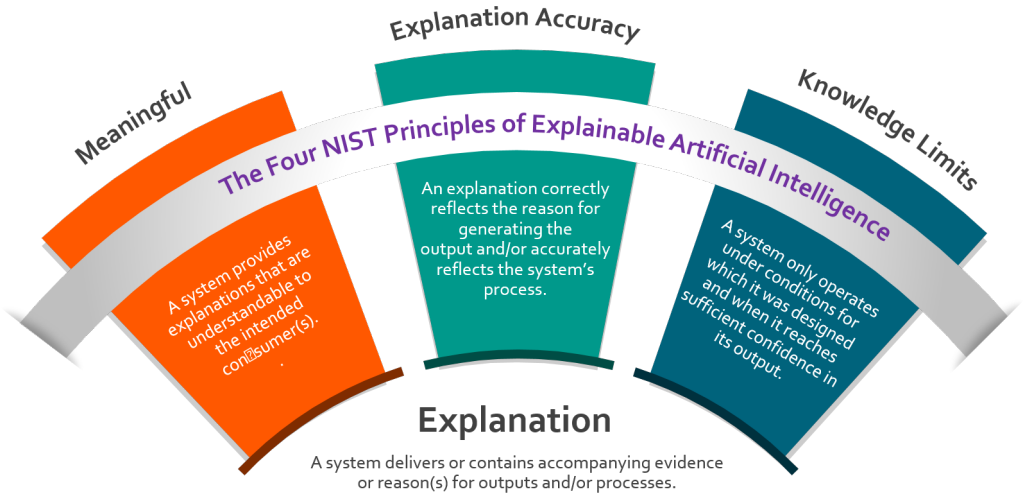


Figure G.2: the NIST four principles of XAI

needed to:

- generate trust, transparency, and understanding;
- ensure compliance with regulations and legislation;
- mitigate risk;
- generate accountable, reliable, and sound models for justification;
- minimize or mitigate bias, unfairness, and misinterpretation in model performance and interpretation; and
- validate models and validate explanations generated by XAI.

Thus, XAI is one of several properties that characterize trust in AI systems which needs a principled method that characterizes a good explanation from an AI system. First, the characterization needs to be human-centered, because humans consume them. Second, they need to be understandable to people. Third, explanations should correctly reflect the system's process for generating the output. To foster confidence in explanations, the system should indicate when it is operating outside its designed conditions. These core concepts of a good explanation are the basis for the NIST four principles of explainable AI Phillips et al. (2020) (Fig. G.2).

Explainability is by far one of the most rich and complex to assess feature in recent research concerning ML/DL topics. Several reasons justify such fact, in particular because explainability aims to provide answer as to why ML/DL algorithms succeed or fail, which is rather a challenge given their heuristic nature and intricacy. Explainability is also a high-level demand in several domains aiming to transfer safety-critical human-based tasks to autonomous systems, e.g. for accountability purposes, a basic explainability requirement for a self-driving vehicle being to sufficiently characterize the contribution of the ML/DL network branches during pedestrian/obstacle detection. Last, yet not the least, explaining the behavior of ML/DL models can be tihgtly coupled to solve apparent conflicts/inconsistencies between certain attributes/features. To illustrate this point, we recall the counter-intuitive properties of a ML/DL model referred in Szegedy et al. (2013) which are rather related to robustness, *e.g.* misclassification of statistically indistinguishable points, misclassification of the same points after adding new training data and updating model parameters, etc. Such apparently inconsistent outcomes make designers raise questions about ML/DL models' foundations and call for methods to characterize them. Thus, explain-

ability is a term used to encapsulate and refer to all previous needs and aims and is therefore a qualitative attribute of AI-based systems.

Definition 64 (Explainability) *Explainability deals with the capability to provide the human with understandable and relevant information on how an AI application is coming to its result.*

There are no unified methods or scales to evaluate explainability. Recent surveys, as the one offered by Xie et al. (2020), suggest that explainability can be decomposed by the methods used to evaluate it. A brief description of the main families found in literature is provided in the below.

- **Visualization methods** pursue the characterization of a ML/DL network by visual observation of the levels of activation/deactivation according to the input data and their influence in the classification performance, sensitivity, and other functional / structural properties. Representative instances in this family are:
  - ***Back-propagation*** helps to observe relevance of data in terms of the activation / deactivation gradients observed at different layers in the network during training, *e.g.* Activation Maximization Erhan et al. (2009), Deconvolution Zeiler and Fergus (2014), Layer-wise Relevance Propagation Montavon et al. (2017).
  - ***Perturbation-based*** methods provide means to observe and compare its impact in the network w.r.t. non-perturbed input, *e.g.* Occlusion Sensitivity Zeiler and Fergus (2014), Representation Erasure Li et al. (2016), Meaningful Perturbation Fong and Vedaldi (2017).
- **Distillation methods** aim to represent (distill) the knowledge encoded in the ML/DL network after training via a more human-readable format suitable for both user interpretation and logic/machine reasoning. Some representative instances in this family are:
  - ***Local Approximation*** methods mimic the input/output behavior of the target ML/DL model on smaller datasets, and using approximation functions, *e.g.* linear functions. Local Approximation methods are based upon the hypothesis that the ML/DL behavior can be better and more easily characterized on local areas rather than over the entire dataset, *e.g.* LIME Ribeiro et al. (2016), Anchors Ribeiro et al. (2018).
  - ***Model Translation*** methods aim to mimic input/output behavior of the target ML/DL model however considering the whole dataset over a symbolic model, *e.g.* Graph-based Zhang et al. (2018), Rule-based Harradon et al. (2018).
- **Intrinsic methods** search to integrate the means for explainability as part of the design of the ML/DL model. The explainability of ML/DL networks should be intrinsic and thus input/output behavior should be explicitly justified by the ML/DL model itself. Representative instances in this family are:
  - ***Attention Mechanisms*** rely upon contextual vector and attention mechanisms used to learn a conditional distribution over data inputs which provide an interpretation on the behavior of the weights of the operations of activation and deactivation, *e.g.* Single Modal Weighting Hendricks et al. (2016), Multimodal Interaction Anderson et al. (2018).
  - ***Joint Training*** consists in introducing an additional task in the ML/DL model, asides the original one, in charge of providing direct or indirect explanations for the main task behavior, *e.g.* Text Explanation Liu et al. (2018), Explanation Association Iyer et al. (2018).

Being a qualitative feature, explainability in turn requires criteria to evaluate the quality of explanations. This presupposes a non-negligible intervention of humans in the assessment process. Some methods proposed to evaluate explanations can be found in Xie et al. (2020). The evaluation of the appropriateness of explainability libraries and methods is performed per use case.

