Safer Autonomous Systems

SAS Report (Deliverable D1.1): Report on first version of a benchmark for safety monitoring of machine learning components

Raul S. Ferreira [1], Jean Arlat [1], Jeremie Guiochet [1], Helene Waeselynck [1], Harita Joshi [2], Mario Trapp [3]

[1]**LAAS-CNRS, Universite de Toulouse - UPS**
[2]**Jaguar Land Rover, UK**
[3]**Fraunhofer ESK, Munich, Germany**

European Training Network on Safer Autonomous Systems (SAS)

This project has received funding from the European Union's EU Framework Programme for Research and Innovation Horizon 2020 under Grant Agreement No. 812.788

Prolegomena

This Deliverable 1.1 was initially, as stated in the SAS Grant Agreement Description of the Action Annex I Part B, foreseen to be submitted on October 31, 2020. However, and due to the 2nd lockdown in France regarding the ongoing COVID-19 pandemic, some experiments that were crucial within the scope of this Deliverable 1.1 had to be postponed. As a consequence of the above, the submission date for this Deliverable 1.1 has been shifted to December 7, 2020.

Abstract

High accurate machine learning (ML) classifiers cannot guarantee that they will not fail at operation. Several works propose adding surveillance mechanisms to ensure systems correctness, such as safety monitors (SM) at runtime. However, the performance of such monitors, even for simple image classification tasks, tends to be optimistic. Actually, many works do not explore how to completely validate their efficiency. Hence, developing frameworks for assessing SM is an important research question due to the increasing adoption of ML in perception tasks for safety-critical systems. Therefore, this deliverable aims at establishing a baseline methodology for benchmarking SM mechanisms for ML classifiers. We propose a framework based on a traditional approach known as ”FARM”, that is adapted for systems with ML components. As a result, our approach can assess SM performance encompassing a broader set of metrics and configuration than usually proposed in the literature. Moreover, we show that our framework can easily highlight drawbacks in current methods and provide less biased analysis.

TABLE OF CONTENTS

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

1 Introduction

Machine learning (ML) models make decisions based on trained past data. A common way to validate it is by analyzing the discrepancy between the predicted values and the ground-truth (labels). If the results are satisfactory, it goes to production. However, even modern ML techniques, such as deep learning (DL), have a high uncertainty level. That is, a model can be wrong in its predictions even with 100% confidence [10], possibly leading safety-critical systems to hazardous situations. Hence, researchers are investing efforts in redundant techniques for monitoring these algorithms.

However, most researchers focus on increasing ML robustness during training, while just a few consider runtime safety strategies. One of these approaches is the safety monitor [21] (SM), a fault-tolerance mechanism widely applied in safety-critical domains [4]. The SM is responsible for maintaining the system in a safe state despite the occurrence of hazardous situations. It is usually composed of detection and a recovery system.

Recent works propose to monitor deep neural networks (DNN) inputs [20], outputs from the last layer [11] or from the intermediate ones [5, 12]. However, there is no consensus in the literature on the minimum requirements to be tested. Besides, many works focus on measuring the overall detection performance through positives/negatives rates or accuracy. Such an approach ignores other essential metrics or does not consider different perspectives, such as the system alone. Moreover, they usually do not take into account important statistical analysis regarding data and the results. Including such metrics, perspectives, and analysis is essential to assess safety monitor performance.

We aim at answering two research questions (RQ):

1. **RQ1 - Better measurements for SM regarding detection**: how efficient is the SM in detecting problematic data?
2. **RQ2 - The overall impact of the SM on the SUT**: is the SM improving or worsening the overall performance?

Therefore, this work proposes a baseline methodology for benchmarking SM for ML components. A benchmark framework has the main goal to assess the performance of an object by performing a set of standard tests against it [9]. Our main contributions are:

- *A new baseline methodology for benchmarking SM for ML classifiers*
  To the best of our knowledge, this is the first work that proposes a methodology that measures the efficiency of SM for ML components in different perspectives.
- *Three independent modules that can also work together*
  Data profile; system under test; and evaluation. These modules can be individually incorporated into a custom pipeline or even work as an entire pipeline for unit testing SM for image classifiers with ML.
- *Adaptation of well-established approaches*
  Our framework is an adaptation of the method FARM (Fault-Activity-Readouts-Measurement) [3] for evaluating fault tolerance mechanisms. Besides, it uses established metrics and dataset variations that bring more statistical power to the results.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

Our first benchmark results indicate that the runtime monitors based on observing neuron activation values are not more accurate than a random monitor. Additionally, most of them are statistically similar regarding detection accuracy.

This work is organized as follows: in Section 2, we provide background and related works on ML threats, safety monitoring, and current benchmark framework for SM. In Section 3, we present an overview of our framework, along with its main objectives. In Section 4, we show the first experiments, datasets, and results performed in the framework. Finally, in Section 5, we present our final considerations, limitations, and research directions.

