EC4.40

The RUM Methodology

Methodological Guideline for RUM (Robustness, Uncertainty and Monitoring)

Document reference number for ANR

contact@confiance.ai | www.confiance.ai

Document reference: 440A

Contributors

	Name	Organisation	Role
Responsible for the deliverable	Martin Gonzalez	IRT SystemX	AS Co-Lead
Scientific responsible	Martin Gonzalez	IRT SystemX	AS Co-Lead
Co-authors	Paul-Marie Raffi	IRT SystemX	Main Co-author
	Kevin Pasini	IRT SystemX	Main Co-author
	Lucas Mattioli	IRT SystemX	Main Co-author
	Mohamed Ibn-Khedher	IRT SystemX	Chapters C-D
	Katarzyna Kapusta	Thales	Chapter C
	Juliette Mattioli	Thales	Chapter D
	Hatem Hajri	Safran	Chapter C
	Luca Mossina	IRT Saint-Exupéry	Chapter C
	Corentin Friedrich	IRT Saint-Exupéry	Chapter C
	Eiji Kawasaki	CEA	Chapter C
	Fabio Arnez	CEA	Chapter C
Reviewers	Loic Cantat	IRT SystemX	-
	Georges Jamous	Airbus Protect	-
	Karla Quintero	IRT SystemX	-
	Juliette Mattioli	Thales	-
	Nathan Rajohnson	IRT SystemX	-

Document Control

Revision	Date	Commentary	Author
v0.1	01-03-2024	Initiate Document	Martin Gonzalez
v0.2	01-08-2024	First Draft	Martin Gonzalez
v1.0	31-08-2024	Ready for Review	Martin Gonzalez
v2.0	30-09-2024	Delivered	Martin Gonzalez

Contents

A. Introduction

A.1. General introduction to trustworthy AI challenges

Trustworthiness in 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.

A.2. Rationale for this methodological guideline

This document aims to provide to a non-expert engineer, somewhat familiar with the work done regarding Trustworthiness in the confiance.ai program, a synthetic and unified account on what has been accomplished regarding Robustness (R), Uncertainty Quantification (UQ) and Monitoring (M) tools in the EC3 and EC4 teams by the end of the program. Additionally, it aims at providing perspectives for developments in programs that will be sequels of confiance.ai such as CSI and IAG Core.

A.3. Target audience and disclaimer

The target audience of this document consist of machine learning engineers, not necessarily expert but familiar with ML trustworthiness techniques and best practices.

A.4. How to use this document

Throughout the document, the abbreviation RUM denotes the co-consideration of R, UQ and M while we will use the notation R/U/M when these domains are thought separately one from the other. There are three different uses of this document:

1. The reader may use the first chapter as a whirlwind tour of the R/U/M disciplines, their main questions, challenges with the objective of of understanding the basic motivation for constructing the RUM methodology;

2. The reader may use the second chapter as a centralized directory detailing the Robustness, Uncertainty Quantification and Monitoring attributes, metrics, and indicators that are available within the confiance.ai program;

3. The reader may use the third chapter as an in-depth introduction to the RUM methodology, particularly useful in the context of the End-to-End activities, to acquire a unified and organized view of the interactions between different confiance.ai components, their trade-offs and limits.

B. Survey of the RUM attributes and methodology

B.1. High-level definitions

In this section we lay the ground for the definitions of Robustness, Uncertainty and Monitoring that will be used in this paper. These are closely tied, although not identical, to the taxonomy reference also developed in the Confiance.ai program. We refer the reader to documents such as EASA Concept Paper, EUROCAE ED-324, DEEL White Paper, Ashmore et al. (2021) and Confiance.ai's Methodological Guideline for Trustworthy AI Assessment. and the references therein for such account.

In particular, the taxonomy documents developed by the Confiance.ai program concerning the Operational Design Domain (ODD) can be found in SAE J3016 (2018) and Kaakai et al. (2023). Those concerning robustness are in F. et al. (2021), SAE (2022). Those concerning UQ are in Confiance.ai Taxonomy V2. Finally, in this document, all considerations on Robustness, Uncertainty and Monitoring are delimited in the context of Machine Learning. Unless stated otherwise, terms such as Monitoring will refer to the specific case of ML Monitoring and so on.

B.1.1 Robustness

Machine learning robustness addresses the question of how well a machine learning model can maintain its performance and make accurate predictions in the face of various challenges, perturbations, or uncertainties. Robustness is concerned with the ability of a model to generalize well to new, unseen data and to withstand changes in the input, such as noise, outliers, adversarial attacks, or variations in the distribution of the data.

In essence, the key question addressed by machine learning robustness is: *"How resilient is the model's performance when faced with different conditions or perturbations in the input data?"* Robust machine learning models are desirable because they are less likely to be affected by unexpected variations in the data, leading to more reliable and trustworthy predictions in real-world scenarios.

Definition 1 *Robustness in AI refers to the ability of a model to perform well across various conditions, including different types of input data, noise, or perturbations.*

Role in AI:

- Ensures that a model is not overly sensitive to small changes in input data.
- Increases the model's reliability in real-world applications.
- Helps prevent unexpected failures or vulnerabilities.

Importance: Robust models are less susceptible to adversarial attacks, changes in data distribution, or noisy input, leading to more consistent and dependable performance.

Techniques: Regularization methods, data augmentation, adversarial training, and model architecture design are techniques used to enhance robustness.

B.1.2 Uncertainty Quantification

Machine learning uncertainty quantification addresses the question of how confident or uncertain a machine learning model is in its predictions. It seeks to understand and quantify the uncertainty associated with the model's output, providing a measure of the model's confidence in its predictions. This is particularly important when dealing with real-world applications where uncertainties are inherent or when the model encounters situations that differ from its training data.

The main question addressed by uncertainty quantification in machine learning is: *"How well does the model know what it doesn't know?"* This involves assessing the uncertainty in predictions and understanding the model's limitations. Uncertainty can arise from various sources, including limited data, model complexity, and inherent variability in the underlying processes. By quantifying uncertainty, practitioners and users can make more informed decisions and take appropriate actions based on the reliability of the model's predictions.

Definition 2 *Uncertainty quantification is the process of estimating and characterizing the uncertainty associated with predictions made by a model.*

Role in AI:

- Provides insights into the confidence or reliability of model predictions.
- Helps users and stakeholders understand the limitations of the model in different scenarios.
- Facilitates decision-making by considering uncertainty.

Importance: Understanding uncertainty is vital for making informed decisions, especially when deploying models in real-world applications where uncertainty can have significant consequences.

Techniques: Bayesian methods, dropout during training, ensemble methods, and probabilistic models help quantify and characterize different sources of uncertainty.

B.1.3 Monitoring

Machine learning monitoring addresses the question of how well a machine learning model is performing over time, and whether it continues to meet the desired criteria for accuracy and effectiveness in a dynamic environment. The key question addressed by machine learning monitoring is: *"Is the model behaving as expected, and is its performance consistent with the intended objectives?"*

Definition 3 *Monitoring involves tracking the performance and behavior of a system or model*

over time. In the context of AI, it refers to the continuous observation of a model's outputs, inputs, and other relevant metrics during its deployment.

Role in AI:

- Helps in detecting anomalies or deviations from expected behavior.
- Aids in identifying performance degradation or shifts in data patterns.
- Allows for timely intervention and updates to the model.

Importance: Models can degrade over time due to changes in the environment, data, or underlying assumptions. Monitoring ensures early detection of such issues, enabling timely intervention and model maintenance.

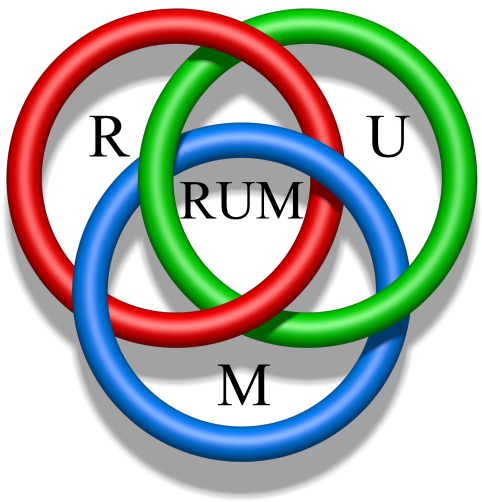
Techniques: Performance metrics tracking, concept drift detection, anomaly detection, and model explainability methods contribute to effective monitoring.

B.1.4 Links between Monitoring, Uncertainty, and Robustness

Robustness (R), Uncertainty (U), and Monitoring (M) are interconnected concepts in Machine Learning (ML). Let's explore the relationships between these three aspects. Confiance.ai proposes the "**RUM methodology**" that aims at providing means to assess AI component robustness beyond the robustness of the ML model. The following principles underpin the methodology:

- a joint monitoring of robustness attributes and uncertainty quantities;
- the robustness joint assessment of monitored observables and uncertainty quantities;
- the uncertainty joint quantification of monitored observables and robustness attributes.

This relationship should be apprehended as displayed intuitively in the figure below, as three 3D loops topologically linked where any two such loops are linked by a third one.



Robustness, Uncertainty and Monitoring methods can only successfully address their different

challenges by working together.

B.1.4.1 Monitoring for Robustness

- Regular monitoring can help identify scenarios where a model's performance deviates from expectations.
- Monitoring can detect situations where the model may exhibit reduced robustness, such as when faced with previously unseen data.

B.1.4.2 Monitoring for Uncertainty Detection

- Continuous monitoring can be used to identify cases where the model's uncertainty estimates may be high, signaling potential areas of concern or scenarios where the model lacks confidence.

B.1.4.3 Robustness for Uncertainty Mitigation

- Robust models are often better at handling uncertainties in the data, making them more reliable in uncertain or changing environments.
- Robustness measures, such as data augmentation or regularization techniques, can contribute to better uncertainty quantification.

Overall, the RUM methodology provides means to articulate and characterize different ODD zones to better detect possible failure modes, assess possible trade-offs or overall system-level compensations. This is not possible if robustness, uncertainty quantification or monitoring are considered independently.

Finally, when working in harmony, R, U and M incur into better risk management and more adaptive AI systems.

Risk Management. Robust models, along with uncertainty quantification and continuous monitoring, collectively contribute to effective risk management. Understanding and managing uncertainties reduce the likelihood of unexpected failures or inaccuracies.

Adaptive Systems. A unified view promotes the development of adaptive machine learning systems that can dynamically adjust to changing conditions, ensuring sustained performance and reliability over time.

B.2. Unified Life-cycle Perspective

Ensuring a model's robustness during development, quantifying uncertainties during inference, and monitoring its performance in production are continuous processes that follow the Confiance.ai's End2End Methodological Approach. A first architecture of the overall mechanism is presented in Figure B.1.

First, the Specified ODD, the component specifications and other higher-level requirements have an impact on the selection of data and model to be used during training. After these have been set, and a data split has been done, the model is trained.

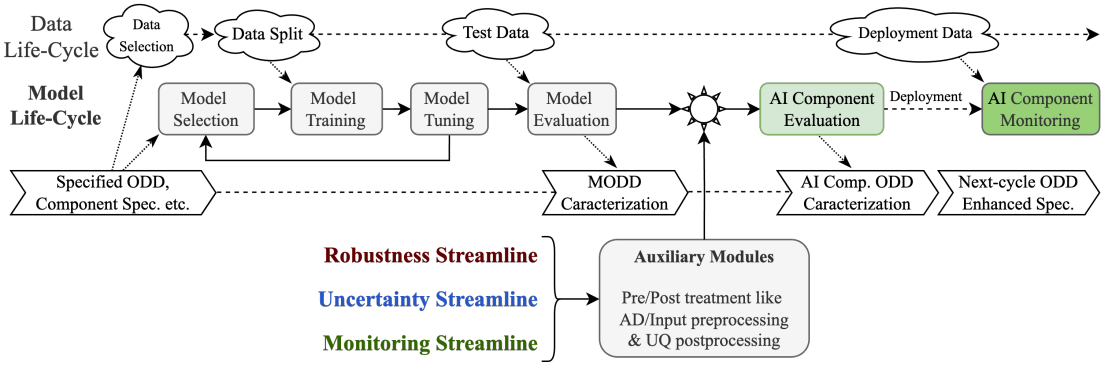


Figure B.1: High-level overview of the R/U/M stream-line's relationship with the model's life-cycle.

The model tuning phase consists on intermediate evaluations of the trained model that may have a retroactive impact on possible chances in the model architecture, the training algorithm or the data selected and/or its split.

Once this loop gives a satisfactory trained model according to the higher level specified ODD & specifications, the model is evaluated on chosen test data, which might be the same i.i.d. test data used during the intermediate evaluations of the model in the tuning phase, or it can consist on completely different test data such as OOD data (synthetic or real), unbalanced and non-i.i.d data and so on. The choice of the test data reflects the specified ODD requirements.

Overall, evaluating the model on such test data gives a Model-ODD characterization, usually very close but never completely identical to the part of the Specified ODD characterization concerning *only the model*.

In parallel to the streamline concerning these stages, three other streamlines, each one concerning robustness measures, Uncertainty quantification modules and monitoring systems is developed in parallel. Some of these might have dependencies with the model's streamline as they will be *inserted* into the model itself. Otherwise, they will consist on auxiliary modules that are plugged to the initial model, which we will call the *central model*, and which will usually take the form of pre-treatment modules such as anomaly detectors that will filter out anomalies so that the central model dies not make inference on them, or post-treatment modules such as Enriching the central model's outputs with information about its uncertainties.

The AI component will then be the resulting aggregate of the central model along with all the implemented auxiliary modules around it.

Now that we have composed such AI component, a further evaluation must be made concerning its trustworthiness attributes, but this time, *as a component*. Indeed, the trustworthiness attributes of the AI component will in general be different from those of the central model alone, and while one can expect that the robustness or uncertainty of the AI component will be better than that of the central model alone, *there might be cases in which implemented robustness and UQ measures will harm each other,* creating trade-offs between robustness and uncertainty that could not have been foreseen or evaluated on the central model alone.

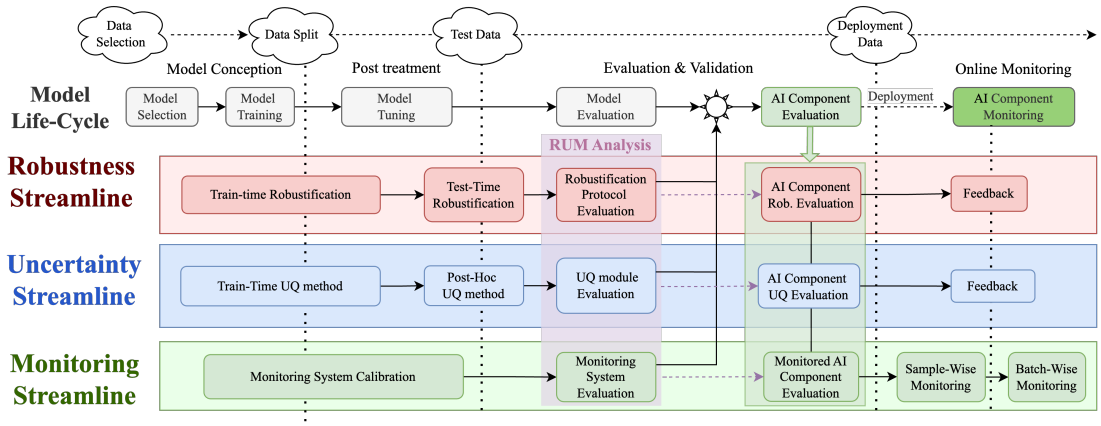


Figure B.2: Simplified view of the RUM life-cycle and operational delimitation of the RUM analysis.

The AI component's evaluation will then incur into an AI component ODD characterization that will then be compared to the initial specified ODD. In case the resulting AI component is considered to comply with the requirements, it will be embedded in the system that will host this component. We emphasize that although many evaluations are of absolute necessity in this stage, there are not the subject of the RUM methodology and will not be treated here.

Finally, once the system hosting the AI component is deployed, the monitoring functions that are included in the AI component will allow to online monitor many aspects of the deployed and hosted AI component, either in a sample-wise or batch-wise fashion, and which may use informative feedback coming from the UQ and Robustness streamlines to do so.

B.2.1 Life-cycle RUM Integration

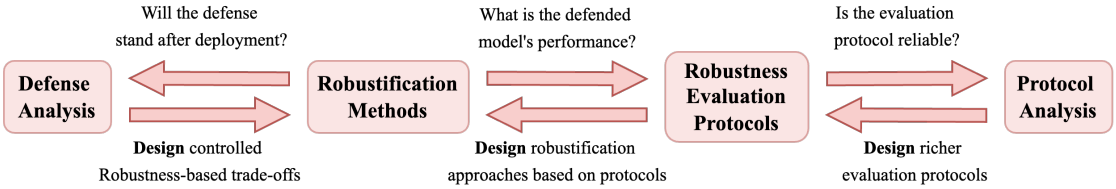
Robustness, uncertainty quantification, and monitoring are interconnected throughout the machine learning life-cycle. Ensuring a model's robustness during development, quantifying uncertainties during inference, and monitoring its performance in production are continuous processes as shown in Figure B.2.

In this process, the displayed green & violet rectangles articulate the difference between evaluating RUM of the resulting AI component and evaluating the quality of the robustification protocol itself, the UQ module itself, and the monitoring system itself. The latter are considered in the violet rectangle and are part of the subject of the RUM methodology. Consequently, the action of *Robustness* on the RUM methodology implies evaluating both the robustness improvement and the evaluation of the UQ module - rather than only the central ML model, as well as the evaluation of the monitoring system.

A holistic approach that integrates robustness, uncertainty quantification, and monitoring is essential for building resilient and trustworthy machine learning systems, particularly in applications where accuracy, reliability, and interpretability are critical. The principle of RUM is the integral consideration of these 3 streamlines and the means to deploy it will depend on the specific constraints and characteristics of the AI component and context of operation. Specific trade-offs and aggregations stem from the consideration of the RUM method; these trade-offs are then RUM attributes per se.

B.2.2 The RUM analytic approach

B.2.2.1 Robustness



Confiance.ai's robustness assessment consists on the workflow in the above figure, based on:

- **Robustness Evaluation Protocols:** These transform the theoretical evaluation concept into an algorithmic protocol. For instance, crafting adversarial attacks over the test sets realizes the idea of worst-case In-Distribution analysis while Monte-Carlo sampling is used to estimate a theoretically continuous region around a sample to be certified. It usually serves as a design guide to craft robustification approaches.

- **Robustification Methods:** The very structure of each evaluation protocol usually serves as a design guide to craft defense approaches. Such robustification methods are tightly bounded with the evaluation protocols from which they were crafted and their goal is to improve that protocol's metrics. For instance, adversarial training is crafted to improve a model's performance in the context of worst-case evaluation (i.e. adversarial attacks).

- **Protocol Analysis:** Beyond addressing different questions, protocols are themselves implementations of theoretical methods and need to be the subject of a validation/certification process. For instance, worst-case evaluation is thought to be conducted with the strongest available adversarial attacks within a predefined cyber-security breach context. As such, if the protocol only uses weak adversarial attacks against a well-defended model, it will generate an over-estimated robust accuracy due to a poor quality evaluation protocol rather than due to a poor defense mechanism.

- **Defense Analysis:** analyzing the quality of a robustification method goes beyond metric improvement and its process is tightly related to protocol analysis. For instance, it is a fact that too strong adversarial training overfits and poorly generalizes to adversarial threats which were not seen during training. As a consequence, the worst-case analysis protocol must be enriched to account for such strength-generalizability trade-off.

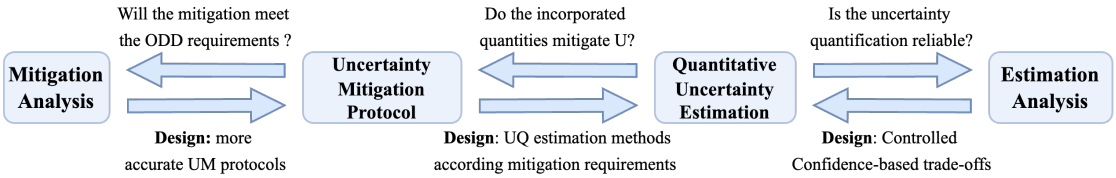
The next Chapter will focus mainly on the two middle blocks while the last Chapter will focus mainly on the blocks at the ends. The underlying idea:

- The existing trusted ones have mainly done robustification and evaluation, which are determined above all on the target model (the classifier let's say);

- Batch 1-2-3 mainly did benchmarks *per protocol* without analyzing in depth the reliability of the existing protocols. This is where the RUM methodology will come in.

Before entering into a short presentation of the above points, let us present the analog of this story for UQ and Monitoring Evaluation and Analysis.

B.2.2.2 Uncertainty



Confiance.ai's uncertainty management methods consists on the workflow in the above figure, based on:

- **Quantitative uncertainty estimation:** The initial uncertainty quantification step enables us to assess the impact of various sources on the outputs of the AI component by approximation using estimators fit on observations. It then produces a measure of uncertainty which will be used to better control model output, and thus increase our confidence. Estimators can have various forms (monitoring probes, statistical modeling, model with uncertainty quantification by design, model dedicated to uncertainty prediction), and be focused on specific sources of uncertainty.

- **Uncertainty Estimation analysis:** An evaluation step is essential to ensure the correctness and relevance of uncertainty measures. Evaluation protocols are specific to the nature of the uncertainty being measured, are based on uncertainty metrics (e.g. (e.g Likelihood, Empirical coverage, Entropy, Calibration curve area) and protocols (data alteration, noise injection, sensibility analysis) that aim to palliate to the absence of ground truth which is often inaccessible (unknown noise) or too costly (rare event).

- **Uncertainty Mitigation protocol:** From an uncertainty measure and according to risk aversion specifications, it is possible to set up an uncertainty mitigation process that will act on inputs, outputs, or the model itself, to calibrate the risk according to the specification. Mitigation often consists of reduce or introduce a bias that leads the errors to be less costly or less likely even if it may slightly degrade performance on average on training data.

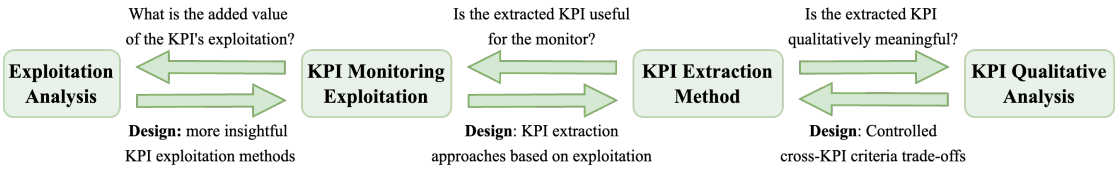
- **Mitigation Analysis:** The mitigation analysis aim to evaluate the benefits of the mitigation process on the AI component outputs according to business KPIs. As the mitigation process can be based on estimators, trained on data that may differ slightly from real data, We need to make sure that on new data, mitigation prevent or reduce the cost of errors made by the AI component. Note that there is some real-time mitigation analysis (e.g. Real time OOD detection), that rely on an human operator who must interpret model confidence alert through UQ-KPI produce by a mitigation protocol from uncertainty measure that express one-off drop or continuous deterioration of model confidence.

One very striking point concerning the comparison between the robustness and uncertainty analysis is that **the middle arrows go in opposite directions**. The underlying reason of this is that Uncertainty and Robustness estimations **are of inherently different nature**.

- On the one hand, uncertainty analysis produces quantities that have an intrinsic quantitative value that can be exploited

- On the other hand, robustness analysis produces numerical values that can only reflect on qualitative attributes and are not exploitable in a direct manner

B.2.2.3 Monitoring

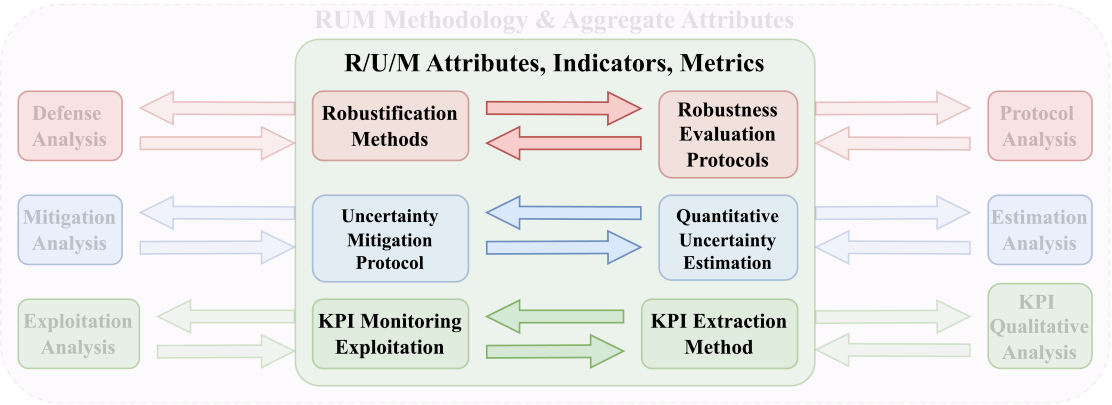


Monitoring evaluation workflow can be decomposed in 4 blocks:

- **KPI Extraction Method:** Extract the evaluation KPIs at different monitoring phases: model ODD characterization, monitor functions calibration and monitor functions validation.
- **KPI Monitoring Exploitation:** Select the KPIs that can be exploited to improve the monitoring process and KPIs extraction.
- **KPI Qualitative Analysis:** Inspect the quality of each KPI in regard to the end user of the monitor. Is it comprehensible for most end user, does it need to be transformed into a graph, should it be complemented with other KPIs to be more meaningful.
- **Exploitation Analysis:** How can each KPI, or an ensemble of KPIs improve the monitoring process ? Do the monitoring KPIs increase the trustworthiness in the AI component or in the AI workflow ?

B.2.3 R/U/M Attributes, Indicators and Metrics during Batches 1 to 3

During Batches 1 to 3 of the Confiance.ai program, several components were made in the involved teams concerning either R, UQ or M. Along with several actors and developers of such tools and components, we established a classification of their components as matching at least one of the following 6 axes.



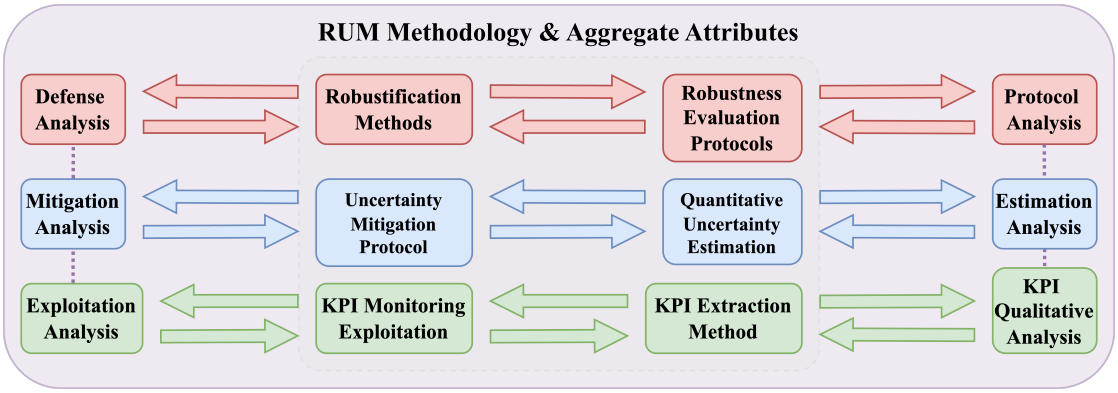
Such axes actually are structured as 3 different couples, each one concerning R, U or M, along the lines of the above subsection. In Chapter C, we give a systematic, structured, and centralized

account on all of the attributes, indicators and metrics that are currently utilized by these components. An important remark here is that, several components already elucidate links between R, U and M along the model's life-cycle. For instance, many components start by providing UQ associated to a model, associate to it an uncertainty mitigation approach and then switch to the treatment of the obtained quantities as KPIs to be exploited by the monitoring system. We will give a detailed account on all the concerned components from the Confiance.ai program in the last section of Chapter D, after elucidating the RUM methodology and its unified life-cycle perspective.

B.3. Overview on the RUM methodology & aggregate attributes

The idea behind the RUM methodology is to leverage the R/U/M enhanced analysis presented in the above subsection to provide a systematic critical - yet constructive - analysis, on each of both ends of the couples presented above:

- Robustness evaluation protocol + Robustification Method
- Quantitative Uncertainty estimation + Uncertainty mitigation protocol
- KPI extraction method + KPI monitoring exploitation



This will lead to the above figure resuming the added value of the RUM methodology and which is broke down into two families of new attributes, for which we present several experiments on both standard benchmarks and industrial Confiance.ai Use-Cases to shed evidence on the significance of these attributes, and we propose subsequent and novel recommendations for industrial usage of the confiance.ai components in light of these new attributes.

1. **RUM-based analytic attributes**: there are attributes that are the result of the analysis of the above couples, such as those explained in subsection B.2
2. **RUM aggregate attributes**: these are attributes that are the result of **mixed** considerations between R-U-M attributes in the conceptual form of a Borromean knot and which are illustrated in the above figure by the outer vertical dotted lines linking the outer RUM boxes.

In particular, for RUM aggregate attributes, we started by enumerating already existing such attributes and reframing them in the RUM methodology and we would call them in this case

nominal attributes. In order to keep this survey as simple as possible, we only make a survey on one of such attributes:

- Joint Monitoring of Model Robustness of UQ modules: novel adversarial attack protocols have been proposed to attack a model's UQ module *without changing its performance*. Concretely, this means that the produced adversarial attacks aim to maximally damage the UQ module associated to the central model, making the latter abnormally overconfident or underconfident on its predictions. Such attacks are proposed both for the Bayesian/Ensemble and the CP approaches for UQ. Consequently, recent work has proposed UQ-aware adversarial training, proposing a metric to control the trade-off between robustness and uncertainty in the couple consisting of a model and its associated UQ module.

Next, since these new family of attributes seems to be an emergent field of research, we expect that more of such attributes will become explicit in the future. But in the meanwhile, by leveraging the attributes we already have at hand, we propose a MCDA methodology, based on Tropical Algebra, for obtaining new attributes in a formal way. Then, the obtained RUM aggregate attributes will be called *non-nominal*.

B.4. Towards the systemic industrial adoption of the RUM method

The following two chapters show a detailed account on all the content mentioned in this chapter. By the end of this methodological guideline, one will see that Robustness, Uncertainty Quantification and Monitoring have been re-framed in what we called the RUM Methodology. Through extensive investigation on the available literature on these subjects, and numerous experiments we shed light on new attributes. These attributes are of two different natures.

First, there are "RUM-based analytic attributes" that enhance and propose a critical view on the already available attributes concerning Robustness, Uncertainty Quantification and Monitoring. Second, we have the "RUM aggregate attributes" whose evaluation protocol and its analysis, as well as their improvement methods and their analysis, need the co-consideration of Robustness, Uncertainty Quantification and Monitoring simultaneously.

We framed these to new families of trustworthiness attributes in a versatile methodology that welcomes ongoing research on the subject, as well as a unified and enriched life-cycle perspective in order to take into account our novelties. We strongly believe that the proposed framework has the potential to be used as a solid basis for a systematized methodological tool for helping to address trustworthiness in Machine Learning.

Finally, although we restrained to the sole consideration of Robustness, Uncertainty Quantification and Monitoring, we believe that many other faces of trustworthiness such as data considerations and particularly explainability tools for ML can be analyzed through the RUM methodology and extend our systematic tool to these further ML domains.

C. R/U/M - Definitions, Attributes, Indicators and Metrics

In this chapter we will put together all definitions, attributes, indicators and metrics concerning robustness, Uncertainty Quantification and Monitoring as they were developed and/or used by the available components of the Confiance.ai program during Batches 1 to 3. These can be globally conceived as those concerning the middle columns in Figure C.1.

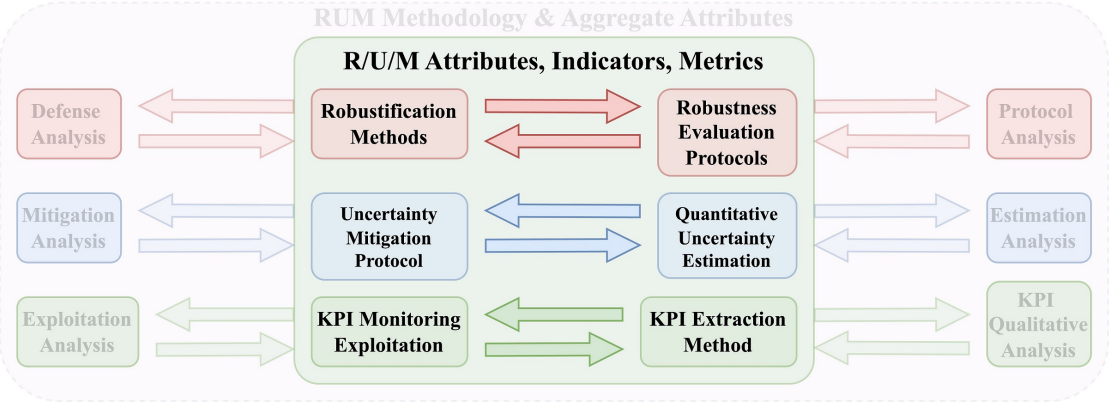


Figure C.1: The R/U/M Attributes Indicators and Metrics within the RUM Methodology.