According to explainability methods, several metrics are identified Confiance.ai et al. (2022):

- **Faithfulness**: measures the degree to which an interpretation method accurately reflects the reasoning of the model it interprets. It is important to note that explanations provided by an unfaithful method can conceal any biases that exist in the model's judgments, which may result in unwarranted trust or confidence in the model's predictions. Faithfulness is calculated using the following formula Du et al. (2019): $Faithfulness = 1/N \sum(y_{x^i} - y_{x^i|a})$, where y_{x^i} is the predicted probability for a given target class using the original inputs, and $y_{x^i|a}$ is the predicted probability for the target class for the input with significant sentences/words removed. According to Arya et al. (2022), faithfulness is the inverse of the Pearson Product-Moment correlation and ranges from -1 to 1. A negative correlation of 1 indicates a perfect correlation, a positive correlation of -1 indicates the opposite, and 0 indicates no correlation. Faithfulness is calculated as follows: $Faithfulness = -\sigma_{xy}/(\sigma_x + \sigma_y)$, where σ_x^2 (resp. σ_{xy}^2) represents the variance of x (resp. the co-variance of (x,y)). $= \sqrt{E(xy) - E(x)E(y)}$ and $\sigma_x = \sqrt{E(x^2) - E(x)^2}$.

  Faithfulness is interchangeable with MuFidelity. This metric corresponds to the pearson's correlation coefficient between the predicted logics of each modified test point and the average explanation for only the subset of features Confiance.ai et al. (2022).
- **Monotonicity**: applies only to some explainable methods. It consists in progressively adding the values of x to a null vector, then looking if the probability of predicting the correct class with it is increasing Ribeiro et al. (2016).
- **Sensitivity**: measures the degree of explanation changes to subtle input perturbations using Monte Carlo sampling-based approximation Yeh et al. (2019).

G.2.2 Interpretability

Interpretability Delseny et al. (2021) relates to *the capability of an element representation (an object, a relation, a property...) to be associated with the mental model of a human being. It is a basic requirement for an explanation.*

To complete that definition, ISO/IEC DIS 22989 (2021a) defines such attribute as the *property of an AI system to express important factors influencing the AI system results in a way that humans can understand.*

Interpretability assesses how easily human can understand the internal workings of an AI system; interpretable explanations need to use a representation that is understandable to humans, regardless of the actual features used by the model Ribeiro et al. (2016). In the context of ML systems, interpretability is defined as the ability to explain or to present in understandable terms to a human Doshi-Velez and Kim (2017).

Fidelity measures how well the explanations provided accurately reflect the AI system behavior Yeh et al. (2019). Fidelity metrics measure the efficiency of the methods to explain models. Fidelity is also defined Plumb et al. (2020), when the explainer's output space is $(\varepsilon_s, (\varepsilon_s := (g \in G | g : X \to Y))$, the explanation is defined as a function $g : X \to Y$, and it is natural to evaluate how accurately g models f in a neighborhood $N_x : F(f,g,N_x) := E_{x' \sim N_x}[(g(x') - f(x'))^2]$

which refer to the neighborhood-fidelity (NF) metric. This metric is sometimes evaluated with N_x as a point mass on x, this version is called the point-fidelity (PF) metric. In NLP (Natural Language Processing) domain, the fidelity measures how well an interpretable method can approximate the performance of a black-box model. Underlying this definition is the assumption that a method that better approximates a black-box also must use a similar reasoning process to underlying model. As such, this last definition is a more specific form of faithfulness as applied to interpretability methods that construct models approximating an underlying black-box model.

Definition 65 (Interpretability) *Interpretability relates to the capability of an element representation (an object, a relation, a property...) to be associated with the mental model of a human being. It is a basic requirement for an explanation.*

G.3. Controllability

One way to know that something is wrong is through the feedback we receive by observing the system's properties from the exterior. If the system is not observable, it's hard to know where we need to take action to control it.

G.3.1 Definition

Definition 66 (Controllability) *Controllability is the property of an AI system which allows a controller to intervene in the functioning of the AI system ISO/IEC TS 8200 (2023).*

It stands for the ability level to avoid harm and is one of the parameters that determine the Safety Integrity. In this sense, controllability focuses on how to build advanced AI systems that do not harm humans Aliman and Kester (2019). ISO/IEC TS 8200 (2023) standard indicates that controllability can be estimated in relationship with the suitability, relevance and reliability of the control mechanisms designed. The definition of what is deemed "suitable" and "relevant" may depend on the context, and the standard does not provide details on that; reliability may require that appropriate testing procedures be designed. Controllability of an AI system may be an attribute that needs to be checked at different stages of the engineering workflow.

G.3.2 Motivation

Without adequate control mechanisms, AI systems can pose significant risks to human safety. For instance, an AI-powered robot or machine's decision might put human co-workers in danger if they happen to come in between a process or task. Similarly, autonomous vehicles with uncontrolled AI might make dangerous decisions on the road, placing the lives of road users (drivers and pedestrians) at risk. Considering the wide range of risks involved, it is essential to carefully assess and take proactive measures to ensure the safe integration of AI to avoid having uncontrolled and possibly very risky AI systems being developed without oversight.

G.3.3 Example

Below, controllability is illustrated through two examples from respectively the automotive and the military domains.

In Habli et al. (2016), the notion of controllability in the automotive domain is illustrated by considering the following vehicle features: an engine control system, an electric driveline incorporating in-wheel motors, and an air suspension system. Fuel economy, performance and cost and emissions legislation all contributed to engine control systems evolving from a mechanical design to a complex programmable control system. The early electronic control systems typically mimicked previous mechanical designs, while modern engine control systems use rather complex control algorithms expressed in the torque domain. This gives rise to a complex relationship between the acceleration pedal position and the resultant vehicle acceleration. Generally, layered monitoring strategies eGAS Working Group (2013) are employed to continuously monitor the relationship between the driver's torque demand (via the accelerator pedal) and the delivered engine torque (estimated from measured engine parameters); any hazardous discrepancy between demanded and delivered torque is mitigated by limiting the engine's performance or ultimately by shutting it down.

For an experienced driver, the longitudinal control task is generally a subconscious process, with the driver modulating the accelerator pedal position, based on their current driver strategy and given their knowledge of the current traffic environment to attain the desired vehicle acceleration. If vehicle acceleration (Complete Vehicle Behaviour) is too slow then the accelerator pedal can be pressed harder. Contrarily, if acceleration is too great then the accelerator pedal can be released, and once completely released the brakes applied. This attribute was particularly well demonstrated in the Dedicated Road Infrastructure for Vehicle Safety in Europe (DRIVE) project Jesty P.H (1993). The latter presented thus the notion of controllability from an automotive system safety perspective and defined it as the probabilistic attribute linking a hazardous event to an accident by indicating how likely the driver will be able to control the hazardous situation and hence avoid harm. The DRIVE project classified controllability into five levels linking it directly to integrity levels, as shown in Fig. G.3.

According to this classification, a hazard is categorized as "nuisance only" if there is a zero probability that a hazardous event becomes an accident. While, a hazard is categorized as uncontrollable if a human intervention cannot avoid harm and thus cannot control the hazardous situation Habli et al. (2016).

