Coupling algebraic topology theory, formal methods and safety requirements toward a new coverage metric for artificial intelligence models

Faouzi Adjed, Mallek Mziou-Sallami, Frédéric Pelliccia, Mehdi Rezzoug, Lucas Schott, Christophe Bohn, Yesmina Jaafra

Noname manuscript No.
(will be inserted by the editor)

Coupling Algebraic Topology Theory, Formal Methods and Safety Requirements Towards a New Coverage Metric for Artificial Intelligence Models

Faouzi Adjed* · **Mallek Mziou-Sallami** · **Frédéric Pelliccia** · **Mehdi Rezzoug** · **Lucas Schott** · **Christophe Bohn** · **Yesmina Jaafra**

Received: date / Accepted: date

Abstract Safety requirements are among the main barriers to the industrialization of machine learning based on deep learning architectures. In this work, a new metric of data coverage is presented by exploring the algebraic topology theory and the abstract interpretation process. The algebraic topology connects the cloud points of the dataset and the abstract interpretation evaluates the robustness of the model. Thus, the coverage metric evaluates simultaneously the dataset and the robustness, and highlights safe and unsafe areas. We also propose the first complete process to evaluate, in terms of data completeness, the machine learning models by providing a workflow based on the proposed metric and a set of safety requirements applied on autonomous driving. The obtained results provide an interpretable coverage evaluation and a promising line of research in the industrialization of artificial intelligence models. It is important to mention that the proposed metric is not dependent on the specific data. In other terms, it can be applied on 1 to $n-$dimensional data.

Keywords Coverage Metric · Topological Data Analysis · Deep Reinforcement Learning · Abstract Interpretation for Artificial Intelligence

F. Adjed · F. Pellicia · M. Rezzoug · L. Schott · C. Bohn · Y. Jaafra
IRT-SystemX, 2 boulevard Thomas Gobert, Palaiseau 91120, France
E-mail: firstname.lastname@irt-systemx.fr

M. Mziou-Sallami
CEA, The French Alternative Energies and Atomic Energy Commission, 2 Rue Gaston Cremieux, 91000, France
E-mail: mallek.mziou@cea.fr

F. Adjed · Y. Jaafra
Expleo France, 3 Avenue des Près, Montigny-le-Bretonneux 78180, France
E-mail: firstname.lastname@expleogroup.com

F. Pelliccia
Apsys, 36 rue Raymond Grimaud, 31700 BLAGNAC, France
E-mail: frederic.pelliccia@apsys-airbus.com

*Corresponding Author(faouzi.adjed@irt-systemx.fr)

1 Introduction

Over the last few decades, deep learning models have been developed in several areas. However, their validation and certification remain very challenging for the scientific community. Indeed, as illustrated by Goodfellow et al. [1], deep neural network models are vulnerable to adversarial examples . The authors investigated linear and non linear explanations of adversarial examples, which constitute an obstacle for the deployment of artificial intelligence (AI) models based on deep neural network architectures. Furthermore, the deep learning models are also vulnerable to privacy attacks as summarized by Liu et al. [2].

Arrieta et al. [3] draw the chronology of the developments of AI models understanding, by exploring several notions such as explainability and interpretability. The authors proposed the concept of responsible artificial intelligence where a review of recent literature around explainable AI models is presented. However, they only targeted taxonomy/classification models.
Holzinger et al. [4] proposed the use of a new approach by integrating network topology in the explainable AI process. Authors focused their contribution on human interactive explainability using graph neural networks.

Nowadays, according to our knowledge, there is no common process which allows the industrialization of AI models based on deep learning architectures. In the current work, we propose a common methodology to

understand, validate and contribute to the certification of deep learning models. First of all, we define Deep Learning Architectures (DLA), which include supervised, unsupervised and reinforcement learning under deep learning architecture paradigms. Subsequently, we explore a specific case of Deep Reinforcement Learning (DRL) applied to autonomous driving.

The proposed approach aims to interpret, explain and secure a decision-making DLA model. Three scientific domains were used to implement the approach - (i) topological data analysis (TDA) to analyze the sparcity of the input data, i.e representativeness of application space and completeness of scenarios, (ii) abstract interpretation for artificial intelligence (AI^2) to estimate the maximum acceptable perturbation with the required robustness (avoiding all adversarial examples) and, finally, (iii) driving rules in relation with the SOTIF* approach to evaluate the safety of the model's decisions. The approach aims to encompass safety and data-science skills.

The choice of techniques mentioned above to build the process is based on the following reflections:

- The input data of AI models could be presented as a set of n-dimensional cloud points connected by signatures and features. The use of algebraic topology to study this set of input data could be a powerful theory to extract the data features and estimate data coverage. The novelty and the main contribution of this paper is the proposed new metric based on the use of a topological data analysis approach.
- A DLA model is an algorithm and an automatic program with inputs and outputs. Moreover, abstract interpretation is widely used in computer science for debugging, understanding and interpreting computer programs. A recent adaptation of this formal method to verify the robustness of AI models, named AI^2, is used to evaluate robustness in the current work.
- A DLA model is also an autonomous decision system which requires the verification and validation of a set of requirements and specifications. Therefore, driving rules in relation with Safety in a SOTIF perspective are integrated in the proposed approach.
- The application environment, which could be real world or simulated, hosts and encompasses the input data, the model and the safety. The highway-env† simulator is chosen for the current work.

The rest of the paper is organized according to the next seven sections. Section 2 lays the definition of some concepts used in the industrialization process and Section 3 draws up the state of the art. Section 4 details the theoretical formulations of TDA, AI^2 and SOTIF, followed by a presentation of the proposed approach in Section 5. The details of the implementation are given in Section 6. The reporting of the results and the discussion are detailed in Section 7, and finally the conclusion is drawn in Section 8.

*See section 4.3

†See section 6.1

2 Terminology

One of the main issues causing the poor scalability of learning techniques in the real world is the lack of understanding of safety concepts by the research community, especially as regards machine learning. Therefore, safety related terminology should first be defined and clarified. In the current work, we focus on five concepts, which are interpretability, explainability, verification followed by validation and certification. In basic machine learning models (white box), the concept of transparency is fully used to interpret and explain the functionality of the model, which is opposite to black-box models such as deep learning models [5]. The two following subsections define the five concepts stated above, starting with interpretability and explainability followed by verification, validation and certification.

It is important to highlight that the understanding of model decisions might need to use other approaches such as causality and causability [4] to fully interface with humans.

2.1 Interpretability & Explainability

As reported by Arrieta et al. [3], there are notable differences between interpretability and explainability in the context of machine learning decisions. The authors explained that interpretability refers to the passive characteristics of the model, which allow humans to make sense of the decision. On the other hand, explainability refers to the active characteristics detailing the procedure and the functionality of each step (internal function). In other words, we can see interpretability as the intuitive explanation, and explainability as the mathematical and rigorous explanation. The explainability contributes also to the transparency and the traceability. As mentioned in [4], the quality of the explanation can be measured by causability. Both concepts are applied adaptively according to the complexity of the studied learning model.

2.2 Verification, Validation & Certification

Verification involves examining a set of requirements. It ensures compliance with the specifications which allow the validation, in terms of safety, of a product. It is important to highlight that the validation presented in this paragraph is different from the statistical validation commonly used in the machine learning during model training stage. Certification could be referred to as a standard process recognized by the community (nationally or internationally). This process could include the different steps related to the state of the art such as transparency, causality and confidence levels.