We will start by presenting this for Robustness. In this case, we will differentiate different theoretical formalization of the notion of robustness, emphasize their limits and shed light on specific notions of robustness that are not readily formalizable. This will makes us have as a starting point the robustness evaluation protocols which implement these theoretical considerations, link such protocols to their corresponding metrics and attributes which they aim to address, and list the available robustification methods that were made available during the Confiance.ai program.

Next, as Uncertainty Quantification gives quantities which are themselves the starting point for uncertainty mitigation protocols, we will start by distinguishing the different natures of uncertainty quantities studied in the program and the protocols for using them to mitigate uncertainty.

Finally, we will present the Monitoring analogs of this story along the lines of how we presented the UQ part, as the way monitoring systems extract and exploit KPIs follows a somewhat similar procedure as that of uncertainty quantities, the latter actually being among such KPIs.

Disclaimer - Target Audience. This Chapter was written with as designated audience ML engineers whose goal is to have a compendium of the R/U/M attributes indicators and metrics formally defined and their implementation protocols.

C.1. Robustness

ML Robustness was addressed more than 10 years ago in Szegedy et al. (2013) and has a very rich literature on its methods. We refer the reader to the Confiance.ai deliverable 45A for a systematic account on such methods.

C.1.1 Theory versus practice

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.

Definition 4 *For $\delta > 0$, we will say that f is δ-locally stable at x w.r.t. d iff*

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

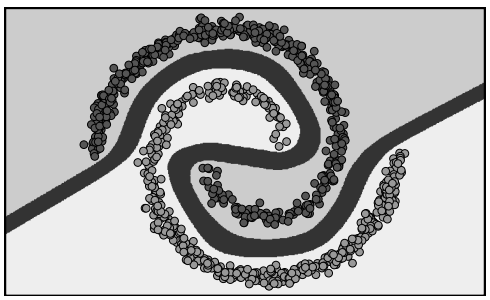
If property (C.1) holds for all $x \in D$, we say that f is δ-globally stable over D w.r.t. d.

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. Notice however 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 should be interpreted as the measuring 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*

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

We choose not to do 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.2) as the formal intersection of property (C.1) with the generic property $[f(x) = y]$.

Remark 2 *Most work on robustness verification has focused on this local robustness property; but the global robustness captures 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. A model f that is δ-locally stable at $x \in D$ w.r.t. d can be trivially transformed into a δ-globally stable model by simply restricting D into D^{res}, where we excluded all inputs that are at a distance less than δ to the decision boundary (see Leino et al. (2021) for details)*



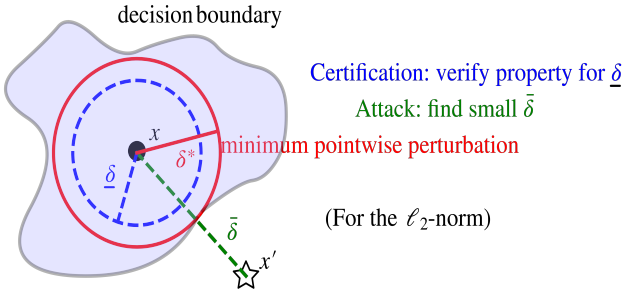
C.1.1.0.1 Robustness Evaluation Protocols. To invalidate property (C.1), for given δ and x, amounts formally to find an *adversarial example* x' satisfying:

$$\exists x' : d(x,x') \leqslant \delta \quad \text{and} \quad f(x) \neq f(x'). \tag{C.3}$$

As such, the minimal distance δ^* such that (C.3) 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 δ^*:

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

Optimizing $\underline{\delta}$ and $\bar{\delta}$ do not answer to the same problems: while finding $\bar{\delta}$ implies crafting a *threat model* to find a *single* sample representing the *empirical* evaluation of our model's *lack* of robustness, finding $\underline{\delta}$ implies finding a *neighborhood* of x in where we can *formally* demonstrate property (C.2) *regardless the kind of attacks* we may have used while optimizing $\bar{\delta}$.



Finally, robustness evaluation methods might design ways of better constructing bounds $\underline{\delta} \leqslant \delta^* \leqslant \bar{\delta}$. without actually changing the value δ^*. But this value can very well change, with *robustification methods* that will modify the model f in a tractable way (usually adding regularization components).

C.1.2 Worst-case In-Distribution Evaluation Protocols

The usual protocol of testing empirically the robustness of a model f is to

- Choose test set S
- Compute the **clean accuracy** $\mathscr{A}_f(S)$: it corresponds to the standard accuracy of f on S. Other performance metrics can be used depending on the use case and the task, e.g. F1 score, mIoU for segmentation task or mAP for object detection.
- Translate the whole test set S into S_Δ^{adv} of adversarial examples under a threat model Δ. Several hyper-parameters for the specific case of *normed-based* adversarial attack must be fixed:
  - the perturbation norm L_p, usually L_2 or L_{inf},
  - the perturbation budget ε: for a data point x and a norm L_p, we constrain the adversarial example x_{adv} to respect $\|x_{\text{adv}} - x\|_p \leq \varepsilon$,
  - the attack method (e.g. FSGM, PGD, CW) and their own parameters (e.g. loss to minimize/maximize, number of iterations, step size)

- For each element x in S, choose x' (among all known adversarial treats) that fools the network.
- Gather all this into an attacked dataset S' (which has all the worst case perturbations of elements of S)
- Compute the **adversarially robust accuracy** $\mathscr{A}_f(S_\Delta^{\text{adv}})$; it corresponds to the **worst found** accuracy. In the specific case of normed-attacks, higher perturbation budgets ε indicate that attacks possess greater potential to fool the model, resulting in a large decrease in empirical robust accuracy. There is no standard values for the perturbation norm and budget; they are specific to the use case and the system requirements. It's important to keep the same hyper-parameter setting to compare the empirical robust performance across multiple models. Slightly modifying these attack parameters can yield important differences in the robustness evaluation. Finally, depending on the task and the use case, other metric can be considered: robust F1 score, robust mIoU, robust mAP, etc. This metric can be computed for all neural networks.
- Characterizing the model's robustness as the difference $\mathscr{A}_f(S) - \mathscr{A}_f(S_\Delta^{\text{adv}})$.

Now, since we are approaching adversarial attacks as a way of stress testing our model to produce a worst-case in-distribution analysis, we can find a better threat model Δ' that will incur into a lower value $\mathscr{A}_f(S_{\Delta'}^{\text{adv}})$. On the one hand, this measure ensures that we are not over-estimating the model's robustness but it does not mean that the model became "less robust" since it didn't change. It only meant that we found a better attack method.

C.1.3 Average-case Out-of-Distribution Evaluation Protocols

As worst-case OOD input perturbations naturally falls outside the specified ODD, no meaningful sensitivity analysis can be done for it. Nevertheless, average-case OOD input perturbations may fall inside the specified ODD, in which case its sensitivity analysis induces challenges on OOD generalization and OOD robustness capabilities.

C.1.3.1 Intrinsic synthetic OOD robustness metrics (Common Corruptions)

As the very notion of a corruption is always relative to a clean counterpart, we find that this metric has a particular weakness for simulated corruptions as it doesn't take into account the following structural principle underlying such corruptions: miss-classified clean images should result on miss-classified simulated corruptions. As such, the rCE answers questions like "how much does the model decline under corruption inputs" but it doesn't detect the corruption error contributions coming from clean miss-classifications.

To take into account this structural hypothesis, we now define an intrinsic relative accuracy metric. Let C be a set of simulated corruptions $c = (\tilde{c}, s)$, where c is a corruption label and s is a severity level. We make the assumption that for each $c \in C$ one can generate (at least) one corruption simulation x_c of a clean image x. We denote y the true label of x, y_{cl} the model's prediction on x and y_c the model's prediction on x_c. Let N be the size of the dataset. Define $M = \sum_{i=1}^{N} \mathbb{I}_{\{y_{\text{cl}}^i = y^i\}}$, $\mathscr{A}_{\text{cl}} = M/N$,and for $c \in C$,

$$\mathscr{A}_c = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_{\{y_c^i = y^i\}}, \quad \mathscr{A}_c^{\text{rel}} = \frac{1}{M}\sum_{i=1}^{N} \mathbb{I}_{\{y_c^i = y^i \,\&\, y_{\text{cl}}^i = y^i\}}$$

where $\mathbb{I}_{\{y_{\text{cl}}=y\}} = 1$ if $y_{\text{cl}} = y$ and 0 else. We find that this metric addresses more accurately (in the statistical analysis sense) questions like "how do corruptions intrinsically behave for this model". *Methodology point*: the clean accuracy $\mathscr{A}_{\text{cl}}$ is used to save the model's parameters during training; the absolute corruption accuracy $\mathscr{A}_c$ gives the model's accuracy for a corruption c; the relative corruption accuracy $\mathscr{A}_c^{\text{rel}}$ computes how many corrupted simulations x_c were correctly classified among the correctly classified clean counterparts; the positiveness of the rCE remains a relevant sanity check for debugging and verifying that the above hypothesis is preserved by the corruption simulator.

Our experiments show that $\mathscr{A}_c^{\text{rel}}$ doesn't overestimate the model's robustness and abnormal behavior (e.g. $\mathscr{A}_c > \mathscr{A}_c^{\text{rel}}$) implies that a selected corruption simulation is ill-posed. On the other hand, we find that the Relative mCE increasingly underestimates it, as highly accurate models will see a greater proportion of their miss-classified corruptions to come from miss-classified clean samples, making $\mathscr{A}_c$ to decrease while clean accuracy increases, as show in Fig. 3 of Hendrycks and Dietterich (2019). This phenomenon is confirmed in Figures 5–8 of Mu and Gilmer (2019) where miss-classified clean images represent 12% of the shown examples and none of them incur into well-classified associated corruptions.

Baseline Normalization: Usual corruption metrics such as the mCE^f and the relative mCE^f for a model f are computed with respect to a baseline (e.g. AlexNet) error in order to normalize it over those corruptions that are known to be particularly challenging. For $(c,s) \in C$, they are the average over all 15 corruptions labels c of the clean and corrupted top-1 error rates (averaged first across all 5 severity levels):

$$\text{CE}_c^f = \left(\sum_{s=1}^{5} E_{s,c}^f\right) / \left(\sum_{s=1}^{5} E_{s,c}^{\text{AlexNet}}\right), \qquad \text{rCE}_c^f = \left(\sum_{s=1}^{5} E_{s,c}^f - E_{\text{cl}}^f\right) / \left(\sum_{s=1}^{5} E_{s,c}^{\text{AlexNet}} - E_{\text{cl}}^{\text{AlexNet}}\right) \tag{C.4}$$

We do not normalize our metrics for two reasons. First, the mCE^f was proposed as an attempt to unify robustness bench-marking under a unique number across many models, which we do not do here. But most importantly, it has been shown that feature aggregating networks and deeper nets markedly enhance robustness. Thus, as NODEs consist on a paradigm shift of the notion of depth, which becomes adaptive even while testing, we find that testing such network against a baseline fixed depth network such as AlexNet could be harmful for our comparative analysis. This is somewhat concordant with the last recommendation in Croce et al. (2022) for generating adversarial attacks for adaptive test-time defenses. We do not average our metrics across severity levels to analyze their behavior at increasing severity.

Corruption simulators: Instead of testing our models on static corrupted datasets (MNIST-C, CIFAR10-C, ImageNet-C), we run the exact same corruption simulator[1] on the datasets of all our experiments (and which is the same one used by the authors that proposed the above-mentioned static corrupted datasets). For simplicity, corruptions that were specially tailored for MNIST (such as zig-zag or canny edges) that only make sense to be conducted on that dataset will be left out from our comparative study, as well as fully formalizable corruptions (such as rotations, translations..) that one can use as auxiliary data augmentation techniques such as adversarial and S&P noise augmentations. This should be taken into account as a partial

[1] Available at https://github.com/bethgelab/imagecorruptions.

reproducibility issue: while the performance of the considered models should decrease when tested on compressed JPEG corrupted datasets such as ImageNet-C, the comparative results conducted in this work do not show any qualitative distinction (although the overall model's accuracies decrease). Also, our objective is not to propose an architecture capable of achieving state-of-the-art robust performances in either of the mentioned datasets. Our choice to fix a common corruption simulator for different datasets is somehow a model-driven choice. It has been shown that classification error patterns between robust models and those coming from human perception are fundamentally different. As such, focusing on understanding a model's behavior around simulated corruptions can be improved by fixing one simulator and generating corruptions among different datasets. This allows to release some part of the randomness of a generated simulation and prevents different data augmentation techniques to guess a corruption simulator's parameters, which in a sense can be seen as information leakage. By using the corruption simulator as a white-box component of our generated corruption dataset we hope this will promote better model's transferability to some degree.

C.1.4 Probabilistic Certification & Testing

C.1.4.1 Theoretical approach

Given a base classifier f, probabilistic certification creates a smoothed version of this classifier, denoted as g, which predicts the most likely class c that the base classifier f will predict for a noised version of the input $x+\varepsilon$, where $\varepsilon \sim \mathcal{N}(0,\sigma^2 I)$. Thus, g can be defined as:

$$g(x+\varepsilon) = \arg\max_{c\in[C]} P(f(x+\varepsilon)=c), \text{ where } \varepsilon \sim \mathcal{N}(0,\sigma^2 I). \tag{C.5}$$

Cohen et al. Cohen et al. (2019a) was the first paper to prove tight robustness guarantees for a randomized smoothing classifier against adversarial attacks constrained under the l_2 norm. Given the top two classes, c_A and c_B, that the base classifier is most likely to predict for the sample, the robust radius r is given by:

$$r = \frac{\sigma}{2}(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)), \tag{C.6}$$

where $p_A = P(f(x+\varepsilon)=c_A)$ and $p_B = P(f(x+\varepsilon)=c_B)$, with the condition that $p_A > 0.5$ and Φ represents the CDF of the standard Gaussian distribution. This demonstrates that if g correctly predicts the input x, we can provide provable l_2 robustness of our model in a probabilistic sense. The main limitation for this approach was that the base classifier had to be robust to noise, meaning it had to be trained with noisy inputs. This limitation was first resolved in Salman et al. (2020) by passing inputs through trained denoisers before feeding them to the model, which then could be taken "off the shelf".

Given a clean image x, the denoised smoothing procedure creates a smoothed classifier by appending a denoiser to any pretrained classifier so that the pipeline predicts in majority the correct class under Gaussian noise corrupted-copies of x. The resulting classifier is certifiably robust against L_2-perturbations of its input.

Still, such denoisers had to be trained but such limitation was also lifted by using pretrained diffusion models as denoisers.

C.1.4.2 Implementation protocol

We follow the evaluation protocol as presented in Cohen et al. (2019b).

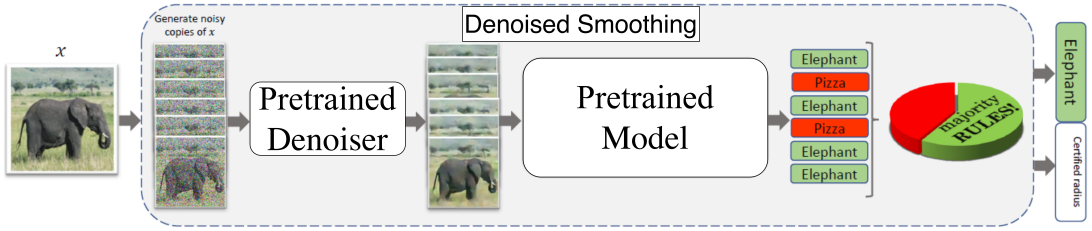


Figure C.2: Denoised Smoothing Salman et al. (2020)

C.1.4.2.1 Point-wise Certification Protocol Given a fixed input x and a RS classifier described by (f, σ), the protocol is divided into two tasks

1. Return the prediction $\hat{g}(x)$;
2. Return the radius $\hat{R}$ in which this prediction is certified robust.

In order to establish the radius in (C.6) for a chosen input x, there are 4 terms that can only be *approximated*: the classes A and B, and the probability estimates of p_A and p_B:

- (Monte-Carlo 1) Generate n noisy samples of the form $x' = x + \varepsilon$ in the ℓ_2-ball centered on x with $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$ for σ a value that is only used here to approximate A;
- Make a majority vote on the predictions $f(x')$ to obtain the class $\hat{A}$, which is supposed to be equal to the class A with very high probability[2];
- (Monte-Carlo 2) Generate $n_0 = 100000$ noisy samples to estimate an accurate approximation of p_A on a statistically significant coverage of the chosen l_2-ball;
- Return the class $\hat{A}$ and the radius $\hat{R}$.

C.1.4.2.2 Set-wise Certification Protocol The Set-wise Certification Protocol is different from the point-wise one as here we ask the following different question:: Given a dataset S made of m inputs $x \in S$ and for a fixed radius $\hat{\sigma}$, what is the percentage of points in where simultaneously:

1. the classifier f returns the correct class associated to x;
2. the certified radius $\hat{R}$ associated to x is greater or equal to $\hat{\sigma}$.

As such, the **probabilistically certified robust accuracy** of f on S will consist on the ration of inputs satisfying simultaneously these two conditions among all inputs (i.e. elements of S).

Notice that **Probabilistic Testing** generalizes the above protocol to other kinds of transforma-

[2]The standard practice is to use a hypothesis test to calibrate an abstention threshold so that the majority vote abstains if it cannot guarantee that the majority class has high enough probability to the equal to A. Inputs where abstention is flagged are also not taken into consideration when computing the probabilistically certified robust accuracy.

tions in order to probabilistically verify a mathematically formalized property.

C.1.5 Deterministic Certification

To verify a robustness formal property of a ML model, there are tools implementing formal methods for algorithmic verification which are being optimized for the specific case of DNNs. For using such tools, we defined a set of terms primarily related to the time required to perform a robustness evaluation and the tool's ability to certify the model's robustness. It should be noted that among all the samples in the dataset, only *C* samples are correctly classified, and it is only these *C* samples that we use for evaluating the robustness. These terms into consideration are:

- **T:** the average time,in second, needed to evaluate the correctly classified images (*C*)
- **V:** the number of the *Verified* robustness samples among all the evaluated images (*C*)
- **F:** the number of the *Falsified* robustness samples among all the evaluated images (*C*)
- **U:** the number of the *Unknown* robustness samples among all the evaluated images (*C*)
- **E:** the number of evaluation *Error* among all the evaluated samples (*C*)

Tools aim to answer the question "is the sample robust or not?" with the correct answer as quickly as possible. Additionally, the best tool maximizes the term (U + E).

$$\max_{U,E}(U+E)$$

C.1.5.1 The Protocol

The main question addressed is: "what does it take to *guarantee* that DNNs are not vulnerable to adversarial attacks?" formulated as *verifying* the I/O properties of DNNs in a way that it leverages existing DNN infrastructures.

- Choose radius ε
- Choose 1000 test samples to validate
- Choose a threshold in seconds
- Embed the non-linear and non-convex optimization procedure as a MIP/LP
- Solve it with off-the-shelf NN-tailored combinatorial solvers
- Create a scoring process: give 10 points to the verification tool if it correctly classified the input as being robust or not (TP or TN) and retrieve 150 points to the verification tool if it renders either a False Positive (claims that an input is robust when the ground truth says it is not robust) or a False Negative (claiming to have found a counter-example to the robustness property that is incorrect)

Attributes. The following attributes are usually used for adressing different attributes of the verification tools.

- Efficiency: A positive integer (the greater the better). This attribute quantifies the performance of the formal verification tools by counting the number of instances — defined by a property specification (pre- and post-condition), a network, and a timeout - conclusively verified within a its timeout. Notice that for VNN-COMP 2023[1] the timeout of the *verification alone* was fixed to five minutes and such competitions have not yet demanded for the competing verification tools to be endowed with certificates of their own.

- Selectivity: A positive number between 0 and 1 (the greater, the better). It correspond to the ratio between the number of truly negatively predicted instances TN (instances that are correctly falsified) and the number of all the negative instances that are presented to the formal method. A high sensitivity means that the formal method is effective at correctly falsifying negative instances.

- Accuracy: A positive number between 0 and 1 (the greater, the better). It correspond to the proportion of correctly verified by the formal method. Two attributes here: accuracy, evaluating the number of instances correctly verified among all predictions and sound-completeness, evaluating the number if instances correctly verified among all explored instances.

C.1.6 Intrinsic Model Property Testing (indirect robustness)

Architectural specification concerns choices at the level of the model's *skeleton*, roughly belonging to specific families of skeleta (ConvNets, CNNs, ResNets, Transformers), that have theoretical properties related to trustworthiness, making them more suitable to be used in a specific scenario.

On the one hand, these choices can concern geometric properties: there are transformer architectures which are equivariant to permutations, certain ConvNets are equivariant to translations. If we denote T such spatial transformation, and f the chosen architecture, equivariance on an input x means that

$$f(T(x)) = T(f(x)).$$

On the other hand, other choices may concern stability properties: many of these architectures, when seen as L-Lipschitz functions (for a usually intractable L), may be turned into 1-Lipschitz functions.

Most of the time, for such properties to be preserved during training, a well-adapted training algorithm must be used in conjunction with these choices. The choice of this algorithm is data-agnostic and, when it exists, will be such that the selected property will remain valid after training.

While equivariance and Lipschitzness are properties, models can also be endowed with extra-structure such as uncertainty quantities associated to its outputs and whose qualitative distinctions (e.g. aleatoric and epistemic) remain valid after training.

Besides the above, there are properties that are guaranteed to remain fixed for most all learning algorithms (provided they do not modify the model's squeleton), trivially verifying *designability*. For instance, in Peng et al. (2023), it is empirically proven that the Width-Depth ratio of an n-

stage CNN f, given by

$$\mathrm{WDR}_f := \frac{1}{n-1}\sum_{i=1}^{n-1}\frac{W_i}{D_i}$$

is a metric which is shown to have an optimal value for improving adversarial robustness.

It is worth mentioning that these so-called *by-design* features of models are not universal in the sense that they are still relative to a specification parameter, environment and task, and their ability to address specific trustworthiness challenges should always be experimentally evaluated or verified.

C.1.6.1 Model Simulated Stability

Let $X = \{x^i\}_{i=1}^N$ be a test set, denote $T(x^i,t)$ a (usually non-unique) **simulated** transformation of x^i of type T and intensity t. The "stability curve" of a model f on X with respect to T is a function over the intensity:

$$S_{f,T}(t) := \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}_{\{f(x^i)=f(T(x^i,t))\}}.$$

For instance, x^i is an image, T adds blur to it, at intensity t. Then, over X, $S_{f,T}(t)$ computes how many of the blurred image's predictions changed their top-1 class with respect to their clean counterparts.

C.1.6.1.1 Remarks:

- The performance of f *does not* correlate with its stability
- Since T is a simulated transformation, a high $S_{f,T}(t)$ in general *does not* imply that f will be stable in deployment under the real context corresponding to the simulated transformation T.
- A globally unstable model can be very adversarially robust and an adversarially vulnerable model can be very stable. Example: take T as adding Gaussian noise:
  - Since adversarial attacks are a "sub-transformation" T_{adv} that adds *adversarial* gaussian noise to an image with *low* intensity t, then one can have simultaneously $f(x^i) = f(T(x^i,t))$ and $f(x^i) \neq f(T_{\mathrm{adv}}(x^i,t))$ for most of the x^i's.
  - Vice-versa, suppose that f has been found to be quickly unstable for low t, that it has been adversarially retrained for a *single* threat model, and that T_{adv} adds adversarial noise along that *same* threat model. Then one can simultaneously have $f(x^i) \neq f(T(x^i,t))$ and $f(x^i) = f(T_{\mathrm{adv}}(x^i,t))$ for most of the x^i's.
- As such, $S_{f,T}(t)$, neither as a curve or as a quantity for fixed t, is neither necessary or sufficient metric for evaluating the model's robustness.
- Nevertheless, it may provide useful heuristics to be monitored: specializing transformations may render stability curves more sensitive to specific robustness attributes.

C.1.6.2 1-Lipschitzness

1-Lipschitz neural networks are designed to have a Lipscitz constant equal to 1. Their significance is that Lipschitzness is a property involving stability in which knowing the Liptshitz constant is fundamental to be able to produce effective theoretical stability bounds. Recall that a continuous function $f : \mathscr{X} \to \mathbb{R}$ is globally L-Lipschitz on $\mathscr{X}$ w.r.t. the norm $\|\cdot\|_2$ if

$$\|f(x_1) - f(x_2)\|_2 \leqslant L\|x_1 - x_2\|_2, \qquad \forall x_1, x_2 \in \mathscr{X}.$$

If so, it is locally L-Lipschitz with respect to all $x \in \mathscr{X}$ w.r.t. the norm $\|\cdot\|_2$

$$\|f(x) - f(x')\|_2 \leqslant L\|x - x'\|_2, \qquad \forall x' \in \mathscr{X}.$$

Fixing $x \in \mathscr{X}$ and taking $x_{\text{adv}} := x + \delta \in B_{l_2}(x, \bar{\delta})$ and $L = 1$, incurs into

$$\|f(x) - f(x_{\text{adv}})\|_2 \leqslant \|\delta\|_2 \leqslant \|\bar{\delta}\|_2.$$

Computing pointwise certified stability requires only logit values $f(x)$. Indeed, let $f : \mathscr{X} \to \mathbb{R}^K$, and let $\text{top}_1 f(x) > \text{top}_2 f(x)$ be the logits (before softmax) of the two top predicted classes. It takes the attacker to choose δ such that $\text{top}_1 f(x_{\text{adv}}) < \text{top}_2 f(x_{\text{adv}})$ while leaving all other predictions unchanged. As such, we obtain (before softmax which is L-Lipschitz with $L \leqslant \frac{\sqrt{K-1}}{\sqrt{K}} < 1$):

$$\frac{\text{top}_1 f(x) - \text{top}_2 f(x)}{\sqrt{2}} := \delta_m(x) \leqslant \|f(x) - f(x_{\text{adv}})\|_2 \leqslant \|\delta\|_2.$$

Remarks.

- 1-Lip networks are "stable by design" and robustness cannot be directly inferred from this, but only stability w.r.t. ℓ_2 norm. Experimentally, FSGM in ℓ_∞ norm is enough to break this defense as shown in Table 1 of Serrurier et al. (2021). As such, 1-Lip may inherit somewhat natural stability attributes (such as no vanishing or exploding gradients, yet to be verified) but it worst-case analysis shows it is not competitive in comparison with other adversarial defenses.

- Although they benefit from an intrisic stability property in theory, there are no available implemented robustness certificates specific to 1-Lipschitz networks, and the latter are only be certified via the standard protocols for probabilistic and deterministic certificates at present. In particular, they do not produce any instance of what a continuous ℓ_2 ball is, and thus they can't be evaluated on such mathematical objects, they can only express robustness certificates by means of sampling from this ball, or using verification tools as presented above.

C.1.7 Attacks against model Watermarking (indirect robustness)

ML watermarking is a family of techniques aiming at marking ML models during their training in order to be able to identify their origins when needed. Two watermarking setting are possible: a white-box setting in which a secret change is introduced to the model parameters and a black-box setting where the model behavior is changed for some secret inputs. An introduction to ML

watermarking can be found in Kapusta et al. (2024). An analysis of ML watermarking applied to a welding inspection use case was published in Kapusta et al. (2024).

ML watermarking can be subject to various attacks, aiming at removing the watermark from the model or disabling the possibility of efficient watermark verification. Therefore, robustness analysis in the context of ML watermarking has for goal to quantify the resistance of ML watermarking against known attacks.

Here is a non-exhaustive list of attributes that are used in the literature (see Kapusta et al. (2024) and Kapusta et al. (2024)) to characterize watermarking techniques along with a proposition of some metrics (when possible) that could be used to compare the watermarking solutions. Note that some of the attributes are hard to quantify. We consider a model M that becomes $\hat{M}$ after being marked using a black-box watermarking technique with a set of watermarks $\theta = (I_{key}, L_{key})$ containing k key inputs I_{key} and their corresponding labels L_{key}.

The verification function $Verify(M, \theta)$ tests a model M^* for the presence of the watermarks. It queries the model using the I_{key} inputs and obtains thus the L_{ext} labels. It outputs 1 if the watermark detection ratio WDR is above a certain threshold $\mathscr{T}$ and 0 otherwise. The WDR function compares for each input i in the key input I_{key} if the extracted label l_{ext} is the same as the expected label l_{key} and outputs the number of matching labels.

$$\text{Verify}(M^*, \theta) = \begin{cases} 1 \text{if WDR}(M^*, \theta) \geq \mathscr{T} \\ 0 \text{otherwise} \end{cases} \tag{C.7}$$

We note the accuracy of a model M^* on the test data set D^{test} as $Acc(M^*, D)$.

- **Robustness** Robustness of ML watermarking is usually understood as the resilience of the watermarking technique against removal attacks based on different model modifications such as pruning, fine-tuning, or WM overwriting. The primary goal of the attacker is to derive a surrogate model from the watermarked model that does not contain the watermark. At the same time, the attacker expects the model to perform as well on the main task as the original model. These two goals need to be reached with a reasonable amount of resources (significantly less than building a new comparable model from scratch). In other words, the attack aims to build a model M_A from $\hat{M}$ following the two conditions
  - $\text{Verify}(M_A, \theta) = 0$
  - $|Acc(M_A, D^{test}) - Acc(\hat{M}_{WM}, D^{test})| < \varepsilon$
- **Fidelity** The prediction quality of the model on its original task should not be degraded significantly after marking a model. This requirement can be defined following two conditions. The Watermark detection Rate needs to be above the threshold to certify the IP of the model. The second condition corresponds to the impact of the watermark on the main task learned by the model. This condition is evaluated by comparing the accuracy of $\hat{M}$ with a non-watermarked model M using the test set D^{test}. If the difference between both accuracies is less than a tolerated error ε the watermarking technique meets the fidelity requirement. These two conditions can be summarized as follows
  - $\text{Verify}(\hat{M}, \theta) = 1$

  - $|Acc(\hat{M}, D^{test}) - Acc(M, D^{test})| < \varepsilon$

- **Reliability** Watermark extraction shall yield minimal false negatives; WM shall be effectively detected using the set of θ pertinent watermarks.

  - $WDR(\hat{M}, \theta) \geq \mathscr{T}$

- **Integrity** Watermark extraction shall yield minimal false alarms (a.k.a., false positives); the watermarked model should be uniquely identified using the pertinent keys. In other words, a non-marked model M should not pass the verification procedure.

  - $WDR(M, \theta) < \mathscr{T}$

- **Capacity** The capacity in ML watermarking is often defined as the number of bits conveyed by a white-box watermarking technique. For black-box watermarking, it is usually understood as the number k of black-watermarks embedded into a network.

- **Secrecy** Secrecy means that the presence of the watermark should be secret and the watermark should be undetectable. For some techniques it can be quantified using resistance against exhaustive search.

- **Efficiency** Communication and computational overhead of watermark embedding, and extraction shall be negligible. This can be defined only with regard to a specific watermarking technique.

- **Generality** The watermarking methodology should be ideally applied to various DNN architectures and dataset. Generality is easier to achieve for white-box watermarking, but hard for black-box techniques. Even if black-box techniques apply the same idea of watermarking from backdooring, their implementation details will vary depending on the network.

- **Non-repudiation** This attribute determines if the model owner cannot successfully dispute its ownership and the validity of the watermark. It means that a link is created between θ and the model owner identity.

C.2. Uncertainty

Uncertainty Quantification (UQ) in machine learning is a field focused on assessing and managing the uncertainty in predictions and model parameters. It plays a crucial role in enhancing the reliability, robustness, and interpretability of machine learning models, especially in critical applications such as those provided by the Confiance.ai program. UQ addresses two main types of uncertainties:

1. **Aleatoric Uncertainty (Data Uncertainty)**: This type of uncertainty arises from the inherent noise or randomness in the data. It cannot be reduced by increasing the amount of data or improving the model. For example, in medical imaging, the inherent variability in images due to different scanning conditions or patient movements contributes to aleatoric uncertainty.

2. **Epistemic Uncertainty (Model Uncertainty)**: This uncertainty stems from the lack of knowledge about the best model to describe the data. It can be reduced by gathering more data, improving the model, or using better modeling techniques. For instance, in predictive maintenance, the uncertainty about the underlying physical processes can be reduced as more sensor data becomes available.

Mitigating uncertainty requires a comprehensive approach throughout the entire ML model life-cycle from data collection to model deployment and monitoring. Uncertainty measurements can serve different purposes such as:

- **Predictive uncertainty**, which involves assessing the confidence of a model's predictions. This can be crucial for applications where knowing the reliability of a prediction is as important as the prediction itself.
- **Out-of-distribution (OOD) detection,** which involves identifying inputs that are significantly different from the training data, for which the model might not provide reliable predictions
- **Anomaly detection**, which identifies instances that deviate significantly from the norm, which might indicate errors, fraud, or novel insights.