2 Background and related work

Despite the ML field has several sub-fields such as object detection, regression, clustering, reinforcement learning, and so on, this work focuses on benchmark SM for ML classifiers when these classifiers are exposed to specific data that can threaten their performance at runtime. In the next subsections, we give a brief explanation of these concepts. Besides, we also show some related works regarding benchmarking detectors for image classifiers with ML when facing threatening data.

2.1 ML classifiers and out-of-distribution data

An ML classifier is a software component that uses an ML algorithm for identifying, given an observation, in which category it belongs [2]. ML classifiers take an input, such as an image or a vector of numerical values, and outputs categorical values (classification) according to a category (class) previously seen during the training process. This component, called the ML model, is usually validated by analyzing the discrepancy between the classified values and the ground-truth (labels). If the results are satisfactory, it goes to production.

ML models tend to be biased by the training data [22], resulting in a natural inability of a model to generalize 100% of time even with all available data correctly could be collected.

Beyond this fundamental model generalization issue, there is another real problem: data incompleteness. That is, rare conditions tend to be underrepresented since the training data represents a subset of all real-world possibilities [27]. It means that runtime data represents the same target class from training data, but that has different characteristics, different enough for threatening the ML performance. Such data is known as out-of-the-distribution (OOD) data. There are at least five types of OOD data when an ML model is operating at runtime:

- Novel classes: new classes are introduced during runtime. For example, an ML model can be trained to identify dogs but fails to classify a new dog type not present in the training data [24].
- Adversarial inputs: ML fails with high confidence due to small modifications in an input [17].
- Distributional shifts: joint distribution of inputs and outputs of the training dataset may differ from the runtime data. Such a situation is also known as a concept-drift [8]. Examples of distributional shift include a change in class attributes such as dimension, physical characteristics, contrast, brightness, and other pixel-related variations in the image.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

- Noise: deteriorated images due to small failures in exteroceptive sensors or unexpected environmental interference.
- Anomalies: severe failures from exteroceptive sensors capable of severely corrupt images. However, contrary to noise, such failures corrupt data so that such images lose the semantic value. It means that whatever it is the ML decision, it can correspond to an inaccuracy. Examples of such images are black images, images with several shifted pixels, and so on.

Next, we give details about current approaches for monitoring ML models when exposed to these types of OOD data.

2.2 SM for ML components

The provision of SM for ML components is a technique with a growing interest in the autonomous systems field, and the current literature proposes two different approaches to do it:

2.2.1 SM based on a functional specification

This approach is generally based on a physical model of the system or the environment and on properties they should guarantee. Some techniques use a set of rules for verifying the safety of ML components, considering this component as a *black-box*. For example, safety rules can be applied to verify if a vehicle can completely stop before reaching an obstacle ahead [23], or to trigger an intervention for avoiding a collision when an ego vehicle is approaching a leading vehicle [1]. In both cases, external (exteroceptive) sensors, such as distance sensors, are observed along with some internal (proprioceptive) sensors, such as speed, to evaluate if a safety property will be violated.

Some techniques have the advantage of automatically generating safety strategies such as SMOF [21]. However, analyzing the ML as a black box can be considered as a drawback. Accessing a certain level of properties that lead an ML to give a particular decision is essential for SM in complex scenarios. Besides, these techniques tend to be infeasible in a complex scenario [1].

Another technique based on a functional specification is the use of code assertions, also called model assertions [15]. This technique is an adaptation of the classical program assertions to monitor and improve ML models. The idea is to verify properties in the program logic to point possible failures. For example, monitoring temporal stability during operation verifies if the DNN output regarding an object is flickering in and flickering out in the camera, indicating a possible failure. Even if these techniques can be applied during design and operation, they alone cannot guarantee that a DNN decision is safe. The reason is that, for some corner cases, ML outputs wrong decisions that cannot be verified by inspecting the code logic or the sensor values and lead to hazards. Such corner cases come from regions of uncertainty [7].

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

2.2.2 SM based on training data

This approach is usually purely based on data instead of using a physical model for building the solution. For example, one approach is to monitor the *neuron patterns* observed from the layers of the DNN. For instance, an SM can monitor the values in the last layer of a DNN [11], verifying the decision's confidence level. However, it is not reliable since DNNs can output a wrong decision even with a high confidence level.

For avoiding the problem mentioned above, Liang et al. [19] proposed enhancing the reliability of out-of-distribution image detection in neural networks by applying techniques that decrease the confidence values outputted by the DNN. Previous results showed that such a technique is more reliable than the work proposed by [11]. This method is considered one of the state-of-the-art techniques for OOD detection. However, this method uses OOD data for finding optimal values for the parameters. Such an assumption is not realistic in most cases and tends to be very biased. A generalized version of this algorithm (also known as ODIN) was proposed by [13] and do not use OOD data to choose its parameters.