In the context of vehicle handling, controllability refers to the relationship between the driver's ability and the vehicle's handling qualities Abe (2015).

Moreover, controllability is tackled in the military domain. Indeed, according to Boardman and Butcher (2019), military systems must be able to function effectively and safely under a wide range of highly dynamic environments and use cases that are hard to anticipate or predict during the design phase. Such systems must also be resilient not only to failure, but also to uncertain, complex, and unpredictable events and situations where the dynamics of the military domain necessitate complex judgments with respect to acceptable actions based on rules of engagement, judgments, and international law over proportionality, legality and risk. Given this, the maintenance of Human Control through a combination of specification, training, design, operating procedures, and assurance processes is regarded as critical in many, if not all military systems.

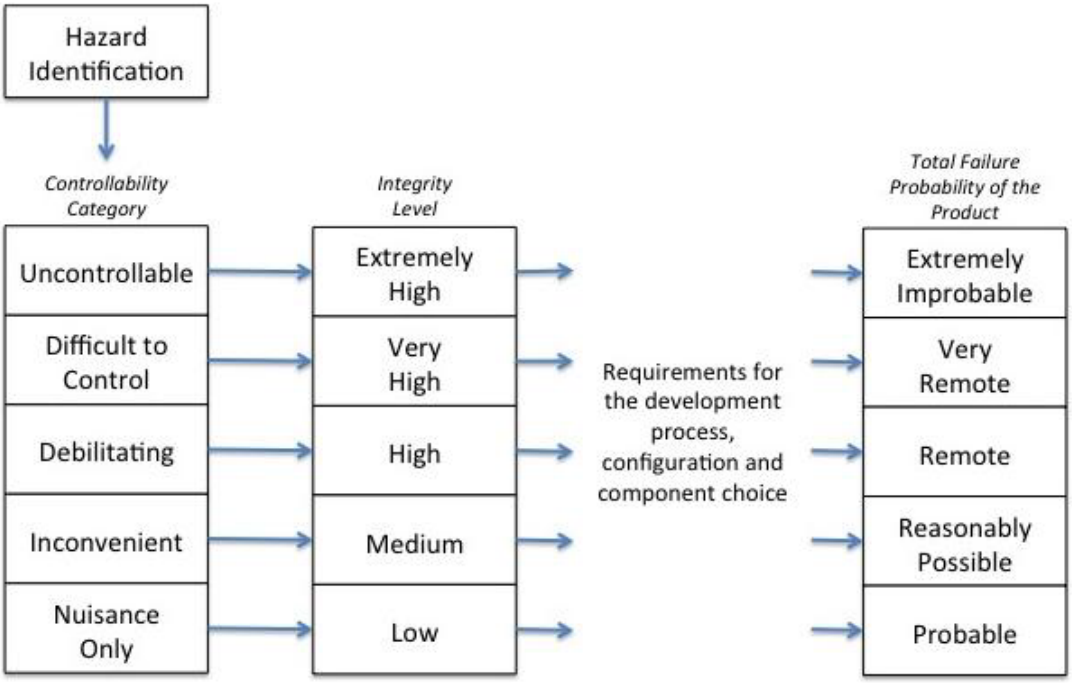


Figure G.3: Controllability and Integrity Categories from DRIVE Habli et al. (2016)

G.4. User controllability

G.4.1 Definition

User controllability raises the following question: How can humans remain in control of AI-based systems designed to perform tasks autonomously? Central to user controllability is Meaningful human control (MHC) referring to "the ability to make timely, informed choices to influence AI-based systems that enable the best possible operational outcomes" Boardman and Butcher (2019). The idea behind MHC is that humans should retain control and moral responsibility over autonomous systems. Although MHC was first coined in the debates on autonomous weapon systems Santoni de Sio and Van den Hoven (2018) Scharre and Horowitz (2015), it has become increasingly relevant in other domains, especially in the context of autonomous vehicles Mecacci and Santoni de Sio (2020) Beckers et al. (2019) Calvert et al. (2020).

G.4.2 Conditions for Meaningful Human Control: What makes Human control meaningful?

Referring to Van den Hoven (2013) and Friedman and Hendry (2019), there are two necessary conditions for meaningful human control viz., tracking and tracing:

- **Tracking condition**: In the tracking condition, control is considered as meaningful when the system's performance co-varies with the reasons of the relevant person(s), in the same way that a mercury column in a thermometer co-varies with the temperature in the room. When air humidity varies, but the temperature remains constant, we expect no change in the mercury column, as it only tracks temperature. In light of this, to be considered under

meaningful human control, a human-AI system should be responsive to the human moral reasons relevant in the circumstances.

- **Tracing condition**: In order for a human-AI system to be considered under meaningful human control, it is necessary that its capabilities, behavior, and possible effects on the world should be traceable to a proper moral and technical understanding on the part of at least one relevant human agent who designs or interacts with the system. Based on this, the tracing condition requires individuals with a proper moral and technical understanding of the system. That would not be the case if a supervisor of an automated vehicle did not understand traffic rules.

G.4.3 Means for achieving Meaningful Human Control: How to achieve MHC?

Human control over a system is complex and dynamic, and this for at least two reasons. First, it is not bound to one particular moment in time. Second, the required level of control is highly situation-dependent. Given this, means for assisting in the achievement are needed. Below is a set of enablers for MHC as proposed by van Diggelen (2019).

G.4.3.1 Dimensions/ characteristics

The proposed set of dimensions/ characteristics that contribute to meaningful human control are the following:

- **The human has freedom of choice**: At the heart of this dimension is the degree to which the human user can choose from all the possible available courses of action. At one extreme the human has complete, unconstrained freedom to choose any course of action, including illegal or detrimental ones, with no influence or assistance from the system. At the other extreme, the machine has constrained humans to a single course of action, with no freedom to deviate from it. The constraining of human freedom of choice can result from different reasons: e.g., the system does not make a particular element of system functionality or course of action available to the user; Information is incomprehensible or inaccessible to the user. However, it is worth noting that not all actions and decisions require complete freedom of choice. Indeed, due to workload or the speed at which decisions must be made, it may not be preferable in some situations to consider all of the potential courses of action or options.
- **The human has the ability to impact the behavior of the system**: This dimension allows to capture the extent to which the user is equipped with the functionality to change the behavior of the system either in real-time or in advance through constraining allowable actions and behaviors or through the setting of bounds. At one extreme the user/ human has complete control over system behavior e.g., the ability to override an autonomous system. At the other extreme, there is no built-in feature (functionality built into the system) to allow the user to change the system behavior.
- **The human has time to decide to engage with the system and alter its behavior**: Central to this dimension is the question: Does the system give the user sufficient time to process information, make decisions, and alter its behavior if necessary? This dimension therefore captures the temporal aspect of user interactions with the system. In this vein, it should be noted that there are situations where it is not desirable or feasible for the human to be "in-the-loop" for example in the case of defensive systems, so they have to intervene

"before the–loop" for instance by setting constraints on the action in order to ensure that human control is maintained.