3 Related Work

Several approaches have been developed in the literature to evaluate the robustness of DRL summarized by Urban and Miné [6]. Some research studies have demonstrated the sensitivity of Neural Networks against attacks and proposed neural networks verifiers using formal methods. Verifiers can be classified into two categories: complete or incomplete verifiers. Complete methods are exact. They do not generate false positives, but they often have scalability issues like SMT based methods [7,8]. Incomplete verifiers produce false positives but can solve scalability problems using the Abstract interpretation theory [9,10,11]. The three main challenges of this class of methods are finding abstract transformers which are scalable, sound and precise for the different existing activation functions such as Tanh, sigmoid, Relu, etc.

In order to certify or provide evidence of software quality assurance for AI-based systems, for instance the certification of critical software, it will be necessary to increase test scenarios. The proof for ML techniques differs from the approaches proposed in conventional methods. Indeed, for ML-based techniques, there are two kinds of methods used to evaluate the possible cases of coverage:

1. Structural evaluation (in the model itself)
2. Assessment focused on the input domain, i.e the diversity and multiplicity of the input data.

To analyze the coverage in the model, Pei et al.[12] introduced the notion of neuron coverage (i.e., the number of neurons activated by a set of test inputs) to systematically trigger inconsistencies between multiple Deep neural networks (DNNs). Tian et al. [13] used the same neuron coverage metric for guided test generation to identify erroneous behaviors without requiring multiple DNNs. Yu et al. [14] suggested to use coverage metric to improve the accuracy of DNNs by activating more inactivated neurons. Coverage metrics serve as an indicator of the relevance of the decision systems evaluation, which strongly depend on the quality and diversity of the scenarios constituting the training and validation dataset. In the literature, the existing criteria to assess the level of coverage can generally be classified into two types, qualitative and quantitative criteria. The first class of criteria seems to be very dependent on the nature of the scenarios and field of application, and frequently requires expertise and knowledge of the physics quantities and all parameters defining each scenario. The second class is less dependent on the structure of the scenarios. In other words, quantitative metrics are able to estimate the level of coverage regardless of the nature and size of the scenarios. This work takes place within this context.

4 Theoretical Background

We propose a quantitative evaluation of the completeness of a dataset compared to the possible real cases by combining the following tools:

1. Abstract interpretation for artificial intelligence
2. Topological data analysis
3. Driving rules in relation with SOTIF approach

The first two tools are independent from the dimension of the input data which make the coverage metric generic for any DLA model. The focus of this paper is to propose a new metric to measure the completeness of a data set and its representativeness regarding its learned model. The main objective of the approach is to ensure the effectiveness of an AI-based model facing a new situation and its capacity of generalization. The following three subsections provide a brief theoretical state of the art of abstract interpretation, topological data analysis, SOTIF process and its application, and finally an overview of deep reinforcement learning.

4.1 Abstract interpretation

Abstract interpretation, introduced by Cousot and Cousot [15], is a theory which consists in determining, from computer programs, the semantics related to abstract relations in order to demonstrate their stability [16]. It is used for automatic debugging, compiler optimizing, code execution and certification of programs against certain bug classes. Recently, abstract interpretation has been adapted to verify the robustness property of neural network models [17,18,19,20] under the name of

AI^2. To sum up how the robustness problem is formulated and adapted, using the theory of abstract interpretation in the context of deep learning, we introduce and define the overall idea in the next paragraph.

Abstract interpretation is based on a solid and complete approximation of each operation in a given program [21]. The theory of abstract interpretation uses Galois connections between two ordered sets, the approximation set (abstraction function) and the original set (concretization function). As mentioned above, this approach has been adapted for models based on artificial intelligence by designing specific abstract transformers.

The adaption of the approach is applied to a set of perturbations around the original input as introduced by Singh et al. [17]. Indeed, let $\bar{X}$ be a given set. $\bar{X}$ may undergo a deformation or even an attack. To overcome this issue, it is necessary to verify and validate the perturbed input. In such a case, we denote $R_{\bar{X},\varepsilon}$ the set of perturbed inputs around $\bar{x} \in \bar{X}$, with a small constant $\varepsilon > 0$, and we denote C_L the output sets with the same label L representing the robustness given by the following equation.

$$C_L = \{\bar{y} \in \bar{Y} | \arg\max \bar{y}_i = L\}$$

Therefore, using the appropriate abstract transformers, if the set of outputs resulting from the perturbed set $R_{\bar{X},\varepsilon}$ are included in C_L, the robustness property is validated. In other terms, the robustness property is not validated if at least one perturbed input in $R_{\bar{X},\varepsilon}$ has an output $o_{\varepsilon'}$ different from label L ($o_{\varepsilon'} \notin C_L$).

In the state of the art, Singh et al. [17] proposed an approach, called DeepZ analyzer, to deal with the scalability problem of AI^2. DeepZ makes it also possible to certify the robustness of a neural network using precise transformers for different activation functions. It should be mentioned that DeepZ is based on abstract domains and more particularly on zonotopes [22]. To improve the performances of the approach, Gehr et al. [18] introduced another analyzer, called DeepPoly. The latter is based on a novel abstract domain that merges polyhedron with floating points and intervals. This analyzer can automatically prove the robustness of different neural networks architectures, including convolutional neural networks. It is characterized by its high arithmetic precision in floating points and manages several activation functions. This method has been used to verify complex perturbations, including 2D rotation [20,23] and filtering [24,25].

4.2 Topological Data Analysis

TDA is a recently emerged field which aims to understand and exploit the structure of data by extracting topological and geometric characteristics from the manifold to which the discrete data belongs. Holzinger [26] summarizes the theoretical concepts of TDA, and details the state of the art of techniques and domains of TDA implementation, such as text mining and medical domain.

4.2.1 Simplicial Homology

TDA combines algebraic topology and other tools of pure mathematics to enable a rigorous mathematical study of "form". The main tool is called persistent homology [26], an adaptation of homology to point cloud data. Persistent homology has been applied to many types of data in various fields. In addition, its mathematical basis also has theoretical importance and is based on the notion of simplicial complexes. A simplicial complex is a set made up of points, line segments, triangles and their $n-$dimensional equivalents [27]. Simplicial complexes should not be confused with the more abstract notion of a simplicial set appearing in modern theory of simplistic homotopy. We can generate a simplicial complex by connecting the nearest points according to a metric, called an abstract simplicial complex (see Fig. 1). This is a way of representing data in the form of a graph, in order to facilitate the processing of large data. The transition from a point cloud to a simplicial complex is called filtration. This corresponds to the first stage of the TDA pipeline. Persistent homology enables us to deduce signatures and features capable of describing the manifold.

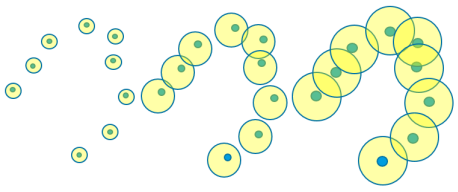


Fig. 1: The ε parameter defines the radius of the ball around each point

4.2.2 Filtration and feature extraction

The objective of TDA is to extract invariant and relevant characteristics capable of characterizing and understanding information included in data. Topological primitivities are invariant with respect to the origin of

the coordinate system, and with respect to plane similarity deformations and compression. These three criteria make the topological primitives very important. Several types of diagrams have been proposed to represent these features: persistence diagram, barcode diagram and Betti numbers. The persistent diagram is capable of coding the evolution of data topology through different scale factors and is based on another diagram called "bar code" as illustrated in Fig 2. The Betti numbers refer to the number of holes in the topology.