Cheng et al. [5] suggest comparing the recorded patterns of the DNN activation functions during the training with the ones related to the inputs during runtime. After the standard training process, a runtime monitor is created by feeding the training data to the network to store the neuron on-off activation patterns using binary decision diagrams (BDD). During runtime, the monitor rejects the network's decision if the current BDD pattern is not similar to the ones stored in the monitor during the training. One of the advantages is that it partially addresses the novelty class problem by signaling variations in the DNN activation functions' patterns. However, choosing a suitable threshold for determining which variation could be signaled as a novel class can be difficult, varying between the datasets. Besides, the memory required for constructing the BDD overgrows in respect to the DNN's size.

To overcome the problems above, Henzinger et al. [12] propose to verify neuron activation patterns values, but using a 2D projection instead of a BDD. This projection is an abstraction box built by taking the maximum and minimum activation function values during training. It inspects if the output of an activation function, after a new input passes through it, falls inside of this abstraction box during runtime. If not, it raises the alarm considering this input as a novel input. By doing that, the authors reduced the memory issue and sped up construction and detection. However, 2D projections tend to lose information when applied to high dimensional data, impacting the detection's accuracy. Moreover, applying linear projections to nonlinear activation functions tends to impact the detection performance negatively.

Another approach is to monitor the DNN inputs based on a radius distance threshold calibrated during the training [20]. The idea is to perturb the DNN inputs, observe the correct answer, and to determine how considerable is the distance regarding the DNN decisions. By doing that, the authors observed that the radius distance of a DNN that correctly classified the perturbed inputs is much larger than those that misclassified such inputs after a perturbation. The authors use this intuition to protect the DNN decisions from adversarial attacks. The advantage of this approach is that it does not need to inspect the internals of the DNN. The drawback is that it tends to be biased to the data used during the training.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

2.3 Benchmarking SM for ML-based components

Despite several works for monitoring machine learning at runtime, just a few try establishing better benchmark methodologies. One of these works proposes a less biased evaluation of out-of-distribution detectors, called OD-test [26]. Same as our work, the authors present a methodology that divides the data into three parts: training/validation, in-distribution testing set, and out-of-distribution testing set. Another similarity with our methodology is that the authors also test the detectors using different ML architectures across other datasets.

Nevertheless, our work differs from OD-test in three different aspects. Firstly, we propose a methodology that covers the entire pipeline, from data generation to evaluation. Secondly, we apply metrics beyond the traditional accuracy and positive/negative rates, with additional statistical analysis in the results. Finally, we perform tests using five different types of OOD data across a more significant amount of generated datasets. In the next Section, we explain each part of this framework and its applicability to assess the monitoring of neuron activation patterns in Section 4.

3 A baseline methodology for benchmarking SM with ML

The proposed methodology is an adaptation of the FARM [3] methodology. It is divided into three modules, as illustrated in Figure 1.

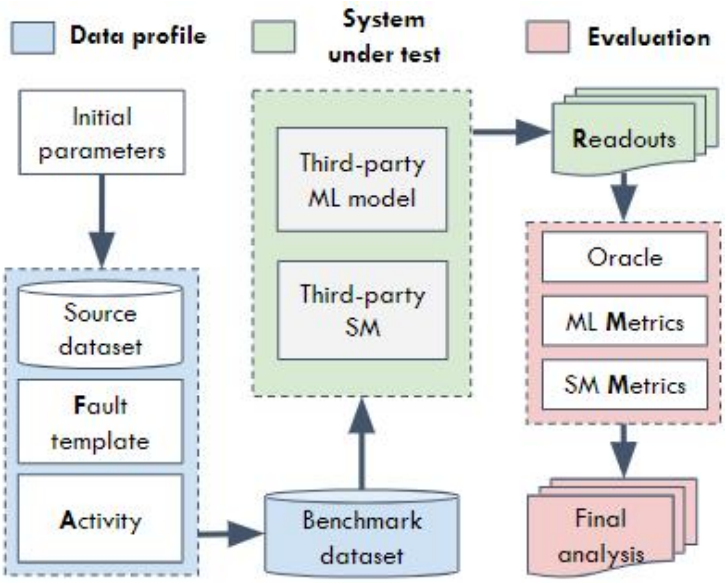

Figure 1: A high-level overview of the framework.

After setting some initial parameters such as type of data generation, the amount of data to be tested, which ML models and SM to include, the methodology starts with the first module, called data

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

profile. This module is responsible for generating benchmark datasets – encompassing *workload (activity)* and *faultload (fault template)* – used as inputs to the system under test (SUT).

The SUT is composed of the ML and the SM. The test runs and generates results (readouts) at the end of the process. These readouts are inputs for the evaluation.

The evaluation module is responsible for analyzing the readouts through several metrics for different components of the system. In the end, it provides the final analysis for each type of readout. Next, we explain each module and its components.

3.1 Data profile

This is the first module and it is composed of the following items:

- **Activity**: The source dataset containing *in-distribution (ID)* data, that is, instances from a distribution known by the ML.
- **Fault template**: it contains rules for generating *out-of-distribution (OOD)* data, that is, instances that can threaten the ML.