- **The human has sufficient situation understanding**: This dimension focuses on the degree to which the user has a sufficient understanding of the system state, in order to understand the quality, provenance, and accuracy of the information as well as the rationale of the decisions and recommendations made.
- **The human is capable of predicting the behavior of the system and the effects of the environment (physical and information)**: This dimension captures the extent to which the user is able to predict how the system will behave in different circumstances.

G.4.3.2 Activities and stakeholders

When the focus is on how the systems might be designed to deliver an adequate level of Human control/ meaningful human control, processes/ activities and stakeholders involved in the fielding of systems are of great importance. Examples of these activities include systems specification, systems design, systems verification and validation, training of users, training of AI and ML, systems of systems integration, interoperability and operational use. To ensure that meaningful human control is provided by fielded systems, these activities require the involvement of several stakeholder groups such as system acquirers whose role in providing appropriate human control in future systems consists in specifying, contracting, and accepting AI-based systems that support MHC; system designers whose role lies in conducting, analyzing, using Human Centred Design approaches, and applying best practices to systems design and testing; organizational users whose role is to integrate AI-based systems within their larger system of systems and organizational structures so that they enable Meaningful Human Control; and end users who support, train, employ human-machine team, develop practices and operating procedures to deliver MHC.

G.5. Accountability

Accountability is a cornerstone of the governance of AI. However, it is often defined too imprecisely because its multifaceted nature and the sociotechnical structure of AI systems imply a variety of values, practices, and measures to which accountability in AI can refer. The terms accountability, responsibility, and liability are closely related yet different and also carry different meanings across cultures and languages. Generally speaking, "accountability" implies an ethical, moral, or other expectation (e.g., as set out in management practices or codes of conduct) that guides individuals' or organizations' actions or conduct and allows them to explain reasons for which decisions and actions were taken. In the case of a negative outcome, it also implies taking action to ensure a better outcome in the future. "Liability" generally refers to adverse legal implications arising from a person's (or an organization's) actions or inaction. "Responsibility" can also have ethical or moral expectations and can be used in both legal and non-legal contexts to refer to a causal link between an actor and an outcome.

Given these meanings, the term "accountability" best captures the essence of this principle. In this context, "accountability" refers to the expectation that organizations or individuals will ensure the proper functioning, throughout their lifecycle, of the AI systems that they design, develop, operate or deploy, in accordance with their roles and applicable regulatory frameworks, and for demonstrating this through their actions and decision-making process (for example, by

providing documentation on key decisions throughout the AI system lifecycle or conducting or allowing auditing where justified).

In the context of machine learning, accountability refers to the responsibility and answerability of the individuals or organizations involved in developing, deploying and using machine learning systems. It is about being accountable for the decisions of the AI system, the potential impact on stakeholders, and any consequences that may arise. Accountability includes being transparent, explainable, and understanding who owns the AI system's behavior.

H. Conclusion

This document offers and overview of the assessment of trustworthiness in Artificial Intelligence (AI) systems, in particular for critical systems. Trustworthiness is essential for the wider acceptance and safe implementation of AI in various domains, especially where AI decisions can significantly impact human lives.

This document helps understand the different attributes forming the roots of trustworthiness and how to take them into account during the development and throughout the life of an AI. It provides clear definitions and, when possible, concrete metrics. It highlights the need to consider trustworthiness attributes on different engineering items: the system itself, of course, but also the ML model, an ODD, a dataset or a given data item within a dataset.

This document is also an entry point for other guides such as the Confiance.ai end-to-end methodology, the Confiance.ai data lifecycle methodology, or the Confiance.ai labelling evaluation framework. An important challenge will be to keep these different guides consistent, with a good articulation of quality and risks, and with a clear understanding of how to tailor AI assessment to a particular context of use.

While it has been chosen to keep in this document a system-centric view, it will also be interesting to better show how AI development processes can be assessed.

Alphabetical Index

Bibliography

Abe, M. (2015). *Vehicle handling dynamics: theory and application*. Butterworth-Heinemann.

Ackoff, R. (1989). From data to wisdom. *Journal of applied systems analysis*, 16(1):3–9.

Adam, J.-L., Adedjouma, M., Aknin, P., Alix, C., Baril, X., Bernard, G., Bonhomme, Y., Braunschweig, B., Cantat, L., Chale-Gongora, G., et al. (2022). Towards the engineering of trustworthy AI applications for critical systems - The Confiance. ai program.

Aerospace, S. (2010). Aerospace recommended practice ARP4754 Issued Revised REV. A 1996-11 2010-12 Superseding ARP4754 (R) guidelines for.

Akoumianakis, D. and Stephanidis, C. (1989). Universal Design in HCI: A critical review of current research and practice. *Engineering and Construction*, 754.

Al-Sabaawi, A. (2022). Cryptanalysis of block cipher: Method implementation. In *2022 International Conference for Advancement in Technology (ICONAT)*, pages 1–7.

Alam, K., Md Rokibul Alam, K., Faruq, O., and Morimoto, Y. (2016). A comparison between RSA and ElGamal based untraceable blind signature schemes. In *2016 International Conference on Networking Systems and Security (NSysS)*, pages 1–4.

Aliman, N. and Kester, L. (2019). Transformative ai governance and ai-empowered ethical enhancement through preemptive simulations. *Interdisciplinary review of emerging technologies*, 1.

Allemang, D. and Hendler, J. (2011). *Semantic web for the working ontologist: effective modeling in RDFS and OWL*. Elsevier.

Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L. (2018). Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6077–6086.

ANSI/ IEEE Std 729-1983 (1983). Ieee standard glossary of software engineering terminology.

Arya, V. et al. (2022). AI Explainability 360: Impact and design. In *Proceedings of the AAAI Conf.*, volume 36 (11).

Avižienis, A., Laprie, J.-C., and Randell, B. (2004). Dependability and its threats: a taxonomy. In *Building the Information Society*, pages 91–120. Springer.

Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE transactions on dependable and secure computing*, 1(1):11–33.

Bai, C., Mielczarek, B., Krzymien, W. A., and Fair, I. J. (2007). Improved analysis of list decoding and its application to convolutional codes and turbo codes. *IEEE Transactions on Information Theory*, 53(2):615–627.

Bakhtiyor, A., Orif, A., Ilkhom, B., and Zarif, K. (2020). Differential Collisions in SHA-1. In *2020 International Conference on Information Science and Communications Technologies (ICISCT)*, pages 1–5.

Bana e Costa, C. A. and Oliveira, M. (2012). A multicriteria decision analysis model for faculty evaluation. *Omega*, 40:424–436.

Bana e Costa, C. A. and Vansnick, J.-C. (1994). A theoretical framework for Measuring Attractiveness by a Categorical Based Evaluation TecHnique (MACBETH). In *Proc. XIth Int. Conf. on MultiCriteria Decision Making*, pages 15–24, Coimbra, Portugal.