C.2.1 Predictive uncertainty

Uncertainty is an abstraction (formalized through a mathematical framework in our cases) used to incorporate in a modeling scope the consequences of things that are not observed, not understood, or that cannot be caught.

Applied to the modeling part of an ML model, it primarily aims to express the consequence of uncertainties coming from different sources (Noise measurement, Irreducible phenomenon variability, lack of observations, modeling approximation) on the AI Component output (for a modeling scope link our AI Component). More specifically, it allows assessing (or quantifying) an irreducible variability and/or increased risk of model error that impacts model prediction due to noise or anomaly in inputs, distance to data domain distribution, lack of observations and data representativeness issues, or even modeling weakness on the data subset to which inputs belong. In this regard, it contributes to both robustness and safety attributes.

An ML model with uncertainty quantification performs a primary task (ex: Classification, Prediction, Anomaly detection) by providing operational indicators (that solve the task). Such indicators can be refined/corrected/or supplied by complementary indicators that aim to characterize how one or more (intertwined) sources of uncertainty may impact model prediction at the inference of a sample (x) or a sample subset. Depending on the application, the indicators can express:

- **Prediction Uncertainty** measure on model output for in-of data distribution (In-ODD) sample where the model with UQ (by design or by component) has captured the observed irreducible variability.
- **OOD measure** which characterizes a model risk due to out-of-data distribution, meaning modeling isn't relevant for such a sample. It can be seen as an approximate way to handle

data-density.

- **Anomaly measure** on sample that discriminate abnormal sample from a nominal behavior. Indeed, the notion of outliers/anomaly can be seen the impact of a source of uncertainty characterized by a strong but very punctual impact. In this scope, uncertainty and OOD measures can be used as criteria to separate anomalies from nominal modeling (ex Z-score) who can still contain some nominal uncertainty.

Uncertainty indicators are often raw uncertainty measurements that may require some post-processing to be processed into a human-interpretable form. Post-processors may require estimations of specific parameters linked to operational requirements. Such refined indicators have to be evaluated on dedicated metrics that also can incorporate operational requirements. A few examples of refined and complementary operational indicators built from uncertainty measures are presented:

- Complementary uncertainty indicators:
  - A prediction uncertainty measure (ex: variance) can be turned by a post-processor into error margins or confidence intervals that characterize, for In-ODD input, the uncertainty around prediction according to the model.
  - An OOD measure (ex: pseudo likelihood) can be turned by a post-processor into a model confidence level that expresses, according to the model, that input differs from an expected distribution learned on the training set.
- Refined operational indicators:
  - A forecast can be refined with a prediction uncertainty measure (ex variance) by a post-processor into a new forecast indicator that limits under-prediction risk according to the predicted uncertainty measure.
  - A residue-based anomaly score (Difference between model prediction and observation) can be refined with a prediction uncertainty measure (ex variance) by a post-processor into an "anomaly score considering uncertainty" to better handle the nominal variability of data (heteroscedasticity).
  - An anomaly score can be refined with an OOD measure (ex: pseudo log-likelihood) by a post-processor into a Model-risk-aware anomaly score to better handle OOD-risk in model-based anomaly detection.

C.2.1.1 Link between UQ & OOD, data science step and confiance.ai Attributes

These three types of indicators (Prediction uncertainty, OOD and anomalies measures) can be used in several steps of the data science pipeline to contribute to confiance.ai attributes:

C.2.1.2 Link between UQ & OOD, data science step and confiance.ai Attributes

These three types of indicators (Prediction uncertainty, OOD and anomalies measures) can be used in several steps of the data science pipeline to contribute to confiance.ai attributes:

- For Data-quality assessment through model with UQ in a data-qualification step
  - To aliment data-characterization metrics with prediction uncertainty measures.
  - To aliment data-representativeness with OOD measures.
  - To detect and process outliers with anomaly measures.
- For ML-model Robustness & ML-risk assessments UQ-based during conception step
  - In the learning of robust model with UQ/probabilistic components that provide uncertainty measure used in training set augmentation/selection considering prediction uncertainty and/or OOD measure
  - In the learning of robust model with UQ/probabilistic task-formalization using probabilistic loss and/or sampling strategy.
  - For design of robust model that used UQ/framework to provide complementary uncertainty indicators.
- For monitoring a model with UQ in a model evaluation step.
  - Ensure of the effectiveness of risk-assessment ability.
  - Ensure of the effectiveness of task under risk-requirement.
  - Ensure of the effectiveness of OOD-assessment ability.
  - Ensure of the effectiveness of anomaly detection.
- For Monitoring data using an ML model with UQ in operation step:
  - to express uncertainty of prediction for a given input.
  - to express OOD-risk link to the prediction of a given input.
  - to perform refined anomaly detection based on a 'nominal model'
- For Model-Monitoring based on uncertainty-KPI in operation step:
  - Monitor model-likelihood or/and OOD-measure to detect drop indicating distribution-shift in input;
  - Monitor UQ-metrics to detect drop indicating over/under-confidence of UQ-components.

C.2.1.3 Indicator & metric for UQ in regression

Let f be a model with uncertainty quantification which from a sample x_i link to a target y_i, predict the distribution $f(x_i) = \widehat{p}_i$ upon of the target space $\mathscr{T}$. The distribution $\widehat{p}_i$ could be also represented in another form, for example as a prediction and gaussian uncertainty measure: $(\widehat{y}_i, \widehat{\sigma}_i)$.

Accuracy: Express average cost of error (according to the loss L) of average model prediction.

- Generic expression: $\sum_i L(p_i, y_i)$
- RMSE: $\sqrt{\frac{1}{n}\sum_i^n |\mathbb{E}[\widehat{p}_i] - y_i|^2}$

UQ çavalidity: Express the relevance of predictive uncertainty according to observations.

- Such measures need to define the set called α-lvl confidence interval of a distribution p as

$$IC^{\alpha}(p) = \{\mathbb{I}_{p(x)>s} | \int_x \mathbb{I}_{p(x)>s} * p(x) = \alpha\}$$

- The α-lvl Empirical Gap coverage express for a predicted distribution p, according to a

dataset X the gap between expected coverage of a α-level confidence interval of p and it empirical coverage observed on X. it caught an empirical over or under confidence of the predicted probability at the α-lvl of confidence.

$$Emp-Gap-Cov^{\alpha}:\alpha-\frac{1}{n}\sum_{i}\mathrm{I}_{\{y_i\in IC^{\alpha}(\hat{p}_i)\}}$$

- The area between curve calibration express for a predicted distribution $\hat{p}$, according to a validation dataset X, is the sum of Empirical gap coverage of a probability p over all α-lvl. It expresses the absolute of over and under confidence of predicted distributions.

$$\int_{\alpha=0}^{1}(|Emp-Gap-Cov^{\alpha}(\hat{p}_i,y_i)|)$$

UQ optimality: Express how much estimated uncertainty is theoretically informative

- Shanon entropy $H_{\mathscr{T}}(\hat{p})=\frac{1}{N}\sum_{i}^{N}-\mathbb{E}[log(\hat{p}_i)]$
- Variance: $Var[\hat{p}]=\frac{1}{N}\sum_{i}^{N}\mathbb{E}[\hat{p}_i-\mathbb{E}[\hat{p}_i]))^2]$
- α-lvl sharpness of predicted distribution,

$$Sharpness^{\alpha}(\hat{p})=\frac{1}{N}\sum_{i}^{N}\mu(IC^{\alpha}(p_i))$$

with μ measure (ex: euclidian distance for Gaussian IC) .

Model Log likelihood: express model relevance which combine Accuracy and UQ entropy

- Generic expression: $-\frac{1}{n}\sum_{i}log(\mathscr{L}(\hat{p}^{i}(y_i))$
- Normal Negative log likelihood $\frac{1}{n}\sum_{i}ln(2\pi\hat{\sigma}_i)+0.5(\frac{y_i-\hat{\mu}_i}{\hat{\sigma}_i})^2$

Model risk: Express average error-risk (Predicted probability of each values * cost) of model prediction by consider uncertainty.

- Generic expression: $\frac{1}{n}\sum_{i}\int_{v}^{\mathscr{T}}\hat{p}^{i}_{(v)}*L(v,y)$
- Gaussian uncertainty with RMSE: $\hat{p}_i=(\widehat{\mu}_i,\widehat{\sigma}_i):\frac{1}{n}\sum_{i}\int_{v}^{\mathscr{T}}PDF_{(v)}^{\mathbb{N}(\widehat{\mu}_i,\widehat{\sigma}_i)}*|v-y_i|^2$

C.2.1.4 Indicators & metrics for OOD measure from model epistemic uncertainty

From UQ-ML paradigm focused on the model uncertainty, we can obtain a set of diverse and relevant model parameters $\Theta=\{(\theta_1,\ldots,\theta_k)|\forall\theta_k,\mathscr{L}(f_k^{\theta}(X_{test}),Y_{test})>s\}$ given by an UQ formalism from a dataset X_{train},Y_{train}. This can be done by learn several models on subset of dataset (Model ensemble paradigm) or by approximate a distribution of models (allowing to drawn models) using bayesian or evidential paradigms. From a set of model parameters Θ, we can then infer for each sample i a set of prediction $\hat{Y}_i^{\Theta}=\{\hat{y}_i^{\theta}\}^{\theta\in\Theta}$ whom variability express models uncertainty.

If we consider disentangled UQ-paradigm that also provide data uncertainty, we also infer a set of uncertainty measure $\hat{\Sigma}_i^{\Theta}=\{\hat{\sigma}_i^{\theta}\}^{\theta\in\Theta}$ which is more link to data-variability.

- metamodel-based OOD indicator: $I^{\Theta}_{ood}(x_i) = Var(\hat{Y}^{\Theta}_i)$
- metamodel-based σ-norm OOD indicator: $I^{\Theta}_{\sigma-ood}(x_i) = \frac{Var(\hat{Y}^{\Theta}_i)}{E[\hat{\Sigma}^{\Theta}_i)]}$

We can then use such indicators and their distribution to make model-based characterization of datasets, or samples according to a nominal dataset of reference (X_{ref}).

- Dataset-representativeness metrics: $D(\mathscr{D}_{I_{ood}(X)}, \mathscr{D}^{ref}_{I_{ood}})$
- Meta-model-based ood measure:$1 - P_{I^{\Theta}_{ood}(X_{ref})}(0 > v > I^{\Theta}_{ood}(x_i))$

C.2.1.5 Indicators & metrics for unsupervised anomaly detection in regression

Uncertainty quantification can be used to refine unsupervised anomaly detection in regression, in order to perform contextual anomaly detection that consider uncertainty to ponder ate anomaly score.

Unsupervised meta-model-based residual anomaly measure with Gaussian UQ:

- Average error (without UQ) $s_a(x_i, y_i) = |E(\hat{Y}^{\Theta}_i) - y_i|$
- z-score (normalized by total UQ): $z_a(x_i, y_i) = \frac{|E(\hat{Y}^{\Theta}_i) - y_i|}{\hat{\sigma}_i} = \frac{|E(\hat{Y}^{\Theta}_i) - y_i|}{\sqrt{E[\hat{\Sigma}^{\theta_i}]} + \sqrt{Var[\hat{Y}^{\theta}_i]}}$
- z-score with ML-risk aversion (epistemic penalization): $z^{\alpha}_a(x_i, y_i) = \frac{|E(\hat{Y}^{\Theta}_i) - y_i| + \alpha\sqrt{Var[\hat{Y}^{\theta}_i]}}{\sqrt{E[\hat{\Sigma}^{\theta}_i]} + \sqrt{Var[\hat{Y}^{\theta}_i]}}$

C.2.2 UQ under the Bayesian inference approach

UQ expresses uncertainty as a probability. In Bayesian inference, the epistemic uncertainty is described by the posterior distribution $p(\theta|\mathscr{D})$ where θ are the parameters of a model and $\mathscr{D} = (y_i, x_i)_{i=1}^N$ is the training dataset. We refer the reader to the Confiance.ai document n. 431E for further details and references of this section. Define:

- $p(\theta|\mathscr{D})$: posterior distribution over the parameters θ of f after observing $\mathscr{D}$
- $p(\mathscr{D}|\theta)$: likelihood of $\mathscr{D}$ given by f_θ
- $p(\theta)$:prior distribution over the parameters

Set $\mathscr{D} = \{(x^i, y^i)\}_{i=1}^N$. There are related via Bayes as

$$p(\theta|\mathscr{D}) = \frac{p(\theta, \mathscr{D})}{p(\mathscr{D})} = \frac{p(\theta)p(\mathscr{D}|\theta)}{p(\mathscr{D})} = \frac{p(\theta)}{p(\mathscr{D})}\prod_{i=1}^{N} p(y^i|x^i, \theta) = \frac{e^{-\mathscr{L}_{\mathscr{D}}(\theta)}}{p(\mathscr{D})}$$

Prediction of a label y^*, corresponding to a new input x^*, must contain both the aleatoric and epistemic uncertainty. It is provided by the **predictive posterior distribution** (PPD) $p(y^*|x^*, \mathscr{D})$ that is computed using

$$p(y^*|x^*, \mathscr{D}) = \int p(y^*|x^*, \theta)p(\theta|\mathscr{D})\mathrm{d}\theta = \mathbb{E}_{p(\theta|\mathscr{D})}[p(y^*|x^*, \theta)] \tag{C.8}$$

where $p(y^*|x^*, \theta)$ is the likelihood of an observation given a model with weights θ.

Good predictions are both precise and accurate, UQ metrics measure the prediction reliability/trustworthiness.

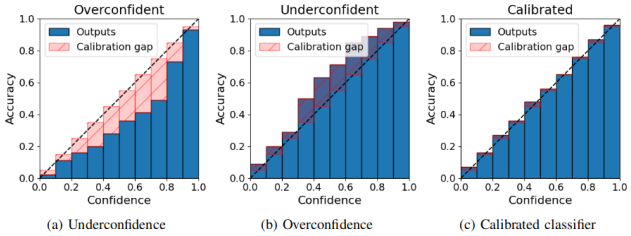
- The predictive power/capacity is measured by the likelihood $p(y^*|x^*, \mathscr{D})$. Using a test data set, we can compute the negative log-likelihood metric: $-\sum_i \log p(y_i^*|x_i^*, \mathscr{D})$
- The calibration of the predictions is probed by computing a reliability diagram. Calibration offers a powerful method for testing the quality of uncertainty estimates because well-calibrated predictions can be interpreted as objective probabilities.

Example on **classification** from Guo et al. (2017). To estimate the expected accuracy from finite samples, we sort and group predictions $p(y^*|x^*, \mathscr{D})$ into intervals and calculate the accuracy of each bin, meaning

$$\text{accuracy} = \sum_i \mathbb{I}(y_i - \hat{y}_i) \tag{C.9}$$

where $\hat{y}_i$ is the predicted label $\hat{y}_i = \arg_y \max p(y_i|x_i, \mathscr{D})$. The confidence for the same bin is defined as

$$\text{confidence} = \sum_i p(y_i = \hat{y}_i|x_i, \mathscr{D}) \tag{C.10}$$



(a) Underconfidence (b) Overconfidence (c) Calibrated classifier

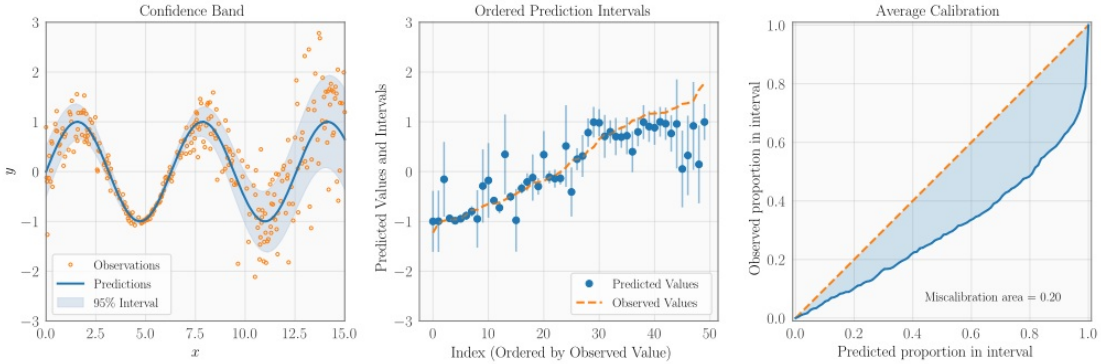
Figure C.3: From Gawlikowski et al. (2021). These diagrams plot expected sample accuracy as a function of confidence.

Other metrics can be computed based on these fundamental principles, see Psaros et al. (2022):

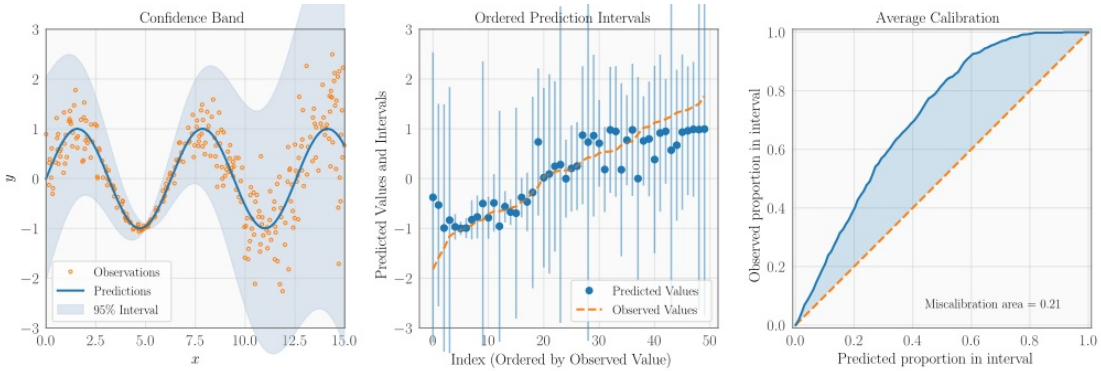
- Expected Calibration Error: $\langle|\text{accuracy} - \text{confidence}|\rangle$ with the average taken over bins.
- Maximum Calibration Error: $\max|\text{accuracy} - \text{confidence}|$ with the max taken over bins.
- Prediction Interval Width: for instance, $\langle\mathbb{V}[y]\rangle = \langle\mathbb{E}_{\text{ppd}}[(y - \mathbb{E}_{\text{ppd}}[y])^2]\rangle$ where the expected value are taken over the predictive posterior distribution.
- Prediction entropy $\mathbb{H}_{\text{ppd}} = \mathbb{E}_{\text{ppd}}[-\log p(y|x, \mathscr{D})]$

Example on **regression** from Chung et al. (2021), the resulting analogs for the calibration curves are as follows.

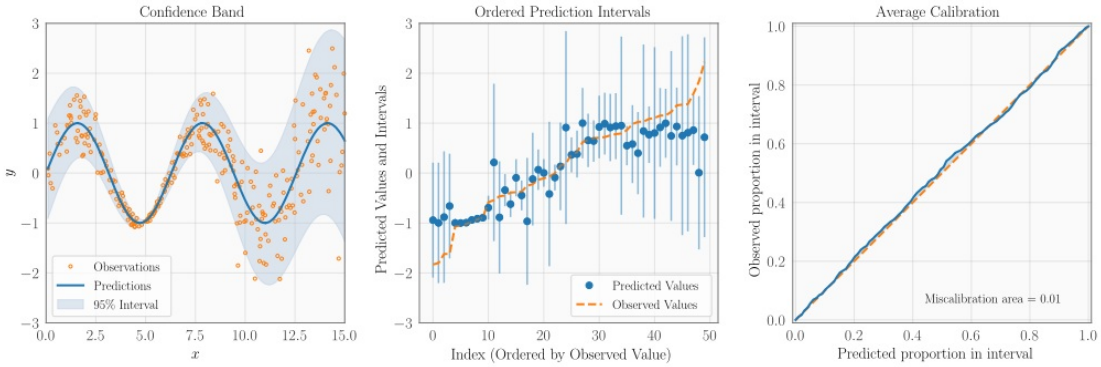
Overconfidence (too little uncertainty):



Underconfidence (too much uncertainty):



Well Calibrated:



C.2.3 UQ under the Conformal Prediction Approach

One of the major approaches to UQ relies on *prediction sets*, such as prediction intervals for regression problems. Let $f(\cdot)$ be a predictor for a target variable Y (scalar, discrete, etc.) given a generic input X. In a frequentist context, for instance, one can fix a significance level α (small, e.g. 0.05) and its relative confidence level $1-\alpha$, and look for the prediction interval $C_\alpha(X;f)$ that contains the true (unknown) value of Y with probability, say 95%. We refer the reader to the Confiance.ai document n. 431B for further details and references of this section.

In formulas, this is commonly expressed in two ways. The first, Eq C.11, known as "marginal validity", holds on average over all possible values of (X,Y). The second (and practically more useful) in Eq. C.12 holds "conditionally" on the realization of the random features X: this is what we generally try estimate with ML models, although it often requires strong hypotheses on the data and the model[3]. In the deliverables of the Confiance.ai program, the reader can find multiple references to CP: this approach guarantees to satisfy Eq. C.11 but not Eq. C.12, which is generally impossible without major hypotheses, hard to make for complex ML tasks.

$$\mathbb{P}\{Y \in C_\alpha(X)\} \geq 1-\alpha \tag{C.11}$$

$$\mathbb{P}\{Y \in C_\alpha(X) | X = x\} \geq 1-\alpha \tag{C.12}$$

C.2.3.1 Metrics for prediction sets

A natural metric of success is that of *empirical coverage*, that assesses on the test data whether the nominal validity of $1-\alpha$ set by the user is attained.

$$\text{Empirical coverage: } Cov(D_{\text{test}}) = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \mathbb{I}\{Y_i \in C(X_i)\} \tag{C.13}$$

Coverage gap: Let $\hat{r}_y$ be the class-conditional coverage (for instance, one can use empirical **risk**, e.g. cars having empirical risk of $\hat{r}_y = 0.923$ over all test data.).

$$\text{Coverage Gap}(D_{\text{test}}) = |(1 - Cov(D_{\text{test}})) - \alpha| \in [0, 1-\alpha] \quad \text{(0 is best).} \tag{C.14}$$

C.3. Monitoring

C.3.1 Rule Based Model ODD Characterization

Rule based Model ODD Characterization is obtained by testing ML Model robustness by sampling and perturbation.

Parts of this subsection are subject to a patent that was made within the Confiance.ai program Thales (2024).

C.3.1.1 Model ODD Characterization

- Let p be the chosen ODD parameters of interest to characterize.
- Let S be a subset of the training dataset
- For each ODD parameter p, let $M(p)$ be the chosen method to perturb the subset S.
- For each ODD parameter p let $R(p)$ be the range of intensity levels of perturbation. The intensity level should correspond to an input parameter of the perturbation method $M(p)$. This range should be wide enough to contain all the possible values that could be found in operation.

[3] For instance, this is the case for the usual linear model, with normally distributed errors and a correctly specified (linear) model. These hypotheses are often not verifiable for production contexts.

- Le $f(x)$ be a metric for inference evaluation. This metric can depend on the type of data and the type of problem. For image classification, F1 score or inference score of the predicted class can be used. For object detection, mean average precision can be more appropriate. This metric should be an indicator of the AI Model performance in the task(s) that it has been designed to resolve.

- Let T be the tolerance threshold: a percentage that the metric $f(x)$ should maintain for safe operation of the system.

There are 4 steps to characterize the Model ODD for p:

- Generate $N = n \times R(p)$ samples with the range $R(p)$ of perturbation from the subset S of n samples

- Run inference on those N perturbed samples

- Compute the metric of inference evaluation for each perturbed sample. For each range step k from 1 to R, compute the mean or median value of the metric on the N samples of the subset. This will give a curve C containing the reference metric on the range $R(p)$.

- Analyze the curve C for trends of stability, instability and zones where the tolerance threshold is maintained.

Steps 1, 2 and 3 are presented in Figure C.4 and give as output a Curve C representing the evolution of the reference metric per perturbation intensity.

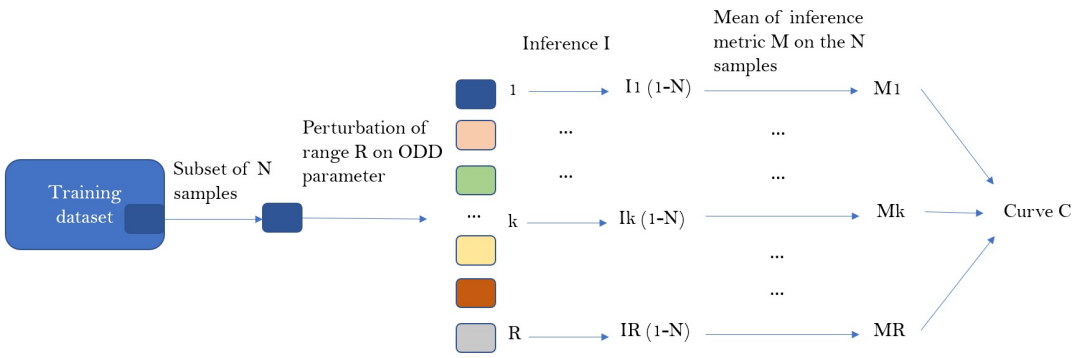


Figure C.4: Computing inference metric curve

This curve C is then analyzed by a set of rules to identify high level and low level zones.

High level zones:

- Model ODD Zone: where the model performance is mostly above the tolerance threshold.

- Out of Model ODD Zone: where the model performance is mostly below the tolerance threshold.

Low level zones:

- Lack of stability: instability zone due to oscillations or sudden change of regime.
- Local lack of robustness: zone where the model performance is below the tolerance threshold
- Model stability zone: zone where the model performance is above the tolerance threshold and relatively close to the base value, without instability or sudden change of regime
- Opportunistic ODD zone: zone where the model performance is above the tolerance threshold but this zone is separated from the stability zone(s) close to the base by one or several local lack of robustness zone(s).

Figure C.5 shows those high level and low level zones in a fictive curve C. When possible, the Business-driven ODD should be defined on the same range of perturbation intensities of the ODD parameter. This would allow to compare the Business-driven ODD with the high level zones. Some comparison zones could then be defined where:

- The Model performs as expected (B): When a portion of the Business-driven ODD zone fits with a portion of the Model ODD zone. The Model performs as expected except locally in (E) and (F) due to small lacks of stability.
- The Model under performs (A): When a portion of the Business-driven ODD zone fits with a portion of the Out of Model ODD zone.
- The Model over performs (C): When a portion of the Business-driven Out of ODD zone fits with a portion of the Model ODD zone.
- The Model is not robust (D): When a portion of the Business-driven Out of ODD zone fits with a portion of the Model Out of ODD zone. The Model performance is not trusted in this zone, even in an opportunistic ODD zone such as (G).

C.3.1.2 Rule Based OOD Detection empirical metrics

The detected measure d of a given ODD parameter in the input data is interpolated on a curve $C(d)$ of detection measures made on the augmented dataset for this ODD parameter during Rule Based Model ODD Characterization detailed in C.3.1. This interpolation gives a value I of ODD parameter intensity. This value I is compared with the ODD limits A and B with $A < B$ of this parameter and the uncertainty ranges $U(A)$ and $U(B)$ of those limits.

Fig. C.6 shows an example of curves (Model ODD curve C on the left and $C(d)$ (on the right) for the ODD parameter Brightness. On C, the ODD zone is starting at intensity 23.58 and ending at intensity 328.68. On $C(d)$, interpolating those values on the perceived brightness medians gives the calibrated thresholds starting at 10.75 and ending at 138.25. The perceived brightness detected on an input image from production will then be compared with those thresholds and an arbitrary uncertainty range to give an alarm, no alarm or potential alarm status.

The estimated detection uncertainty is obtained by computing the standard deviation of the calibrated measures. It can be visualized on the detection curve on Fig. C.7.

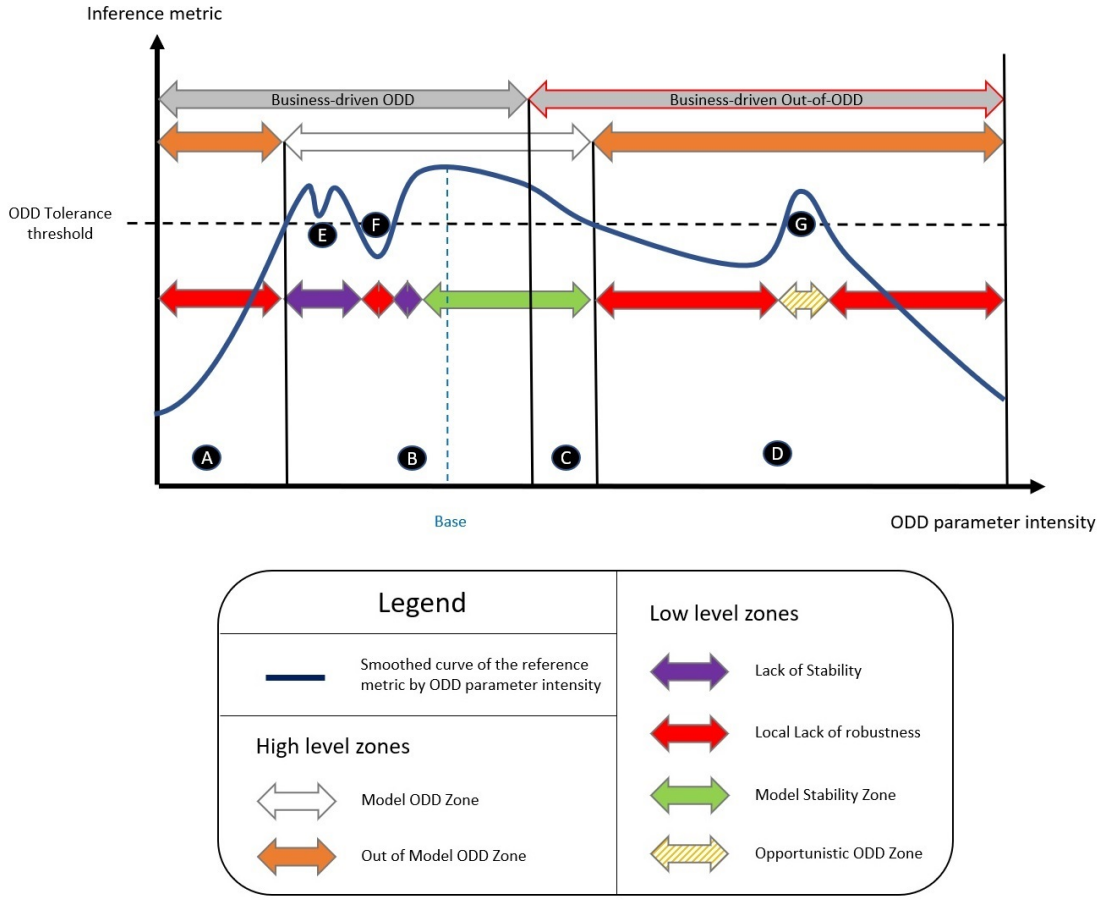


Figure C.5: Model ODD zones compared with Business-driven zones

C.3.2 The case of Classification/Regression

C.3.2.1 Time series window of detection

The detection is made on a window of detection with a size that should be close to the window used by the model during inference.

C.3.2.2 Images area of detection

The detection can be made on all pixels, or on a zone of interest chosen to contain most of the pixels used in average by the model during inference.

C.3.2.3 Smart monItoR VErdict SyNthesis (SIREN)

The SIREN metric is computed as follow:

- If $A + U(A) < I < B - U(B)$: The parameter is inside the ODD on the input data
- If $I < A - U(A) or B + U(B) < I$: The parameter is outside the ODD on the input data
- Else: The parameter is in the uncertainty range of the ODD on the input data

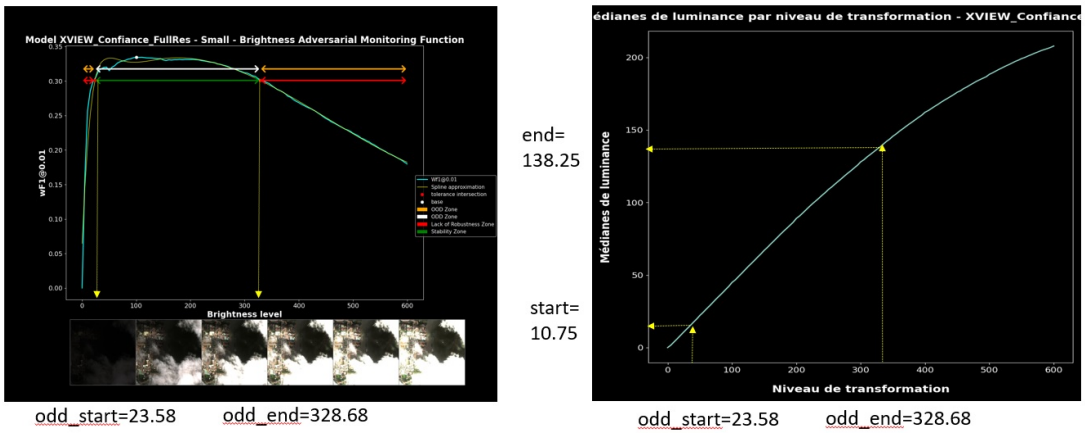


Figure C.6: Curves used to interpolate an estimation of an ODD parameter intensity

C.3.2.4 Input Integrity Monitoring Score (I2MS)

The I2MS metric is the level of trust that should be impacted on the Model output confidence based on the position on the Model ODD curve of the list of studied ODD parameters. It can be computed for each ODD parameter p by interpolating the value $V(p)$ on the Model ODD curve to retrieve the estimated impact $I(p)$ on the inference score with an uncertainty range $U(p)$. Then those impacts can be combined with weights $W(p)$ given to each ODD parameter into the I2MS. Each weight represent the priority given to the corresponding ODD parameter and the sum of all weights should be equal to 1.