The benchmark dataset receives ID and OOD data to test how the ML and the SM behave when exposed to expected and unexpected data. Even though all types of OOD can be part of a unique benchmark dataset, in this work, we generated datasets divided by each category of OOD data.

An important premise is that the generated benchmark dataset is not applied to build or validate the SM or to train the ML model. This guarantees that the experiments do not yield biased results and is more realistic since, in real scenarios, there are no guarantees about which type of data will arrive at runtime.

3.2 System under test

The system under test (SUT) receives the benchmark dataset as input, and produces readouts for each component. This module is composed of two main items: the **ML model**, that is, a third-party ML classifier already trained; and the **safety monitor**, a third-party SM containing a detection mechanism. We call the ML and the SM third-party components since they can be built separately from the framework.

This module simulates a stream of images that can be ordered sequentially or randomly. We started with datasets with random sequences of images for this work since the primary image datasets available in the literature use such a setting. In Algorithm 1, we outline the pseudo-code for the testing procedure.

The simulation works as a stream of images coming from the benchmark dataset set D that contains images X and labels y, arriving one of each at a time. The ML model receives this image and makes a $\hat{y}$ classification.

Next, the system triggers the SM, which checks a set of predefined properties. These properties can be the ML classification $\hat{y}$ with the associated confidence level, intermediate layers, the input X, or even a combination of two or more of these properties. In the case of this pseudo-code example,

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

Algorithm 1 System under test

Input: Generated dataset D; SM θ; ML model ϕ
Output: Set of $readouts$ for each instance $X, y \in D$

$readouts \leftarrow \{\}$
for each $X, y \in D$ **do**
  $\hat{y} \leftarrow \phi(X)$
  $\hat{s} \leftarrow \theta(X, \phi)$
  $\hat{m} \leftarrow eval(\hat{y}, \hat{s})$
  $readouts \leftarrow add(y, \hat{y}, \hat{s}, \hat{m})$
end for
return $readouts$

the SM inspects the ML model's internal properties (DNN's hidden layer) during the classification, as recently suggested in the literature [5, 12].

First, the SM detector mechanism raises a detection alarm or not $\hat{s}$ regarding the OOD detection. Second, the SM intervenes invalidating the ML classification result if an alarm is raised; otherwise, it accepts the ML classification. We observe whether such intervention produced a desirable outcome for the system or not. We call it overall detection $\hat{m}$. By desirable outcome, it means that canceling an ML output or agreeing with it was beneficial to the SUT. This result is independent from the fact if the SM specific detection was correct or not in respect of the OOD detection. The ground truth y, the ML classification $\hat{y}$, the SM specific detection $\hat{s}$, its overall detection $\hat{m}$, and other *readouts* are divided into three different categories:

1. ML readouts: it contains a classification $\hat{y}$ represented as a class number, confidence value, and intermediate values from hidden layers.
2. SM readouts: it contains two types of detection: a) specific ($\hat{s}$), which outputs 1 for OOD detection or 0 otherwise; and b) overall ($\hat{m}$), which after OOD detection, it cancels the ML decision (1) or 0 otherwise.
3. General readouts: it contains general outputs from all components above, such as processing time and memory.

As can be noted, the SM readouts contain results in two dimensions: OOD detection and SM decision. Considering two dimensions for detection is more realistic and complete than just analyzing the detection rate for the OOD data. The reason is that while the SM tries to detect OOD data, it also can avoid that the ML gives an answer that is different from the ground truth. The opposite is also true. The SM can incorrectly detect ID data as OOD data, hindering the correct decisions of ML. Next, we detail the last module, the one that evaluates all readouts produced by the SUT module.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

3.3 Evaluation

This module receives the readouts as inputs containing false positives/negatives and true positives/negatives regarding OOD data detection and the SM decision. This module provides *metrics* evaluating two main aspects:

1. *The SM's decision impact:* the objective is to determine if the SM improves or worsens the overall accuracy. We analyze the SUT using the ML alone (baseline) and ML with the SM. The considered readouts are the overall decisions made by the SUT with ML alone and ML with SM, processing time, and memory.
2. *The detection performance between SMs:* the objective is to assess the SM results regarding the OOD detection. The readouts include SM OOD detection (e.g., raises/does not raise the alarm), memory, and processing time.