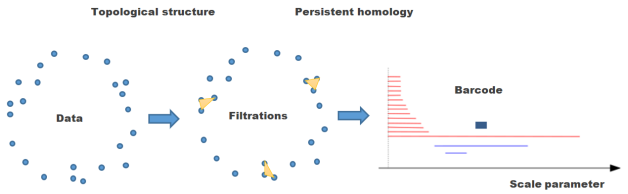


Fig. 2: "Pipeline" for the study of simplicial complexes

There are several ways to build simplicial complexes such as *witness filtration* or *Rips Filtration*. The *Geometry score* [28] is a metric that uses the results of persistent homology and more specifically persistence barcodes. By looking at the statistics on its components, we can determine for how long each lived during the period of the topological study: by this we mean that each number of holes is weighted by the maximum value of ε, which gives the relative value lifetime of this number of holes. This is a confidence measure since a number of holes that lasts for a while (in terms of ε evolution) indicates that this number reflects the shape of the data variety. Its computation requires repeating the experiment several times, and the values obtained for each relative lifetime are averaged, giving what we call their $Mean\ Relative\ Living\ Time$, or $MRLT$. For two datasets X and Z, the geometry score is then calculated using the quadratic error between the two $MRLTs$ of each distribution. Then, the lower the geometry score, the more it reflects the proximity of the manifold. Indeed, this would mean that the number of holes does not decrease. In short, this metric and the TDA make it possible to evaluate deep learning models in terms of coverage and model collapse, regardless of the size and dimension of the data used.

4.2.3 Abstract simplicial complex

Depending on the value of the filtration function and a set of vertices (point cloud), several abstract simplicial complexes could be built. For example, the Čech simplicial complex is built by the non-empty set of the intersection of ε-balls around its vertices. Furthermore, using the Nerve theorem, the Čech simplicial complex provides a topologically correct reconstruction of the topological space [29]. In other words, two connected vertices in the Čech simplicial complex guarantees that the whole topological range is covered between the two vertices. This main property is explored in the current work to build the covering metric of the datasets used in DLA. It is important to mention that this property is independent of the data and their dimension, i.e. the theory could be applied on n-dimensional ($n \in \mathbb{N}$) data.

4.3 Safety of the Intended Function process

The Safety of the Intended Functionality (SOTIF) is a notion defined in a dedicated ISO Standard (ISO/DIS 21448). It offers a methodology to grant the absence of unreasonable risk due to hazards resulting from the functional insufficiency of the intended functionality or its implementation for road vehicles. It is an additional approach to classical functional safety approach which considers the system safety without failures. The first step of the SOTIF process consists in an analytic approach to identify functional insufficiency considered as a triggering condition in accident scenarios. This first step is not taken into account in the present paper. The second step is an evaluation of known scenarios regarding hazard. The third step of the SOTIF process consists in evaluating unknown hazardous scenarios around known ones in order to improve system robustness.

To illustrate the second and third step, we consider that there are a known and an unknown area. The known area includes scenarios where system behavior is safe and the unknown area includes scenarios where system behavior leads to potential harm (safe and unknown scenarios are not useful for safety purposes) (see Fig 3). In reality, as illustrated in Fig. 4, these areas overlap.

The objective of automotive development is to reduce known and potentially dangerous (area 2) and unknown (area 3) behaviors at an acceptable level of residual risk such as illustrated in Fig. 5. The following development objectives can be inferred from the given scope:

- Scope 1: Maximize safe function/system conduct.
- Scope 2: Reduce the known risk behavior.
- Scope 3: Minimize the unknown area.

According to a set of specifications and requirements, this article gives methodologies to improve system behavior and evaluate it in known scenarios and to evaluate unknown scenarios around known ones. These method-

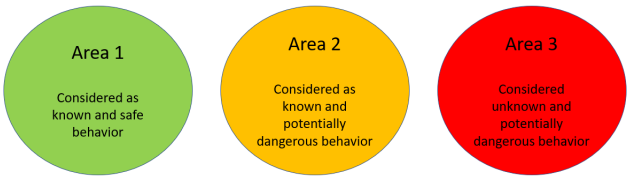


Fig. 3: Venn's Diagram and definition for Possible Function/System Behavior: separate areas view

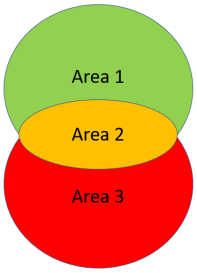


Fig. 4: Venn's Diagram and definition for Possible Function/System Behavior: overlapped areas view

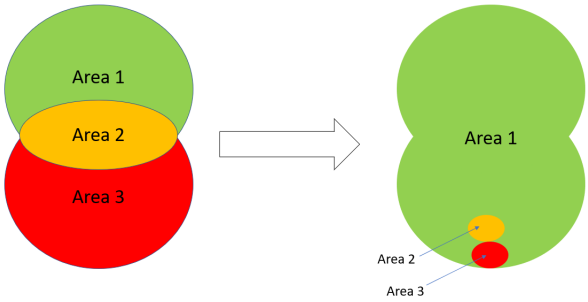


Fig. 5: Venn's Diagram and definition for Possible Function/System Behavior: overlapped areas view

ologies help improve systems robustness regarding SOTIF approach.

4.4 Operational Metrics

In order to evaluate the ability of the system to drive safely in a SOTIF point of view, the three following operational metrics have been taken into account; (i) Time Inter Vehicles (TIV), (ii) Time To Collision (TTC) and (iii) Braking Time. The following subsections further detail these metrics.

Let C_1 and C_2 be two cars where C_1 is behind C_2 as illustrated in Fig. 6. We define the three functions $P(x)$, $V(x)$ and $A(x)$, where $x \in \{C_1, C_2\}$, to describe the position, the velocity and the acceleration/deceleration of each car, respectively.

Fig. 6: Disposition of cars used to compute TIV, TTC and Braking Time

4.4.1 Time Inter Vehicles (TIV)

When a human driver follows another car, they keep a safe distance between their car and the car in front of them. In the rules of the road, this distance allows drivers to react in case of an emergency; this takes into account the reaction time of the driver and the braking distance of the car. Both of these notions depend on the speed of the car. That is why the safe distance between the two cars C_1 and C_2 moving at close speeds, is defined as a TIV by the following definition:

$$TIV = \frac{|P(C_2) - P(C_1)|}{V(C_1)} = \frac{\text{Distance between cars}}{\text{Speed of the following car}}$$

For human-driven cars, the rules of the road recommend 2 seconds of TIV for safe driving.

4.4.2 Time To Collision (TTC)

When C_1 is faster than C_2, then the distance between the two cars decreases. When the speed gap is important, TIV is not sufficient to manage the safe distance. In such a situation, the TTC is used to compute the time to collision.

Therefore, TTC enables us to know how much time car C_1 has to react before the collision with C_2 by braking or changing trajectories.

The TTC is defined by the following formula:

$$TTC = \frac{|P(C_2) - P(C_1)|}{V(C_1) - V(C_2)} = \frac{\text{Distance between cars}}{\text{Speed difference between the two cars}}$$

4.4.3 Braking Time

In addition to the two previous notions and especially with TTC, it is important to take into account braking time. This is the time the car takes to reach the speed of the slower car in front of it. Braking Time is defined by the following formula (considering a constant average deceleration value for C_1):

$$\text{Braking Time} = \frac{V(C_2) - V(C1)}{A(C_1)} = \frac{\text{Speed difference}}{\text{Average braking deceleration}}$$

We can consider that when the Braking Time is higher than the TTC, there is no more safety margin to prevent a crash from occurring.