C.3.2.5 The Prediction Uncertainty Monitoring Score (PUMS)

The Object Detection Probability (ODP) also known as *confidence score* or *objectness loss* indicates the epistemic probability given by the model that the object or input is of the predicted class. It can be obtained with a probe in the model or independently to remove any common mode of failure between the monitoring function and the monitored model, for instance by using a surrogate model.

$$PUMS = I2MS \times ODP$$

C.3.3 Object Detection (images)

For images, the I2MS is applied on a grid and computed for a given bounding box by combining the scores of grid cells intersecting with the bounding box based on their intersection over union.

For a given bounding box i, $PUMS_i$ metric is simply computed by the following equation:

$$PUMS_i = ODP_i \times I2MS_i \tag{C.15}$$

where ODP_i is computed by the model for this bounding box i, and $I2MS_i$ is a local metric detailed in C.3.2.4

The ODP indicates the epistemic probability given by the model that the object is present in a bounding box, and that this object is of the predicted class.

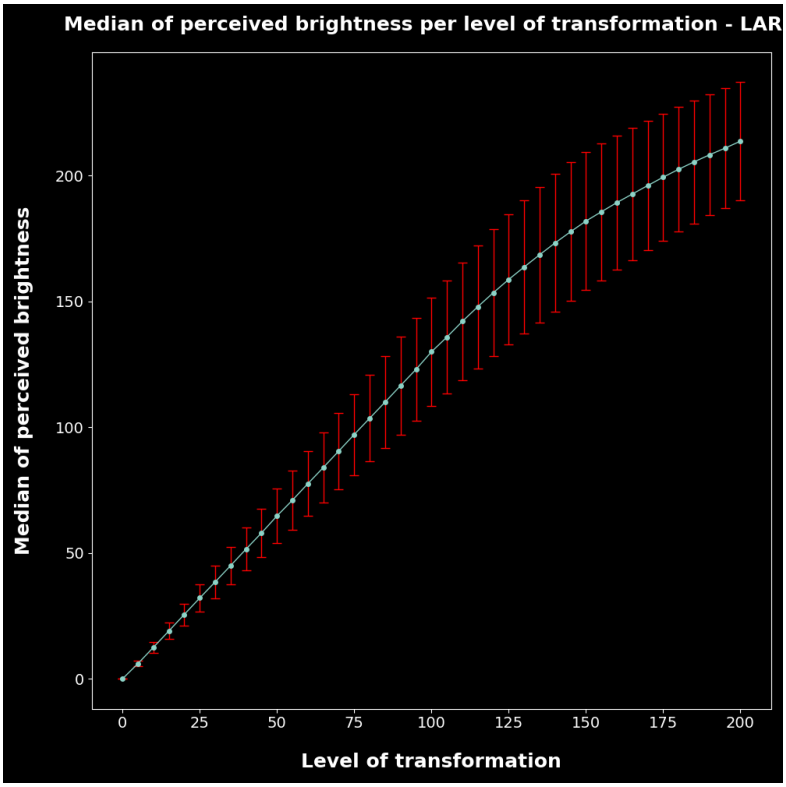


Figure C.7: Uncertainty on detection curve

For a bounding box $i \in \mathbb{N}^*$ in intersection with $j \in \mathbb{N}^*$ cell(s) of the image grid, for $k \in \mathbb{N}^*$ different types anomalies, and for $\alpha_1, \ldots, \alpha_k$ corresponding to real weights ($\alpha_k \in \mathbb{R}_+$ assigned to the different types of anomalies such that $\forall p \in [1,k], 0 \leq \alpha_p \leq 1$ and $\sum_{p=1}^{k} \alpha_p = 1$, the $I2MS_i$ metric can be computed by the following equation:

$$I2MS_i = 1 - \frac{1}{OBB_i} \times \sum_{p=1}^{k} \alpha_p \times \sum_{j} (PA_{j,p} \times (OBB_i \cap AMB_{j,p}) \tag{C.16}$$

where OBB_i is the area of bounding box i, $PA_{j,p}$ is the probability to have the anomaly p in the grid cell j, and $AMB_{j,p}$ is the area of the virtual grid cell j for the anomaly p.

C.3.4 OOD Monitoring

C.3.4.1 Definitions

Out-Of-Distribution (OOD) data is essentially data samples from a different distribution than what the model has encountered during its training phase. In the literature, OOD data is usually categorized between covariate shift and semantic shift:

- Covariate shift corresponds to a shift in input distribution but with the same output distribution (or labels). For instance, suppose a binary classifier for real images of cats and dog, blurry images of cats or sketch drawings of dogs are considered as covariate shifts: labels are preserved (cats and dogs) but the input distribution is different because of blur or different image domains.

- Semantic shift corresponds to variations in the output distribution, e.g. new labels. For example, images of other animals like penguins, or even completely different data like satellite images.

Moreover, it is also common to split OOD data from near-OOD (or hard-OOD) to far-OOD (or easy-OOD).

- Far-OOD corresponds to data that are very far from the ID distribution, making it easier to detect (like penguin images).
- Near-OOD data are more difficult to perceive because they are closer to the initial distribution, e.g wolf close to dog or cheetah close to cat.

OOD detection. A reliable AI Component should not only perform well on known in-distribution (ID) data but also be able to detect any OOD input as "unknown". OOD detection is the process to discern if an input belongs to the ID or OOD category.

Post-hoc OOD detection is a particular class of OOD detection based on pretrained neural networks. There is no need to train a specific OOD detector or to re-train an existing network. Post-hoc OOD detection relies on internal features and/or outputs of the neural network to estimate if the input is ID or OOD.

OOD score. The OOD detection is a binary decision. Detectors return an OOD score. If this score is higher than a threshold defined by the end user, the input is considered as OOD.

C.3.4.2 Metrics

Since OOD detection is a simple binary decision (In-Distribution vs. Out-Of-Distribution) based on a score and a threshold, standard metrics can be applied to assess detectors' performance: accuracy, confusion matrix, etc. The community has focused on three main metrics:

- **AUROC** (Area Under the Receiver Operating Characteristic curve) is a widely used metric for assessing binary classifier performance. It quantifies the detector's ability to differentiate between positive and negative samples across all possible decision thresholds. This metric is thus independent from a given threshold. A perfectly separable score between ID and OOD yields an AUROC of 1 (best reachable AUROC). However, a completely random OOD detector gives an AUROC of 0.5.
- **FPR at $x\%$ TPR**. The idea behind this metric is to measure the FPR (False Positive Rate) when one chooses a threshold that gives $x\%$ TPR (True Positive Rate). In simpler terms, it answers the question: "If I want to be really sure I catch $x\%$ of OOD data, how often will the In-Distribution (ID) data be wrongly classified as OOD?". We usually choose $x = 95\%$ and call the metric **FPR95TPR**. The lower FPR95TPR, the better the detector.
- **TPR at $x\%$ FPR**. This is equivalent to the previous metric but considering a fixed FPR value, e.g. 5% for **TPR5FPR**. 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?". The higher TPR5FPR, the better the detector.

It is important to note that FPR$\times$TPR and TPR$\times$FPR must be carefully chosen based on system

specifications. Depending on the error type (false positives or false negatives) and on the error tolerance, the user must choose the best corresponding metric for his/her use case.

C.3.5 Distribution shift monitoring

C.3.5.1 Data Distribution Shift

Distribution shift refers to changes in the data distribution that can manifest in various ways. In particular, Yang et al. (2021)proposed a taxonomy and coined the term generalized *Out-of-Distribution* (OoD) detection to encapsulate five related sub-tasks: anomaly detection, novelty detection, open set recognition, OoD detection, and outlier detection. According to the authors, common to these tasks is the definition of *In-Distribution* (InD) data distribution and the goal of detecting OoD samples. However, the proposed taxonomy focuses mainly on perception (computer vision) classification tasks.

Alternatively, Ruff et al. (2021)provide a definition that can be extended beyond classification tasks by formalizing the distinction between *normal* and *anomaly*. In this regard, the concept of *normality* is defined with a distribution $\mathbb{P}^+$ on the dataset space $\mathscr{X} \subseteq \mathbb{R}^D$ to represent the normal data behavior in a given task or application. In this context, a sample that deviates from such normal distribution—*an anomaly*—is then a sample $x \in \mathscr{X}$ that lies in a low probability region under $\mathbb{P}^+$. If we assign a probability density function (pdf) $p^*(x)$, the set of anomaly samples is defined as $\mathscr{A} = \{x \in \mathscr{X} \mid p^*(x) \leq \tau\}, \tau \geq 0$, where τ is a threshold in a way that the probability of $\mathbb{A}$ under $\mathbb{P}^+$ is sufficiently small to make a distinction a clear distinction between normal and anomaly samples. Besides, this taxonomy emphasizes the difference between *low-level sensory* anomalies and *high-level semantic* anomalies in the context of deep learning. Examples of *low-level* anomalies can be texture or artifacts on images. Instead, *semantic anomalies* can be images or objects from non-normal classes, i.e., anomalies in the label space.

Following previous taxonomies Ruff et al. (2021); Yang et al. (2021), we can define the data distribution as a joint distribution $P(X,Y)$ over the input (e.g., sensor) $\mathscr{X}$ and label $\mathscr{Y}$ spaces. Distribution shift can occur in the marginal distribution of the input data $P(X)$, the label distribution $P(Y)$, or both. In this section, the focus is on two types of shifts:

- **Covariate-Shift.** It refers to changes in the marginal distribution of the input (sensory) space $P(X)$. This can occur due to various factors, such as adversarial examples, domain shifts, or style changes. Covariate shift is commonly used to evaluate a model's generalization and robustness performance and assumes that the label space $P(Y)$ remains the same during testing.

- **Semantic-Shift.** Pertains to changes in the label space $P(Y)$. This can be due to the occurrence of new classes or categories or a new data modality in a regression task. Detection of semantic distribution shift is a primary focus in many detection tasks considered in this framework. In these tasks, the label space $\mathscr{Y}$ can be different between in-distribution InD and OoD data, and thus, the model should refrain from making predictions on OoD data to avoid potential errors.

Figure C.8 shows an example of a data distribution. In the figure, the InD dataset is the GTSRB[4] dataset, where reference (anchor) examples from the InD dataset are located in high-density

[4]GTSRB–German Traffic Sign Recognition dataset.

(dark) regions. Instead, covariate-shift samples occupy low-density regions (high-contrast or blurry InD samples). On the other hand, samples from completely different datasets (CIFAR-10, EMNIST) that have a different semantic space occupy extremely low-density regions.

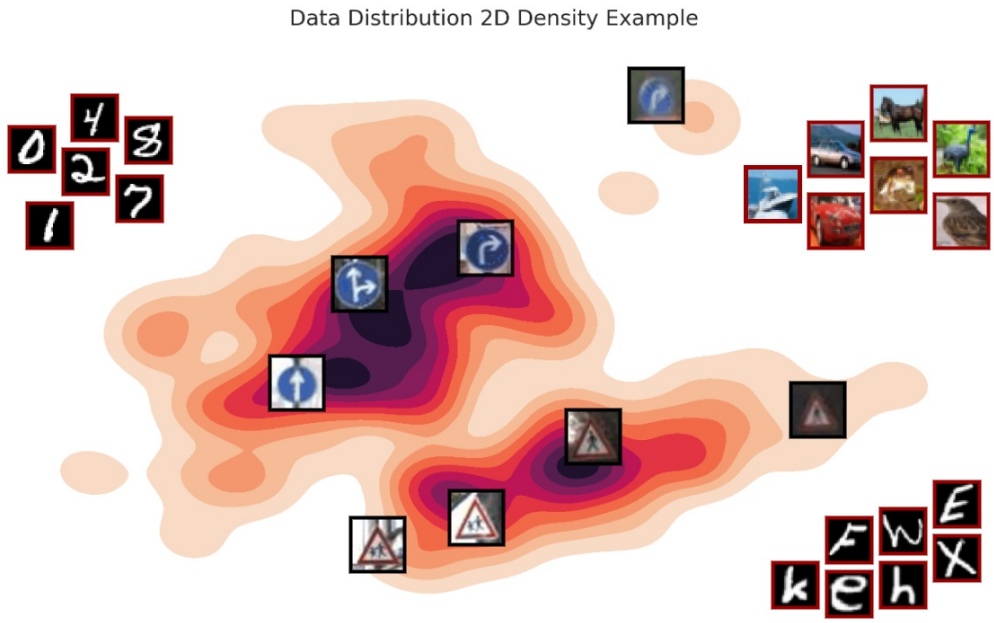


Figure C.8: Data distribution example. A didactic representation of data distribution shift with GTSRB as the InD dataset, covariate-shift samples in low-density regions, and semantic shift samples with OoD datasets such as CIFAR-10 and EMNIST.

Alternatively, Bayram et al. (2022), propose a fine-grained terminology focusing on data streams. In particular, the authors use the terms concept drift, concept shift, and data shift to express the data distribution changes $P_t(X,Y) \neq P_{t+w}(X,Y)$ two-time instances t and $t+w$—distribution shift in the previous taxonomies. The sources of drift are defined as follows:

- **Concept drift.** Indicates a change in the underlying target concept, $P_t(y \mid X) \neq P_{t+w}(y \mid X)$. This type of drift presents three subcategories. *Flicker concept drift* when some data samples belong to two different classes at two different times (ambiguity). *Severe or full concept drift* when the target data classes of all data samples change. Finally, *subconcept drift* occurs when a subspace of the data samples changes its target classes.

- **Covariate shift.** Indicates a change in the input data distribution without affecting the target concept $P_t(X) \neq P_{t+w}(X)$. This definition is the same as in the previous taxonomies.

- **Class prior probability shift.** Indicates a change in the classes distribution over time $P_t(y) \neq P_{t+w}(y)$. This drift can have two subcategories, *concept evolution* to refer to a situation in which new classes emerge, and *concept deletion* to refer to the disappearance of classes. This definition can be linked with *semantic-shift* definition from the previous taxonomies.

C.3.5.1.1 Data Distribution Shift Detection Problem Formulation Data distribution shift detection can be framed as a binary classification task. The classifier Ω aims at using a confidence score $\mathscr{S}$ with a corresponding threshold τ to determine (at inference time) whether a new input sample $\mathbf{x}^*$ belongs to the training data distribution or not (OoD, anomalous samples), as presented in Equation (C.17):

$$\Omega(\mathscr{S}(\mathbf{x}^*), \tau) \begin{cases} 1 & InD \quad \mathscr{S}(\mathbf{x}^*) \geq \tau \\ 0 & OoD \quad \mathscr{S}(\mathbf{x}^*) < \tau \end{cases} \tag{C.17}$$

Therefore, following the equation above, the goal is to derive a confidence score such that—*by convention in the literature*—positive InD samples have higher confidence scores and vice versa for OoD or anomalous input samples. Then, the classifier Ω uses the confidence score $\mathscr{S}$ to get a notion of trust in the DNN and elicit its verdict.

C.3.5.1.2 Distribution Shift Detectors based on Confidence Measures In the literature, several methods exist to detect data distribution shifts. However, post-hoc methods are of interest given that they aim at creating confidence scores (or measures) that have a minimal impact on a DNN model and its training process without altering the loss function. Some prominent post-hoc methods categories and their corresponding confidence scores are listed below. For more details, we refer the reader to the work from Yang et al. (2021):

- **Output based methods.** In this category, we have uncertainty-based scores such as predictive uncertainty (with Bayesian deep learning methods) and uncertainty metrics such as predictive entropy and mutual information. Other widely used confidence scores are the max softmax probability (MSP), max-logit, and energy score.

- **Feature processing.** These methods aim at processing representations or embeddings to improve existing output-based confidence scores, such as the energy score. Common methods in this category are ReAct, ASH, and DICE.

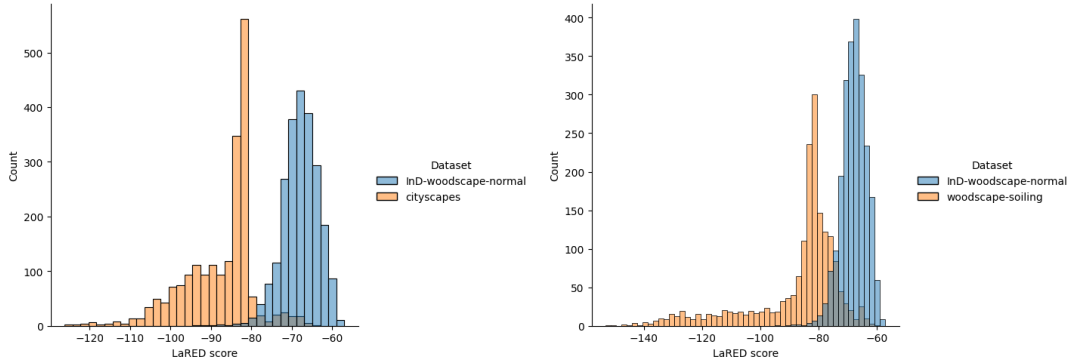
- **Distance-based methods**. This family of methods aims at using a distance to the empirical training embedding distribution of the penultimate DNN layer as a confidence measure. Common methods in this family are the Mahalanobis distance (parametric embedding density assumption) and the KNN k^{th} distance (non-parametric embedding density assumption).

- **Density-based methods**. This family is often employed in combination with generative models. Typically, the confidence measure is represented by the log-likelihood density of the feature or embedding density obtained with generative models.

Most of the methods described above can be related to the *Data distribution detectors* methods and approach described in Bayram et al. (2022). Note that this family or type of detector does not require labels or a concrete reference, as in the performance-based approach.

C.3.5.1.3 Distribution Shift Detectors Performance Evaluation For a quantitative evaluation, threshold-independent metrics are employed to assess the performance of distribution shift detectors. These metrics are:

- **FPR95** measures the false positive rate (FPR) of OoD samples when the true positive rate (TPR) of InD samples is 95%.



(a) Woodscape vs Cityscapes - Different dataset in the same context

(b) Woodscape vs Woodscape-Soiling: clean images vs images w/anomalies

Figure C.9: DeepLabv3+ trained w/Woodscape (InD): CEA's LaRED confidence score comparison for InD data and data with covariate shift. C.9a presents the confidence score comparison between InD and samples from a different dataset in the same context (Cityscapes–driving scenarios in a city). C.9b presents the confidence score comparison between InD images and sample images from the same training distribution that present anomalies due to environmental factors that impacted the camera lens during data collection (rating, fog, mud). In both figures, the InD confidence score is in blue, and the shifted samples (OoD) confidence score plot is in orange.

- The area under the receiving operating characteristic curve **AUROC**.
- The area under the precision-recall curve **AUPR**.

For a qualitative evaluation, the confidence score empirical distribution for both InD and OoD samples is plotted to show the confidence score separability. A recent performance metric, the Area Under the Threshold Curve (**AUTC**) Humblot-Renaux et al. (2023), proposes to quantify the overlap between the InD and OoD confidence scores distributions.

C.3.5.1.4 Distribution Shift Detectors During Operational Time During operation, a threshold τ is defined on InD data empirical distribution of the proposed confidence score $\mathscr{S}$, such that a high fraction of the InD data samples (e.g., 95 %) is correctly classified. Note that, in this family of detectors, the threshold selection does not depend on OoD data. To illustrate this point, Figure C.9 presents a comparison of the confidence score $\mathscr{S}$ (CEA's `LaRED` in this case) distribution for samples that come from the training InD dataset and samples with covariate shift. Both plots in Figure C.9 show that the proposed confidence score $\mathscr{S}$ respects the definition from Equation (C.17) that aligns with the convention in *SOTA* literature Yang et al. (2021, 2022).

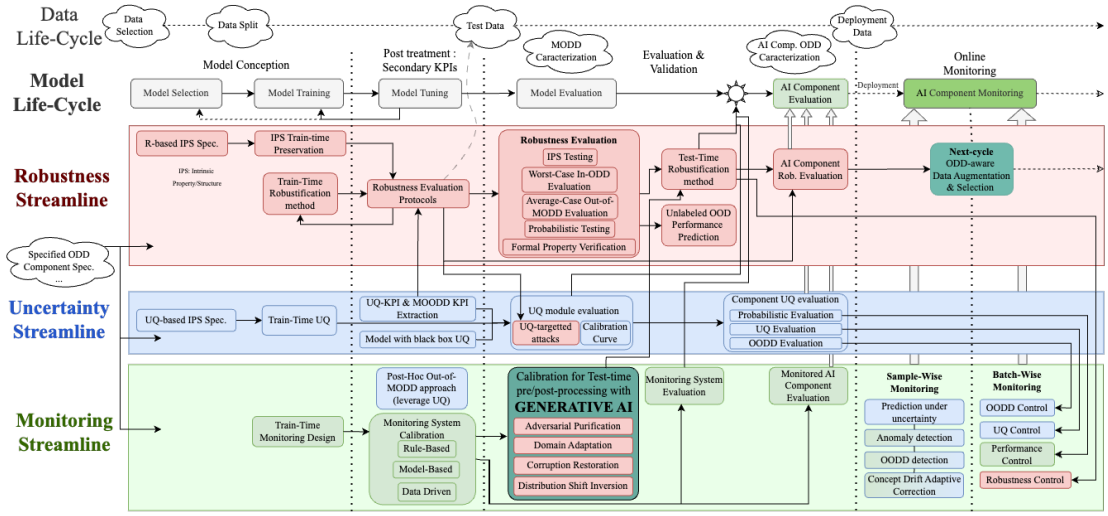
From a safety engineering perspective, the previous approach for threshold selection is naive because it does not consider that different contexts and situations from the operational design domain (ODD) definition might lead to different thresholds. Therefore, hazard and risk analysis (HARA) must be carried out for safe real-world deployment to identify high-risk situations and their relation with the proposed confidence score values. In this regard, the HARA should deliver safety invariants that can be formally expressed with predicates to obtain safety rules Machin et al. (2016), e.g., by employing a formal specification language such as signal temporal logic Leung et al. (2023) or more traditional approaches like statistical process control Bayram

et al. (2022), where the notion of time is also taken into account explicitly. Moreover, feedback collected from real-world deployment should be used to update the confidence score thresholds, eventually improve dataset ODD coverage, and later update the AI-based model.

D. The RUM Methodology

D.1. Introduction

In Chapter B, we sketched the functioning of the unified model's and AI component's life-cycle as illustrated in Section B.2. The detailed account of the AI component's life-cycle including the different attributes, indicators, metrics and KPIs developed in the last Chapter are illustrated in the following figure:



In comparison to its simplified version, this figure shows two main novelties that will extensively be developed in this chapter

1. There is a **multiplicity of lower-level concepts** that are grouped under the hood of the concepts presented in Section B.2.

2. There are elaborate **dependencies and intricacies** between the model's life-cycle with the R/U/M streamlines but most importantly, **between the RUM streamlines themselves**.

This will lead us to first explain the methodology organizing the lower level concepts that are brought up above and the necessity of the inter-dependencies in the RUM streamlines.

The structure of this chapter is as follows:

- In Section 2, we make a short preliminary on the concept of ODD Coverage that will be used later on;

- In Section 3, we present the RUM methodology as part of the AI Component's process. We show how the analysis method explained in Chapter B sheds light into more elaborate R/U/M attributes and we develop on how these new concepts help to better evaluate an AI component with respect to a specified ODD associated to it;

- In Section 4, we present the currently existing (to our knowledge) RUM aggregate attributes, which are strictly the result of evaluation analysis involving at least two out of the three R/U/M dimensions. We also present an algebraic formalism based on $(\max, +)$-algebra to obtain non-nominal RUM aggregates through MCDA;

- In Section 5, we develop an explanation on the necessity of the interlinks between the RUM streamlines by analyzing how the absence of one of these three aspects has a negative impact on the two others;

- In Section 6 we embed the different functionalities of the available Confiance.ai components into the RUM life-cycle integrated process.

D.2. A preliminary approach to the concept of ODD Coverage

Robustness as a scientific question: craft models that are robust relative to the biggest possible collection of perturbations. Robustness as a technical question: craft models that are robust relative to specified operation environment. As such, questions arising when considering techniques such as *universal adversarial attacks* will not be tackled here, as *universality* is inherently a scientific rather than a technical challenge.

Moreover, usually scientifically addressed questions need to assume strong conditions in order to better formalize this challenges as preliminary toy problems (such as adversarial training formalized in terms of ℓ_p-balls) which are already difficult to solve. One of the main assumptions done data-wise is that of independent and identically distributed (i.i.d.) data, as seen by the model at all training, validation and test phases. Nevertheless, this assumptions is rarely fulfilled and even two versions of what will be considered as *the same dataset* may distribute features in such a way that the model will show different behavior towards it.

As such, the technical and non i.i.d. approaches bring us naturally to consider, for a given performance metric and threshold, the extent to which the model covers data along ODD and perturbation parameters i.e. the parametric domains that can be validated as being *covered by the AI component*.

The taxonomy documents developed by the Confiance.ai program concerning the Operational Design Domain (ODD) can be found in SAE J3016 (2018) and Kaakai et al. (2023).

D.2.1 Scenarios preliminary terminology

We will articulate the distinctions between 3 different families of scenarios:

- *Potentiality scenarios*: background scenarios which have the potential of reuniting the global determination of possibility conditions. Usually determined in vernacular language.

- *Possibility scenarios*: scenarios determined as collections of possibility conditions. Usually determined in formalized terms (logical formulas, bounds, ...).

- *Effectivity scenarios*: scenarios seen as realized instances in the scope of the possibility conditions above-mentioned. Usually determined as a specific instantiation along the

predetermined logical terms.

Such distinction is inspired by the notions of "functional", "logical" and "concrete" scenarios from the self-driving car's (SDC) community, on which we built this more general version to handle more generic kinds of contexts beyond SDC. Potential, possible and effective scenarios can be dialectically seen in two opposite ways:

1. As a proliferation (bottom-up): here we interpret a potential scenario as a root from which there is a proliferation of multiple possible scenarios, each from which there are subsequent proliferations of effective scenarios.

2. As model extensions (top-down): here we interpret a potential scenario as the meet (or necessity) of all possibility scenarios and each possibility scenario as encompassing collections of effective scenarios.

For example, a potential scenario in Autonomous driving determined as "Right-turn" can be seen as a single scenario from where multiple possible scenarios proliferate (in the form of admissible spans of angle for turning, type of vehicle, speed,...) and from which effective scenarios will further proliferate as instances of possible combinations (45 degrees turn, bus, 20km/h,...). On the other hand, one can thing of the latter as elements of a set given as the subset of admissible spans, themselves as subsets of all potential scenarios that might start by a right-turn. In this case, the potential scenario "Right-turn" is not seen as a single scenario but as the set encompassing all the above-mentioned scenarios and virtually more than just those effectively sampled ones.

D.2.2 Structural constraints for scenario based on AI Component design

Pre-acquired

1. Data-level: the specifics of the data handed to the component (its type, characterization, qualitative assessment and metric).

2. Component infrastructure: the place of the central model with respect to auxiliary models that constitute the AI component (such as anomaly detectors, UQ modules, ...).

3. Component supra-structure: the AI system level specifications, hardware and physical components. This last point implies a reflection on the AI Component's embedability and deployment, and will not be treated in the RUM methodology, which will stop at the AI Component's level.

D.2.3 Structuring hierarchical levels of scenario determination

Our central object of study is that of an AI component. We characterize such a component as taking a pre-specified choice of type of inputs, a task, an output and having a core (or central) ML model performing such task.

1. (Level 1) Real scene description where the task is performed: global semantical aspects, part of the *Specified* ODD (ODDS) i.e. the ODD determined in a top-down approach from earler stages of the "W-cycle" of the End-to-End methodology developped in the Confiance.ai program.

2. (Level 2) Perturbation environment on which trustworthiness is tested: nominal perturbations as a part of the OODS

An important remark on what makes the difference between level 1 and 2 parameters is that level 2 parameters are infinitesimal with respect to level 1 parameters. In other words, change of level 2 parameters should not describe a progressive change from one scenery to another. This follows the intuition that, for instance, blur as an ODD parameter does not change a day-time scene to a night-time scene. On the other hand, brightness as an ODD parameter may change these scenes and so should be rather considered as a level 1 ODD parameter, unless one is given very little magnitude bounds on it so that it is "forced down" to a level 2 parameter. Since no such domain change is provided by the blur transformation, determining it as a level 2 parameter doesn't depend on its transformation magnitude.

In this case, by denoting the AI component as an application $f : X \to Y$, where X is a dataset relative to a scenario instantiation (given by level 1 ODD parameters) and Y are the component's responses relative to X, we consider

1. a thickening X' of X consisting on the perturbations in level 2

2. a thickening Y' of Y consisting on the responses of the component with respect to the perturbations

As such, given acceptability thresholds on the certainty of the responses, the coverage problem turns into specifying intermediate $X \subset X_{\text{rob}} \subset X'$ and $Y \subset Y_{\text{conf}} \subset Y'$, where Y_{conf} is an acceptable level of response uncertainty associated to a robustness zone X_{rob}. Such a pair $(X_{\text{rob}}, Y_{\text{conf}})$ will be called a "conformal thickening" of (X,Y).

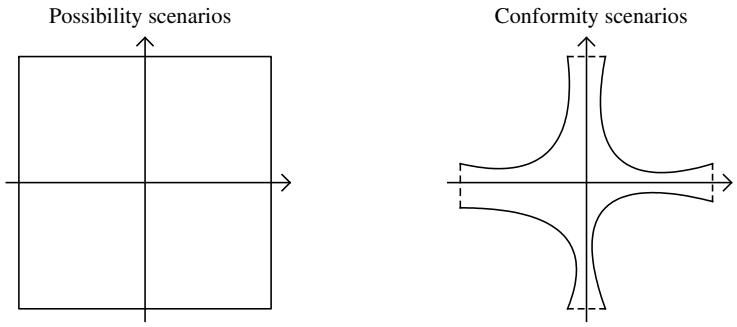
Under the assumption of a homogeneous treatment of the obtained parameters from these two levels, one has to think of level 1 as determining some sort high dimensional manifold $\mathscr{M}$ (possibly with corners, with multiple connected components,...) and level 2 parameters as a *thickening* of $\mathscr{M}$. In the 1D case where $\mathscr{M}$ is given by curves in the 2D plane, these thickenings are to be seen as **tubular neighborhoods**.

The advantage of this interpretation will be made explicit below, in terms of what is formally sampled as scenario instantiations so that the verification problem remains tractable.

Definition 5 *We define conformity scenarios as those possibility scenarios which are extensions of effectivity scenarios by adjoining conformal thickenings along predetermined level 2 perturbation ODD parameters.*

The objective for determining conformity scenarios is to reduce the high-dimensional sampling space from where to test effective scenarios to a shrunk subspace consisting of level 2 tubular neighborhoods along level 1 ODD parameters. Intuitively, the rationale is that a level 2 parameter such as blur will occur at a lesser magnitude in daytime than at night time, so if luminosity is considered as a level 1 parameter, the blur occurring at decreasing luminosity values is expected increase starting from small values. This implies that sampling sunny day scenarios with high blur are not conformity scenarios and can be discarded as such while raising a new specification relating low blur to "normal" conditions and leaving the specification of high blur in sunny days to what would be considered as "abnormal" conditions. The former should be part of the ODD

while the latter shouldn't, as they may correspond to situations such as blur resulting from smoke due to fire in a car's engine. For simplicity, consider that the specified ODD consist on two commensurable level 1 parameters and, associated to each, two distinct (but possibly overlapping) perturbation parameters. Then sampling effective scenarios in order to cover all possibility scenarios would amount on simulating below the left-hand side rectangle while sampling them to cover conformity scenarios amounts to simulating the left-hand shrinkage:



It is then straightforward that the more level 1 ODD parameters are needed to determine logical scenarios, the complexity needed to sample from whole volumes will increase exponentially. But in parallel, the ratio between the volume of the shrunken space of conformity scenarios and that of possibility scenarios sharply decreases.

D.2.4 Formalizing possibility and effectivity coverages

Under the assumption that the correspondence between exigencies, system-level ODD parameters and functional scenarios has been established and is consistent, this in turn can be seen as a problem of checking a well-defined property and quantifying the coverage in the tested data in which such property is satisfied.