This module contains two major items: the **oracle**, and the **metrics for ML and SM**. The oracle determines whether a test has passed or failed. Thus, the oracle considers a correct specific detection when the SM correctly detects an OOD data. Besides, the oracle considers a correct overall detection for the SUT when the SM correctly avoids a wrong ML classification for ID or OOD data. Therefore, the oracle takes into consideration the following scenarios to determine whether a result is correct or incorrect:

- **ID data arrives in the stream**
  - **If the SM detects OOD data:** it means a *false positive for the specific task* of OOD detection.
    - **If the ML classification is equal to the ground truth:** it means a *false positive for the overall task* of avoiding a failure since the SM intervened without necessity.
    - **If the ML classification is different from the ground truth:** it means a *true positive for the overall task* of avoiding a failure since it canceled the ML misclassification for ID data.
  - **If the SM does not detect OOD data:** it means a *true negative for the specific task* of OOD detection.
    - **If the ML classification is equal to the ground truth:** it means a *true negative for the overall task* of avoiding a failure since the ML gave a correct classification even though the SM did not detect OOD data.
    - **If the ML classification is different from the ground truth:** it means a *false negative for the overall task* of avoiding a failure since the ML misclassified the OOD data and the SM did not detect OOD data, which could cancel the wrong ML decision.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

- **OOD data arrives in the stream**
  - **If the SM detects OOD data:** it means a *true positive for the specific task* of OOD detection.
    - **If the ML classification is equal to the ground truth:** it means a *false positive for the overall task* of avoiding a failure since the ML gave a correct classification but the SM intervened in the decision.
    - **If the ML classification is different from the ground truth:** it means a *true positive for the overall task* of avoiding a failure since the ML gave an incorrect classification and the SM correct canceled the ML decision.
  - **If the SM does not detect OOD data:** it means a *false negative for the specific task* of OOD detection.
    - **If the ML classification is equal to the ground truth:** it means a *true negative for the overall task* of avoiding a failure since the SM detection for OOD data also avoided an ML misclassification.
    - **If the ML classification is different from the ground truth:** it means a *false negative for the overall task* of avoiding a failure since the SM detection for OOD data also avoided an ML misclassification.

The only exception for the above rules is for novelty class detection. Since the ML does not utilize the new classes for training, it will always output the wrong classification for new classes at runtime. For this type of OOD data, when an OOD data arrives in the stream, if the SM correctly detects OOD data, it will always be interpreted as true positive since the SM always correctly cancels the ML classification, independently of the ground truth. Conversely, if the SM does not detect OOD data, it will always be considered as a false negative. If the SM lets the ML deal with a class that it was not trained in before can be regarded as unsafe. That is, the ML will always give an incorrect classification.

Regarding the metrics, we apply eight types of metrics. The first six metrics are based on positives and negatives rates, while the last two are based on statistical methods:

- *Matthews correlation coefficient (MCC)*: it ranges from -1 if (ML or SM is always wrong) passing through 0 (ML or SM is accurate as random) to 1 (ML or SM is always right). This metric is more reliable than traditional metrics such as accuracy. It yields a high score only if the ML or SM can be assertive in all of the four confusion matrix categories [6]. In contrast, accuracy only considers the portion of the right answers. In a scenario in which the number of a class is 80%, the accuracy could yield a score of 80% for an SM that did not detect any of the other classes.
- *False positive rate (FPR)*: also known as type-I error. It indicates how many false alerts the SM raises for the specific task of detecting OOD data. If this value is high, the SM tends to indicate a problem when there are not in most cases.
- *False negative rate (FNR)*: also known as type-II error. It indicates how often the SM misses detecting OOD data. If this value is high, it indicates that the SM does not recognize the difference between the ID and OOD data.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

- *Precision and recall*: the fraction of correct detection and the fraction of available OOD data. These metrics help to indicate how well the SM detected OOD data through the benchmark dataset.
- *Micro-F1*: harmonic mean (global) for the prediction x recall. It helps assess the quality of multi-label binary problems, which makes them suitable to be applied in the SUT evaluation.
- *Percentage of relative change*: it shows the relative difference between two methods expressed in percentage. When comparing the baseline and the SMs, this metric shows how much the SM improved or worsened the overall task of detecting OOD in the SUT.
- *Critical difference diagram*: it is applied for all MCC results through all datasets. It shows how statistically significant are the results between the SMs.
- *Wilson score*: it gives a range of probabilities in the detection error results with a level of confidence. It shows how much the detection error is expected to vary in a real scenario.

Next, we apply this benchmark framework to run experiments for OOD detection.

4 Experiments and results

To validate our framework, we chose ML models that classify images and SMs that inspect these models internal outputs. Hence, the benchmark dataset is composed of images.

4.1 Data profile

We use three traditional image datasets as the *activity* of our framework:

- GTSRB [25]: German traffic signs with 43 classes, divided into 39200 instances for training and 12600 for testing.
- BTSC [14]: Belgium traffic signs with 62 classes, divided into 7000 images for training and testing.
- CIFAR-10 [16]: images with ten general classes (e.g., dog, car ...), containing 50000 instances for training and 10000 for testing.

Next, we apply a *fault template* for novelty class detection. This fault template applies one dataset as ID dataset and another dataset with new classes as OOD data. Therefore, it produces three benchmark datasets:

- GTSRB as ID data, and BTSC as OOD data: this combination tests whether the SM can distinguish new classes that have similar characteristics to the known ones.
- GTSRB as ID data, and CIFAR-10 as OOD data: this combination tests whether the SM can distinguish new classes that are very different from the known ones.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

- CIFAR-10 as ID data, and GTSRB as OOD data: this combination tests the same as the aforementioned combination. However, since the ID dataset is different, the ML model and the SM are built with different data. That is, this permutation between these two datasets produces different outcomes.