Beckers, G., Sijs, J., van Diggelen, J., van Dijk, R. J., Bouma, H., Lomme, M., Hommes, R., Hillerstrom, F., van der Waa, J., van Velsen, A., et al. (2019). Intelligent autonomous vehicles with an extendable knowledge base under meaningful human control. In *Counterterrorism, Crime Fighting, Forensics, and Surveillance Technologies III*, volume 11166, pages 73–89. SPIE.

Bevan, N. (2008). Classifying and selecting ux and usability measures. In *International Workshop on Meaningful Measures: Valid Useful User Experience Measurement*, volume 11, pages 13–18. Institute of Research in Informatics of Toulouse (IRIT) Toulouse, France.

Blatchford, M. L., Mannaerts, C. M., and Zeng, Y. (2021). Determining representative sample size for validation of continuous, large continental remote sensing data. *International Journal of Applied Earth Observation and Geoinformation*, 94:102235.

Boardman, M. and Butcher, F. (2019). An exploration of maintaining human control in ai enabled systems and the challenges of achieving it. In *Workshop on Big Data Challenge-Situation Awareness and Decision Support. Brussels: North Atlantic Treaty Organization Science and Technology Organization. Porton Down: Dstl Porton Down*.

Borsci, S., Malizia, A., Schmettow, M., Van Der Velde, F., Tariverdiyeva, G., Balaji, D., and Chamberlain, A. (2022). The chatbot usability scale: the design and pilot of a usability scale for interaction with ai-based conversational agents. *Personal and Ubiquitous Computing*, 26:95–119.

Bousquet, O. and Elisseeff, A. (2002). Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526.

Branco, M., Zaluska, E., De Roure, D., Salgado, P., Garonne, V., Lassnig, M., and Rocha, R. (2008). Managing very-large distributed datasets. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 775–792. Springer.

Braunschweig, B., Gelin, R., and Terrier, F. (2022). The wall of safety for AI: approaches in the confiance.ai program. In *Proceedings of the Workshop on Artificial Intelligence Safety 2022 (SafeAI 2022)*, volume 3087 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Brooke, J. (1996). Sus: a “quick and dirty’usability. *Usability evaluation in industry*, 189(3):189–194.

Brundage, M., Avin, S., Wang, J., Belfield, H., Krueger, G., Hadfield, G., Khlaaf, H., Yang, J., Toner, H., Fong, R., et al. (2020). Toward trustworthy ai development: mechanisms for supporting verifiable claims. *arXiv preprint arXiv:2004.07213*.

Calvert, S. C., Heikoop, D. D., Mecacci, G., and Van Arem, B. (2020). A human centric framework for the analysis of automated driving systems based on meaningful human control. *TheoreTical issues in ergonomics science*, 21(4):478–506.

Chapman, A., Simperl, E., Koesten, L., Konstantinidis, G., Ibáñez, L.-D., Kacprzak, E., and Groth, P. (2020). Dataset search: a survey. *The VLDB Journal*, 29(1):251–272.

Chaudhary, P., Gupta, R., Singh, A., and Majumder, P. (2019). Analysis and comparison of various fully homomorphic encryption techniques. In *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*, pages 58–62.

Cho, J.-H., Xu, S., Hurley, P. M., Mackay, M., Benjamin, T., and Beaumont, M. (2019). Stram: Measuring the trustworthiness of computer-based systems. *ACM Computing Surveys (CSUR)*, 51(6):1–47.

Choquet, G. (1954). Theory of capacities. In *Annales de l'institut Fourier*, volume 5, pages 131–295.

Clements, R. (1991). *Handbook of Statistical Methods in Manufacturing*. Prentice-Hall: Upper Saddle River.

Colquitt, J. A., Brent, S. A., and Jeffery, L. A. (2007). Trust, trustworthiness, and trust propensity: a meta-analytic test of their unique relationships with risk taking and job performance. *Journal of applied psychology*, 92(4):909.

Confiance.ai et al. (2022). Explainability benchmark v2 - the confiance.ai program.

Confiance.ai et al. (2023). Ec3.21: Methodological guidelines for oodeel library.

Confiance.ai EC5 (2023). Methodological Guideline for Formalization of the data lifecycle in the context of machine learning and critical systems.

Confiance.ai EC6 (2023). Operational Design Domain: definition, quality model and relation to trustworthiness.

Delseny, H., Gabreau, C., Gauffriau, A., Beaudouin, B., Ponsolle, L., Alecu, L., Bonnin, H., Beltran, B., Duchel, D., Ginestet, J.-B., et al. (2021). White paper machine learning in certified systems. *arXiv preprint arXiv:2103.10529*.

Dereziński, M. (2019). Fast determinantal point processes via distortion-free intermediate sampling. In *Conference on Learning Theory*, pages 1029–1049. PMLR.

Devroye, L. and Wagner, T. (1979). Distribution-free inequalities for the deleted and holdout error estimates. *IEEE Transactions on Information Theory*, 25(2):202–207.

Díaz-Rodríguez, N., Del Ser, J., Coeckelbergh, M., de Prado, M. L., Herrera-Viedma, E., and Herrera, F. (2023). Connecting the dots in trustworthy artificial intelligence: From ai principles, ethics, and key requirements to responsible ai systems and regulation. *Information Fusion*, page 101896.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Du, M. et al. (2019). On attribution of recurrent neural network predictions via additive decomposition. In *The WWW Conf.*, pages 383–393.

EASA (2020). EASA Artificial Intelligence Roadmap 1.0 published. A human-centric approach to AI in aviation.

EASA (2021). Concept paper first usable guidance for level 1 machine learning applications.

ED-76A (2015). Standards for processing aeronautical data.

eGAS Working Group (2013). Standardized e-gas monitoring concept for gasoline and diesel engine control units.

Eigner, O., Eresheim, S., Kieseberg, P., Klausner, L. D., Pirker, M., Priebe, T., Tjoa, S., Marulli, F., and Mercaldo, F. (2021). Towards resilient artificial intelligence: Survey and research issues. In *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 536–542. IEEE.

Erhan, D., Bengio, Y., Courville, A. C., and Vincent, P. (2009). Visualizing higher-layer features of a deep network. Technical report, Département d'Informatique et Recherche Opérationnelle, Univ. of Montreal.

ETSI (2021). Securing Artificial Intelligence (SAI); Mitigation Strategy Report.

EUROCAE WG114 – SAE G34 (2021). A joint standardization initiative to support Artificial Intelligence revolution in aeronautics.

European Commission (2021). 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.

Fang, X. and Wu, Y. (2017). Investigation into the elliptic curve cryptography. In *2017 3rd International Conference on Information Management (ICIM)*, pages 412–415.

Felderer, M. and Ramler, R. (2021). Quality assurance for ai-based systems: Overview and challenges (introduction to interactive session). In *International Conference on Software Quality*, pages 33–42. Springer.