4.5 Deep Reinforcement Learning

Reinforcement Learning (RL) consists in an agent which learns to perform a task by maximizing cumulative discounted rewards [30]. The agent acts by sequentially choosing actions from observations over a sequence of time steps. The problem is modeled as a Markov Decision Process (MDP) [31] expressed by (S, A, P, R), where S is the observation space, A is the action space, $P : S \times A \times S \rightarrow [0; 1]$ $(P(s_{t+1}|s_t, a_t))$ is the transition probability and $R : S \times A \times S \rightarrow \mathbb{R}$ $(R(s_t, a_t, s_{t+1}))$ is the reward function.

RL algorithms attempt to learn a policy $\pi_\theta : S \times A \rightarrow [0; 1]$ which calculates the probability $\pi_\theta(a_t|s_t)$ of selecting the action a_t given the observation s_t, where θ denotes the parameters for the policy π. The goal is to maximize the expected cumulative discounted return $\mathbb{E}_{\tau\sim\pi_\theta} \sum_{t=1}^{\infty} \gamma^t r_t$ with r_t the reward obtained at step t in episode τ and $\gamma \in]0, 1[$ being the discount factor. The policy π_θ is learned from sequential experiences in the form of transition tuples $(s_t, a_t, r_{t+1}, s_{t+1})$. Deep Reinforcement Learning (DRL) uses Neural Networks to learn $\pi_\theta(s_t)$.

5 Proposed approach

The proposed approach in this work deals with the explainability of machine learning model decisions. As mentioned above, the explainability of black-box models using conventional approaches is intricate. In fact, it is difficult to define a separate method dealing only with transparency or causality for example. Therefore, we propose a new approach from a different angle where the black-box-ness of the model will not be an issue for its interpretability and explainability. This approach uses the three domains presented in the last section, which are (i) AI^2, (ii) TDA and (iii) the SOTIF process. On the basis of these techniques, we developed a new metric called "Covering metric" where for a given model and dataset test, local and global covering evaluations are proposed.

5.1 Common Approach

The workflow of the approach is divided into 3 main steps:

1. Estimating the minimum disturbance tolerated around each sample using abstract interpretation AI^2. This ensures that all cases varying from the original to the minimum disturbance do not contain any adversarial examples.
2. Calculating the persistence diagram on the cloud dataset using TDA. With this technique, we deduce the minimum radius that guarantees coverage of the cloud dataset and completely fills all Betti numbers.
3. Estimating the ratio between the overall volume and the volume occupied by the radius balls generated with the Abstract Interpretation. We define two ratios: global for the ratio between AI^2 and TDA radius, and local for the ratio between the number of Betti numbers filled by AI^2 and the total Betti numbers detected by TDA.

The intuition behind the approach is to identify when we can trust or doubt the decision of the model. Fig. 7 illustrates the safe and the unsafe areas by red and black circles, respectively. The ideal situation is the case where the red and the black circles are covering the same area, as illustrated by the two examples in Fig. 8. Sub-figures (a) and (b) of Fig. 8 show the possible perspectives to improve the safe area of AI^2 through model precision improvement and/or database enrichment.

To quantify the safe area, a process building a covering metric is proposed. The workflow presented in Fig. 9 illustrates the three steps for a typical classification model. From the figure, the right side (AI^2) computes the safe scenarios which represent the *known-known*‡ (real) and *unknown-known*§ (inside AI^2 circle in Fig. 7) scenarios and the left side computes the spots and holes in the dataset used for the evaluation. They are represented by the *unknown-unknown*¶ (holes created by AI^2 covering) and *known-unknown*‖ (real scenarios where the model failed) scenarios.

5.2 Metric evaluation

As presented in the last section, the coverage metric is based on TDA and AI^2. From the dataset, a simplicial

‡scenarios evaluated by the model and considered as safe

§Scenarios not evaluated by the model but considered as safe by AI^2 process

¶Scenarios not evaluated by the model and its decision could be unsafe

‖Scenarios evaluated by the model and considered as unsafe

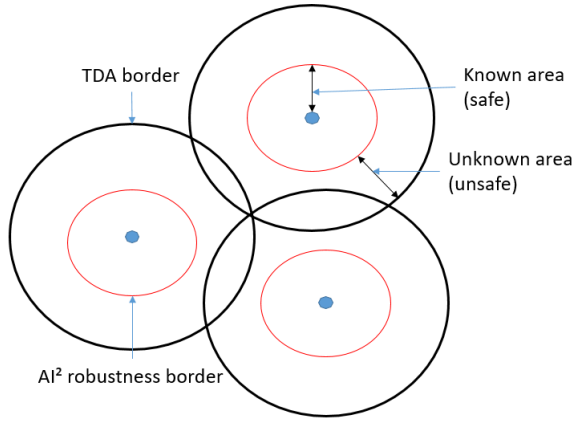
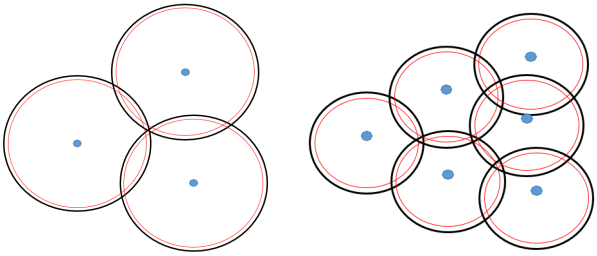


Fig. 7: Safe and unsafe areas illustrated by AI^2 and TDA techniques



(a) improvement of safe area by model precision (b) improvement of safe area by data increase

Fig. 8: Illustration of the improvement of safe areas exploring two aspects

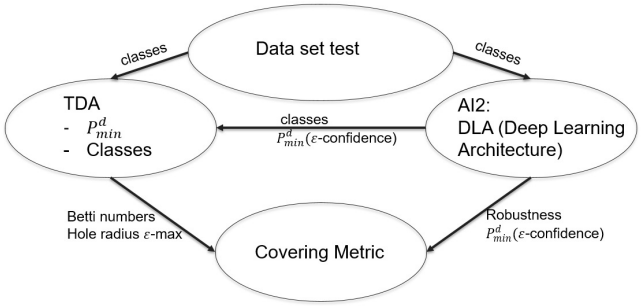


Fig. 9: Workflow of coverage metric building: from the test dataset, AI^2 and TDA

complex using Čech filtration is built to extract the minimum radius covering all Betti numbers. Using AI^2, we compute the maximum perturbation respecting the safety requirements as follows.

Let E be the set of $x \in \mathbb{R}^M$ and *net* define the model to evaluate. We define $K(f)$ the Čech simplicial complex from the cloud points of E under the filtration f, and $\{B_n : K \to \mathbb{N}\}$ the function which computes the sum of all Betti numbers.

– The topology of the data is given by the following equation:

$$\exists \varepsilon_{min}, \forall\ \varepsilon > 0 \text{ where },$$
$$\{B_n(\varepsilon_{min}) = 0 \text{ and } B_n(\varepsilon_{min} - \varepsilon) > 0\}$$

We denote the obtained results (ε_{min}) by ε^M_{TDA}.