The string of scenarios explained above is denoted

$$U_{\text{eff}} \subset U_{\text{cf}} \subset U_{\text{pos}} \prec U_{\text{pot}}$$

where U_{eff} is a (countable) set of effective scenarios, U_{cf} is the conformal thickening of U_{eff} by level 2 type of ODD specifications, U_{pos} is the (virtually infinite-continuous) set of possibility scenarios and U_{pot} is a universe of potential scenarios. Knowing that, by our assumptions, the relation between potential and possibility scenarios is not addressed in this document, one has non-symmetric notions of

- Covering $\mathscr{C}_{p,c}$ of possible scenarios by conformal scenarios i.e. U_{pos} by U_{cf}
- Covering $\mathscr{C}_{c,e}$ of conformal scenarios by effective scenarios i.e. U_{cf} by U_{eff}

D.2.4.1 Leveraging Diffusion Models for understanding the geometry of data manifolds and their Tubular Neighborhoods

In the recent work Sakamoto et al. (2024), an innovative use of Diffusion Models allowed a better understanding of the tubular neighborhoods surrounding the studied data manifold. In particular, they establish a observable relations between the behaviour of the score function underlying the diffusion models and the tubular neighborhoods, showing promising avenues

to leverage diffusion models as a way of generating samples lying precisely in such tubular neighborhoods. Although the approach we propose in the RUM methodology directly links to tubular neighborhoods of level 1 ODD parameters, the latter are tightly related to such data manifolds. As such, the prospect of utilizing diffusion models for smart synthetic data generation both along the lines of tubular neighborhoods in the data manifold (outside the case of singularity points) and corresponding to such neighborhoods in the ODD parameter space should be the subject of further research.

D.2.4.2 Other sampling methods for ODD effectivity scenarios

As we also assume that there is a well-specified correspondence between the above scenarios that concern the ODD and the scenes that concern the inputs of the AI component f

$$\begin{array}{ccc} U_{\text{eff}} \subset U_{\text{cf}} \subset U_{\text{pos}} & \Longleftrightarrow & X_{\text{eff}} \subset X_{\text{cf}} \subset X_{\text{pos}} \\ u & \longleftrightarrow & p \end{array}$$

where u is a single scenario in correspondence with a single input parameterized by p. Now assume further that the problem of sampling from X_{eff} and from X_{cf} follows a tractable procedure, then the characterization of both $\mathscr{C}_{p,c}$ and $\mathscr{C}_{c,e}$, for the purpose of AI component's *qualification* concerns two phases: testing and falsifying

$$\begin{array}{r} \text{check } \Phi(f,p,u) \vDash \psi_{\text{eff}}, \text{ for all } u \in U_{\text{eff}} \\ \text{check } \Phi(f,p,v) \vDash \psi_{\text{cf}}, \text{ for all } v \in U_{\text{cf}} \\ \text{find } u \in U_{\text{eff}} \text{ such that } \Phi(f,p,u) \nvDash \psi_{\text{eff}} \\ \text{find } v \in U_{\text{cf}} \text{ such that } \Phi(f,p,v) \nvDash \psi_{\text{cf}} \end{array}$$

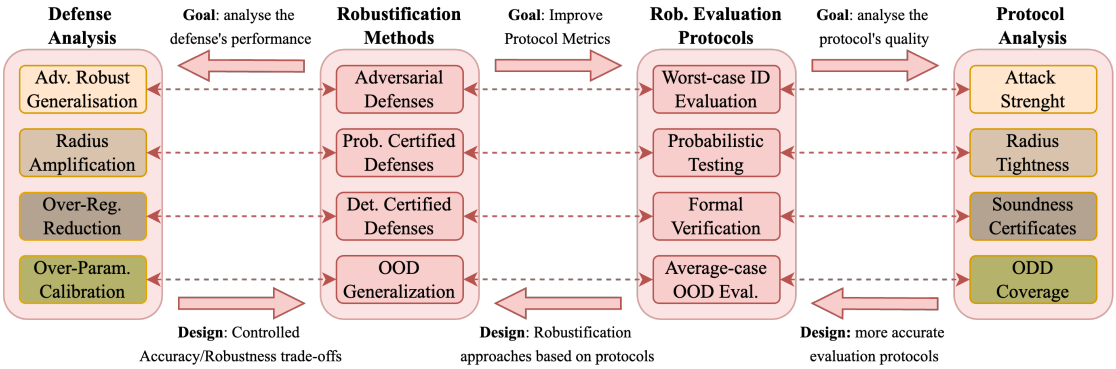
Gibbs sampling has shown to propose a good balance between speed and coverability for sampling scenarios this way, although this approach might not be able to access some disconnected regions of interest lying inside the space of possibility scenarios.

D.3. The RUM methodology as part of the AI Component's process

In order to give a account on the first point announced in the introduction of this chapter, we need to dive into the detailed development of the RUM analysis for R, U and M as was presented in Section B.2. Recall that the RUM analysis consists on four general and interlinked categories. As a reminder, the corresponding categories for robustness evaluation were coined "Defense analysis", "Robustification methods", "Robustness Evaluation Protocols" and "Protocol Analysis".

The key idea to retain, as will be shown in the subsections below, is that there are different sub-categories of these 4 different categories but most importantly, **the link between these subcategories are organized in the form of chains of correspondences**, where, for instance, the first subcategory of the first category is in mutual correspondence to the first subcategories of each of the remaining categories.

D.3.1 RUM-based analytic Robustness attributes



Before going through the specifics of each of the four above-mentioned correspondences, let us explain the global mechanism concerning robustness, and which will *not* be the same concerning UQ and Monitoring.

Here we start from a specific evaluation protocol which outputs metrics in the form of numbers that are results of the evaluation protocol. These numbers are not the same as quantities (such as those in UQ), and are not exploited directly. Rather than that, robustification methods are designed based on these protocols and whose goals are to improve the corresponding protocol metrics. For the same couple (robustification method, evaluation protocol), a second couple is created which analyses the actual ability of the first couple to truly address the conceptual attribute it aims to address. Therefore, taking the first line, for a couple (adversarially trained model, adversarial attack protocol), its analysis will be made on

- how much the adversarial defense actually generalises to available unseen adversarial threats,
- how much the adversarial attack protocol actually addresses “worst-case ID robustness”.

D.3.1.1 Attack strength and adversarially robust generalizability

In the last chapter, we saw that a worst-case ID robustness evaluation protocol implies crafting adversarial attacks targeting the model’s performance in the worst possible way. As such, evaluating the same model against a chosen threat model, say PGD, analyse its drop in performance in comparison to an adversarially trained one, and study the performance degradation evolution at increasing noise budgets constitutes an intermediate analysis which is neither worst-case nor average-case, but which gives a first characterization of the attribute we want to call *attack strength* and which measures how much a specific evaluation protocol approaches or not the currently available worst-case attack existing at present and which constitutes the baseline of what *worst-case ID robustness* means at present time. The important point here is that this is an attribute specific to an evaluation of the protocol itself, in its capability to such address worst-case ID robustness.

What follows induces an impact on the correct usage of the RobustML component from Confiance.ai’s program.

D.3.1.1.1 Application to the Confiance.ai Naval Group Use-Case. To give a concrete example of this, we split Naval Group Anomaly Detection UC data into a training and a test datasets. Then we trained a CNN-based architecture on the training dataset in two ways: first with normal training, and, second, with a specific adversarial training only consisting on weak FGSM attacks. We denote (f, f') these two models respectively. Notice that f' is not using the current state of the art defense mechanism either, meaning that its defense strength is not *optimal*. The below figure shows the evolution of the performance of both trained models against PGD attacks of increasing noise budget on the test dataset. The implemented code can be found at https://git.irt-systemx.fr/confianceai/ec_4/uc-naval-robustesse.

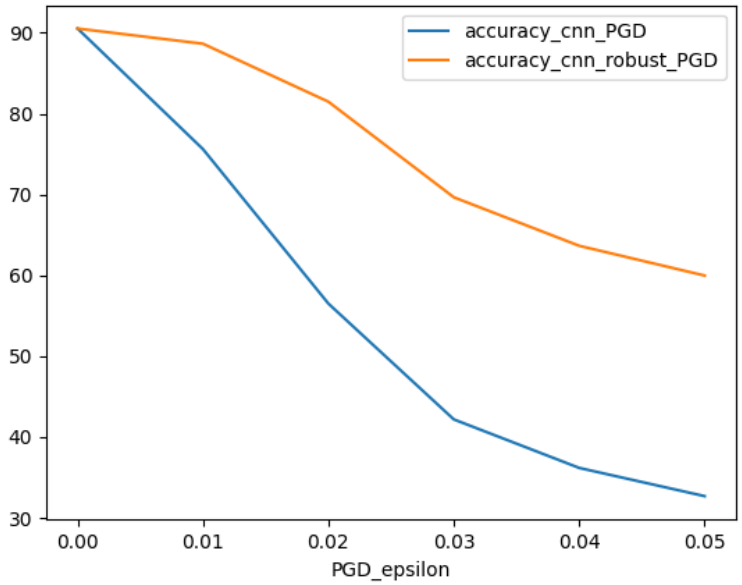


Figure D.1: Evolution of accuracy of the couple (f, f') while varying PGD attack intensity.

D.3.1.1.2 Attributes. As we saw above, on the one hand, finding better verification methods for property (C.1) is not the same of finding methods that will enlarge δ^* (and thus improving the local robustness of a system). A same model f can be shown to have a bigger stability locus because the verification method approaches δ^* in a better way than previous methods but this does not mean that f has been robustified. On the other hand, it provides an improvement towards the evaluation of defense methods that might over-estimate their effectiveness.

This motivates the following trust attributes concerning uniquely the quality of the evaluation protocol and of the defense method, and which has been the subject of the works Rice et al. (2020); Tsilivis et al. (2024).

- **Benign overfitting**: The observed phenomenon of deep learning where a model tends to generalize well even if it was trained for many epochs, to the point of perfectly fitting the training data.
- **Robust Generalization Gap**: The observed phenomenon of robust overfitting as an ob-

struction to *robust generalisation* ie: the fact that overfitting to the training set does in fact harm robust performance in adversarial training. See figure D.2.

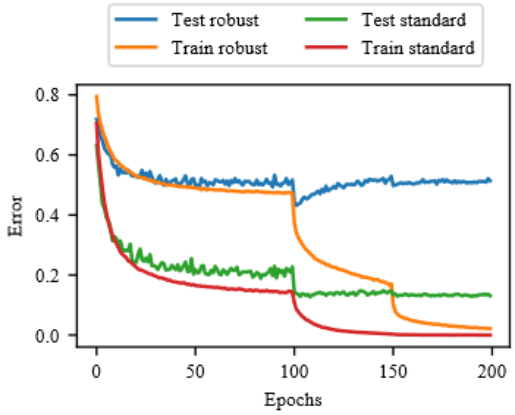


Figure D.2: From Rice et al. (2020), The learning curves for a robustly trained model replicating the experiment done by Madry et al.(2017) on CIFAR-10. The curves demonstrate "robust overfitting"; shortly after the first learning rate decay the model momentarily attains 43.2% robust error, and is actually more robust than the model at the end of training, which only attains 51.4% robust test error against a 10-step PGD adversary for ∞ radius of $\varepsilon = 8/255$. The learning rate is decayed at 100 and 150 epochs.

Remark: Classical approaches to combat overfitting like early-stopping, l_2 weights normalization, data augmentation ... seems to be effective against this phenomenon.

- **Adversarial training**: A defense method used to improve the robustness of a model against a **set** of adversarial attacks. It consist of a two-fold optimization process: a maximization of the strongest perturbation inside a L_p-norm ball B and a minimization of a loss-function over the perturbed data.

Robustness-Accuracy Trade-off

In adversarial robustness literature, not limited to RS, there is a focus on increasing the robustness bounds of a classifier. While this seems to be an optimal goal, there has been literature suggesting that increasing robustness comes with a tradeoff in accuracy Zhang et al. (2019); Tsipras et al. (2019). It has been studied for adversarial robustness in much detail, including theoretical considerations. In Zhang et al. (2019), the authors study the accuracy costs of improving robustness in a model. This was further explored in Tsipras et al. (2019). Both empirically and theoretically, it is seen that robustness and accuracy exist in a trade-off, contrary to what one would expect intuitively. Randomized smoothing is also vulnerable to this trade-off.

Adversarial robustness generalizability

Models trained to improve their robust accuracy with respect to attacks corresponding to chosen threat models do not in general improve their robust accuracy to *unseen* attacks i.e. attacks that were not seen during training. As such, one can define **adversarial robustness generalizability** as the capacity of a model to generalize their robustness beyond attacks that were seen during training.

D.3.1.1.3 Metrics: Let D be a data distribution, S a data set of n samples drawn i.i.d. from D, z a subset of S and M a model with a set of parameters θ.

- **Empirical risk:** $\mathbb{E}_{z\in S}[l(\theta,z)]$
- **Adversarial loss:** $h(\theta,z) = \max_{z'\in B_\varepsilon(z)} l(z',\theta)$, with ε the adversarial budget.
- **Adversarial population risk:** $R_D = \mathbb{E}_{z\in D}[h(\theta,z)]$
  It is a theoretical quantity corresponding to the expected adversarial loss on the true data distribution.
- **Adversarial empirical risk:** $R_S = \mathbb{E}_{z\in S}[h(\theta,z)]$
- **Adversarial Robust Generalization Gap:** $\varepsilon_{gen} = R_D(\hat{\theta}) - R_S(\hat{\theta})$, with $\hat{\theta}$ the set of parameters after the adversarial training.

  The Adversarial Robust Generalization Gap represent a measure of how well a neural network can perform on unseen adversarial data, from a list of perturbations, potentially unseen during adversarial training.

D.3.1.1.4 Validation threat model attributes and quality metrics

- **Effectiveness**: strength of the attack as measured by the ratio between clean and robust accuracy on the target AI Component. Effective threat models incur into reliable worst-case analysis and ensures that adversarial robustness is not being over-estimated.
- **Imperceptibility**: extent to which a threat model bypasses implemented detection mechanisms as given by the ratio of attacks that bypass them. Contributes to the overall effectiveness of the threat model.
- **Adaptiveness**: ability of a threat model to operate by leveraging information about the implemented defense mechanisms. Measured as a ratio between their attack success rate and that of their underlying *static threat model*, which leverages only information about the target ML model.
- **Efficiency**: inverse of the computational complexity necessary for the threat model to perform its attack.
- **Reproducibility /transferability**: ability of a threat model to generalize its effectiveness across different target AI Components performing the same task. Transferable attacks have a high potential risk of appearing during deployment.
- **Realizability**: existence of at least an instance of an adversarial input belonging to the threat model in the context of the real operating system conditions. Realizable attacks have an effective risk of appearing during deployment.

Overall, the above attributes and corresponding metrics allow to assess the quality of the threat model. This in turn allows to better approach a solid worst-case analysis of input perturbation sensitivity of a ML model seen as a the central model inside of an AI Component.

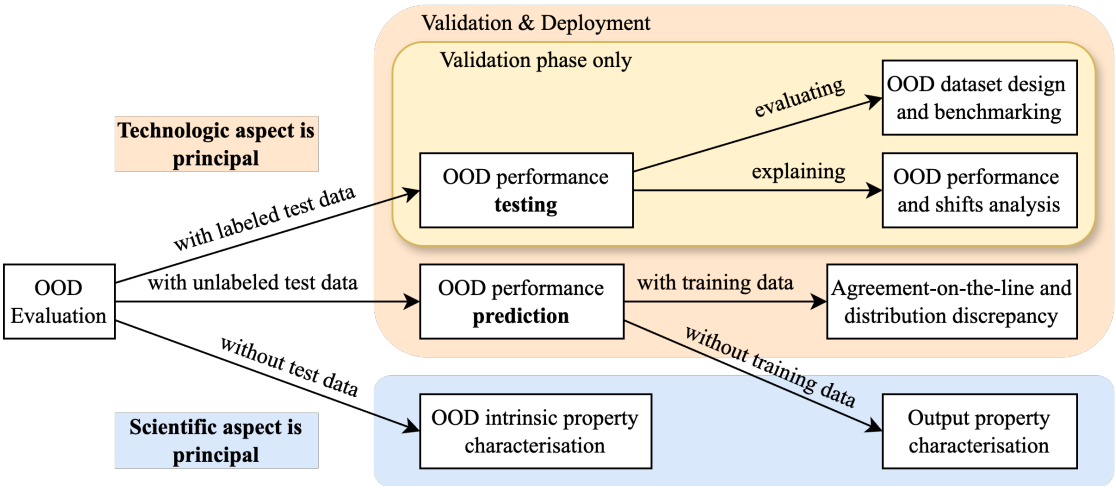
D.3.1.1.5 Validation defense mechanism attributes and quality metrics

- **Relative strength**: effectiveness of the defense mechanism against a specified threat model as measured by the ratio between the system's robust accuracy before and after the defense mechanism is in place.
- **Generalizability**: ability of the defense mechanism to generalize across unseen threat models as measured by the difference in the model's error between initial threat model according to which the defense was crafted and possibly numerous and distinct validation threat models.
- **Genericity**: (in the context of foundation models) the ability and extent to which a defense mechanism transfers to down-stream tasks Schlarmann et al. (2024).
- **Fidelity**: the ability of the defense mechanism to maintain the initial model's performance as measured by the ratio between clean accuracies before and after the defense mechanism is in place. In the context of norm-based attacks, there is a theoretically principled trade-off between accuracy and robustness
- **Relative δ-Certifiability**: ability of the defense mechanism to provide robustness certificates relative to an evaluation formalism (usually dependent to a normed ball with a pre-specified radius) given a set of inputs. Here, the hyper-parameter δ signifies the probability at which the certificate holds (usually 99.95% for randomized-based defenses). When $\delta = 100\%$, we say that the relative certificate holds deterministically (usually the case for IBP-related defense mechanisms, see Kotha et al. (2024)).
- **Relative resilience**: the ability of the defended system to slowly decrease its performance as the threat model perturbation budget grows. It is important to notice that recent methods have shown to exploit the structure of certifiable defenses in order to make these methods not resilient at all i.e. with a robust accuracy quickly decreasing to 0 right past the certification radius.
- **Relative effectivity**: the ability for a defense mechanism to successfully defend against realizable adversarial attacks. Notice that models (not necessarily endowed with defense mechanisms) may already provide non-trivial certified accuracies for small enough certification radius in some norm. The utility of certificates is tightly related to their relative effectivity.
- **Efficiency**: computational overhead of the defended mechanism compared to the undefended one.
- **Adaptiveness**: ability of a defense mechanism to modify its behavior towards inputs in test-time.

D.3.1.2 Average-case Out-of-Distribution Robustness Analysis and ODD coverability

Among the challenges, one that poses a substantial obstacle is the problem of generalization against distribution shifts, often referred to as Out-of-Distribution (OOD) generalization. This is most prevalent because current algorithms heavily rely on the IID assumption, i.e. test data and training data should be independent and identically distributed, but distribution shifts are

almost everywhere and the work Hendrycks et al. (2021) gives ample evidence that there is no known single method that consistently improve OOD generalization. Moreover, shifts of different nature may appear simultaneously but there is no notion of a shift encompassing all such simultaneous shifts at once.



- OOD Performance Testing: the main concern is about exploiting available labeled test data (either real or synthetic) to test the model's performance and degradation, and analyze the underlying reasons of it and interpret distribution shifts. Ensure:
  - principled environment partitions according to the existing distribution shift
  - there is no test data leakage from pre-training and oracle model selection
  - Performance analysis is aware of subpopulation worst shifts
- OOD Performance Prediction: in the presence of unlabeled test data, no direct testing is possible. The main concern here is thus to *predict* the model's performance on such data.
- OOD Intrinsic Property Characterization: in the absence of *any* test data, neither testing nor prediction are applicable. Thus, the main concern here is to check whether the model satisfies intrinsic properties that are hypothesized to facilitate generalization under distribution shifts. Among methods, we find distributional robustness, invariance, flatness and, to some extent, sharpness. It is worth noticing that recent works call for a reserved view on these methods by purposely crafting OOD test sets that violate the universality of their claims.

D.3.1.2.1 Refactoring ODD evaluation through the lens of scientific and technological research aspects Simply put, this concerns the dialectic opposition between the particular and the general, that is, between searching to exploit the most of a specific situation and placing its generalizability potential as a second priority, and searching to elucidate the generality of a specific phenomenon seen in a situation in a way that is best independent of that particular situation. As a rule of thumbs, the more general an approach is, the less it can exploit available particular data so the weaker the result will be. Needless to say, either fully neglecting generalizability or specificity yield limited or negative industrial interest. For instance, Sharpness-Aware Minimization was believed to indicate OOD generalization without the presence of any test data. But

recent works have shown to contradict a negative correlation between sharpness and OOD generalization precisely by showing that strong data augmentation may improve OOD generalization while sharpening the landscape. Finally, other definitions of sharpness such as in Zou et al. (2024) do provide a theoretically grounded correlation between sharpness (in their definition) and generalization.

Nonetheless, there are beneficial and non-beneficial trade-offs between these two aspects concerning the industrial usability of what they produce. We shall make the assumption here that industrial use-cases in need for OOD evaluation have at their disposal

- a source training set and a target trained model to be evaluated
- collections of proxy labeled OOD datasets (usually publicly available) and models also trained on the same training data
- a target unlabeled OOD dataset, corresponding to in-deployment data, which is only partially accessible (if not though samples, through a data specification process)

In light of this hypothesis, it becomes clear that

- OOD intrinsic property characterization and property characterization put the scientific aspect as principal. For the former, their techniques are aimed to cover the most OOD theoretically possible scenarios. For the latter, their techniques aim to be independent of the given training data, an assumption that corresponds to the unrealistic industrial scenario of not having the training data at our disposal.
- Agreement-on-the-line and distribution discrepancy, as OOD performance prediction methods, are fully aligned with the above hypothesis and maintain their technological aspect as the principal one.

Notice that these two principal aspects can be switched. On the one hand, a technique that puts the scientific aspect as principal can immediately be applied to a particular scenario and will show non trivial technological interest as long as the latter is not strictly covered by a stronger & less general technique. On the other hand, for each method that puts the technological aspect as principal, one can ask questions about different levels of generalizability of such techniques. For instance, one can ask about the extend to which AGL can address ODD intrinsic property characterization.

While applied research usually sees novel methods an intricate mixes between there two families of techniques, the fact that they will fall in one of these two categories remains unchanged as it solely depends on whether train and/or test data was used or not in any part of the process.

Finally, we highlight the fact that the focus here is assumes the technological aspect as principal and if we were to change this, the interests would change too as it is straightforward that general methods about OOD evaluation tend to dig deeper into the scientific question about the core essence of what OOD generalization is theoretically.

D.3.1.2.2 Test Sets Protocols & Design Testing a model's performance on a specified OOD dataset and interpreting the pertinence of the results of such tests are directly dependent on spe-

cific properties of such test sets. In order to choose among the numerous OOD test sets publicly available and exploit them in a principled way, the properties and types of such test sets should reflect the properties that are specified in the ODD-S. Most importantly, different evaluation protocols will lead to benchmarks with different raking positions of compared as shown in Yu et al. (2024). In other words, even with well-adapted test sets, bad evaluation protocols (such as inadvertently allowing data information leakage) can lead into suboptimal model selection. Let's quickly list different dimensions one would want to consider (for simplicity we only concentrate on visual datasets):

- **Types**: style change, background alteration, location modification, different sources, demographic attributes, and so on. These are usually separated and should be appropriately combined

- **Subpopulation shift**: datasets reflecting a domain-imbalanced context, intentionally establishing scenarios where the majority group exhibits a clear spurious correlation between the group label and the category label. These datasets mainly focus on assessing the performance of the worst-performing group, typically the minority group. As such, Evaluation protocols put particular emphasis on assessing the worst group performance

- **Domain generalization**: datasets are created with relatively balanced data across domains within the ODDS with emphasis of them *essentially not being seen* during training, emphasizing the average performance across all test instances. Evaluation protocols consist here on average-case performance after employing a leave-on-domain-out strategy, where each domain serves as the test domain while the remainder are utilized as training domains. Nonetheless, particular vigilance is needed to prevent data leakage Yu et al. (2024).

D.3.1.3 Radius Tightness and amplifiability

When visiting probabilistic certification protocols, we saw that these are themselves tied up with probabilistically certifiable defense methods, such using randomized or diffusion denoised smoothing to probabilistically certify that a normed ball around a point is robust (at very high probability). As for all certifiable methods, the effectively obtained radius for each point is an under-estimation of the optimal certifiable radius for the same point. As such, analysis the quality of the probabilistic testing protocol lies around its capacity to be near the optimal radius for that point or even using novel techniques Rumezhak et al. (2023) to enlarge the certified area around that point, in the form of ellipsoids. On the other hand, analysis of probabilistically certifiable defenses has at least two different aspects.

- On the one hand, vanilla Randomized Smoothing (RS) requires that the base model is obtained by noisy training (making it a training-based method), while Diffusion Denoised Smoothing does not need retraining neither the model or the diffusion model used to denoise inputs before preprocessing. In this regard, training-free methods are qualitatively more effective as they can be readily used on very large pretrained models without any need of retraining.

- On the other hand, the model that is used as a denoiser may have different performances *as a denoiser for RS*. For instance, Xiao et al. (2023) proposes better denoising mechanism allowing to amplify the radius of the certified ball around the input (under the same

evaluation protocol).

As such, analysing both the evaluation protocol and the defense will give us two different results: the protocol can be better if for the same evaluated model it can enlarge the area of certification while the defense can be better if for the same evaluation protocol producing a certified area, its radius can be extended by leveraging better denoisers, and if the latter were picked off-the-shelf or not.

Analysis of diffusion-based probabilistically certified robustness Recent work has further expanded the analysis to the consideration of more refined uses of diffusion models in the protocol such as Chen et al. (2024) and the use of consistency models for single-step certification Li et al. (2024). By analyzing their results, one can uncover a new trade-off between these methods, as the single-step methods, which are faster in inference, seem to show greater certifiable accuracy at lower σ values while multi-step methods, which are slower in inference, decrease slower than single step methods at increasing σ values.

Last, but not least, notice that the protocol for evaluating probabilistically certified robustness set-wise does not fully exploit the approximate radius computed for each sample, and simply takes into consideration those that are above a certain threshold. As such, the protocol is not aware of heterogeneity in the data and a richer protocol which informs of the statistics of the actual computed input-wise radiuses might be better to asses how data is distributed between parts having larger certified radiuses and parts having little such radiuses.

D.3.1.4 Soundness certifiability

Formal verification methods are prone to errors which raise the question on when is the guarantee, provided by the tools implementing formal verification, on whether an input formally satisfies a property or not, is actually true. In other words, formal verification tools have consistently mistakenly guaranteed that inputs are robust when, in fact, they are not. Thus, the analysis of the formal verification tool becomes the subject of a certification *of the verification tool itself*.

- Certificate Efficiency: A positive integer (the greater the better). This attribute quantifies the performance of the certification of the formal verification tools by counting the number of instances - defined by a property specification (pre- and post-condition), a network, and a timeout - conclusively verified within a its timeout. Notice that for VNN-COMP 2023 the timeout of the *verification alone* was fixed to five minutes and such competitions have not yet demanded for the competing verification tools to be endowed with certificates of their own.

- Certificate explainability: if the certification tool outputs a formal proof reflecting the way that the verification tool arrived to its conclusion, these should be understandable by humans as easily as possible.

- Certificate Integrity: the certification tool should certify the verification tool only on points that are either TP or TPs. In particular, the certification tool should not certify an incorrectly verified input.

- Certification Speed: the overhead of the couple (verification tool, certification tool of the verification tool) in comparison to the inference of the model whose point-wise robustness

is being verified.

On the side of the proposed formal robustification defense methods, criteria should be advanced on their ability to propose accuracy/robustness trade-offs *at the level of the certification tools*, specially at scale. To our knowledge, no systematic account of this question has been explored in the literature, and the work Desmartin et al. (2024) has left this question as a promising direction for future works.

D.3.1.5 Architectural and Algorithmic Designability (indirect robustness)

Definition 6 *Architectural and Algorithmic designability is the ability for a model to inherit theoretical properties or structure during its training stage.*

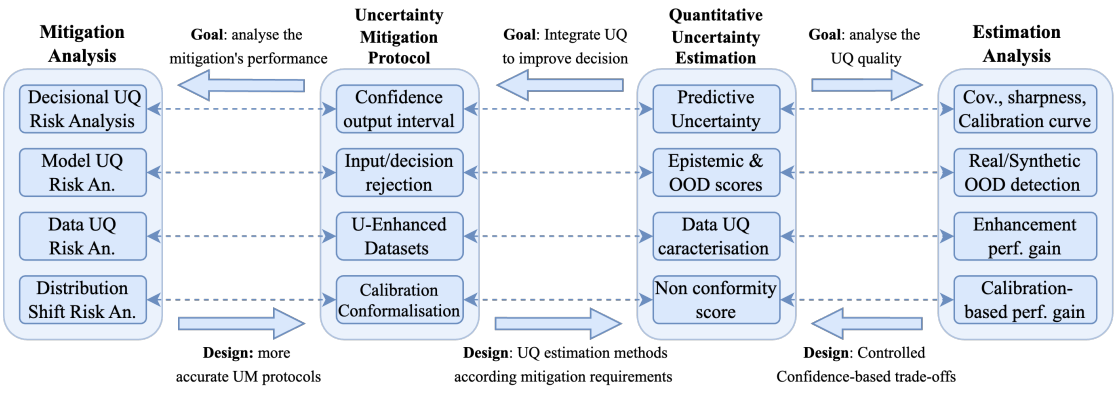
The idea behind this definition is that some theoretical properties cannot be enforced on specific ML architectures. For instance, attention mechanisms do not have a 1-Lipschitz counterpart and thus 1-lipschitzness as an inherent model property specification is not reachable by ML architectures with attention.

On the side of the defenses, since robustness is only indirectly addressed by 1-Lipschitz DNNs (by means of local stability), it is not clear what is the best adapted analysis for the evaluation protocol for this component. In particular, worst-case analysis shows that such DNNs are less robust than vanilla adversarially trained networks against FGSM on ℓ_∞-norm, but at the same time 1-Lip DNNs propose guarantees only for ℓ_2-normed attacks. As we saw in the section concerning attack strength, testing 1-Lip networks against ℓ_2-normed PGD attacks is a weak protocol if one wishes to range these networks as adversarially robust. On the other hand, these networks do not propose any particular secondary property which would make dedicated protocols (other than those already developed above) addressing new robustness attributes (other than those already developed above), to our knowledge. In particular, the certificates are done by means of

$$\varepsilon \geq \frac{p_A - p_B}{2}$$

where the top-2 class probabilities p_A, p_B are yet to be determined via Monte-Carlo, just as is done in the probabilistic certification protocol.

D.3.2 RUM-based analytic Uncertainty attributes



An AI Component that include ML-model has to cope with hazards link to the environment, the

data collection system, labelling process, and many others sources, that directly impact model conception and operation. Integrates into AI Component some uncertainty management mechanisms will confer more control and confidence about our model through the uncertainty attributes.

Before all, as we can statute that it is the aggregation of several forms of model robustness to disturbances of different natures that forge the many faces robustness, in the case of uncertainty, it's the management of several forms of uncertainty linked to sources of different natures, via dedicated mechanisms and mitigation processes, that gives to an AI Component the different dimensions available to deal with "uncertainty".

Then set up on AI Component some uncertainty management process related to specific sources of uncertainty, will allow to better handle the risks of error related to theirs impacts. In the following, after the common step as a unified procedure, we will quickly evoke four examples uncertainty management.

D.3.2.1 Uncertainty Management in General Case

As mentioned previously, integration of uncertainty managements in a AI Component rely on 4 stages:

1) Quantitative Uncertainty Estimation: aiming to produce an uncertainty measure that quantify the impact of some uncertainty sources on model output, or upon KPI metrics by performing an approximation using estimators fitted on empirical observations. Uncertainty estimators are design to catch the impact of some source of uncertainty on specifics quantities as uncertainty measures. For a same model impact of uncertainty sources, such as the uncertainty of a numerical model output, it is possible to quantify it in different forms, using different estimators:

- two bounds obtained by quantile regression
- a variance measured empirically on conditional error of empirical data.
- A distribution predicted by a model with uncertainty quantification by design.

Estimators take various forms: Cross validation scheme, monitoring probes, statistical modelling, model with uncertainty quantification by design, model dedicated to uncertainty prediction. Some model impact linked to specific uncertainty can be more difficult or even impossible to quantify in isolation in certain configurations. For example, it's impossible to distinguish between noise intrinsic to the phenomenon, and sensor measurement noise, without having control over the sensor. Epistemic uncertainty, impacting model relevance due to poorly represented data, is significantly more difficult to capture than aleatoric uncertainty, impacting uncertainty around model decisions. Some types of impact are also more extensively studied in the literature (OOD detection, Predictive uncertainty) than others, even if new fields are gradually emerging (Conformal prediction, Gap generalization). But in all case, the aim is always the same: to produce a measure that can be used for risk assessment of model error.