Fernandez, A., Insfran, E., Abrahão, S., Carsí, J. Á., and Montero, E. (2012). Integrating usability evaluation into model-driven video game development. In *Human-Centered Software Engineering: 4th International Conference, HCSE 2012, Toulouse, France, October 29-31, 2012. Proceedings 4*, pages 307–314. Springer.

Floridi, L. (2019). Establishing the rules for building trustworthy ai. *Nature Machine Intelligence*, 1(6):261–262.

Fong, R. C. and Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3449–3457.

Friedman, B. and Hendry, D. G. (2019). *Value sensitive design: Shaping technology with moral imagination*. Mit Press.

Frøkjær, E., Hertzum, M., and Hornbæk, K. (2000). Measuring usability: are effectiveness, efficiency, and satisfaction really correlated? In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 345–352.

Gao, Z., Li, Z., and Tu, Y. (2004). Design and completion of digital certificate with authorization based on pki. In *Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration, 2004. IRI 2004.*, pages 462–466.

Ge, M. and Helfert, M. (2006). A framework to assess decision quality using information quality dimensions. In *ICIQ*, pages 455–466.

Gol Mohammadi, N., Paulus, S., Bishr, M., Metzger, A., Könnecke, H., et al. (2013). Trustworthiness attributes and metrics for engineering trusted internet-based software systems. In *International Conference on Cloud Computing and Services Science*, pages 19–35. Springer.

Gong, Z., Zhong, P., and Hu, W. (2019). Diversity in machine learning. *IEEE Access*, 7:64323–64350.

Grabisch, M. and Labreuche, C. (2010). A decade of application of the choquet and sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, 175(1):247–286.

Gundersen, O. E. (2019). Standing on the feet of giants—reproducibility in ai. *Ai Magazine*, 40(4):9–23.

Gundersen, O. E. (2021). The fundamental principles of reproducibility. *Philosophical Transactions of the Royal Society A*, 379(2197):20200210.

Gyllenhammar, M., Johansson, R., Warg, F., Chen, D., Heyn, H.-M., Sanfridson, M., Söderberg, J., Thorsén, A., and Ursing, S. (2020). Towards an operational design domain that supports the safety argumentation of an automated driving system. In *10th European Congress on Embedded Real Time Systems (ERTS 2020)*.

Habli, I., McDermid, J. A., and Monkhouse, H. (2016). The notion of controllability in an autonomous vehicle context. In *CARS Workshop Critical Automotive Applications: Robustness & Safety*.

Halstead, M. H. (1977). Halstead. elements of software science.

Hamon, R., Junklewitz, H., and Sanchez, I. (2020). Robustness and explainability of artificial intelligence. *Publications Office of the European Union*.

Hariprasad, T., Vidhyagaran, G., Seenu, K., and Thirumalai, C. (2017). Software complexity analysis using halstead metrics. In *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, pages 1109–1113. IEEE.

Harradon, M., Druce, J., and Ruttenberg, B. E. (2018). Causal learning and explanation of deep neural networks via autoencoded activations. *CoRR*, abs/1802.00541.

Heinrich, B., Hristova, D., Klier, M., Schiller, A., and Szubartowicz, M. (2018). Requirements for data quality metrics. *Journal of Data and Information Quality (JDIQ)*, 9(2):1–32.

Hendricks, L. A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., and Darrell, T. (2016). Generating visual explanations. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, pages 3–19. Springer International Publishing.

Hexmoor, H., Castelfranchi, C., and Falcone, R. (2003). *Agent autonomy*, volume 7. Springer Science & Business Media.

High-Level Expert Group on Artificial Intelligence (2019). Assessment list for trustworthy artificial intelligence (altai). Technical report, European Commission.

IEEE (2022). CertifAIEd.

ISO 26262-1 (2011). Road vehicles — functional safety — part 1: Vocabulary.

ISO 9000 (2015a). Quality management systems — Fundamentals and vocabulary.

ISO 9000 (2015b). Quality management systems — Fundamentals and vocabulary.

ISO 9241-210 (2019). Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems.

ISO/DIS 5725-1 (2020). Accuracy (trueness and precision) of measurement methods and results — Part 1: General principles and definitions.

ISO/IEC 21827 (2008). Information technology — Security techniques — Systems Security Engineering — Capability Maturity Model© (SSE-CMM©).

ISO/IEC 2382 (2015). Information technology — vocabulary.

ISO/IEC 25010 (2011a). *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*. ISO/IEC JTC 1/SC 7 Software and systems engineering.

ISO/IEC 25010 (2011b). Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models.

ISO/IEC 25012 (2008). Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Data quality model.

ISO/IEC 25023 (2015). Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality.

ISO/IEC 27000 (2018). Information technology — Security techniques — Information security management systems — Overview and vocabulary.

ISO/IEC DIS 22989 (2021a). Information technology — Artificial intelligence — Artificial intelligence concepts and terminology.

ISO/IEC DIS 22989 (2021b). *Information technology — Artificial intelligence — Artificial intelligence concepts and terminology*. ISO/IEC JTC 1/SC 42/WG 1 Foundational standards.

ISO/IEC DIS 42001 (2022). Information technology — Artificial intelligence — Management system.

ISO/IEC TR 24028 (2020). Information technology — artificial intelligence — overview of trustworthiness in artificial intelligence.

ISO/IEC TR 24029-1 (2021). Artificial intelligence (ai) — assessment of the robustness of neural networks — part 1: Overview.

ISO/IEC TR 29119-11 (2020). Software and systems engineering — software testing — part 11: Guidelines on the testing of ai-based systems.

ISO/IEC TS 5723 (2022). Trustworthiness — Vocabulary.

ISO/IEC TS 8200 (2023). Information technology - Artificial intelligence - Controllability of automated artificial intelligence systems.

ISO/IEC/IEEE 15026-1 (2019). Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary.

ISO/IEC/IEEE 24765 (2017). Systems and software engineering — vocabulary.

Ivanković, M., Petrović, G., Just, R., and Fraser, G. (2019). Code coverage at google. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 955–963.

Iyer, R., Li, Y., Li, H., Lewis, M., Sundar, R., and Sycara, K. (2018). Transparency and explanation in deep reinforcement learning neural networks. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, page 144–150. Association for Computing Machinery.

Jakubik, J., Vössing, M., Kühl, N., Walk, J., and Satzger, G. (2022). Data-centric artificial intelligence. *arXiv preprint arXiv:2212.11854*.

Jarrahi, M. H., Memariani, A., and Guha, S. (2022). The principles of data-centric ai (dcai). *arXiv preprint arXiv:2211.14611*.

Jenn, E., Albore, A., Mamalet, F., Flandin, G., Gabreau, C., Delseny, H., Gauffriau, A., Bonnin, H., Alecu, L., Pirard, J., et al. (2020). Identifying challenges to the certification of machine learning for safety critical systems. In *European congress on embedded real time systems (ERTS 2020)*.