– Let $\mathcal{R}$ be the robustness function evaluating the model *net*, and η the robustness required:

$$\exists \varepsilon_{max}, \mathcal{R}(net(\varepsilon_{max})) < \eta$$

where ε_{max} represents the perturbation applied on the input data of the model *net*. We denote the obtained results (ε_{max}) by ε^M_{AI2}.

– The coverage metric then is given by:

$$\rho = \frac{\alpha + \beta}{2}$$

where,

$$\alpha = \frac{\varepsilon^M_{AI2}}{\varepsilon^M_{TDA}},$$
$$\beta = \frac{B_n(0) - B_n(\varepsilon_{AI2}) + 1}{B_n(0) + 1}$$

The metric ρ is divided into two parts α and β presenting local and global coverage, respectively. In fact, α represents the sparsity of the data, and β represents the filled and unfilled areas (holes) in the data. It is important to mention that the dimension of the hole is not considered in the current metric.

5.3 DRL adaption

The workflow presented in Fig. 9 illustrates a classification model that contains a labeled test dataset as ground truth. However, for deep reinforcement learning where there is no ground truth, an adaption of the workflow is required. In fact, in deep reinforcement learning, the agent learns from its environment without ground truth to verify the model decision compared to the classification task.

To ensure the verification of DRL models, we analyzed its three outputs which are actions, rewards and policies. By introducing some safety requirements, model actions are the most relevant for the verification of the decision [32]. In fact, only the actions output are linked to each scenario which could be independent from other scenarios.

In DRL, for a given scenario, the action taken is not binary, i.e. two or more actions could be relevant for the same scenario. However, only binary verification could be established: critical and non-critical actions in terms of safety. Therefore, two classes are performed as follows:

– Critical Actions class defines the set of actions that puts the agent in a dangerous situation. For example, for the autonomous car, the action "accelerate" where the car in front is breaking is considered as dangerous, then critical.
– Tolerated actions class defines a set of considered safe actions for the agent. We consider that the optimal action (with highest reward) belongs to the tolerated actions class.

The implementation and the application of these classes are handled for an autonomous car in Highway environment (see section 6.1). The workflow depicted in Fig. 10 illustrates the adaptation of the common workflow for DRL models. The AI^2, implemented by ERAN** and ELINA†† libraries, is also adapted for DRL architecture. Gudhi‡‡ library is used for TDA computation. The next section (Section 6) details the different steps of the implementation.

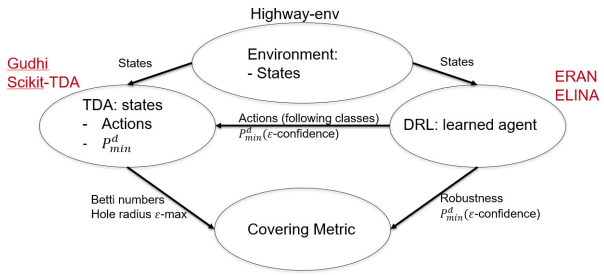


Fig. 10: Workflow of coverage metric building adapted for DRL

6 Implementation

6.1 Simulator environment

Highway-v0 [33] is a 2D open-source autonomous car driving simulation environment. In this environment, the agent drives a car on an infinite four lanes unidirectional highway. The vehicle, piloted by the agent (the ego-vehicle), is inserted in the traffic flow of other vehicles (the exo-vehicles). All exo-vehicles follow a basic driving algorithm. The goal of the agent is to drive as fast as possible without having an accident.

The configuration implemented in the environment to acquire the states of the model represented by the tuple (S, A, P, R) is illustrated by Fig. 11 and formulated by the vector s given below:

$$s = [\ y_{ego}, v_{ego}, \\ x_{bl}, \ v_{bl}, \ x_{fl}, v_{fl}, \\ x_b, \ v_b, \ x_f, \ v_f, \\ x_{br}, \ v_{br}, \ x_{fr}, v_{fr}\ ] \tag{1}$$

Where y_{ego} is the transverse position of the agent in the width of the road, v_{ego} is the velocity of the agent, the x_s are the positions of the exo-vehicles relative to the agent in the longitudinal direction of the road. v_s are the velocities of the exo-vehicles relatively to the agent velocity. bl, fl, b, f, br, and fr represent the closest exo-vehicles to the agents, in the back-left, front-left, back, front, back-right, and front-right positions, respectively. At each time step the agent observes only three lanes, its lane and the two adjacent lanes, left and right, and observes two exo-vehicles on each lane.

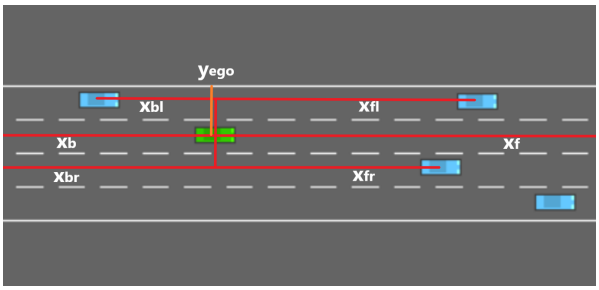


Fig. 11: State representation (without velocities) in Highway-v0

The action of the agent $\{a_t \in A\}$ is a discrete choice between five possibilities:

0: Turn Left : change lane to the left
1: Nothing: stay on the same lane, at the same velocity
2: Turn Right: change lane to the right
3: Accelerate ($+5m \cdot s^{-1}$)
4: Decelerate ($-5m \cdot s^{-1}$)

The episode ends when the agent has a collision with another vehicle. The goal of the agent is to drive as long as possible without having a collision, and as fast as possible to get the maximum reward at each time step.

6.2 Operational reward function

The basic reward function implemented in Highway-env rewards the agent when it reaches a high speed by avoiding collisions. The equation (2) illustrates the behavior agent reward.

$$R_v = \max\left(-1, \frac{V_{ego} - \frac{\sum_{i=0}^{n} V_{exo_i}}{n}}{V_{max} - \frac{\sum_{i=0}^{n} V_{exo_i}}{n}}\right) \tag{2}$$

**https://github.com/eth-sri/eran
††https://github.com/eth-sri/ELINA
‡‡https://gudhi.inria.fr/

However, with only the speed-based reward function, the agent decreases its performances, in terms of the number of collisions and episodes duration, by increasing traffic density in the environment. Furthermore, the driving behavior of the agent does not respect the operational metrics presented in Section 5.2, that could be considered as dangerous and risky. Thus, to improve the performances of the agent, the operational metrics are integrated in the reward function as presented in equation (3), where the agent takes into account, at the same time, the high speed and the operational metrics.

$$R = \min\left(R_v, R_o\right) \tag{3}$$

R_v being the velocity reward defined in the equation (2) above and R_o being the operational reward given by the following equation (4)

$$R_o = \begin{cases} \min(r_f, r_{ft}, r_{bt}) & \text{if change lane} \\ r_{fb} & \text{else,} \end{cases} \tag{4}$$

where r_f, r_{ft}, r_{bt} and r_{fb} being functions of TTC and TIV defining the risk from exo-cars in front same lane, front target lane, before target lane and braking time in same lane of ego-car, respectively.

The application of this new reward function induces a significant improvement in the agent behavior, for both the duration of scenarios without accident and the visual behavior in a simulation environment. During iterative engineering processes to specify reward function, an analytical approach gives more interpretability to agent behavior. Using a simulation environment to replay accidents, it was easier to understand which part of the reward function contains a mistake or needs further improvement.