2) Estimation analysis & evaluation: ensure that the uncertainty measures produced by estimators are relevant. Evaluation protocols are specific to the nature of the uncertainty measure. They may use various uncertainty metrics (e.g Likelihood, Empirical coverage, Entropy, Cal-

ibration curve area ..., OOD Detection) combined with specific protocols (data generation, disturbance/alteration, noise injection, sensibility analysis) to palliate to the absence of ground truth which is often inaccessible (unknown noise) or too costly (rare event). In general, the evaluation plan is designed to ensure if the uncertainty measure behaves as expected (discriminate sample presenting a high variability or ambiguities, outliers & out of distribution).

3) Uncertainty Mitigation: integrate into the AI Component to exploit uncertainty quantity to mitigate risk according to a risk aversion specification. Uncertainty mitigation process that will act on training data, inputs, outputs, or the model itself, to calibrate the risk according to the specification. Mitigation often consists of reducing or introducing a bias that makes the errors less costly or less likely, even if it may slightly degrade performance on average on training data. Note that there is some real-time mitigation (e.g. Real time OOD detection), that rely on a human operator who must interpret model confidence alert through UQ-KPI produced at the mitigation step from uncertainty measure expressing one-off drop or continuous deterioration of model confidence.

4) Mitigation analysis & evaluation: Finally, a mitigation effectiveness analysis must show that the uncertainty management of the AI Component is fully effective, based on uncertainty evaluation protocols combined with operational metrics requirements. It to evaluate the benefits of the mitigation process on the AI Component outputs according to business KPIs. As the mitigation process can be based on estimators, trained on data that may differ slightly from real data, it need to make sure that on new data, mitigation prevent or reduce the cost of errors made by the AI Component.

D.3.2.2 Model Output Uncertainty Management

Numerous sources of uncertainty of different kinds impact on the data upstream of the modeling phase. Within the modeling framework, these uncertainties may be associated with an aleatoric nature, i.e. they are irreducible within the modeling framework. The impact of these sources of uncertainty can be quantified through the model's decision uncertainty. Listing a few arbitrary sources, we can mention:

1. Uncertainties intrinsic to the task or the environment.
2. Measurement noise related to the collection process.
3. Loss or failure to collect some useful features.

Following the 4 general steps of the general uncertainty management process, Aleatoric uncertainty management consists of:

1. **Predictive uncertainty:** Uncertainty quantification estimators can predict or quantify, in conjunction with the model, the conditional uncertainty of the model output according to the model input. Estimators can take many forms that include model with uncertainty quantification by design (e.g. probabilistic or Bayesian estimators) or third-party estimators dedicated to caught data-uncertainty (e.g. quantile regressors).
2. **Predictive uncertainty metrics:** To determine the relevance of predictive uncertainty estimations, we have to evaluate two Properties on test data not used for model training and

uncertainty estimators. The first one is UQ-validity expressing the suitability of the measurement according to empirical observation. It can be evaluated using Coverage metrics or Calibration curves. The second's one is UQ-optimality expressing the informativeness of the uncertainty measurement through Entropy or Sharpness metrics. Some metrics, such as the likelihood of the pair of model estimators, can include both properties.

3. **Confidence output intervals:** It is possible to translate a quantity of uncertainty into a confidence interval expressing according to a specified confidence level the set of choices/-values that are possible according to the model (i.e. probabilistic prediction). In addition, if we have a risk-aversion specification for certain types of error, we can mitigate the risk on model's decision by induce a bias based on the predicted uncertainty to reduce the risk of costly errors. This results in a risk-aware model decision.

4. **UQ risk analysis:** Finally, on the validation database, it is possible, with an business cost function, to analyse the benefit of model decisions with and without mitigation, to ensure the correct generalization of the model and UQ-estimators pairs.

5. **UQ Monitoring:** For the purposes of monitoring AI components, the continuous analysis of UQ metrics on the different operational contexts can make it possible to identify a data shift with an increase/decrease in the predicted uncertainty or even a degradation of the UQ faculties leading to an over /under confidence of uncertainty measurements.

D.3.2.3 Model Epistemic-Uncertainty Management

Models are train on finite dataset that often presents problems of representativeness, linked to events that are rare or difficult to collect. The impact of epistemic-uncertainty sources (link to lack of observation) on the representativeness of learning dataset induce a risk that the model may make irrelevant decision learning to critical errors. To try to handle underconfident model risk, we can set up the following epistemic-uncertainty management:

- **Epistemic / OOD Scores:** There are several paradigms for estimating whether a model is suitable for handling an input that might not belong to its learning distribution, and induce an irrelevant output. In simplified terms, the task is related to the estimation of inputs distribution density. In practice, with high-dimensional data, this is a very complex task, which can be approximated in different ways. We can split approaches that try to exploit by design the main model with that using a third-party model dedicated to out-of-distribution detection. For those based on the main model, there are also various approaches based on model latent-space distribution, metamodeling output instability (Model ensemble, Bayesian Model), model output instability by last layers disturbance, or model outputs density estimations. In all cases, these approaches produce epistemic/ood scores that express a risk linked to a lack of confidence in the model's ability to handle atypical input.

- **Real/Synthetic OOD performances:** With a database of atypical elements representing OODs that the model is likely to encounter, we can evaluate the detection performance of OOD scores using standard metrics (AUROC, F1Scores, precision-recall curves). In absence of OOD evaluation data, it is possible to design alteration (e.g. noise injection) or data generation procedures, that integrate expert knowledge about the nature and form of the OODs that the model is likely to encounter in operational use, in order to evaluate if

the Epistemic/OOD score can be used to identification of out of distribution or abnormal inputs.

- **Input/Decision Rejection:** Based on a score providing information on the presumed confidence/capacity of the model to process input data, the mitigation process consists of establishing one or more critical thresholds beyond which palliative measures must be taken. Among the palliative measures, we can mention the replacement of the model output by a default output, or even in the presence of a human operator, the alert raising, aimed at informing an operator to pay increased attention or to decide without considering the model output.

- **Model UQ Risk analysis:** The analysis of the benefits of mitigation linked to model confidence can be complex due to the lack of real OOD cases, and the difficulty of automating palliative measures on rare events not handled by the model. An analysis of the behaviour of the components (alert raised, indicator status) confronted with OOD scenario simulations based on expert knowledge is only the solution in the absence of a real OOD situation scenario.

- **UQ Monitoring:** For the purposes of monitoring AI components, continuous analysis of epistemic/OOD scores can make it possible to identify a data shift with an increase/decrease in predicted uncertainty or a deterioration in the confidence of the model in its ability to predict, suggesting a need for re-learning.

D.3.2.4 Training Dataset Uncertainty

The training set of a machine learning model plays a crucial role on the impact that the multitude of sources of uncertainty will have on the model. Qualification, cleaning, and improvement work can allow to better handle some risks associated with model uncertainty.

- **Data Uncertainty Quantification:** A number of processes can be implemented to identify traces of uncertainty and quality problems in a dataset. Some labelling errors can be detected by analysing model-based or metric-based local uncertainty measures. Unsupervised anomaly detection approaches can identify outliers that may hinder model learning. Data coverage analysis can identify representativeness problems based on metrics, or model-based approaches that exploit epistemic confidence measures, for example.

- **Performance Gain:** Data qualification can be assessed qualitatively, by noticing that the analyses have identified data-quality issues. In more quantitative ways, it is also possible to compare the quality of the original dataset, and dataset cleaned up on the basis of data qualification findings. To do that, we can use data quality metrics-based comparison or model-based characterization for example by compare performance gap between both datasets.

- **Uncertainty Enhanced Datasets:** Here the uncertainty mitigation process, will consist to perform all the process that aim to improve the dataset to form an uncertainty-enhanced dataset. It including labelling correction, noise filtering, data-augmentation, data-generation in order to increase data quality and to better cover the target data domain. The aim of all these treatments is to reduce the impact of certain sources of uncertainty, or to make it easier to capture the impact of certain other noise-related sources.

- **Data UQ Analysis:** Similar to the evaluation of data quality measures, we can judge the effectiveness of the data-set quality improvement process by comparing the performance of a model trained on the data-set without processing vs. the model trained on the Uncertainty-enhanced data.

D.3.2.5 Gap Generalization Uncertainty

Model training databases often present a gap with the real data distribution, for a various causes (slight sensor fatigue, slight operational shift, poor representativeness). We can assimilate this to uncertainty sources impacting the model through the bias of this training data. Uncertainty management can help reduce the risk of error linked to this generalization gap:

- **Residual/Non-conformity Measures:** Estimators will quantify model errors on calibration data to correct the models bias induced by the training data. They quantify bias through residuals (calibration), or non-conformity measures (conformalization) which can be more or less refined (average measurements, conditional average measurement, measurement considering uncertainty).

- **Calibration Gain:** we can analyse on test set, if a calibration/conformalization processes done using a calibration set induce benefits according to performances metrics.

- **Calibration/Conformalization**: If a calibration/conformalization process allow a reduction the generalization error, it has to be integrated in a AI components as post-processing of model outputs. This post-processing thus values the residue/non-conformity measures acquired on the calibration set. Certain advanced calibration/conformalization procedures update these measures continuously to better consider the evolution and drift of the data.

- **Distribution Shift Risk Analysis:** On the validation data, we can then analyse the benefit between the model without calibration and the model calibrated according to a standard performance metric or a business cost metric.

- **UQ-Monitoring:** For monitoring purposes, a continuous analysis of residual/Non-conformity measures on different data contexts can make it possible to anticipate a data shift which leads to a deterioration in model performance on one or all operational contexts.

D.3.3 RUM-based analytic Monitoring attributes

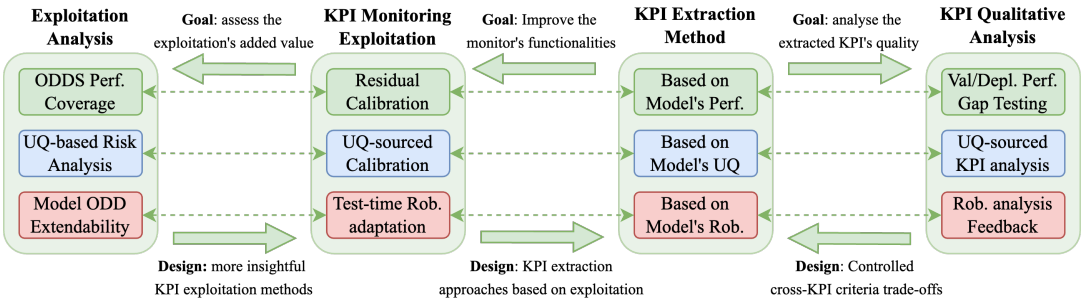


Figure D.3: RUM-based Enhanced Monitoring Analysis.

The analysis of monitoring properties, uncertainty KPIs and robustness indicators contribute to increase the confidence in the overall AI component. Figure D.3 shows a streamline illustrating protocols to conduct such analysis.

The proposed approach is based on the concept of C.3.1. The model performance can be estimated to characterize the model ODD. Comparing it with the specified ODD gives visibility on the current coverage for a given ODD parameter. The analysis of D.4 can give directions on how to improve the model performance and the ODD specified performance coverage.

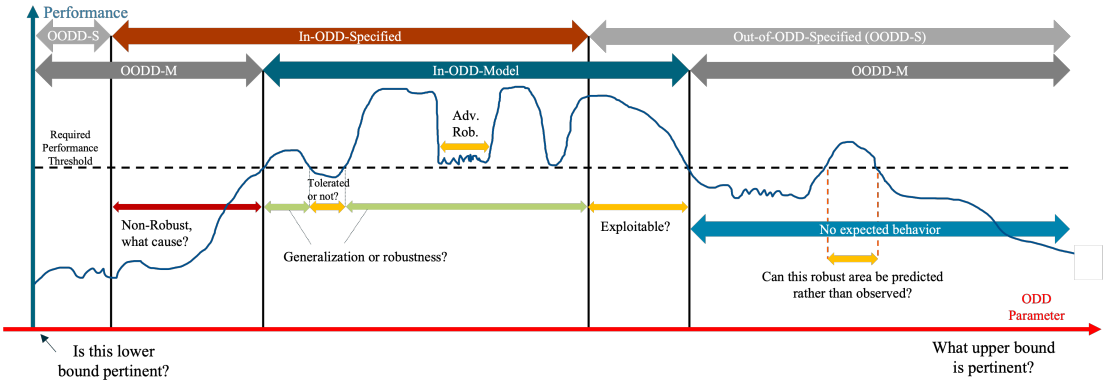


Figure D.4: Basic approach to model ODD. The blue curve represents the evolution of the model's performance along an ODD parameter variation.

The enhanced vision shown in D.5 is based on the previous approach D.4. It adds the dimension of data density in order to give indicators on the data representativity. The analysis can help improving the model performance and ODD coverage by giving hints on how to increase the diversity in the training dataset.

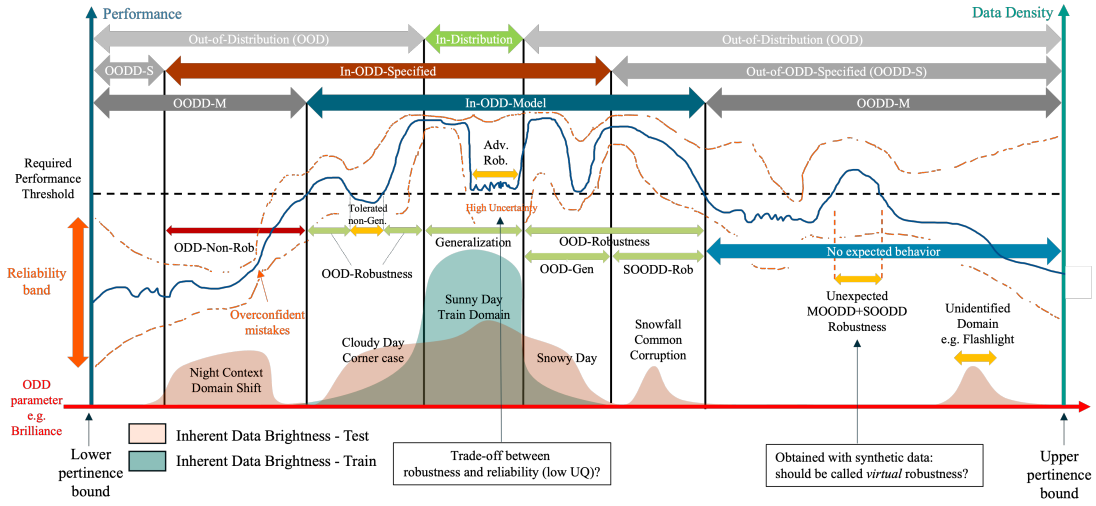


Figure D.5: RUM-Enhanced model ODD. The blue curve represents the evolution of the model's performance along an ODD parameter variation. Notice that if we took Brilliance strictly as a Level 2 ODD parameter, we would have multiple subdivisions of the above diagram corresponding to the different data domains. For sake of simplicity, we put them all together in the same figure.

D.3.3.1 Specified ODD Performance Coverability

After analysis and comparison between the model ODD and the specified ODD, this gives visibility on the specified ODD performance coverage. Model ODD Characterization also allows the automatic calibration of functions aiming at controlling the intensity of ODD parameters. Those functions can be used for OOD detection and to caliber robustness adaptation. Robustness adaptation is detailed in D.3.3.3. After checking the performance of those calibrated monitors on validation data, the functions can be used to monitor the model ODD performance in validation environment, and then in deployment.

D.3.3.2 UQ-based risk Quantifiability

An AI Component that include ML-model has to cope with hazards link to the environment, the data collection system, and others, that directly impact model conception and operation. Integrates into AI Component some uncertainty management mechanisms will confer more control and confidence about our model.

Uncertainty management mechanisms begin with estimators quantifying the impact of some uncertainties on the model. Such estimators must be validated according to uncertainty metrics to ensure their reliability. From an uncertainty quantification and according to risk aversion specifications, it is possible to set up an uncertainty mitigation process that will act on inputs, outputs, or the model itself, to calibrate the risk according to the specification. A few real-time mitigation process (e.g. Real time OOD detection), can rely on an operator who must interpret model confidence alert through UQ-KPI that express one-off drop or continuous deterioration of model confidence. In these cases, the mitigation action, can be to take a decision ignoring the output of an uncertain model, or to shut-down a model that no longer meets performance or confidence criteria until a model-update.

As already mentioned, for the purposes of monitoring AI components, a continuous analysis of:

- Residual/Non-conformity measures on different data contexts can allow to anticipate data distribution shift which leads to a deterioration in model performance on one or all operational contexts.
- epistemic/OOD scores can allow to identify a data shift with an increase/decrease in predicted uncertainty or a deterioration in the confidence of the model in its ability to predict, suggesting a need for re-learning.
- UQ metrics (Coverage,Entropy,Likelihood) on the different operational contexts can make it possible to identify a data shift with an increase/decrease in the predicted uncertainty or even a degradation of the UQ properties leading to an over /under confidence of uncertainty measurements.

D.3.3.3 Model ODD Extendability

Model ODD Extendability aims at increasing the ratio between the ODD of the resulting AI component and of the central Model alone. It can be achieved by detecting extendable zones of ODD parameters where the model performs poorly inside the ODD specified. When the monitor detects an ODD parameter intensity inside this extendable zone, it communicates to the robustness adaptation module information about this ODD parameter intensity. If the ODD parameter intensity can be recovered to an intensity where the model is known to perform well, and if the robustness module adaptation has been correctly calibrated, then the OOD input will be transformed into an ODD input and reinjected into the model inference module.

D.4. The RUM aggregate attributes

A holistic approach that integrates robustness, uncertainty quantification, and monitoring is essential for building resilient and trustworthy machine learning systems, particularly in applications where accuracy, reliability, and interpretability are critical.

D.4.1 Robustness analysis intrinsic to UQ estimations and UQ-based Monitoring

Robustness evaluation and analysis - in the worst-case ID, the average case OOD, and the probabilistic certification scenarios - can and has also be studied specifically concerning on uncertainty quantification modules.

As we will see below, the objective of presenting and re-framing *ongoing* research on this field as part of the RUM methodology will further consolidate the idea that the latter is versatile yet expressive enough to model chains of correspondences regarding R, U and M that are still in the making.

As an introductory example let's see how this works when trying to address worst-case ID robustness for Bayesian/Ensemble based UQ methods, keeping in mind that analog works have shed evidence on this for Conformal Prediction. Recent works Ledda et al. (2023) have focused on attacking the AI-component through attacking its underlying UQ module. Here, the focus is on a specific adversarial scenario in which the attacker is interested in manipulating the uncertainty estimate, regardless of the correctness of the central model's prediction. Its aim

is to undercut the use of UQ techniques for ML models when their results are consumed by a downstream module or by a human.

Let us survey the formalization of such attacks as well as analyzing its evaluation protocol. Denote $\mathbf{x}_{\text{adv}}$ for an adversarial example for the central model f. The objective of the attacker crafting $\mathbf{x}_{\text{adv}}$ is to lower the model's accuracy $\mathscr{A}_f$, and denote $\mathscr{U}(x)$ for the uncertainty estimate of f at an input x. The objective of a UQ attacker is formalized as follows:

$$\underset{\delta}{\text{argmin}} \quad \gamma \cdot \mathscr{U}(x+\delta) \qquad \text{such that } \|\delta\| < \varepsilon \tag{D.1}$$

where $\gamma \in \{-1,+1\}$ controls the attack objective. Concretely, taking $\gamma = -1$ leads to abnormally increasing the uncertainty measure, making the model underconfident, and taking $\gamma = 1$ leads to abnormally decreasing the uncertainty measure, making the model overconfident. Robustness analysis over such protocol leads to induce that crafting such adversarial attacks address a **worst-case analysis of the UQ module's *integrity*** (in the case $\gamma = 1$) **and *availability*** (in the case $\gamma = -1$). Concretely, the model will not say "I don't know" when it should in the first case, and will say "I don't know" when it shouldn't in the second case

Implementing the attacks in equation (D.1) will be different, depending on if the underlying UQ models are of probabilistic or deterministic nature. The work Ledda et al. (2023) develops both cases although we will here only concentrate on the probabilistic case. On the one hand, they obtain *minimum variance attacks* for probabilistic models, that will mainly leave the central model's prediction unharmed while maximally decalibrating the UQ modules associated to them. On the other hand, they obtain *stabilizing attacks*, which leverage the stabilization of predictions, usually used as a robustness method for the central model, as direct attack against the UQ module. The idea is simple: stability in predictions result in lower variance and average prediction's entropy. We reproduce an illustration of their obtained results for stabilization overconfidence attacks in Figure D.6.

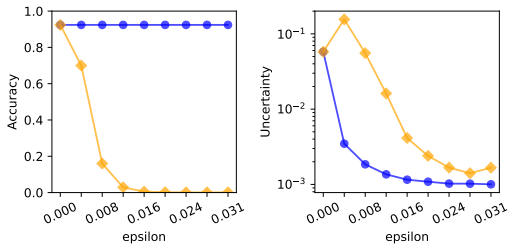


Figure D.6: (From Ledda et al. (2023)) Classification accuracy and uncertainty of a ResNet18 (endowed with a UQ-module) on CIFAR10, under two different UQ-based attacks, as a function of ε. The blue line corresponds to the stabilization overconfidence attack.

The first striking fact is that, while the "robust accuracy" of the model remains unchanged at increasing noise budget ε, it sharply becomes overconfident on the attacked dataset.

The second striking fact is that this attack has a direct incidence on any monitoring system extracting and leveraging uncertainty quantities from such UQ module. For instance, such uncertainties are usually used for OOD detection at the level of the monitoring system. The following show that *the same* stabilization overconfidence attack acts as a *direct attack* on such OOD detection module, greatly lowering its detection accuracy:

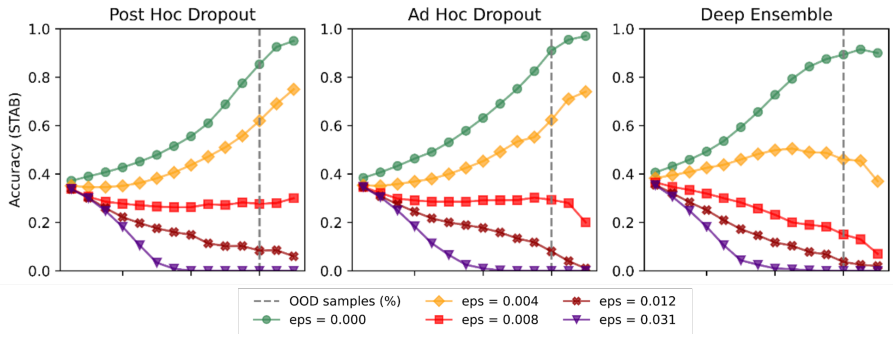


Figure D.7: (From Ledda et al. (2023)) Detection accuracy of UQ-based OOD detection modules in the monitoring system underlying the AI component. The green curve consistently shows how 3 different UQ modules exploited as OOD detectors provide good OOD detection accuracy, and how such accuracy sharply drops while the attacked UQ module becomes overconfident.

The conclusion here provides evidence of our first genuine and nominal RUM aggregate attribute: UQ-based adversarial robustness, provides also a concrete evaluation protocol on how to measure it at the 3 different RUM levels: a measure on the "robust accuracy" (left almost untouched by the stability attack), a measure on the UQ uncertainty and on the resulting UQ-based OOD detection accuracy. Recently, these facts have been generalized in Lorenz et al. (2024) for analysis the robustness of wide families of OOD detectors.

Finally, it is worthwhile to mention that, although the objective in equation (D.1) results in a variety of different and tailored attacks, we cherry-picked the stability attack as it sheds light on the evidence that

1. stabilization, which is usually used as a robustification measure, is*de-facto excluded* as a RUM improvement method

2. No protocol purely based on robustness can evaluate UQ-based adversarial robustness, as there are concrete examples of attacks which do not change the central model's accuracy.

As such, re-framing the presented RUM aggregate attribute in the context of our length 4 chain of correspondences in the RUM analysis, this will add up a further chain in which each of the 4 correspondence boxes will leverage R+U+M information, whether we are at the level of robustness evaluation, its analysis, or at the level of defense methods and their analysis. As a side note, notice that the above considerations are not to be confused with the subject of adversarial robustness and/or formal verification of BNNs such as Wicker et al. (2021) and Batten et al. (2024) or uncertainty reduction on BNNs by exploiting domain knowledge priors such as Sam et al. (2024).

D.4.1.1 The case of Conformal Prediction Robustness

In this section, we will present how the Conformal Prediction literature has proposed different kinds of robustification methods addressing two robustness evaluation protocols: that of worst-case InD analysis in the form of *uncertainty-aware* Adversarial Training, and of probabilistically certified robustness in the form of *uncertainty-aware* Randomized-Smoothing.

First of all, the idea here is that attacks on CP methods make the latter to not be able to produce valid prediction sets under the commonly used ℓ_p-norm bounded attack, meaning that, under attack, the guarantee in equation (C.11) is *not satisfied*. As such, Gendler et al. (2022) generalized this to an "adversarially robust" requirement in the form of

$$\mathbb{P}\{Y \in C_\alpha(X_{\text{adv}})\} \geq 1 - \alpha, \tag{D.2}$$

where X_{adv} is an adversarial attack and proposed a first version of "Randomly smoothed Conformal Prediction" to address *uncertainty-aware* probabilistically certified robustness. Such version presented a flawed robustness guarantee in practice and tended to produce overly large prediction sets. These issues were only recently resolved in Yan et al. (2024).

Yet, these methods were not meant to address a proper *worst-case analysis* on the basis of the strongest available adversarial attacks against CP, and different lines of work have recently tackled the question CP-aware robustness certificates Zargarbashi et al. (2024) by proposing a difference between "average robustness" and worst-case robustness in CP.

Concerning worst-case InD evaluation, evidence was showing that current defense methods, although presenting a good average coverage, where presenting abnormally large prediction sets. As such, only even more recently in Liu et al. (2024b), a proper UQ-aware adversarial training method for CP was proposed and whose analysis show that the defense has as added value the fact of properly reducing the size of the prediction set while maintaining some level of robustness to UQ-aware adversarial attacks. In particular, the authors show that increasingly strong adversarial training (i.e AT with respect to a very strong set of threats of different kind) induces increasingly large prediction sets. Their adversarial objective shows that a trade-off between AT training strength and size of prediction set might be controllable through hyperparameters. The authors also highlight that their work was made under the assumption that all adversarial attacks are known. This means that a proper analysis on the defense generalizability to *unseen and UQ-aware* adversarial attacks is yet to be done.

Research on Conformal Prediction beyond the exchangeability condition started already in 2019 when Tibshirani et al proposed a version of CP under the easiest non-trivial kind of distribution shifts, the covariate type, which shifts the marginal data distribution without changing the conditional distributions of labels given data. They already produced marginal coverage guarantees for this case, and later Park et al in 2021 proposed a training-conditional improved guarantee. Since the, many works have been conducted to handle CP beyond exchangeability gathering results of industrial significance. As such, we highly recommend that the PUNCC component leverages the already existing literature on this subject in order to address more realistic industrial settings in which distribution shifts are ubiquitous. We further refer the reader to Ai & Ren for a recent account on CP under distribution shifts and we recommend that CP-based components such as the PUNCC component to further pursue this avenue.

Although we will not develop it here, other robustness *chains* such as formal verification for CP Jeary et al. (2024) have recently been proposed.

D.4.2 RUM analysis, from model to AI Component level

The idea of this section is to present emerging attributes related to the RUM analysis from the level of the ML model to that of the AI component. To illustrate this, let us consider first the

scenario of worst-case InD analysis in the form of adversarial attacks. We will latter show how recent work quantify uncertainty associated to such new attacks and how the latter happen to be "robust to uncertainty".

Experimentally, the stronger the adversarial attacks, the most accurate the worst-case analysis is. As such, one can measure the comprehensiveness of worst-case analysis by measuring the attack success rate of the combination of threat models lowering the robust accuracy the most.

Robustification techniques might not be intrinsic to the evaluated model f: adversarial purification introduces an input preprocessing module that defends f against adversarial attacks without modifying it. If we denote such system as $S = f \circ P$, where P is the purifier, we call

- model-wise intrinsic adversarial robustness evaluation, the worst-case analysis conducted on f without the preprocessing tool i.e. adversarial attacks on f
- component-wise extrinsic adversarial robustness evaluation, the worst-case analysis conducted on f *with* the preprocessing tool i.e. non-adaptive adversarial attacks on S. Such attacks only leverage the weights of f
- component-wise intrinsic adversarial robustness evaluation, the worst-case analysis conducted on S i.e. adaptive adversarial attacks that leverage both the weights of f and of P

It is worth mentioning that component-wise intrinsic adversarial robustness evaluation answers to an analysis in the context of an attacker that has gained full control of the whole AI component, breaching the highest possible cybersecurity defense implemented for such AI component. Conversely, adaptive attacks will need increasing computational resources to be successfully conducted, dividing adversarial robustness into two different orientations:

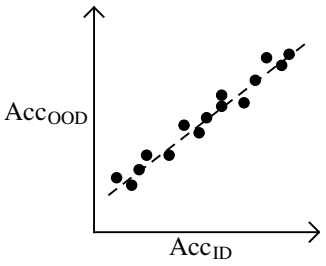
- Addressing the *scientific-oriented challenge* of the current most effective adversarial defense needs to be done by conducting worst-case analysis on S
- Component-wise extrinsic adversarial robustness evaluation may suffice for addressing the *business-oriented challenge* of the best cost-effective adversarial defense under the assumption that the attacker might have at best access to the model f's weights.

In the context of robustness to breaches related to privacy, the work Carlini et al. (2024) has shown also how leakage from different sources on a AI component add to each other to render the AI component much more brittle than the ML model alone. In this same work, the uncertainty associated to such attacks is quantified, which leads the authors to claim that their attack method is robust to uncertainties of a specific nature. The important question asked in the paper directly tying it to the RUM methodology is: "How much the attacker knows about what it doesn't know?".While we will not enter into the details of their construction, we argue here that this constitutes another example of a RUM aggregate and nominal attribute.

D.4.3 OOD Robustness through *On-the-line* phenomena

D.4.3.1 Shifts possessing the "Accuracy-on-the-line" phenomenon

Some shifts possess this remarkable property that their multi-model ID performance is in a linear correlation with their OOD performance:



Recall that for s and t two time stamps, distribution shifts can be mathematically formalized into

- **Covariate shifts**: $P_s(Y|X) = P_t(Y|X)$ and $P_s(X) \neq P_t(X)$, indicating that the feature distribution differs between the source and the target
- **Concept shifts**: $P_s(Y|X) \neq P_t(Y|X)$ and $P_s(X) = P_t(X)$ indicating that there are spurious statistical correlations in the source data that may not hold in the target data

It is worth noticing that for visual and text data, covariate shifts on $P(X)$ are dominant while for tabular data it is concept shifts $P(Y|X)$ that are dominant. This leads to varying relationships between in-distribution and out-of-distribution performances across different settings and datasets. This contrasts sharply with the recently observed accuracy-on-the-line phenomenon, where in-distribution and out-of-distribution performances are suggested to have a strong linear relationship. For concept shifts, the work Liu et al. (2024a) shows how the ACL phenomenon progressively starts to fail as the shifts become more pronounced.

D.4.3.2 The AGL method

In the context of covariate shifts, the Accuracy-on-the-line (ACL) phenomenon consists on a strong correlation between Acc_{ID} and Acc_{OOD} for various models independently trained on the same data set and evaluated on the same (labeled) OOD test-set. Let h, h' be two models independently trained on a dataset following a distribution $\mathscr{D}_{\text{ID}}$. For a distribution $\mathscr{D}$, the expected accuracy and agreement are defined as:

$$\text{Acc}_{\mathscr{D}}(h) = \mathbb{E}_{x,y\sim\mathscr{D}}[\mathbb{I}\{h(x) = y\}], \quad \text{Agr}_{\mathscr{D}}(h,h') = E_{x\sim\mathscr{D}}[\mathbb{I}\{h(x) = h'(x)\}].$$

Notice that $\mathscr{D}$ need not to be labelled for agreement to be well-defined. We will simply denote Acc_{ID} and Acc_{OOD} for the resulting metrics in case $\mathscr{D} = \mathscr{D}_{\text{ID}}$ (labelled) and $\mathscr{D} = \mathscr{D}_{\text{OOD}}$ (unlabelled) respectively. The Agreement-on-the-line (AGL) phenomenon consists on the following

1. When Acc_{ID} and Acc_{OOD} are strongly linearly correlated, so are Agr_{ID} and Agr_{OOD}
2. When both strong correlations hold, they have almost the same slope and bias
3. When Acc_{ID} and Acc_{OOD} are weakly linearly correlated, so are Agr_{ID} and Agr_{OOD}

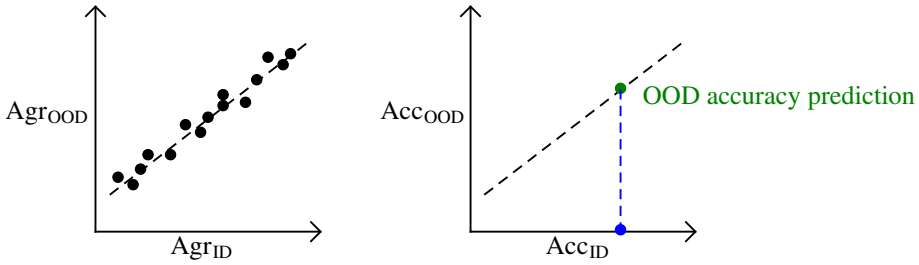
4. When both weak correlations hold, they have different slope and bias in general

Putting 1. and 3. together, we obtain that AGL implies ACL. As such, one immediately deduce

- **Model selection**: check that AGL holds, then since ACL holds, choose the model with highest Acc_{ID}
- **ODD Performance Prediction**: check that AGL holds, then since ACL holds *with same slope and bias*, one can estimate Acc_{OOD} directly from Acc_{ID}. If we have at least 3 trained models, we can get

  $$\text{Acc}_{\text{OOD}} \approx a \cdot \text{Acc}_{\text{ID}} + b$$

  where a, b are obtained by a simple linear regression starting from Agr_{ID} and Agr_{OOD} only.