Jesty P.H, Buckley T.F, W. M. M. (1993). The development of safe advanced road transport telematic software. *Microprocessors and Microsystems*, 17:37–46.

JORF (2018). Jorf n°0285 du 9 décembre 2018.

Kavzoglu, T. (2009). Increasing the accuracy of neural network classification using refined training data. *Environmental Modelling & Software*, 24(7):850–858.

Kearns, M. and Ron, D. (1997). Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. In *Proceedings of the tenth annual conference on Computational learning theory*, pages 152–162.

Kirakowski, J., Corbett, M., and Sumi, M. (1993). The software usability measurement inventory. *Br J Educ Technol*, 24(3):210–212.

Kulesza, A., Taskar, B., et al. (2012). Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286.

Labreuche, C. (2011). A general framework for explaining the results of a multi-attribute preference model. *Artificial Intelligence*, 175(7-8):1410–1448.

Li, J., Monroe, W., and Jurafsky, D. (2016). Understanding neural networks through representation erasure. *CoRR*, abs/1612.08220.

Li, N. (2010). Research on diffie-hellman key exchange protocol. In *2010 2nd International Conference on Computer Engineering and Technology*, volume 4, pages V4–634–V4–637.

Lin, R.-D. and Chen, W.-S. (2005). Fast calculation algorithm of the undetected errors probability of crc codes. In *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers)*, volume 2, pages 480–483 vol.2.

Liu, H., Wang, Y., Fan, W., Liu, X., Li, Y., Jain, S., Liu, Y., Jain, A., and Tang, J. (2022). Trustworthy ai: A computational perspective. *ACM Transactions on Intelligent Systems and Technology*, 14(1):1–59.

Liu, H., Yin, Q., and Wang, W. Y. (2018). Towards explainable NLP: A generative explanation framework for text classification. *CoRR*, abs/1811.00196.

Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., and Zhang, G. (2020). Learning under concept drift: A review. *CoRR*, abs/2004.05785.

Lund, A. M. (2001). Measuring usability with the use questionnaire12. *Usability interface*, 8(2):3–6.

Mace, R. (1990). Definitions: Accessible, adaptable, and universal design. *Raleigh: North Carolina State University, Center for Universal Design. Retrieved June*, 30:2002.

Madni, A. M. and Madni, C. C. (2018). Architectural framework for exploring adaptive human-machine teaming options in simulated dynamic environments. *Systems*, 6(4):44.

Mamalet, F., Jenn, E., Flandrin, G., Delseny, H., Gabreau, C., et al. (2021). White Paper Machine Learning in Certified Systems. Research report, IRT Saint Exupéry ; ANITI.

Mani, S., Sankaran, A., Tamilselvam, S., and Sethi, A. (2019). Coverage testing of deep learning models using dataset characterization. *arXiv preprint arXiv:1911.07309*.

Mattioli, J., Delaborde, A., Khalfaoui, S., Lecue, F., Sohier, H., and Jurie, F. (2022a). Empowering the trustworthiness of ml-based critical systems through engineering activities. *arXiv preprint arXiv:2209.15438*.

Mattioli, J. et al. (2023a). An overview of key trustworthiness attributes and kpis for trusted ml-based systems engineering. In *AI Trustworthiness Assessment (AITA) @ AAAI Spring Symposium*.

Mattioli, J., Robic, P., and Jesson, E. (2022b). Information quality: the cornerstone for AI-based industry 4.0. *Procedia Computer Science*, 201:453–460.

Mattioli, J., Sohier, H., Delaborde, A., Amokrane-Ferka, K., Awadid, A., Chihani, Z., Khalfaoui, S., and Pedroza, G. (2023b). An overview of key trustworthiness attributes and kpis for trusted ml-based systems engineering. In *AAAI Spring Symposium 2023, AITA: AI Trustworthiness Assessment*.

Mattioli, J., Sohier, H., Delaborde, A., Pedroza, G., Amokrane-Ferka, K., Awadid, A., Chihani, Z., and Khalfaoui, S. (2023c). Towards a holistic approach for ai trustworthiness assessment based upon aids for multi-criteria aggregation. In *Safe AI*.

Mazumder, M., Banbury, C., Yao, X., Karlaš, B., Rojas, W. G., Diamos, S., Diamos, G., He, L., Kiela, D., Jurado, D., et al. (2022). Dataperf: Benchmarks for data-centric ai development. *arXiv preprint arXiv:2207.10062*.

Mbelekani, N. Y. and Bengler, K. (2023). Learnability in automated driving (liad): Concepts for applying learnability engineering (cale) based on long-term learning effects. *Information*, 14(10):519.

Mecacci, G. and Santoni de Sio, F. (2020). Meaningful human control as reason-responsiveness: the case of dual-mode vehicles. *Ethics and Information Technology*, 22:103–115.

Mock, M., Schmitz, A., Adilova, L., Becker, D., Cremers, A. B., and Poretschkin, M. (2021). *Management System Support for Trustworthy Artificial Intelligence*. Fraunhofer IAIS.

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., and Müller, K.-R. (2017). Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222.

Mostafa, S. A., Ahmad, M. S., and Mustapha, A. (2019). Adjustable autonomy: a systematic literature review. *Artificial Intelligence Review*, 51:149–186.

Mountrakis, G. and Xi, B. (2013). Assessing reference dataset representativeness through confidence metrics based on information density. *ISPRS journal of photogrammetry and remote sensing*, 78:129–147.

Ngo, D. C. L., Teo, L. S., and Byrne, J. G. (2003). Modelling interface aesthetics. *Information Sciences*, 152:25–46.

Nielsen, J. (1994). *Usability engineering*. Morgan Kaufmann.

O;, C. A. D. G. G. F. and M, R. H. (2013). Pump as turbine (pat) design in water distribution network by system effectiveness. *Water*, 5(3):1211–1225.

Oman, P. and Hagemeister, J. (1992). Metrics for assessing a software system's maintainability. In *Proceedings Conference on Software Maintenance 1992*, pages 337–338. IEEE Computer Society.

O'Neill, O. (2014). Trust, Trustworthiness, and Accountability. In Morris, N. and Vines, D., editors, *Capital Failure: Rebuilding Trust in Financial Services*, page 0. Oxford University Press.

Pendleton, M., Garcia-Lebron, R., Cho, J.-H., and Xu, S. (2016). A survey on systems security metrics. *ACM Computing Surveys (CSUR)*, 49(4):1–35.

Phillips, P. J., Hahn, C. A., Fontana, P. C., Broniatowski, D. A., and Przybocki, M. A. (2020). Four principles of explainable artificial intelligence. *Gaithersburg, Maryland*.

Piorkowski, D., Hind, M., and Richards, J. (2022). Quantitative ai risk assessments: Opportunities and challenges. *arXiv preprint arXiv:2209.06317*.