6.3 Construction of critical and tolerated classes

In the simulator, as mentioned above, the agent could choose one of the five actions for each scenario. To reduce the time computation of the TDA, the covering of each action was evaluated separately, which drastically reduces the dimension of the data. Indeed, for *Turn Right* action, only the data (relative position and velocity) of exo-vehicles in the right side of the agent and its own velocity are considered, as illustrated in Fig. 12. A scenario is considered as *Turn Right* critical class if the requirement of TTC and TIV are not respected. All other scenarios which are not critical are considered as tolerated.

Separating actions allows a more accurate description of the learned agent behavior and focus on its failures. It permits to point out the spot where the model is not sufficient and/or the sparsity of the evaluated dataset.

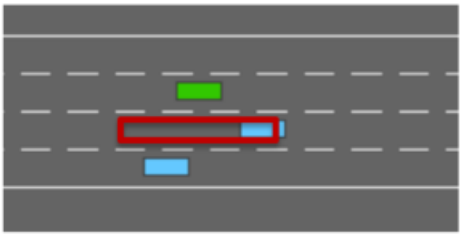


Fig. 12: Data considered in *Turn Right* action

6.4 DRL Model & Dataset

For the experimentation, the Proximal Policy Optimization (PPO) algorithm, which is a recent and simple policy gradient method, is used [34]. The agent is trained in the highway-v0 environment. It is important to highlight that any other RL approach could be considered.

The generation of the dataset used in the current work is done by running the learned agent in the environment over $10,000$ time steps. At each time step, we store the observation generated by the environment, the action taken by the agent giving this observation, and the result obtained after the execution of the action in the environment (if the agent has a collision or not).

Table 1 summarizes the number of critical scenarios for each class used to compute the covering metric. Action 4 (*Decelerate*) is not evaluated in the current work due to the lack of data in critical situations for this action.

action 0	action 1	action 2	action 3	action 4
17,306	207	18466	207	0

Table 1: Critical set scenarios for each action

6.5 AI^2 and TDA implementation

AI^2 implementation follows two algorithms, perturbation then robustness estimation. The perturbation of the data follows the instruction given in Algorithm 1 (The organization of each data vector is detailed by equation (1)). Then, the perturbed data is evaluated following the steps of Algorithm 2.

For each dataset action, TDA optimal radius is computed. The optimal radius is the value of the filtration which allows to connect all the cloud points from the dataset.

Algorithm 1: Perturbation

Data: p (perturbation value), critical_class.
Result: epsilon
if *critical_class* == *LANE_LEFT* **then**
  $epsilon = [\ y_{ego},\ v_{ego},\ x_{bl}+p,\ v_{bl}+p,\ x_{fl}+p,\ v_{fl}+p,\ x_b,\ v_b,\ x_f,\ v_f,\ x_{br},\ v_{br},\ x_{fr},\ v_{fr}\ ]$
if *critical_class* == *IDLE* **then**
  $epsilon = [\ y_{ego}, v_{ego},\ x_{bl},\ v_{bl},\ x_{fl},\ v_{fl},\ x_b,\ v_b,\ x_f+p, v_f+p,\ x_{br},\ v_{br},\ x_{fr},\ v_{fr}\ ]$
if *critical_class* == *LANE_RIGHT* **then**
  $epsilon = [\ y_{ego},\ v_{ego},\ x_{bl},\ v_{bl},\ x_{fl},\ v_{fl},\ x_b,\ v_b,\ x_f,\ v_f,\ x_{br}+p, v_{br}+p, x_{fr}+p, v_{fr}+p\ ]$
if *critical_class* == *FASTER* **then**
  $epsilon = [\ y_{ego}, v_{ego},\ x_{bl},\ v_{bl},\ x_{fl},\ v_{fl},\ x_b,\ v_b,\ x_f+p, v_f+p,\ x_{br},\ v_{br},\ x_{fr},\ v_{fr}\ ]$
if *critical_class* == *SLOWER* **then**
  $epsilon = [\ y_{ego},\ v_{ego},\ x_{bl},\ v_{bl},\ x_{fl}, v_{fl},\ x_b+p, v_b+p, x_f,\ v_f,\ x_{br},\ v_{br},\ x_{fr}, v_{fr}\ ]$

Algorithm 2: Robustness estimation

Data: class_list, domain, data.
Result: Robustness
foreach *critical_class* ∈ *class_list* **do**
  dataset = extractCriticalData(data, critical_class);
  foreach *perturbation_intensity* ∈ $[0, 0.5]$ **do**
    foreach *sample* ∈ *dataset* **do**
      epsilon = perturbation(perturbation_intensity, critical_class);
      specLB = sample + epsilon;
      specUB = sample - epsilon;
      perturbed_label = eranAnalyzer(specLB, specUB, domain)
      if *perturbed_label* == *critical_class* **then**
        failed_sample = failed_sample + 1;
      else
        verified_sample = verified_sample + 1 ;
    robustness[critical_class][perturbation_intensity] = verified_sample/length(dataset);

7 Results and Discussion

To illustrate the link between actions, the interpretation of the results were focused on three actions that could depend on the same set of data. In fact, in safe driving the three actions *accelerate*, *nothing* and *decelerate* should only depend on ego-vehicle velocity and the position and velocity of the vehicle in front. Figures 13 and 14 illustrate scenarios with *accelerate*, *decelerate* and *accelerate, nothing, decelerate* actions, respectively. As we can see, there is a clear separation between the two actions *accelerate* and *decelerate*, the action *nothing* is located between *accelerate* and *decelerate* actions, which is easily interpretable for the considered variables. On the other hand, we can visualize the tolerated action in Fig. 14 where the same scenario (point cloud) could belong to *nothing*, *accelerate* and *decelerate* at the same time. However, other scenarios could be critical such as *accelerate* instead of *decelerate*.

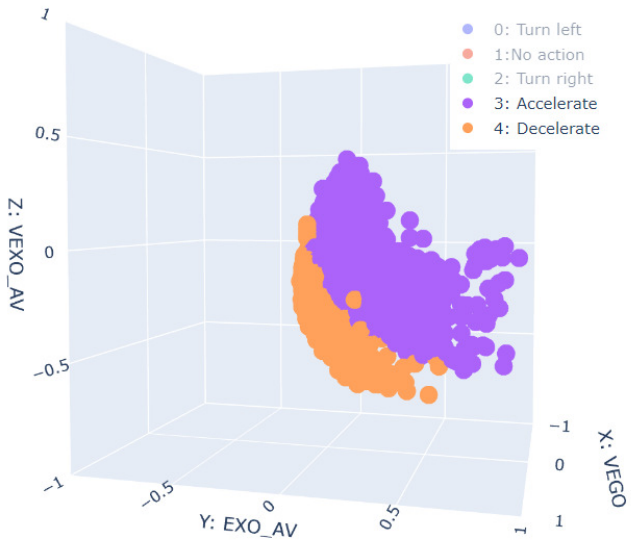


Fig. 13: Results of data for the two actions : accelerate and decelerate. VEGO defines Ego-vehicle Velocity, VEXO_AV and EXO_AV define the velocity and the position of the Exo-vehicle in front of the Ego-vehicle, respectively

As mentioned above, the velocity of vehicles is not continuous ($\pm 5m \cdot s^{-1}$). Fig. 15 and 16 illustrate this behavior where we can see that the velocity is like a staircase. This discontinuity of the velocity will affect the covering metric.