4.2 System under test

We simulate a scenario in which the SM has to detect novel classes by checking one RGB-colored image at a time in a randomly ordered stream of images. Once the SM makes a detection, it cancels the ML classification since this classification is potentially wrong. If nothing is detected, the ML classification is accepted.

For the ML model, we use a LeNet [18] since it is a simple and traditional convolutional neural network (CNN) algorithm containing around 100,000 parameters in total and 128 neurons in the last hidden layer. We test four SM methods: outside-of-the-box abstraction (OOB) [12], and the two SM investigated by us based on this approach. We call them OOB ISOMAP and OOB PCA. Finally, the fourth SM is a state-of-the-art technique also introduced in Section 2, called ODIN [13].

Worth mentioning that these two variants of outside-of-the-box, OOB ISOMAP and OOB PCA, are natural extensions of the original paper since the authors applied their dimensionality reduction method but did not investigate two basic ones: ISOMAP and PCA. ISOMAP is a popular nonlinear dimensionality reduction method, and PCA is also a popular method, but linear. With these two suggested variants, we can analyze if the dimensionality reduction methods influence the outcomes. Besides, the three parameters applied to the original method are kept the same for these two variants.

Regarding the parameters for the SMs, to avoid biased results, we found the best values using only the ID dataset. We applied the same number of clusters for the outside-of-the-box methods, as suggested by the authors. However, we found that enlarging the size of boxes (parameter τ in the paper) as suggested by the authors resulted in more false positives. Finally, we have found that this method has the problem that some points fall precisely over (0, 0) and are not considered inside the box. This behavior also negatively affects this method's performance, so we left this value slightly higher than 0 to avoid this drawback.

Regarding ODIN, we use 1000 as a value for the temperature parameter as suggested by the authors. However, the original paper uses OOD data for choosing optimal values for the other two parameters: magnitude and threshold. To avoid it, we adopted the same strategy proposed by [13]. We searched for the best values of magnitude between a set of values suggested by [13]. After it, we chose the confidence thresholds for determining OOD data by selecting the lower confidence value (using ceiling) outputted from the method when exposed to ID data. We present the chosen parameter values for each SM in Table 1.

4.3 Evaluation

Here we present the results regarding the novelty class detection experiments.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

Table 1: SM parameters chosen for each benchmark dataset.

Outside-of-box	τ	**# of clusters**	**# of dimensions**
GTSRB	0.0001	3	2
CIFAR-10	0.0001	3	2
ODIN	**temperature**	**magnitude**	**threshold**
GTSRB	1000	0.0025	0.0237
CIFAR-10	1000	0.0014	0.1007

4.3.1 OOD detection performance

Here, the measurement is straightforward; OOD data is considered positive, and ID data negative. For evaluating the results, we apply six metrics. Since we measure novelty detection, there is no necessity to include the ML alone since it will always classify it incorrectly, and consequently, there is no necessity to use the percentage of a relative change either. Table 2 shows these results for GTSRB as ID dataset and BTSC as OOD dataset.

Table 2: Comparing data-based monitors for GTSRB as ID dataset, and BTSC as OOD dataset.

Method	MCC	FPR	FNR	Precision	Recall	Micro-F1
OOB	**0.21**	0.84	0.04	0.8	0.96	**0.73**
OOB ISOMAP	0.2	**0.72**	0.11	**0.81**	0.89	0.73
OOB PCA	0.04	0.86	0.11	0.78	0.89	0.68
ODIN	0.03	0.99	**0.0**	0.16	**1.0**	0.06

This benchmark dataset has the challenge of having ID and OOD data with a similar appearance. The results in Table 2 indicate that just the original outside of the box and the ISOMAP variation are only slightly better than a random classifier. Although all monitors have low false-negative rates, we observe that the weakness of those SM is characterized by the high false positive rates observed in the experiment. For this reason, the recall tends to be high. Next, in Table 3, we show the second benchmark dataset results containing CIFAR-10 as ID dataset and GTSRB as OOD dataset.

Table 3: Comparing data-based monitors for CIFAR-10 as ID dataset, and GTSRB as OOD dataset.

Method	MCC	FPR	FNR	Precision	Recall	Micro-F1
OOB	0.06	0.97	**0.0**	0.16	**1.0**	0.09
OOB ISOMAP	0.04	0.98	0.0	0.16	1.0	0.07
OOB PCA	0.17	0.8	0.03	0.2	0.97	0.33
ODIN	**0.23**	**0.61**	0.1	**0.24**	0.9	**0.52**

As opposed to the first one, this benchmark dataset has a significant difference between ID and

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

OOD data. According to Table 3, just ODIN obtained MCC results only slightly better than the random classifier and got the best results across the other metrics. However, a similar behavior to the previous experiment can be observed in the results: a high rate of false positives and a borderline MCC performance. Since the ML performance was not good either for ID data, one can argue that SM based on ML models that do not perform well can have a poor performance. Aiming to investigate it, in Table 4, we show the results for the same pair of datasets, but now using GTSRB as ID dataset and CIFAR-10 as OOD dataset.