Plantard, T., Susilo, W., and Zhang, Z. (2013). Fully homomorphic encryption using hidden ideal lattice. *IEEE Transactions on Information Forensics and Security*, 8(12):2127–2137.

Plumb, G., Al-Shedivat, M., Cabrera, Á. A., Perer, A., Xing, E., and Talwalkar, A. (2020). Regularizing black-box models for improved interpretability. *Advances in Neural Information Processing Systems*, 33:10526–10536.

Pons, L. and Ozkaya, I. (2019). Priority quality attributes for engineering AI-enabled systems. *arXiv:1911.02912*.

Ramos, A., Lazar, M., Holanda Filho, R., and Rodrigues, J. J. (2017). Model-based quantitative network security metrics: A survey. *IEEE Communications Surveys & Tutorials*, 19(4):2704–2734.

Reason, J. (1990). *Human error*. Cambridge university press.

Redman, T. (1996). *Data Quality for the Information Age*. Artech House Telecommunications Library. Artech House.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1135–1144, New York, NY, USA. Association for Computing Machinery.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, page 1527–1535.

Roehr, T. M. and Shi, Y. (2010). Using a self-confidence measure for a system-initiated switch between autonomy modes. In *Proceedings of the 10th international symposium on artificial intelligence, robotics and automation in space, Sapporo, Japan*, pages 507–514.

Santoni de Sio, F. and Van den Hoven, J. (2018). Meaningful human control over autonomous systems: A philosophical account. *Frontiers in Robotics and AI*, 5:15.

Scharre, P. and Horowitz, M. C. (2015). Meaningful human control in weapon systems: A primer. *Center for a New American Security*, 16.

Schmidt, E., Work, B., Catz, S., Chien, S., Darby, C., Ford, K., Griffiths, J.-M., Horvitz, E., Jassy, A., Mark, W., et al. (2021). National security commission on artificial intelligence (ai). Technical report, National Security Commission on Artificial Intelligence.

Science, N. and on Artificial Intelligence, T. C. U. S. C. (2019). *The national artificial intelligence research and development strategic plan: 2019 update*. National Science and Technology Council (US), Select Committee on Artificial ....

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Stanton, B., Jensen, T., et al. (2021). Trust and artificial intelligence. *preprint*.

Sun, L., Dong, H., and Liu, A. X. (2018). Aggregation functions considering criteria interrelationships in fuzzy multi-criteria decision making: state-of-the-art. *IEEE Access*, 6:68104–68136.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv:1312.6199*.

Teruel, M. A., Navarro, E., López-Jaquero, V., Montero, F., and González, P. (2014). A cscw requirements engineering case tool: development and usability evaluation. *Information and Software Technology*, 56(8):922–949.

Tuomisto, H. (2010). A diversity of beta diversities: straightening up a concept gone awry. part 1. defining beta diversity as a function of alpha and gamma diversity. *Ecography*, 33(1):2–22.

Union, S. (2020). Stardog the enterprise knowledge graph platform. *Stardog Union, https://www. stardog. com/platform*.

Van den Hoven, J. (2013). Value sensitive design and responsible innovation. *Responsible innovation: Managing the responsible emergence of science and innovation in society*, pages 75–83.

van Diggelen, J. (2019). Meaningful human control over ai-based systems. In *Symposium Get Connected of NATO C2 Centre of Excellence on 25-27 June 2019, Brussels, Belgium*.

VDE (2022). VCIO based description of systems for AI trustworthiness characterisation. Technical Report VDE SPEC 90012 V1.0 (en), VDE.

Wang, L., Ghosh, D., Gonzalez Diaz, M., Farahat, A., Alam, M., Gupta, C., Chen, J., and Marathe, M. (2020). Wisdom of the ensemble: Improving consistency of deep learning models. *Advances in Neural Information Processing Systems*, 33:19750–19761.

Wang, R. Y. and Strong, D. M. (1996). Beyond accuracy: What data quality means to data consumers. *Journal of management information systems*, 12(4):5–33.

Wing, J. M. (2021). Trustworthy ai. *Communications of the ACM*, 64(10):64–71.

Xie, N., Ras, G., van Gerven, M., and Doran, D. (2020). Explainable deep learning: A field guide for the uninitiated. *CoRR*, abs/2004.14545.

Xu, C., Lien, K.-C., and Höllerer, T. (2023). Comparing zealous and restrained ai recommendations in a real-world human-ai collaboration task. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–15.

Yeh, C.-K., Hsieh, C.-Y., Suggala, A., Inouye, D. I., and Ravikumar, P. K. (2019). On the (in) fidelity and sensitivity of explanations. *Advances in Neural Information Processing Systems*, 32.

Youden, W. J. (1950). Index for rating diagnostic tests. *Cancer*, 3(1):32–35.

Zeiler, M. D. and Fergus, R. (2014). "visualizing and understanding convolutional networks". In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *"Computer Vision – ECCV 2014"*, pages 818–833, "Cham". Springer International Publishing.

Zen, M. and Vanderdonckt, J. (2014). Towards an evaluation of graphical user interfaces aesthetics based on metrics. In *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*, pages 1–12. IEEE.

Zhang, J. M., Harman, M., Ma, L., and Liu, Y. (2020). Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering*.

Zhang, Q., Cao, R., Shi, F., Wu, Y. N., and Zhu, S.-C. (2018). Interpreting cnn knowledge via an explanatory graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pages 4454–4463.

Zhou, T., Zhu, Y., Jing, N., Nan, T., Li, W., and Peng, B. (2020). Reliable soc design and implementation of sha-3-hmac algorithm with attack protection. In *2020 IEEE International Conference on Smart Cloud (SmartCloud)*, pages 88–93.

Title: Methodological Guideline for Trustworthy AI Assessment

Keywords: Quality, Risk, Metrics, Aggregation, System, Data, Process

The document provides a guide to assessing trustworthiness in Artificial Intelligence (AI) systems, particularly focusing on their critical applications where AI decisions significantly impact human lives and safety. The guide elaborates on the various attributes that constitute trustworthiness, offering clear definitions and, where feasible, specific metrics to quantify these attributes. It stresses the importance of evaluating trustworthiness across different engineering components, including the system as a whole, ML models, Operational Design Domains (ODDs), datasets, and individual data items. Furthermore, this document serves as a foundational resource, linking to other detailed guides like the Confiance.ai end-to-end methodology, data lifecycle methodology, and labelling evaluation framework. It underscores the ongoing challenge of maintaining consistency across these guides, ensuring a balanced focus on quality and risk, and adapting AI assessment to specific contexts of use. The document predominantly adopts a system-centric perspective but also acknowledges the value in enhancing the evaluation of AI development processes.

Our partners:


Air Liquide

AIRBUS

Atos

cea

Inria

NAVAL
GROUP

GROUPE RENAULT

SAINT
EXUPÉRY

SAFRAN

sopra steria

SystemX

THALES
Building a future we can all trust

Valeo