The AI^2 results are presented in Table 2. We can see from the results that the agent is less efficient for

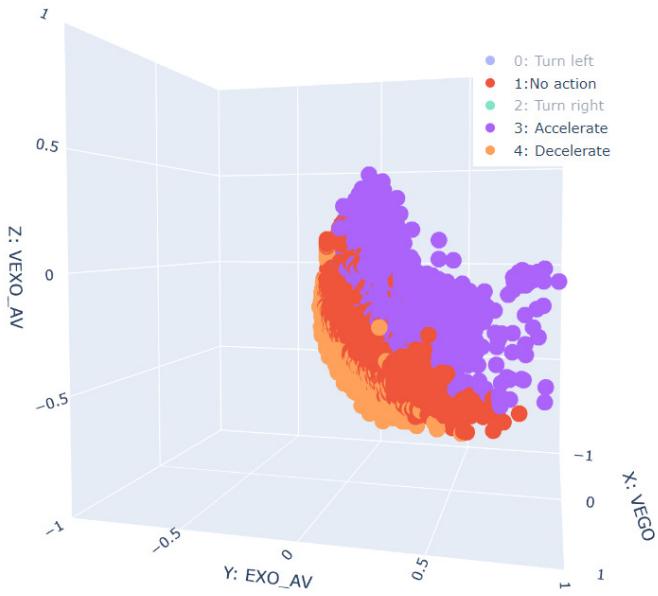


Fig. 14: Results of data for the three actions : accelerate, nothing and decelerate. VEGO defines Ego-vehicle Velocity, VEXO_AV and EXO_AV define the velocity and the position of the Exo-vehicle in front of the Ego-vehicle, respectively

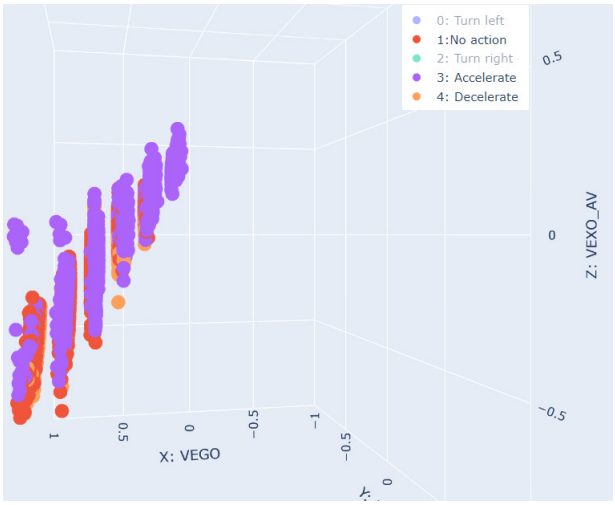


Fig. 15: Results of data for the three actions: accelerate, nothing and decelerate. VEGO defines Ego-vehicle Velocity, VEXO_AV and EXO_AV define the velocity and the position of the Exo-vehicle in front of the Ego-vehicle, respectively

the action *Nothing*, which is interpretable, as the higher the speed of the agent, the more it is rewarded. On the other hand, we can see that the agent is more robust for the two actions *Turn right* and *Accelerate*. However,

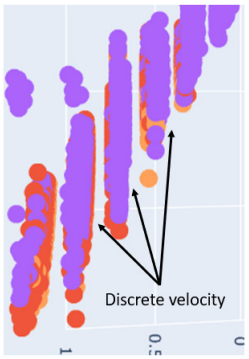


Fig. 16: A zoom of the velocity values obtained from the simulator. An illustration of the non-continuity of the speed of cars.

Decelerate action is not evaluated in the current dataset due to its empty critical class.

ε	action 0	action 1	action 2	action 3
0.005	97.3%	95.1%	99.1%	98%
0.01	94.8%	92.7%	98.2%	97.5%
0.015	92.5%	88.8%	97.3%	97.5%
0.02	90.6%	83.5%	96.9%	96.6%
0.025	89, 3%	83%	96.7%	95.5%

Table 2: AI^2 results for the 4 actions which are: action 0 for Turn left, action 1 for Nothing, action 2 for Turn right and action 3 for Accelerate

	Covering metric for a robustness of 95%
action 0	2%
action 1	4%
action 2	29%
action 3	56%

Table 3: Results of covering metric for the four actions by respecting 95% of robustness as a safety requirement

To evaluate the coverage of the dataset, for each action we consider that the minimum robustness should exceed 95%, then actions *Turn left* and *Nothing* could accept only a disturbance of 0.005, which is equivalent to $0.63m \cdot s^{-1}$ for the velocity and $0.875m$ for the position, and actions *Turn right* and *Accelerate* could tolerate a disturbance of 0.025, which is equivalent to $3.15m \cdot s^{-1}$ for the velocity and $4.375m$ for the position. Table 3 reports the covering metric for each action. As we can see, the cover is higher for the actions *Turn right* and *Accelerate* by 29% and 56% of data coverage

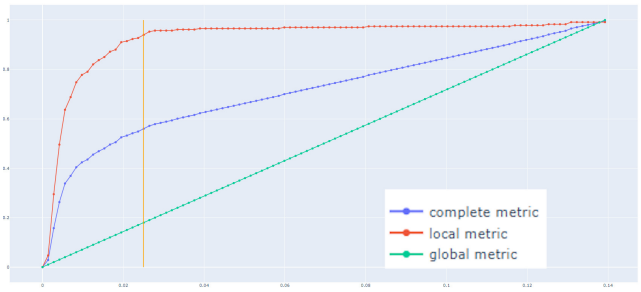


Fig. 17: Example of the evolution of the covering metric. The x axis presents the radius (filtration function), the y axis presents the covering; the yellow vertical bar presents the covering taking into account the requirement of 95% robustness

and only 2% and 4% of data coverage for *Turn left* and *Nothing* actions.

Fig. 17 illustrates the covering metric evolution by increasing the TDA radius. From the figure, it can been seen that for a radius value greater than 0.03, the improvement of the local metric is very low, which presents the heterogeneity of the dataset due to the velocity, i.e there are holes (Betti numbers) needing a large radius to be covered. These areas represent the discontinuity of the velocity illustrated in Fig. 15. In terms of safety, they are harmful zones and can be considered as unknown-unknown scenarios.

As mentioned above concerning the resilient application of the proposed approach, the equivalent workflow could be adapted for more data such as trajectory evaluation [35]. Furthermore, the approach can be useful for data privacy [2,36] to identify the safe and unsafe regions.

8 Conclusion

This work proposes a covering metric for datasets used in machine learning models. The metric is based on the topology of the data and the robustness of the model. In other terms, the proposed approach connects two mathematical domains to evaluate simultaneously the robustness of the model and the coverage of the dataset, by using topological data analysis and Abstract Interpretation methods. Additionally, a set of safety requirements and specifications are integrated to better interpret the covered and uncovered situations. An implementation of the metric, taking into account some safety concepts, is also presented. The approach is applied on an autonomous driving simulator and deep reinforcement learning as decision-making model.

The obtained results highlighted a validation methodology by characterizing the safe and the unsafe decision of machine learning models. As a perspective, an application of the proposed metric on the latent space of the data generated by GAN (Generative Adversarial Networks) models could achieve a connection between the real and synthetic generated data [37], which facilitates the establishment of sensitive data, such as medical or defense data. Another application of the proposed work could be used for recurrent networks [38].