Table 4: Comparing data-based monitors for GTSRB as ID dataset, and CIFAR-10 as OOD dataset.

Method	MCC	FPR	FNR	Precision	Recall	Micro-F1
OOB	0.16	0.73	0.1	0.21	0.9	0.4
OOB ISOMAP	**0.19**	**0.06**	0.81	0.4	0.19	**0.8**
OOB PCA	-0.06	0.99	**0.02**	**0.64**	**0.98**	0.5
ODIN	-0.07	1.0	0.02	0.17	0.98	0.06

As can be seen in Table 4, even the ML model for GTSRB has a high MCC value (0.96), these SMs do not achieve good performance using the DNN internals information. In the case of OOB ISOMAP, it achieved a low false-positive rate, which seems to be something rare. However, it had an equally high false-negative rate, which can be considered worse depending on the scenario. Therefore, all SM had a poor performance. It happens due to the unreliable nature of DNN confidence values and the high nonlinear nature of activation functions, in the case of ODIN and outside-of-the-box, respectively.

Next, we further investigate the SM performance, but this time, for the impact of the SM decision.

4.3.2 The overall impact of the SM in the SUT with ID data

As previously explained, the objective here is to evaluate how much the SM impacts the SUT when it works along with the ML classifier. As explained in Section 3, for novelty classes as OOD data, if the SM correctly detects OOD data, it will always be interpreted as true positive since the SM always correctly cancels the ML classification, independently of the ground truth. On the other hand, if the SM does not detect OOD data, it will always be considered a false negative, and the ML will always give an incorrect classification. Therefore, measuring the OOD detection here is not interesting since the ML would have 0% of accuracy. Similarly, measuring the entire stream is not that informative since the ML accuracy goes to 0 when the amount of OOD data goes to the infinite. For this reason, Table 5 shows the overall SM impact in the SUT when using GTSRB or CIFAR-10 as ID datasets.

The results in Table 5 are expressed as MCC values. For the SM, it also has a percentage of relative change showing how much better/worse it performed compared to the baseline (ML alone).

As can be noted, all SM methods perform worse than the ML alone when exposed to ID data. This result is expected due to the generalization power of ML models. However, a reliable SM is also the one that can perform well when exposed to ID data, avoiding an ML misclassification or simply not raising a false alarm hindering a correct ML classification.

According to the results in Table 5, the best SM performance was obtained by the outside-of-the-box with an ISOMAP and ODIN on GTSRB and CIFAR-10, respectively. It shows that the SM performed

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

Table 5: Overall SM impact in the SUT when using GTSRB or CIFAR-10 as ID dataset.

Method / MCC	GTSRB	CIFAR-10
Baseline	**0.96**	**0.74**
OOB	0.59 (–38.54%)	0.51 (–31.08%)
OOB ISOMAP	0.66 (–31.25%)	0.51 (–31.08%)
OOB PCA	0.51 (–46.87%)	0.59 (–20.27%)
ODIN	0.5 (–47.92%)	0.64 (–13.51%)

consistently better than a hypothetical random classifier (MCC=0). However, no SM surpassed 0.66 for MCC. Thus, it also shows a significant gap between the performance of the ML alone and the SM for ID data. This result is particularly interesting since the tested SM use information from the ML model and build their hypothesis over ID data.

5 Conclusion and research directions

The work presented in this report focuses on benchmarking SM for novelty class detection. The benchmarking approach allowed new insights of SM based on neuron activation patterns proposed in the literature. We conclude that according to preliminary experiments, SM solely based on DNN data seems to be not reliable. They tend to be dependent on the DNN accuracy and with a high false positive rate. Moreover, their performance tends to be only as good as random.

Another conclusion is that there is a need for stronger methodology on how to benchmark SM, including more metrics beyond accuracy and false positives/negatives. As a future work, we will perform more experiments with different ML architectures and types of OOD data.

References

[1] Fadi Al-Khoury, *Safety of machine learning systems in autonomous driving*, Master's thesis, KTH Royal Institute of Technology School of Industrial Engineering and Management, Stockholm, Sweden, 2017.

[2] Ethem Alpaydin, *Introduction to machine learning*, MIT press, 2020.

[3] Jean Arlat, Martine Aguera, Louis Amat, Yves Crouzet, Jean-Charles Fabre, Jean-Claude Laprie, Eliane Martins, and David Powell, *Fault injection for dependability validation: A methodology and some applications*, IEEE Transactions on software engineering **16** (1990), no. 2, 166–182.