Some observations:

- There is no general rule as to how many models are needed for the best prediction of Acc_{OOD}.
- While the ACL phenomenon is common to large classes of models, the AGL phenomenon is *exclusive* to neural networks in the classic, non-foundational, setting. On the other hand, recently, linear models on top of CLIP features for foundation models have shown to also satisfy AGL.
- No assumptions on the distribution shift, networks architecture or training algorithms were made and no empirical measure on the distribution shift was needed
- AGL is a property which is not universally valid: it does not intrinsically characterize OOD generalization properties.
- AGL is sufficiently general as to encompass several Robustness, Uncertainty and Monitoring aspects which will be discussed in the corresponding sections below.

D.4.3.3 Validation OOD dataset quality metrics (Model-centric)

Constructing validation OOD datasets from the data-centric point of view may be done analogously along the lines in [Adjed et al, 2024]. We will give a short model-centric account on attributes and metrics for the behavior of models on such datasets.

- ODD parametric simulated robustness: the ability of a robustification mechanism to match

the required performance along an ODD parametric synthetic variation across the Specified ODD zone.

- Syn-to-Real generalizability: the ability of an OOD robustification mechanism designed by means of synthetic data to generalize to real ODD data [Hendrycks, The many faces of Robustness]

- Effective OOD robustness: ability for a robustification mechanism to outperform on OOD robustness with respect to its predicted OOD robustness [Effective robustness paper+paper 2020]: this is computed as the distance of the point in the (IID, OOD) accuracy plane with the regressed OOD vs IID prediction line.

- Generic OOD robustness: (in the context of foundation models) ability for an OOD robustification mechanism on a foundation model to exhibit OOD generalization capabilities on down-stream tasks. See Kim et al. (2023).

- OOD Detectability: ability for a robustification mechanism to detect OOD inputs and process them according to the system specifications. This is dependent on robustness, reliability of the auxiliary detection module

- Corruption resilience: the ability of the robustification mechanism to slowly decrease the system's performance under an intensity increasing corruption-type OOD model (this might not exactly correspond to variations in the values of ODD parameters).

- Adaptability: the ability of a model to adapt in test-time to OOD inputs. See Kim et al. (2023).

D.4.4 Formal certifiability as monitoring of formal robustness verification tools

The reports from the VNN-COMP competition of neural network verifiers have shown that available formal verification tools *are not sound*. On the one hand, tools have shown to determine incorrectly that, for a specified model and robustness property to be verified, an input would be robust when in fact it is not. Here, providing a counter-example is enough to formally show that the tool made such a mistake. On the other hand, methods have falsified such property without providing a counter-example. Here, things become more subtle because, in the absence of reference ground truth, a majority vote policy is applied to decide whether the method is deemed to be incorrect.

Overall, around 40% of the methods have made mistakes on each year's competition. Up to 2023, the verification tools that made mistakes are: Marabou, Debona, NNV, NV.jl, Venus2, CGDtest, Verapak, α, β-CROWN, NeuralSAT. Most importantly, on the 2023 competition, several benchmarks from 2022 were also reported in parallel from the 2023 scored benchmark for the competition. In this context, methods such as α, β-CROWN, which made no mistakes in the year 2022, started making mistakes in the 2023 venue for the same benchmark : Acas-Xu, vgg-net16 and MNIST-FC (Tab. 13, 27 and 35 in 2023). The fact that the same tool can make no mistakes one year and starts making mistakes on subsequent years raises two questions:

- How to ensure that formal robustness verifiers will not make mistakes in the future, especially if used in critical systems?

Tool (2021)	Nb. of verified instances (tot. 868)	Nb. of incorrect predictions	Accuracy
α,β-CROWN	766	0	100,00
VeriNet	717	0	100,00
ERAN	656	0	100,00
OVAL	636	0	100,00
Marabou	364	26	93,33
nnenum	310	0	100,00
Debona	280	14	95,24
Venus	266	1	99,63
NNV	141	12	92,16
NV.jl	97	5	95,10
RPM	72	0	100,00
DNNF	55	0	100,00
randgen	33	0	100,00

(a) Summary of the VNN Competition results of 2021

Tool (2022)	Nb. of verified instances (tot. 1102)	Nb. of incorrect predictions	Accuracy
α,β-CROWN	950	1	99,89
MN-BaB	812	0	100,00
VeriNet	754	0	100,00
nnenum	515	0	100,00
PeregriNN	478	0	100,00
Marabou	450	1	99,78
CGDTest	405	41	90,81
Debona	341	0	100,00
VeraPak	117	5	95,90
AveriNN	100	0	100,00
FastBATLLNN	32	0	100,00

(b) Summary of the VNN Competition results of 2022

Figure D.8: Summary of the VNN Competition results of 2021 and 2022

Tool (2023)	Nb. of verified instances (tot. 853)	Nb. of incorrect predictions	Accuracy	Worst performance	Corresponding dataset	Existence of errors in 2022	Existence of errors in 2021
α,β-CROWN	721	1	99,86	99,46	ACAS-XU	YES	NO
Marabou	594	0	100,00				YES
NeuralSAT	452	35	92,81	0	traffic-signs-recognition		
PyRAT	416	0	100,00				
nnenum	319	0	100,00			NO	NO
NNV	155	35	81,58	46	collins-rul-cnn		YES
FastBATLLN	32	0	100,00			NO	

Figure D.9: Summary of the VNN Competition results of 2023

Tool (2024)	Nb. of verified instances (tot. 2387)	Nb. of incorrect predictions	Accuracy	Worst performance	Corresponding dataset	Existence of errors in 2023	Existence of errors in 2022	Existence of errors in 2021
α,β-CROWN	2285	0	100			YES	YES	NO
PyRAT	1760	0	100,00			NO		
nnenum	1398	2	99,857143	98,33	Linearizen	NO	NO	NO
Marabou	1394	403	77,573734	62,74	Safenlp	NO		YES
NeVer2	848	0	100,00					
CORA	828	101	89,128095	68,16	Cora			
NNV	736	0	100			YES		YES
NeuralSAT	489	281	63,51	26,11	Cora	YES		

Figure D.10: Summary of the VNN Competition results of 2024. Notice that for this venue **50%** of the competing tools made mistakes in their guarantees, an increase in comparison to previous years.

- If the ground truth is determined by "majority vote" between different checkers - and therefore the determination that a checker has made a mistake - what formal proof status should be associated with this?

Despite encouraging efforts and scientific advances in this field, current verification methods and current evaluation protocols for such methods had yet to address these issues.

Finally, some methods provide counter-examples with incorrect outputs values undermining the formal character of the approach. This can be the case when such counter-examples are obtained via adversarial attack procedures. In the absence of consistent soundness for formal verifiers, the possibility that strong adversarial attacks show a-posteriori that an input might be not robust after-all.

Direct verification of mature verifiers is close to unfeasible since they consist on multi-platform complex libraries. A possible alternative guarantee is to make verifiers to produce human-verifiable proofs, called *proof checkers*, as was first done in Isac et al. (2022) for the Marabou verifier Wu et al. (2024). Yet, Marabou kept making mistakes and the root problem was found to be that the proof checker methodology still didn't take into account floating point errors. As such, the certification of DNN verifiers themselves became a challenge on its own. A solution to this problem was recently proposed in Desmartin et al. (2024), although they emphasized that there exists a trade-off between the reliability and scalability of the proof checking process which itself has the task to formally verify that the formal verifier will not make mistakes. In total, the verifier and the existing tools to certify such verifiers both see trade-offs between reliability and scalability, and the authors in Desmartin et al. (2024) report that proof-checking Marabou's outputs took 106 seconds against the 0.05 seconds that were necessary for the unreliable proof checker constructed in Isac et al. (2022) to complete it's task of certifying the DNN certifier.

Finally, it is worth noting that Jia and Rinard (2021) develop both: a method to produce inputs showing the incorrectness of robustness claims made by a complete verifier (such as MIPVerify); and a method to design neural network architectures and weights inducing wrong results of an incomplete verifier (such as CROWN). One can consider both these methods as adversarial attacks against formal verifiers, as the first one maximally exhibits existing unsoundness and the second increases unsoundness. Notice that the second case is realistic since an attacker with access to the network can modify it according to this method so that its non-robust behavior does not get noticed by a verifier.

D.4.4.0.1 Recommendations for industry-oriented usage. The present evidence have an impact in the following Confiance.ai documents: 19A, 314A, 317A, 32A, 41A, 434A, 435A, 721B, we propose novel usage recommendations for formal robustness verification:

- Formal verifiers are prone to failures which need to be taken in consideration in the perspective of deployment for critical systems: their outputs are not to be considered as always equivalent to the ground truth.
- As a result, confiance.ai formal verification tools are not, in their present form, compliant with the necessary validation & certification requirements of robustness (absence of soundness in the verification tools and absence of certification of the verification tools) for safety-critical systems.
- Empirical risk mitigation might be obtained to some extent by always running multiple formal verifiers and monitoring their disagreement. As such, the CAISAR tool consists on a useful although empirical evaluation platform based on agreements and disagreements of multiple unsound verification tools. Care should be taken so that errors made by verifiers are not propagated to different ones.
- Software validation process at each iteration of each of these verifiers is strongly encouraged
- Tools that made mistakes should propose formal explanations of why their tool reached such mistakes in order to better understand the uncertainty underlying these methods

- If mistakes are not the result of bugs but of *assumed choices* in floating-point reduction strategies to improve the tools efficiency, the theoretical validity of the formal approach including such reductions should be further pursued and an error risk analysis should be conducted to associate quantitative risks to such tools before deployment.

Notice that formal robustness verification addresses a different kind of robustness evaluation than adversarial evaluation. Given a test dataset D, the latter addresses a worst-case analysis of it while the former aims to verify formally, among correctly classified inputs, those satisfying a specific mathematical formalization of the robustness property. On the one hand, worst-case analysis cannot fully address the verification analysis since it can only logically falsify the formal property. On the other hand, verification analysis does not address worst-case analysis, since it usually relies on adversarial attack protocols to exhibit counter-examples and failing to find a counter-example through adversarial attacks *does not imply* that the robustness property holds.

D.4.5 MCDA for non-nominal RUM attributes

In the last sections, we addressed a detailed description and analysis of nominal RUM attributes, which were directly obtained by analysis' specific trade-offs that appear in the present literature. Nonetheless, as such crossed-studies are an emergent field at the time of writing, one can expect many new nominal RUM attributes to appear in the near future but which are not available at the moment.

This raises the question of crafting a formal framework in which the already available R/U/M attributes, under the condition of them being commensurable metrics, indicators can be aggregated without specifying nominally what they address, and then use this formalism to produce trustworthiness non-nominal scores which may present an interest on their own.

We will now present Tropical Algebra as a candidate for such formal framework.

Which aims at creating a methodological framework for reliability assessment that will take advantage of expert knowledge (e.g. in defining thresholds), modelling the application environment (e.g. the influence of the operational design domain on attribute selection), and ease of use in an engineering process (each atomic attribute is associated with a method or metric), covering other AI paradigms to go beyond the current state of the art of nominal trust aggregate attributes such as those presented in the earlier sections of this chapter.

D.4.5.1 Tropical Algebra brief introduction

"Tropical Algebra" is a relatively new mathematics field Gaubert and Max-Plus (1997). Tropical Algebra is based on $(max,+)$-algebra and $(min,+)$-algebra.

In $(max,+)$-algebra, the addition is replaced by maximum: $a \oplus b = \max(a,b)$, and the multiplication is replaced by addition: $a \otimes b = a+b$. Two null elements are defined: $0_{\oplus} = -\infty$ for $\oplus$, while $0_{\otimes} = 0$ for $\otimes$. Similarly, in $(min,+)$-algebra the tropical sum of two numbers returns the minimum among the two numbers while the tropical product of two numbers returns the sum of those two numbers: $a \oplus b = \min(a,b)$, and $a \otimes b = a+b$.

The tropical operation $\oplus$ and $\otimes$ are associative and commutative, and multiplication is left and right distributive over addition. This algebraic structure differs from classical structures, like

fields, in that addition is idempotent: $a \oplus a = a, \forall a$. As with conventional algebra, it is possible to extend the $(max, +)$ (resp. $(min, +)$) algebra to analyze matrices. Indeed, given two matrices A and B with the same dimensions, then: $A \oplus B = C$, where $C_{ij} = A_{ij} \oplus B_{ij}$. If A and B are conformable for multiplication, then $A \otimes B = C$, where $(A \otimes B)_{ij} = \max_k(a_{ik} + b_{kj})$, (resp. $(A \otimes B)_{ij} = \min_k(a_{ik} + b_{kj})$). A complete detailed description and analysis of the mathematics behind such algebra also call "Tropical Algebra" can be found in Cuninghame-Green (2012); Olsder et al. (1992), a review of the basic concepts in Gaubert and Max-Plus (1997).

The Tropical Algebra formalism is particularly well suited to multi-attribute aggregation. Concretely, Tropical Algebra:

1. Is adept at handling non-linear aggregation, which is often required in trustworthiness assessment where attributes may not combine linearly.

2. Provides flexibility in modeling complex relationships between trustworthiness attributes, allowing for more accurate modeling of their respective assessment.

3. The operations are computationally efficient, making it suitable for large-scale problems.

4. Can handle a large number of attributes without significant loss of performance.

5. Is robust against variations in KPIs, ensuring stable assessment even under uncertainty.

6. Maintains consistency in aggregation, which is crucial for reliable assessment.

Accordingly, the Tropical Algebra structure $(S, \oplus, \otimes)$ is an idempotent semi-ring (a.k.a dioid Gondran and Minoux (2008)) with S denoting a set of elements of the concerned trustworthiness dimension. Such aggregation operators satisfy the following axioms: identity when unary, boundary conditions, non decreasing. Besides these basic properties such operators are interesting because they are monotone, symmetric, associative, idempotent Labreuche (2016). Then aggregate trustworthiness KPIs using Tropical Algebra specifies a global robustness score $\mu_{\mathscr{R}}$ to combine the normalized metrics.

For example, if we have three normalized robustness KPIs μ_{R_i} with $i = 1, 2, 3$, the global robustness score can be computed as: $\mu_{\mathscr{R}} = \mu_{R_1} \oplus \mu_{R_2} \oplus \mu_{R_3}$. Alternatively, we can use a weighted sum approach: $\mu_{\mathscr{R}} = \omega_1 \otimes \mu_{R_1} \oplus \omega_2 \otimes \mu_{R_2} \oplus \omega_3 \otimes \mu_{R_3}$ where $\omega_1, \omega_2, \omega_3$ are weights assigned to each metric based on their importance. The resulting $\mu_{\mathscr{R}}$ will give a single value that represents the overall robustness of the model. A higher $\mu_{\mathscr{R}}$ indicates better robustness.

Concretely, the $(max, +)$ algebra is often used for systems where synchronization and timing are crucial, such as train scheduling. Consider a simple example with 3 trains on a single track, where each train needs to wait for the track to be free before proceeding. We denote the times at which each train departs as T_1, T_2 and T_3. The initial departure times are t_1,t_2,and t_3 respectively, and each train might experience a certain delay. Let's assume that the train 1 departs at t_1 with no delay. The train 2 departs at t_2 but can only leave after the train 1 plus a delay d_{12}. Finally, the train 3 departs at t_3 but can only leave after the train 2 plus a delay d_{23}. We can then express the dependencies using $(max, +)$ algebra as follows: $T_1 = t_1$; $T_2 = max(t_2, T_1 + d_{12})$; $T_3 = max(t_3, T_2 + d_{23})$.

By defining a hierarchical aggregation operator μ, this approach can then be generalized to all

the trust attributes Mattioli et al. (2024):

$$\mu = \bigoplus_j \omega_j \otimes \left(\bigoplus_i \omega_{i,j} \otimes \mu_{i,j} \right)$$

where j represents the 6 macro-attributes: robustness, effectiveness, dependability, usability, human agency and human oversight; and i their respective atomic attribute KPIs. Specification of the weights $\omega_{i,j}$ depends on the ODD which captures the applicative importance and dependency between each attributes.

Moreover, in the context of safety-critical systems, it is important to evaluate robustness with respect to a range of attacks rather than just against one. This is why we propose to illustrate using a toy case that employs the $(min, +)$ algebra, corresponding to a worst-case aggregation.

D.4.5.2 Toy use-case: Non-nominal robustness aggregates on MNIST classification

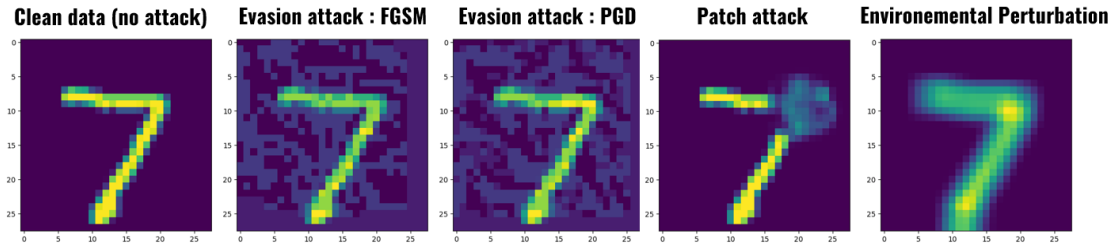


Figure D.11: Various perturbations on a single image of the MNIST dataset

To illustrate the use of Tropical Algebra in the context of aggregation of metrics, we trained a toy-model for image classification on the MNIST dataset Deng (2012). Our aim was to answer the following question: given a list of various robustness attributes and prescribed multiple test scenarios, which attribute should be improved in priority for the model? To answer it, we first gathered all the existing robustness metrics of our use-case, then associated to them a number of scenarios prescribed by a choice of weights and make a majority vote to determine which is the leading metric determined by the $(min, +)$ calculus for each scenario.

Firstly, we evaluated it's clean accuracy, then we evaluated the model's accuracy when presented with various attacks and perturbations. Namely, we used a FGSM attack Goodfellow et al. (2014) (with $\varepsilon = 0.1$) to generate the first attack. For the second attack, we used PGD (with $\varepsilon = 0.1$). For the third perturbation, we used a Gaussian blur (with $\sigma = 3$) and the last one is based on a patch attack (see Figure D.11). We obtained the following results reported in Table D.1.

No attack	FGSM attack	PGD attack	Patch attack	Environmental Perturbation
0.99	0.77	0.44	0.94	0.96

Table D.1: Model accuracy (as a probability between 0 and 1) w.r.t. various attacks on MNIST.

The process of selecting weights and thus indicating preferences is a complex one. Therefore, when utilizing weights to represent the relative importance of the objectives, transforming the functions so that they all have similar magnitudes and do not naturally dominate the aggregate

objective function can assist in accurately setting the weights to reflect preferences. A sensitivity analysis can be employed to compute such weighting parameters based on the ODD, where the relative importance of RUM criteria in the $(min, +)$ aggregator is represented by the vector $(\omega_1, \omega_2, \omega_3, \omega_4, \omega_5)$. The coefficient ω_i is a measure of the importance of criterion i. In order to construct such an operator, it is possible to utilize the two standard levels of 0 and 1. In this context, a value of 0 (respectively, 1) indicates that criterion i has not been met (respectively, has been met in a completely satisfactory manner). The importance of criterion i, denoted by ω_i, is then defined as the added value on the overall score going from the lower level 0 to the upper level 1 on criterion i, with the value on the other criteria being fixed. In order to illustrate the concept, three distinct operational scenarios can be conceived, each emphasizing a particular robustness attribute:

1. Scenario 1: Worst-Case In-Distribution Robustness. In this scenario, we would focus on the model's robustness against the strongest possible adversarial attacks within the distribution (here PGD). Therefore, we could assign the following weights to our metrics: $(\omega_1, \omega_2, \omega_3, \omega_4, \omega_5) = (0.0, 0.0, -0.1, 0.0, 0.0)$;

2. Scenario 2: Average-Case Out-of-Distribution Robustness. This scenario targets the model's robustness to average-case out-of-distribution (OOD) perturbations. The assigned weights for this scenario could be: $(\omega_1, \omega_2, \omega_3, \omega_4, \omega_5) = (0.0, 0.0, 0.0, 0.0, -0.25)$.

3. Scenario 3: Worst-Case in-ODD Robustness for deployment. In this final scenario, we concentrate on the model's robustness in the face of the worst-case in-ODD conditions in deployment. In this particular scenario, we could choose the following weights: $(\omega_1, \omega_2, \omega_3, \omega_4, \omega_5) = (0.0, 0.0, 0.0, -0.2, 0.0)$;

We would thus obtain three different aggregators of the robustness, one for each scenarios:

1. Scenario 1: $\mu_{\mathscr{R}} = \min(0.99 - 0.0, 0.77 - 0.0, 0.44 - 0.1, 0.94 - 0.0, 0.96 - 0.0) = 0.34$;

2. Scenario 2: $\mu_{\mathscr{R}} = \min(0.99 - 0.0, 0.77 - 0.0, 0.44 - 0.0, 0.94 - 0.0, 0.96 - 0.25) = 0.44$.

3. Scenario 3: $\mu_{\mathscr{R}} = \min(0.99 - 0.0, 0.77 - 0.0, 0.44 - 0.0, 0.94 - 0.2, 0.96 - 0.0) = 0.44$.

Here, the majority vote on the aggregation's operator is obvious: - for this toy-case - the PGD attribute is the leading attribute that causes the lack of robustness. This information should be use by the AI scientist to refine the ML design in order to increase the robustness of the model.

D.5. Quantifying the negative impact due to the absence of R, U or M

In this section, we tackle the second point in the introduction of this chapter, concerning the interlinks between the RUM streamlines. These happen to appear already concretely on the available Confiance.ai components. As an example, UQModels starts in the Uncertainty streamline for its first group of functionalities concerning UQ, and then shifts to the Monitoring streamline as it uses the acquired quantities as KPIs which are then exploited for UQ-based OOD detection modules as a part of the monitoring system of the AI component. Beyond the concrete evidence of such links presented by the available Confiance.ai components, there are conceptual

and quantifiable reasons for the necessity of such links. We will present evidence for these in the form of concrete negative impacts observed in the resulting AI component, concerning R, U or M, which happen to be consequences of not taking into account either one out of three of the loops in the RUM Borromean rings.

D.5.1 Absence of Monitoring

Monitoring is a crucial aspect of ensuring the robustness and reliability of systems, especially in the context of complex and dynamic environments. The absence of monitoring can negatively impact robustness and uncertainty quantification in several ways: it hinders the ability to adapt to changing conditions, detect anomalies, maintain system health, implement effective fault tolerance, calibrate models accurately, and gather data for uncertainty quantification. These factors collectively contribute to reduced robustness and increased uncertainty in the performance of a system.

D.5.1.1 Lack of Real-time Feedback

Monitoring provides real-time feedback on the performance of a system. Without continuous monitoring, it becomes challenging to identify and address issues promptly. Shankar and Parameswaran (2022) show that real time feedback through observation of the ML components are necessary to detect issues in ML pipelines and deploy strategies to try to fix them.

In dynamic environments, conditions can change rapidly. Without real-time feedback, the system may fail to adapt to new challenges, leading to decreased robustness.

D.5.1.2 Inability to Detect Anomalies

Monitoring helps in detecting anomalies or unexpected behaviors in a system. Without proper monitoring, anomalies may go unnoticed until they cause significant issues (e.g. Shankar and Parameswaran (2022)).

Anomalies can introduce uncertainties that are not accounted for in the system's design, making it difficult to quantify and manage uncertainties (e.g. Seoni et al. (2023)).

D.5.1.3 Limited Understanding of AI Component Health

Monitoring provides insights into the overall health of an AI Component by tracking various metrics and indicators. In the absence of monitoring, there is limited visibility into the AI Component's state and its components (e.g. Shankar and Parameswaran (2022)).

Understanding system health is essential for robustness and uncertainty quantification because it allows for proactive maintenance and the identification of potential weaknesses.

D.5.1.4 Ineffective Fault Tolerance

Monitoring is critical for fault tolerance mechanisms. Without the ability to monitor and identify faults, the system may not be able to implement effective fault tolerance strategies as explained in Shankar and Parameswaran (2022).

Robust systems are designed to handle faults gracefully, and monitoring is a key element in detecting and responding to faults in real-time.

D.5.1.5 Inaccurate Model Calibration

Models used in systems often require calibration to accurately represent the underlying processes. Monitoring data is crucial for calibrating and updating models to reflect the current state of the system as explained by Feng et al. (2022).

Without ongoing monitoring, the models may become outdated, leading to inaccurate predictions and uncertainty quantification.

D.5.1.6 Limited Data for Uncertainty Quantification

Uncertainty quantification relies on data, and monitoring provides continuous data streams that can be used for assessing uncertainties in real-time as detailed in Mougan and Nielsen (2023).

Without monitoring, there may be limited data available for uncertainty quantification, making it difficult to assess the reliability of predictions and decisions.

D.5.2 Absence of Uncertainty Quantification

The absence of uncertainty quantification can have detrimental effects on both robustness and monitoring in various ways as explained in Seoni et al. (2023). Uncertainty quantification is essential for understanding the limitations and potential variations in system behavior, and its absence can lead to challenges in maintaining robust and effective systems: it can lead to overconfident decision-making, inaccurate risk assessment, ineffective adaptation to changing conditions, misleading monitoring indicators, limited sensitivity analysis, underestimation of errors, reduced confidence in decision support systems, and inadequate resource allocation. These factors collectively contribute to a decrease in the robustness and reliability of systems.

D.5.2.1 Lack of confidence/ML-risk indicators

Machine learning prediction are impacted by numerous sources of uncertainty. In the absence of uncertainty quantification, no information is available about their consequences on the model's prediction/decision, and its confidence in this inference. There is therefore a risk of model error, which is only superficially expressed here as an "average error", without consideration of data-conditionality which reveals more or less ambiguous data, that can deviate from the learning distribution, models weakness due to specifics data-context. Setting up predictive uncertainty mechanisms (e.g Malinin and Gales (2018); Corbière et al. (2019)), to express the increased risk of error associated with different sources (Noise Uncertainty Decision / Stochastic Target / OOS) allows us to produce model confidence indicators at prediction time, or even to valorize into risk-management processing.

D.5.2.2 Poor management of inputs and labels noise

Machine learning models are fed by inputs and labels that may be partly noisy. During training, this can degrade model performance, leading to an underconfident model. Several strategies can be used to face noisy label (e.g Lukasik et al. (2020); Frénay and Verleysen (2013); Northcutt

et al. (2021)) using uncertainty quantification to make label correction, or to produce more robust model.

During operation, it can lead to an increased risk of error due to a noise robustness problem. To face it, there are three complementary ways. The first, upstream of the model, by set up input pre-processing to filter noise. The second, within the model, by solving "tasks under uncertainty" using models that deal with uncertainty or which are robust to noise perturbation. The last, downstream of the model, by implementing predictive uncertainty mechanisms in addition to the model to mitigate it risks of error as a function of predicted uncertainty quantities.

D.5.2.3 Under or Overconfident Decision-Making

Confidence is an important element in a prediction/anomaly detection model (e.g Wang et al. (2021)). When a model expresses a form of confidence (score, pseudo-probability), it is important to ensure that this indicator is reliable. In the absence of a suitable incertitude protocol, a confidence score may be poorly calibrated, leading to the expression of over- or under-confidence. Under-confidence can lead to a loss of precision for AI Components, which may be too conservative and give less informative answers. On the other hand, overconfidence leads to an increased risk of error, threatening the reliability of AI component decisions, especially in a rare or unexpected situation.

The evaluation and calibration of confidence indicators is therefore crucial (e.g Mendil et al. (2022)) to the design of an AI component with properties confidence/risk assessment properties and must be designed and implemented in line with ODD specifications, in particular for the AI Component expected behavior facing rare and critical situations.

D.5.2.4 Ineffective Adaptation to Changing Conditions

Environment and collection system can be may be subject to changes in conditions (usury, sensor modifications, re-calibration) which is very challenging for learning algorithm O'reilly (2013). AI components designe to solve tasks in these environments need to be resilient to certain changes. a totally deterministic system which may struggle to adapt effectively to changing circumstances, reducing its overall robustness in dynamic environments. Uncertainty mechanism can allow a AI Component to better handle the presence of noise in features, and to handle part of influence of irreducible variability on model decision for ML-risk-assessment. This can lead to models that are more robust by design (by learn on noise-augmented datasets, or through learning processes that guarantee some form of invariance to data noise). This can also lead to AI Components internally exploiting risk-assessment model mechanisms to mitigate a model's decisions with regard to a form of aversion to certain types of error.

D.5.2.5 Misleading Monitoring Indicators

Monitoring systems rely on accurate indicators to assess the health and performance of the system Souifi et al. (2022). These indicators are obtained through more or less complex procedures, which may be impacted by various sources of uncertainty likely to introduce variability within these indicators Mauro et al. (2021). Without uncertainties management in monitoring protocol, decisions taken on the basis of non-robustness indicators may under- or over-estimate the actual state of the system, leading to inappropriate risk-taking which may result in system failure or, conversely, sub-optimal utilization.

Uncertainty management can help to better integrate impact of uncertainty sources (either "aleatoric" linked to the input, or "Epistemic" linked to modeling steps intermediate) during monitoring process. For example, model robustness indicators to a perturbation may be based on a data transformation (or generation) process that simulates the perturbation with a certain range of variability. Robust methodologies (Including Statistical test or quantification of indicators variability using cross-validation) can mitigate these variability and limit the risks involved in collecting these indicators.

D.5.2.6 Limited sensitivity analysis

It's not always easy to understand the relevance and therefore to trust deep models, which can act like black boxes. Without suitable qualitative evaluation protocol, models could be validated (based on average performance metrics) without noticing certain suspicious behaviors which require in-depth investigation, and model adjustment. As example, an image classification model can show an abnormal decision or even uncertainty sensitivity to some background pixel to which it should be theoretically invariant according to expert knowledge, or can keep high confidence despite facing data manipulated to be more ambiguous. Sensitivity and uncertainty analysis can help identifying the impact of specified factors on system outcomes (e.g Fel et al. (2021); Gu et al. (2021)), by observing influence of designed inputs modification/perturbations on models outputs. These techniques can be used to verify explainability, robustness and risk-assessment by analyzing the presence of desired behaviors and exhibit some aberrant inferences based on expert knowledge.

D.5.2.7 Reduced Confidence in Decision Support Systems

Machine learning approaches are increasingly used in decision support systems for complex systems (energy, mobility, industry), to synthesize massive data that cannot be interpreted by humans (e.g Yao et al. (2024)). However, decision support systems must manage the impact of sources of uncertainty linked to the system, but also hold uncertainties of models in charge of characterizing and synthesizing the state of the system. The lack of uncertainty quantification can erode confidence in decision support systems, as stakeholders may be unsure about the reliability of the information provided.

The implementation of models capable not only of managing the uncertainty of the environment, but also of expressing the uncertainty inherent to theirs modeling, is mandatory to provide to the decision support system, all the information needed to guarantee its trust.

D.5.2.8 Inadequate experiments planification

Robust AI Components require efficient resource allocation, and uncertainty quantification plays a key role in optimizing resource utilization Nguyen et al. (2022). The conception of AI components is a trade-off between system performance and time/cost constraints, leading to the classic problems of "design of experimentation": in a large research space, how to choose the experiments that will be the most rewarding, according to cost ? There are various works based on uncertainties, which propose methods and entropy criteria to address these issues. Such issues can involve for features, model architecture or even the dataset construction. For example, an active learning procedure can be set up to face poorly labelled dataset, to optimize the costly interactions with operational expert. Without proper uncertainty assessment, there is a risk of

misallocating resources, leading to sub-optimal AI Component. potentially compromising the system's robustness in resource-constrained situations.