Acknowledgements This research work has been carried out in the framework of IRT SystemX, Paris-Saclay, France, and therefore granted with public funds within the scope of the French Investments for the future program (*Programme Investissements d'Avenir "PIA"*). This work is a part of the EPI project (EPI for *Evaluation des Performances de l'Intelligence artificielle* - AI-based Performance Evaluation of Decision Systems). The project is supervised by IRT systemX and its partners, Apsys, Expleo France, Naval Group and Stellantis.

We specially wish to thank Gwenaëlle Berthier for her project management and team leadership and Rosemary MacGillivray for her English proofreading.

Declarations

Conflict of interest The authors have no competing relevant interest to declare about the content of this article.

References

1. I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, arXiv preprint arXiv:1412.6572 (2014)
2. X. Liu, L. Xie, Y. Wang, J. Zou, J. Xiong, Z. Ying, A.V. Vasilakos, Privacy and security issues in deep learning: A survey, IEEE Access **9**, 4566 (2020)
3. A.B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, R. Chatila, H. Franscisco, Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai, Information Fusion **58**, 82 (2020)
4. A. Holzinger, B. Malle, A. Saranti, B. Pfeifer, Towards multi-modal causability with graph neural networks enabling information fusion for explainable ai, Information Fusion **71**, 28 (2021)
5. Z.C. Lipton, The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery., Queue **16**(3), 31 (2018)
6. C. Urban, A. Miné, A review of formal methods applied to machine learning, arXiv preprint arXiv:2104.02466 (2021)
7. G. Katz, C. Barrett, D.L. Dill, K. Julian, M.J. Kochenderfer, Reluplex: An efficient smt solver for verifying deep neural networks, in *International conference on computer aided verification* (Springer, 2017), pp. 97–117

8. M. Sotoudeh, A. Thakur, Correcting deep neural networks with small, generalizing patches, in *Workshop on Safety and Robustness in Decision Making* (2019)
9. M. Naseer, M.F. Minhas, F. Khalid, M.A. Hanif, O. Hasan, M. Shafique, Fannet: formal analysis of noise tolerance, training bias and input sensitivity in neural networks, in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)* (IEEE, 2020), pp. 666–669
10. J. Li, J. Liu, P. Yang, L. Chen, X. Huang, L. Zhang, Analyzing deep neural networks with symbolic propagation: Towards higher precision and faster verification, in *International Static Analysis Symposium* (Springer, 2019), pp. 296–319
11. X. Huang, M. Kwiatkowska, S. Wang, M. Wu, Safety verification of deep neural networks, in *International Conference on Computer Aided Verification* (Springer, 2017), pp. 3–29
12. K. Pei, Y. Cao, J. Yang, S. Jana, Deepxplore: Automated whitebox testing of deep learning systems, in *proceedings of the 26th Symposium on Operating Systems Principles* (ACM, 2017), pp. 1–18
13. Y. Tian, K. Pei, S. Jana, B. Ray, Deeptest: Automated testing of deep-neural-network-driven autonomous cars, in *Proceedings of the 40th international conference on software engineering* (2018), pp. 303–314
14. J. Yu, Y. Fu, Y. Zheng, Z. Wang, X. Ye, Test4deep: an effective white-box testing for deep neural networks, in *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)* (IEEE, 2019), pp. 16–23
15. P. Cousot, R. Cousot, Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints, in *Proceedings of the 4th ACM SIGACT-SIGPLAN symposium on Principles of programming languages* (ACM, 1977), pp. 238–252
16. P. Cousot, R. Cousot, Abstract interpretation and application to logic programs, The Journal of Logic Programming **13**(2-3), 103 (1992)
17. G. Singh, T. Gehr, M. Mirman, M. Püschel, M. Vechev, Fast and effective robustness certification, in *Advances in Neural Information Processing Systems* (2018), pp. 10,825–10,836
18. T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, M. Vechev, Ai2: Safety and robustness certification of neural networks with abstract interpretation, in *2018 IEEE Symposium on Security and Privacy (SP)* (IEEE, 2018), pp. 3–18
19. G. Singh, T. Gehr, M. Püschel, M.T. Vechev, Boosting robustness certification of neural networks., in *ICLR (Poster)* (2019)
20. G. Singh, T. Gehr, M. Püschel, M. Vechev, An abstract domain for certifying neural networks, Proceedings of the ACM on Programming Languages **3**(POPL), 41 (2019)
21. B. Blanchet, Introduction to abstract interpretation, lecture script (2002)
22. K. Ghorbal, E. Goubault, S. Putot, The zonotope abstract domain taylor1+, in *International Conference on Computer Aided Verification* (Springer, 2009), pp. 627–633
23. R. Khalsi, M. Mziou-Sallami, I. Smati, F. Ghorbel, Contourverifier: a novel system for the robustness evaluation of deepcontour classifiers., in *Proceedings of the 14th International Conference on Agents and Artificial Intelligence*, vol. 2 (2022), vol. 2
24. M. Mziou-Sallami, M.I. Khedher, A. Trabelsi, S. Kerboua-Benlarbi, D. Bettebghor, Safety and robustness of deep neural networks object recognition under generic attacks, in *International Conference on Neural Information Processing* (Springer, 2019), pp. 274–286
25. M. Mziou-Sallami., F. Adjed., Towards a certification of deep image classifiers against convolutional attacks, in *Proceedings of the 14th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART,*. INSTICC (SciTePress, 2022), pp. 419–428. DOI 10.5220/0010870400003116
26. A. Holzinger, in *Interactive knowledge discovery and data mining in biomedical informatics* (Springer, 2014), pp. 331–356
27. H. Edelsbrunner, J. Harer, *Computational topology: an introduction* (American Mathematical Soc., 2010)
28. V. Khrulkov, I.V. Oseledets, Geometry score: A method for comparing generative adversarial networks, CoRR **abs/1802.02664** (2018)
29. C. Maria, Algorithms and data structures in computational topology. Ph.D. thesis, Université Nice Sophia Antipolis (2014)
30. R.S. Sutton, A.G. Barto, et al., *Introduction to reinforcement learning*, vol. 135 (MIT press Cambridge, 1998)
31. R. Bellman, A markovian decision process, Journal of mathematics and mechanics **6**(5), 679 (1957)
32. F. Adjed, F. Pelliccia, M. Rezzoug, L. Schott, Certification of deep reinforcement learning with multiple outputs using abstract interpretation and safety critical systems, in *Proceedings of the 31st European Safety and Reliability Conference* (2021), pp. 3185–3191
33. E. Leurent. An Environment for Autonomous Driving Decision-Making (2018). URL `https://github.com/eleurent/highway-env`
34. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, arXiv preprint arXiv:1707.06347 (2017)
35. C. Zhao, Y. Tang, Q. Sun, A.V. Vasilakos, Deep direct visual odometry, IEEE Transactions on Intelligent Transportation Systems (2021)
36. J. Chen, J. Zhou, Z. Cao, A.V. Vasilakos, X. Dong, K.K.R. Choo, Lightweight privacy-preserving training and evaluation for discretized neural networks, IEEE Internet of Things Journal **7**(4), 2663 (2019)
37. T. Karras, S. Laine, T. Aila, A style-based generator architecture for generative adversarial networks, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 4401–4410
38. M. Wu, N. Xiong, A.V. Vasilakos, V.C. Leung, C.P. Chen, Rnn-k: A reinforced newton method for consensus-based distributed optimization and control over multiagent systems, IEEE Transactions on Cybernetics (2020)