[4] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr, *Basic concepts and taxonomy of dependable and secure computing*, IEEE transactions on dependable and secure computing **1** (2004), no. 1, 11–33.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

[5] Chih-Hong Cheng, Georg Nührenberg, and Hirotoshi Yasuoka, *Runtime monitoring neuron activation patterns*, 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE), Florence, Italy, IEEE, 2019, pp. 300–303.

[6] Davide Chicco and Giuseppe Jurman, *The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation*, BMC genomics **21** (2020), no. 1, 6.

[7] Tommaso Dreossi, Alexandre Donzé, and Sanjit A Seshia, *Compositional falsification of cyber-physical systems with machine learning components*, Journal of Automated Reasoning **63** (2019), no. 4, 1031–1053.

[8] Raul S Ferreira, Geraldo Zimbrão, and Leandro GM Alvim, *Amanda: Semi-supervised density-based adaptive model for non-stationary data with extreme verification latency*, Information Sciences **488** (2019), 219–237.

[9] Philip J Fleming and John J Wallace, *How not to lie with statistics: the correct way to summarize benchmark results*, Communications of the ACM **29** (1986), no. 3, 218–221.

[10] Yarin Gal and Zoubin Ghahramani, *Dropout as a bayesian approximation: Representing model uncertainty in deep learning*, International conference on machine learning (ICML), New York, United States, 2016, pp. 1050–1059.

[11] Dan Hendrycks and Kevin Gimpel, *A baseline for detecting misclassified and out-of-distribution examples in neural networks*, CoRR **abs/1610.02136** (2016).

[12] Thomas A. Henzinger, Anna Lukina, and Christian Schilling, *Outside the box: Abstraction-based monitoring of neural networks*, ECAI, Frontiers in Artificial Intelligence and Applications, Santiago de Compostela, Spain, vol. 325, IOS Press, 2020, pp. 2433–2440.

[13] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira, *Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data*, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10951–10960.

[14] Arpan Jain, Apoorva Mishra, Anupam Shukla, and Ritu Tiwari, *A novel genetically optimized convolutional neural network for traffic sign recognition: A new benchmark on belgium and chinese traffic sign datasets*, Neural Processing Letters **50** (2019), no. 3, 3019–3043.

[15] Daniel Kang, Deepti Raghavan, Peter Bailis, and Matei Zaharia, *Model assertions for monitoring and improving ML model*, arXiv preprint arXiv:2003.01668 (2020).

[16] Alex Krizhevsky and Geoff Hinton, *Convolutional deep belief networks on cifar-10*, Unpublished manuscript **40** (2010), no. 7, 1–9.

[17] Alexey Kurakin, Ian Goodfellow, and Samy Bengio, *Adversarial examples in the physical world*, arXiv preprint arXiv:1607.02533 (2016).

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/

[18] Yann LeCun et al., *Lenet-5, convolutional neural networks*, URL: http://yann. lecun. com/exdb/lenet **20** (2015), no. 5, 14.

[19] Shiyu Liang, Yixuan Li, and R Srikant, *Enhancing the reliability of out-of-distribution image detection in neural networks*, International Conference on Learning Representations, 2018.

[20] Jiangchao Liu, Liqian Chen, Antoine Mine, and Ji Wang, *Input validation for neural networks via runtime local robustness verification*, arXiv preprint arXiv:2002.03339 (2020).

[21] Mathilde Machin, Jérémie Guiochet, Hélène Waeselynck, Jean-Paul Blanquart, Matthieu Roy, and Lola Masson, *SMOF: A safety monitoring framework for autonomous systems*, IEEE Transactions on Systems, Man, and Cybernetics: Systems **48** (2018), no. 5, 702–715.

[22] Claude Nadeau and Yoshua Bengio, *Inference for the generalization error*, Machine learning **52** (2003), no. 3, 239–281.

[23] Umit Ozguner, Christoph Stiller, and Keith Redmill, *Systems for safety and autonomous behavior in cars: The darpa grand challenge experience*, Proceedings of the IEEE **95** (2007), no. 2, 397–412.

[24] Pramuditha Perera and Vishal M Patel, *Deep transfer learning for multiple class novelty detection*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), California, United States, 2019, pp. 11544–11552.

[25] Pierre Sermanet and Yann LeCun, *Traffic sign recognition with multi-scale convolutional networks*, The 2011 International Joint Conference on Neural Networks, IEEE, 2011, pp. 2809–2813.

[26] Alireza Shafaei, Mark Schmidt, and James J Little, *A less biased evaluation of out-of-distribution sample detectors*, 30th British Machine Vision Conference, Cardiff, Wales (2019).

[27] Sina Shafaei, Stefan Kugele, Mohd Hafeez Osman, and Alois Knoll, *Uncertainty in machine learning: A safety perspective on autonomous driving*, International Conference on Computer Safety, Reliability, and Security (SAFECOMP), 2018, pp. 458–464.

This project has received funding from the European Union's
EU Framework Programme for Research and Innovation
Horizon 2020 under Grant Agreement No. 812.788.
https://etn-sas.eu/