D.5.3 Absence of Robustness

The absence of robustness in a system can have adverse effects on uncertainty quantification and monitoring, creating challenges in maintaining reliable and effective operations: it can lead to uncertain system responses, inadequate model calibration, unreliable monitoring indicators, increased false alarms, difficulty in identifying root causes, challenges in adaptive control, compromised fault tolerance, and limited resilience to environmental changes. These factors collectively undermine the effectiveness of uncertainty quantification and monitoring efforts in maintaining a reliable and well-performing system.

D.5.3.1 Uncertain System Responses

Robustness is the ability of a system to maintain its performance despite variations and uncertainties in inputs or conditions. When a system, or ML model, lacks robustness, it becomes susceptible to unreliable response when faced with perturbations or uncertainty. For the past decade, it as been well established Goodfellow et al. (2014) that deep neural networks are vulnerable to adversarial examples (inputs that are almost indistinguishable from natural data and yet classified incorrectly by the network). Even state-of-the-art machine learning models can be very sensible to perturbations of their inputs. In the absence of robustness, the system may exhibit unpredictable responses to perturbations - either adversarial or environmental - making it difficult to quantify and manage the system's response accurately under uncertainty.

D.5.3.2 Inadequate Model Calibration

Robust systems often require well-calibrated models that accurately represent the underlying processes. Lack of robustness may lead to inaccurate model calibration.

Qin et al. (2021) have investigated the connection between the adversarial robustness of a model and it's calibration. They showed that " input data points that are sensitive to small adversarial perturbations [...] are more likely to have poorly calibrated predictions". Also, when presented with effective adversarial examples - by construction of these inputs - ml-models tends to have a high confidence for their misclassified predictions.

Inaccurate models can compromise the quality of uncertainty quantification by misrepresenting the relationships between inputs and outputs. ML-models are very efficient in underlying correlations between inputs and outputs. These correlations can be spurious and/or cause a lack of robustness to a change of context or environment. Zhang et al. (2020) have shown an improvement in robustness to unseen adversarial attacks (FSGM and PGD) by creating a "deep CAusal Manipulation Augmented model" that can separate the representation of manipulations that are causally invariant (ie: that does not change the relation between inputs and outputs) with other factors that influences the observed data.

D.5.3.3 Unreliable Monitoring Indicators

Monitoring relies on stable and reliable indicators to assess the health and performance of a system. A lack of robustness can result in erratic system behavior, leading to unreliable monitoring

indicators. Moreover, monitoring of a non-robust system can generate misleading performance metrics. In the context of adversarial attacks, a system can perform overly well under nominal circumstances but poorly when faced with specifically crafted perturbations.

Unreliable indicators make it challenging to detect anomalies and assess the system's state accurately. A system lacking robustness will generate more erratic quality indicators (model accuracy, rate of false positives ...) making it less trustworthy in it's predictions, it will also be less straightforward to distinguish between real data anomalies and poor model predictions under some perturbation.

D.5.3.4 Increased False Alarms and Alerts

A system lacking robustness may trigger false alarms or alerts in monitoring systems, as it may exhibit unexpected behaviors that are not necessarily indicative of critical issues. It will also generate false positives when faced with perturbations (either adversarial or environmental) due to poor performances under those conditions. On the other hand, robust models are designed to be less sensitive to minor fluctuations and noise in the input data, thus they are less likely to misinterpret random noise or benign variations as significant anomalies, thereby lowering the false alarm rate. It is interesting to note that, because of that, robust models may be more prone to declare small malicious data perturbations as false negative.

The absence of robustness can lead to a decrease in the effectiveness of monitoring and uncertainty quantification by generating unnecessary alerts and diverting attention from actual problems. Indeed, each false positive alarm triggered by the system generates an operational response that, depending on the context, could take some time to resolve.

D.5.3.5 Difficulty in Identifying Root Causes

Robust systems are designed to handle variations and uncertainties while maintaining stable performance. The absence of robustness can make it difficult to identify the root causes of issues when they arise.

Without a clear understanding of root causes, uncertainty quantification and monitoring efforts may be ineffective in addressing underlying problems.

D.5.3.6 Challenges in Adaptive Control

Robust systems are better equipped to adapt to changing conditions and disturbances. In the absence of robustness, adaptive control mechanisms may struggle to maintain system stability.

This instability can affect the accuracy of uncertainty quantification and compromise the reliability of monitoring systems.

D.5.3.7 Compromised Fault Tolerance

Robustness is closely tied to a system's fault tolerance capabilities. A lack of robustness may result in decreased fault tolerance, making the system more susceptible to failures.

Compromised fault tolerance can impact the system's ability to recover from unexpected events, affecting both uncertainty quantification and monitoring.

D.5.3.8 Limited Resilience to Environmental Changes

Systems lacking robustness may struggle to adapt to changes in the environment, leading to increased uncertainty in their performance. Sensitivity to environmental perturbations is a key part of the robustness evaluation of a ML model. These environmental perturbations can correspond to a natural change in the deployment context of a system. We can cite for example a change in pixel intensity or the presence of an image blur.

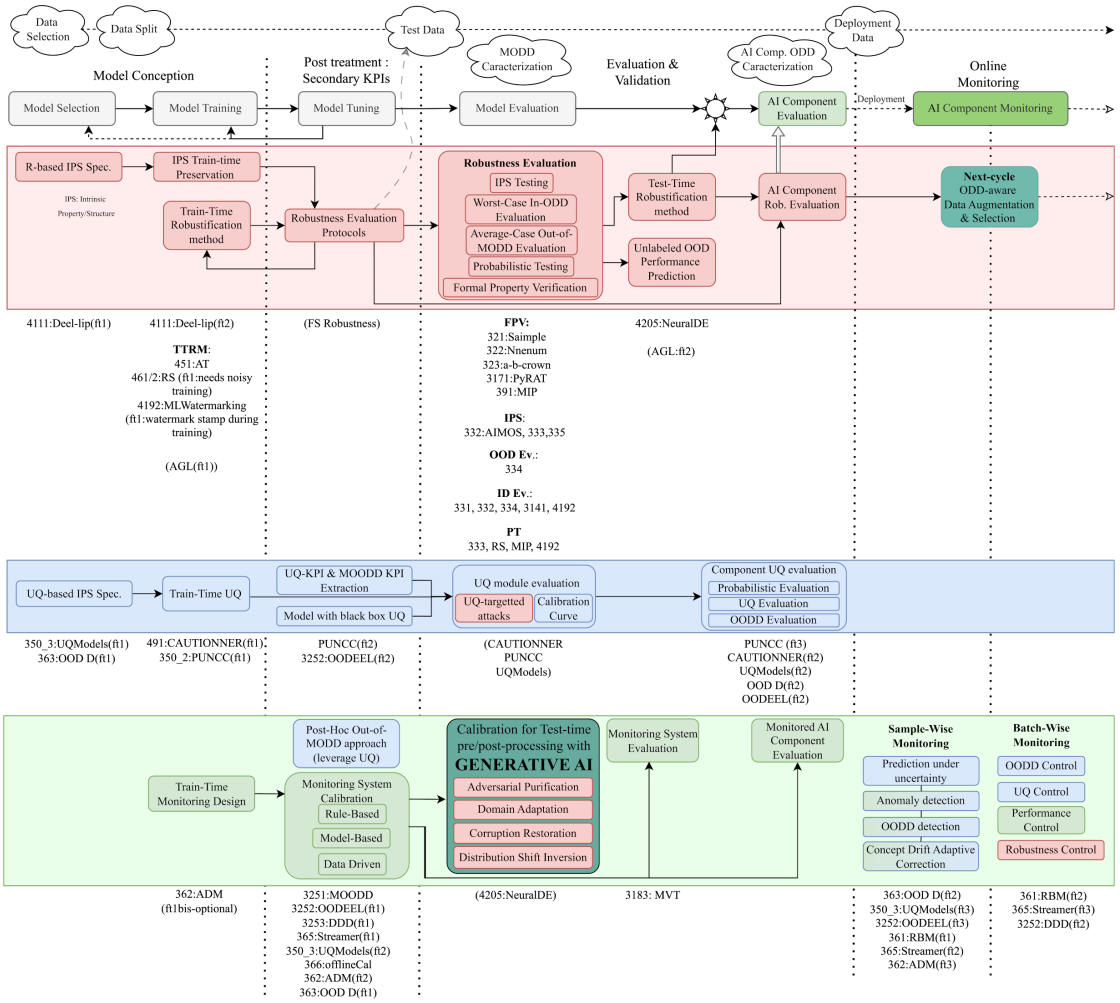
Monitoring and uncertainty quantification efforts may be challenged by the system's limited resilience to external variations and disturbances.

D.5.3.9 Vulnerability of uncertainty quantification

More recently, people have shown the sensitivity of uncertainty quantification estimates to adversarial attacks. For instance, Ledda et al. (2023). showed a new threat model used to specifically manipulate uncertainty quantification estimators without specifically trying to induce a miss-classification of the model.

D.6. The Confiance.ai's R/U/M components reframed in the RUM methodology

In hope to propose a comprehensive and self-contained account on the R/U/M available components from the program Confiance.ai, we sketched a scheme decomposing such components into their different functionalities and we situated the latter on each of their corresponding streamlines and at the moment that we recommend the user of these components to start taking into consideration such functionalities. For instance, although the PUNCC component *does not* depend on the training on the model and can be considered as a post-treatment module, the fact that it necessitates a specific choice in the data split so that the calibration data is indeed exchangeable with the training data, its consideration must be done already when the model is being trained. The following figure encompasses such scheme:


Data Selection
Data Split
Test Data
MODD Caracterization
Evaluation & Validation
AI Comp. ODD Caracterization
Deployment Data
Model Conception
Post treatment : Secondary KPIs
Online Monitoring
Model Selection
Model Training
Model Tuning
Model Evaluation
AI Component Evaluation
Deployment
AI Component Monitoring
R-based IPS Spec.
IPS: Intrinsic Property/Structure
IPS Train-time Preservation
Train-Time Robustification method
Robustness Evaluation Protocols
Robustness Evaluation
IPS Testing
Worst-Case In-ODD Evaluation
Average-Case Out-of-MODD Evaluation
Probabilistic Testing
Formal Property Verification
Test-Time Robustification method
Unlabeled OOD Performance Prediction
AI Component Rob. Evaluation
Next-cycle ODD-aware Data Augmentation & Selection
4111:Deel-lip(ft1)
4111:Deel-lip(ft2)
TTRM:
451:AT
461/2:RS (ft1:needs noisy training)
4192:MLWatermarking (ft1:watermark stamp during training)
(AGL(ft1))
(FS Robustness)
FPV:
321:Saimple
322:Nnenum
323:a-b-crown
3171:PyRAT
391:MIP
IPS:
332:AIMOS, 333,335
OOD Ev.:
334
ID Ev.:
331, 332, 334, 3141, 4192
PT
333, RS, MIP, 4192
4205:NeuralDE
(AGL:ft2)
UQ-based IPS Spec.
Train-Time UQ
UQ-KPI & MOODD KPI Extraction
Model with black box UQ
UQ module evaluation
UQ-targetted attacks
Calibration Curve
Component UQ evaluation
Probabilistic Evaluation
UQ Evaluation
OODD Evaluation
350_3:UQModels(ft1)
363:OOD D(ft1)
491:CAUTIONNER(ft1)
350_2:PUNCC(ft1)
PUNCC(ft2)
3252:OODEEL(ft2)
(CAUTIONNER
PUNCC
UQModels)
PUNCC (ft3)
CAUTIONNER(ft2)
UQModels(ft2)
OOD D(ft2)
OODEEL(ft2)
Post-Hoc Out-of-MODD approach (leverage UQ)
Train-Time Monitoring Design
Monitoring System Calibration
Rule-Based
Model-Based
Data Driven
Calibration for Test-time pre/post-processing with GENERATIVE AI
Adversarial Purification
Domain Adaptation
Corruption Restoration
Distribution Shift Inversion
Monitoring System Evaluation
Monitored AI Component Evaluation
Sample-Wise Monitoring
Prediction under uncertainty
Anomaly detection
OODD detection
Concept Drift Adaptive Correction
Batch-Wise Monitoring
OODD Control
UQ Control
Performance Control
Robustness Control
362:ADM
(ft1bis-optional)
3251:MOODD
3252:OODEEL(ft1)
3253:DDD(ft1)
365:Streamer(ft1)
350_3:UQModels(ft2)
366:offlineCal
362:ADM(ft2)
363:OOD D(ft1)
(4205:NeuralDE)
3183: MVT
363:OOD D(ft2)
350_3:UQModels(ft3)
3252:OODEEL(ft3)
361:RBM(ft1)
365:Streamer(ft2)
362:ADM(ft3)
361:RBM(ft2)
365:Streamer(ft3)
3252:DDD(ft2)

E. Conclusions and Perspectives

The journey towards system-level industrial adoption of the RUM methodology marks a transformative leap in the field of Machine Learning. By re-framing Robustness, Uncertainty Quantification, and Monitoring within the "RUM Methodology," we have unlocked a more comprehensive understanding of these foundational pillars. Through a thorough examination of the literature and a series of rigorous experiments, we've identified new attributes that provide a deeper insight into the dynamics of ML models and the AI components where these interact with auxiliary RUM modules. All in all, the RUM methodology aims to propose a first exploration on what the perimeter of an AI component is about, and which can be illustrated in Figure E.1.

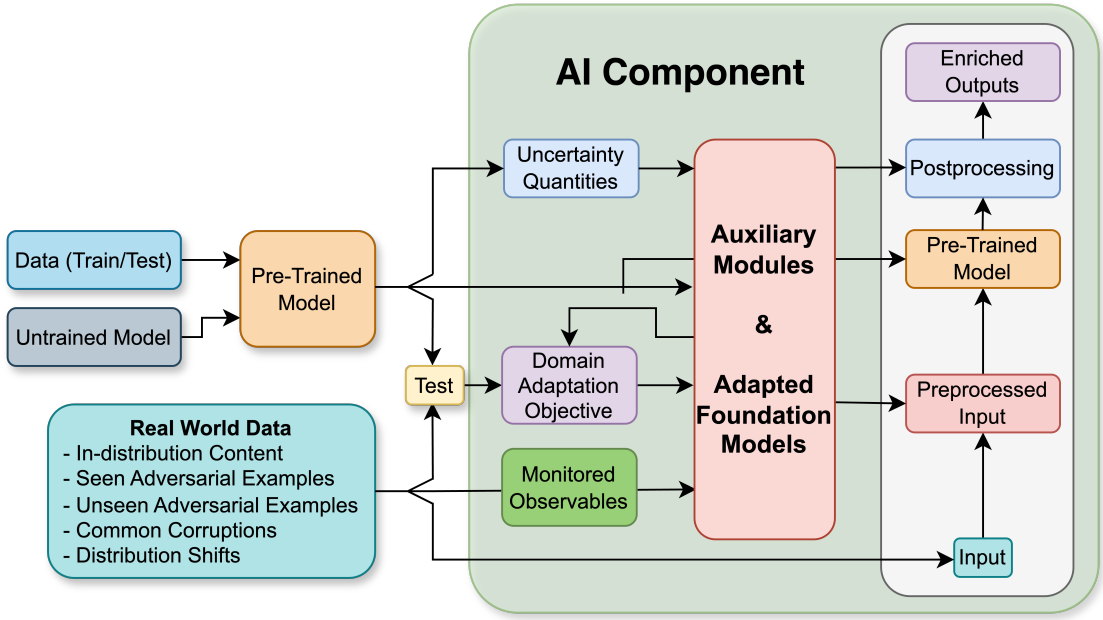


Figure E.1: Anatomy of an AI Component.

These new attributes are categorized into two distinct types: the "RUM-based analytic attributes" and the "RUM aggregate attributes." The former enhances existing metrics by offering a more critical and nuanced perspective on Robustness, Uncertainty Quantification, and Monitoring. The latter requires a holistic evaluation approach, acknowledging that these three components are interdependent and must be analyzed in tandem to fully understand and improve a model's trustworthiness.

Our proposal introduces these two families of attributes into a flexible and adaptive methodology that supports continuous research and embraces a unified, enriched life-cycle perspective. This approach allows for the incorporation of our innovative insights and aims to set a new standard for addressing trustworthiness in ML. We are confident that the RUM framework, implemented

as a tool internally by the Confiance.ai participants, will serve as a solid, systematic tool to guide researchers and practitioners in building more trustworthy models and AI components.

Looking ahead, the potential applications of the RUM methodology extend beyond the confines of its current scope. While we have focused primarily on Robustness, Uncertainty Quantification, and Monitoring, the versatility of our framework suggests that it could be expanded to encompass other critical aspects of trustworthiness in ML, such as data quality and integrity and the burgeoning field of explainability. By extending the RUM methodology to these additional domains within the future sequences of the Confiance.ai program, we could aim to create a more holistic and comprehensive tool that not only addresses the current challenges in ML but also anticipates future developments in the field. With this framework, we pave the way for a new way of tackling trustworthiness in ML, ensuring AI components are not only performant but also reliable in their way of addressing Robustness, Uncertainty Quantification, and Monitoring attributes.

Bibliography

Ashmore, R., Calinescu, R., and Paterson, C. (2021). Assuring the machine learning lifecycle: Desiderata, methods, and challenges. *ACM Computing Surveys (CSUR)*, 54(5):1–39.

Batten, B., Hosseini, M., and Lomuscio, A. (2024). Tight verification of probabilistic robustness in bayesian neural networks. In *The 27th International Conference on Artificial Intelligence and Statistics, AISTATS*, volume 238.

Bayram, F., Ahmed, B. S., and Kassler, A. (2022). From concept drift to model degradation: An overview on performance-aware drift detectors. *Knowledge-Based Systems*, 245:108632.

Carlini, N., Paleka, D., Dvijotham, K. D., Steinke, T., Hayase, J., Cooper, A. F., Lee, K., Jagielski, M., Nasr, M., Conmy, A., et al. (2024). Stealing part of a production language model. In *International Conference on Machine Learning*. PMLR.

Chen, H., Dong, Y., Shao, S., Hao, Z., Yang, X., Su, H., and Zhu, J. (2024). Your diffusion model is secretly a certifiably robust classifier. *arXiv preprint arXiv:2402.02316*.

Chung, Y., Char, I., Guo, H., Schneider, J., and Neiswanger, W. (2021). Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. *arXiv preprint arXiv:2109.10254*.

Cohen, J., Rosenfeld, E., and Kolter, Z. (2019a). Certified adversarial robustness via randomized smoothing. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320. PMLR.

Cohen, J., Rosenfeld, E., and Kolter, Z. (2019b). Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pages 1310–1320. PMLR.

Corbière, C., Thome, N., Bar-Hen, A., Cord, M., and Pérez, P. (2019). Addressing failure prediction by learning model confidence. *arXiv preprint arXiv:1910.04851*.

Croce, F., Gowal, S., Brunner, T., Shelhamer, E., Hein, M., and Cemgil, T. (2022). Evaluating the adversarial robustness of adaptive test-time defenses. *arXiv preprint arXiv:2202.13711*.

Cuninghame-Green, R. (2012). *Minimax algebra*, volume 166. Springer Science & Business Media.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142.

Desmartin, R., Isac, O., Komendantskaya, E., Stark, K., Passmore, G., and Katz, G. (2024). A certified proof checker for deep neural network verification. *arXiv preprint arXiv:2405.10611*.

F., M., E., J., G., F., H., D., C., G., A., G., B., B., L., P., L., A., H., B., B., B., D., D., J.-B., G., A., H., S., P., K., D., C., P., J.-M., G., C., C., S., P., M., D., C., C., L., G., D., G. F., B., L., S., G., and A., A. (2021). White paper machine learning in certified systems. Technical report, DEEL PROJECT.

Fel, T., Cadène, R., Chalvidal, M., Cord, M., Vigouroux, D., and Serre, T. (2021). Look at the variance! efficient black-box explanations with sobol-based sensitivity analysis. *Advances in neural information processing systems*, 34:26005–26014.

Feng, J., Phillips, R. V., Malenica, I., Bishara, A., Hubbard, A. E., Celi, L. A., and Pirracchio, R. (2022). Clinical artificial intelligence quality improvement: towards continual monitoring and updating of ai algorithms in healthcare. *NPJ digital medicine*, 5(1):66.

Frénay, B. and Verleysen, M. (2013). Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869.

Gaubert, S. and Max-Plus (1997). Methods and applications of (max,+) linear algebra. In *STACS 97: 14th Annual Symposium on Theoretical Aspects of Computer Science*, volume 14, pages 261–282. Springer.

Gawlikowski, J., Tassi, C. R. N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A., Triebel, R., Jung, P., Roscher, R., Shahzad, M., Yang, W., Bamler, R., and Zhu, X. X. (2021). A Survey of Uncertainty in Deep Neural Networks. *arXiv:2107.03342 [cs, stat]*. arXiv: 2107.03342.

Gendler, A., Weng, T.-W., Daniel, L., and Romano, Y. (2022). Adversarially robust conformal prediction. In *International Conference on Learning Representations*.

Gondran, M. and Minoux, M. (2008). *Graphs, dioids and semirings: new models and algorithms*, volume 41. Springer Science & Business Media.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Gu, B., Zhang, T., Meng, H., and Zhang, J. (2021). Short-term forecasting and uncertainty analysis of wind power based on long short-term memory, cloud model and non-parametric kernel density estimation. *Renewable Energy*, 164:687–708.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. Number: arXiv:1706.04599 arXiv:1706.04599 [cs].

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al. (2021). The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349.

Hendrycks, D. and Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*.

Humblot-Renaux, G., Escalera, S., and Moeslund, T. B. (2023). Beyond auroc & co. for evaluating out-of-distribution detection performance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3880–3889.

Isac, O., Barrett, C. W., Zhang, M., and Katz, G. (2022). Neural network verification with proof production. In *FMCAD*, pages 38–48.

Jeary, L., Kuipers, T., Hosseini, M., and Paoletti, N. (2024). Verifiably robust conformal prediction. *arXiv preprint arXiv:2405.18942*.

Jia, K. and Rinard, M. (2021). Exploiting verified neural networks via floating point numerical error. In *Static Analysis: 28th International Symposium, SAS 2021, Chicago, IL, USA, October 17–19, 2021, Proceedings 28*, pages 191–205. Springer.

Kaakai, F., Adibhatla, S. S., Pai, G., and Escorihuela, E. (2023). Data-centric operational design domain characterization for machine learning-based aeronautical products.

Kapusta, K., Mattioli, L., Addad, B., and Lansari, M. (2024). Protecting ownership rights of ml models using watermarking in the light of adversarial attacks. *AI and Ethics*, pages 1–9.

Kim, E., Sun, M., Raghunathan, A., and Kolter, Z. (2023). Reliable test-time adaptation via agreement-on-the-line. *arXiv preprint arXiv:2310.04941*.

Kotha, S., Brix, C., Kolter, J. Z., Dvijotham, K. D., and Zhang, H. (2024). Provably bounding neural network preimages. *Advances in Neural Information Processing Systems*.

Labreuche, C. (2016). On capacities characterized by two weight vectors. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems: 16th International Conference, IPMU*, pages 23–34. Springer.

Ledda, E., Angioni, D., Piras, G., Fumera, G., Biggio, B., and Roli, F. (2023). Adversarial attacks against uncertainty quantification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4599–4608.

Leino, K., Wang, Z., and Fredrikson, M. (2021). Globally-robust neural networks. In *International Conference on Machine Learning*, pages 6212–6222. PMLR.

Leung, K., Aréchiga, N., and Pavone, M. (2023). Backpropagation through signal temporal logic specifications: Infusing logical structure into gradient-based methods. *The International Journal of Robotics Research*, 42(6):356–370.

Li, Y., Chen, Z., Jin, K., Wang, J., Li, B., and Xiao, C. (2024). Consistency purification: Effective and efficient diffusion purification towards certified robustness. *arXiv preprint arXiv:2407.00623*.

Liu, J., Wang, T., Cui, P., and Namkoong, H. (2024a). On the need for a language describing distribution shifts: Illustrations on tabular datasets. *Advances in Neural Information Processing Systems*, 36.

Liu, Z., Cui, Y., Yan, Y., Xu, Y., Ji, X., Liu, X., and Chan, A. B. (2024b). The pitfalls and promise of conformal inference under adversarial attacks. In *International Conference of Machine Learning*.

Lorenz, P., Fernandez, M., Müller, J., and Köthe, U. (2024). Deciphering the definition of adversarial robustness for post-hoc ood detectors. *arXiv preprint arXiv:2406.15104*.

Lukasik, M., Bhojanapalli, S., Menon, A., and Kumar, S. (2020). Does label smoothing mitigate label noise? In *International Conference on Machine Learning*, pages 6448–6458. PMLR.

Machin, M., Guiochet, J., Waeselynck, H., Blanquart, J.-P., Roy, M., and Masson, L. (2016). Smof: A safety monitoring framework for autonomous systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(5):702–715.

Malinin, A. and Gales, M. (2018). Predictive uncertainty estimation via prior networks. *arXiv preprint arXiv:1802.10501*.

Mattioli, J., Sohier, H., et al. (2024). An overview of key trustworthiness attributes and kpis for trusted ml-based systems engineering. *AI and Ethics*, 4(1):15–25.

Mauro, V., Giusti, C., Marchetti, S., and Pratesi, M. (2021). Does uncertainty in single indicators affect the reliability of composite indexes? an application to the measurement of environmental performances of italian regions. *Ecological Indicators*, 127:107740.

Mendil, M., Mossina, L., Nabhan, M., and Pasini, K. (2022). Robust gas demand forecasting with conformal prediction. In *Conformal and Probabilistic Prediction with Applications*, pages 169–187. PMLR.

Mougan, C. and Nielsen, D. S. (2023). Monitoring model deterioration with explainable uncertainty estimation via non-parametric bootstrap. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15037–15045.

Mu, N. and Gilmer, J. (2019). MNIST-C: A Robustness Benchmark for Computer Vision. In *ICML Workshop on Uncertainty and Robustness in Deep Learning*.

Nguyen, V.-L., Shaker, M. H., and Hüllermeier, E. (2022). How to measure uncertainty in uncertainty sampling for active learning. *Machine Learning*, 111(1):89–122.

Northcutt, C., Jiang, L., and Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411.

Olsder, G., Quadrat, J., et al. (1992). Synchronization and linearity.

O'reilly, J. X. (2013). Making predictions in a changing world—inference, uncertainty, and learning. *Frontiers in neuroscience*, 7:105.

Peng, S., Xu, W., Cornelius, C., Hull, M., Li, K., Duggal, R., Phute, M., Martin, J., and Chau, D. H. (2023). Robust principles: Architectural design principles for adversarially robust cnns. *arXiv preprint arXiv:2308.16258*.

Psaros, A. F., Meng, X., Zou, Z., Guo, L., and Karniadakis, G. E. (2022). Uncertainty Quantification in Scientific Machine Learning: Methods, Metrics, and Comparisons. *arXiv:2201.07766 [cs]*. arXiv: 2201.07766.

Qin, Y., Wang, X., Beutel, A., and Chi, E. (2021). Improving calibration through the relationship with adversarial robustness. *Advances in Neural Information Processing Systems*, 34:14358–14369.

Rice, L., Wong, E., and Kolter, Z. (2020). Overfitting in adversarially robust deep learning. In *International conference on machine learning*, pages 8093–8104. PMLR.

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., and Müller, K.-R. (2021). A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795.

Rumezhak, T., Eiras, F. G., S. Torr, P. H., and Bibi, A. (2023). Rancer: Non-axis aligned anisotropic certification with randomized smoothing. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 4661–4669.

SAE, E. . (December 2022). Process standard for development and certification/approval of aeronautical safety-related products implementing ai (draft 4b). Technical report, EUROCAE.

SAE J3016 (2018). Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems.

Sakamoto, K., Sakamoto, R., Tanabe, M., Akagawa, M., Hayashi, Y., Yaguchi, M., Suzuki, M., and Matsuo, Y. (2024). The Geometry of Diffusion Models: Tubular Neighbourhoods and Singularities. *Proceedings of the Geometry-grounded Representation Learning and Generative Modeling at the 41st International Conference on Machine Learning*.

Salman, H., Sun, M., Yang, G., Kapoor, A., and Kolter, J. Z. (2020). Black-box smoothing: A provable defense for pretrained classifiers. *CoRR*, abs/2003.01908.

Sam, D., Pukdee, R., Jeong, D. P., Byun, Y., and Kolter, J. Z. (2024). Bayesian neural networks with domain knowledge priors. In *Knowledge and Logical Reasoning in the Era of Data-driven Learning Workshop, International Conference of Machine Learning*.

Schlarmann, C., Singh, N. D., Croce, F., and Hein, M. (2024). Robust CLIP: Unsupervised adversarial fine-tuning of vision embeddings for robust large vision-language models. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F., editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 43685–43704. PMLR.

Seoni, S., Jahmunah, V., Salvi, M., Barua, P. D., Molinari, F., and Acharya, U. R. (2023). Application of uncertainty quantification to artificial intelligence in healthcare: A review of last decade (2013–2023). *Computers in Biology and Medicine*, 165:107441.

Serrurier, M., Mamalet, F., González-Sanz, A., Boissin, T., Loubes, J.-M., and Del Barrio, E. (2021). Achieving robustness in classification using optimal transport with hinge regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 505–514.

Shankar, S. and Parameswaran, A. G. (2022). Towards observability for machine learning pipelines. In *CIDR*.

Souifi, A., Boulanger, Z. C., Zolghadri, M., Barkallah, M., and Haddar, M. (2022). Uncertainty of key performance indicators for industry 4.0: A methodology based on the theory of belief functions. *Computers in Industry*, 140:103666.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv:1312.6199*.

Thales, I. S. (2024). Dispositif de surveillance prÉdictive du fonctionnement d'une unitÉ d'estimation d'une grandeur physique.

Tsilivis, N., Frank, N., Srebro, N., and Kempe, J. (2024). The price of implicit bias in adversarially robust generalization. *arXiv preprint arXiv:2406.04981*.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. (2019). Robustness may be at odds with accuracy. *ICLR*. cite arxiv:1805.12152.

Wang, D.-B., Feng, L., and Zhang, M.-L. (2021). Rethinking calibration of deep neural networks: Do not be afraid of overconfidence. *Advances in Neural Information Processing Systems*, 34:11809–11820.

Wicker, M., Laurenti, L., Patane, A., Chen, Z., Zhang, Z., and Kwiatkowska, M. (2021). Bayesian inference with certifiable adversarial robustness. In *International Conference on Artificial Intelligence and Statistics*, pages 2431–2439. PMLR.

Wu, H., Isac, O., Zeljić, A., Tagomori, T., Daggitt, M., Kokke, W., Refaeli, I., Amir, G., Julian, K., Bassan, S., et al. (2024). Marabou 2.0: A versatile formal analyzer of neural networks. *arXiv preprint arXiv:2401.14461*.

Xiao, C., Chen, Z., Jin, K., Wang, J., Nie, W., Liu, M., Anandkumar, A., Li, B., and Song, D. (2023). Densepure: Understanding diffusion models towards adversarial robustness. *Proceedings of ICLR 2023*.

Yan, G., Romano, Y., and Weng, T.-W. (2024). Provably robust conformal prediction with improved efficiency. In *International Conference on Learning Representations*.

Yang, J., Wang, P., Zou, D., Zhou, Z., Ding, K., Peng, W., Wang, H., Chen, G., Li, B., Sun, Y., et al. (2022). Openood: Benchmarking generalized out-of-distribution detection. *Advances in Neural Information Processing Systems*, 35:32598–32611.

Yang, J., Zhou, K., Li, Y., and Liu, Z. (2021). Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*.

Yao, Y., Han, T., Yu, J., and Xie, M. (2024). Uncertainty-aware deep learning for reliable health monitoring in safety-critical energy systems. *Energy*, 291:130419.

Yu, H., Zhang, X., Xu, R., Liu, J., He, Y., and Cui, P. (2024). Rethinking the evaluation protocol of domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21897–21908.

Zargarbashi, S. H., Akhondzadeh, M. S., and Bojchevski, A. (2024). Robust yet efficient conformal prediction sets. *arXiv preprint arXiv:2407.09165*.

Zhang, C., Zhang, K., and Li, Y. (2020). A causal view on robustness of neural networks. *Advances in Neural Information Processing Systems*, 33:289–301.

Zhang, H., Yu, Y., Jiao, J., Xing, E., Ghaoui, L. E., and Jordan, M. (2019). Theoretically principled trade-off between robustness and accuracy. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7472–7482. PMLR.

Zou, Y., Kawaguchi, K., Liu, Y., Liu, J., Lee, M.-L., and Hsu, W. (2024). Towards robust out-of-distribution generalization bounds via sharpness. In *The Twelfth International Conference on Learning Representations*.

Title: The RUM Methodology

Keywords: Robustness, Uncertainty, Monitoring, End-to-End Approach, Trustworthiness

This document presents and develops the so-called RUM Methodology, a systemic methodological framework that integrates Robustness, Uncertainty and Monitoring attributes, protocols, and analysis into a unified approach. The RUM Methodology proposes new trustworthiness attributes, evaluation protocols and industrial recommendations with particular emphasis on the available components developed within the Confiance.ai program.

Our partners: