Efficient Computation of Robustness of Convolutional Neural Networks

Paolo Arcaini
National Institute of Informatics
Tokyo, Japan
arcaini@nii.ac.jp

Andrea Bombarda
University of Bergamo
Bergamo, Italy
andrea.bombarda@unibg.it

Silvia Bonfanti
University of Bergamo
Bergamo, Italy
silvia.bonfanti@unibg.it

Angelo Gargantini
University of Bergamo
Bergamo, Italy
angelo.gargantini@unibg.it

***Abstract*—Validation of CNNs is extremely important, especially when they are used in safety-critical domains. In particular, in the latest years, the focus of validation has been put on assessing the *robustness* of CNNs, i.e., their ability to correctly classify perturbed input data. A way to measure robustness is to check the network accuracy over many datasets obtained by altering the input data in different ways, but this is time and resource-consuming. In this paper, we present ASAP, a method to efficiently compute the robustness of a CNN, exploiting a parabola-based approximation which allows to adaptively select only relevant alteration levels. The method is tested on two different benchmarks (MNIST and breast cancer classification). Moreover, we compare ASAP with other techniques based on uniform sampling, numerical integration, and random sampling.**

***Index Terms*—Convolutional Neural Networks, robustness, efficient robustness computation, image classification, alteration**

I. Introduction

Convolutional Neural Networks (CNNs) are increasingly used to perform different activities [3], among which many of them are safety-critical [14] as, e.g., in autonomous driving [10], or in the medical practice [8]. Thus, especially in these cases, validation activities must be performed [20]. The most recent developments in the validation of CNNs are based on the evaluation of the *robustness*, i.e., the ability of the network to correctly evaluate slightly altered inputs. However, the majority of research works are focused on adversarial examples, which are often created by exploiting the internal structure of the network [16], and may not reflect plausible inputs that could occur during real network usage [12], [13], [18]. Instead, the robustness should be defined by considering *real alterations* that may occur to input data as well. In [4], we proposed a robustness definition for CNNs that considers the alterations that are typical of the domain in which the CNN is used. Moreover, in [5], we presented ROBY, a Python tool for automatic robustness analysis, implementing the robustness definition and supporting different types of input data.

Computing robustness can be really expensive in terms of time and resource consumption, since every input of the test dataset has to be repeatedly altered with different alteration levels. One way to obtain a correct estimation of the network robustness is by increasing the number of alteration levels analyzed. However, this has the drawback of increasing the time required for the analysis. Thus, in this paper, we propose ASAP (*Adaptive SAmpling by Parabolic estimation*), a novel method to estimate the robustness of a CNN, aiming at computing robustness more efficiently. It is based on a parabolic approximation of the accuracy curve and allows, by means of specific parameters, to trade off the precision of the computed robustness against the time required to compute it.

We assess ASAP on two different benchmarks, namely the MNIST dataset recognition and the breast cancer classification. Furthermore, we compare the approach with other possible techniques for robustness computation based on uniform sampling, numerical integration, and random sampling.

Paper structure. Sect. II introduces the definition of robustness and highlights the limits of existing approaches for its calculation. Sect. III presents ASAP. Sect. IV introduces the experiments design, with the two benchmarks used for the evaluation. Sect. V evaluates ASAP w.r.t. other available techniques. Finally, Sect. VI reviews some related work, and Sect. VII concludes the paper.

P. Arcaini is supported by ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), JST, and Engineerable AI Techniques for Practical Applications of High-Quality Machine Learning-based Systems Project (Grant Number JPMJMI20B8), JST-Mirai.

II. Background and basic definitions

A *convolutional neural network* (CNN) is a type of deep neural network, mainly used to analyze images, which uses the linear mathematical operation *convolution* (instead of the regular matrix multiplication) in at least one of its layers.

Digital images taken as input of the CNN can be susceptible to modifications during their acquisition. In [4], we proposed a *robustness* definition that assesses how much input modifications influence the classification accuracy; the definition is built on the concept of *alteration* defined as follows.

Definition 1 (Alteration)**.** An *alteration* of type A of an input t is a transformation of t that mimics the possible effect on t when a problem during its acquisition, or in its elaboration, occurs in reality. In the following, we identify with P^{A_l} the set of data obtained by altering all the input data in P with an alteration of type A of *level* $l \in [L_A, U_A]$, where $[L_A, U_A]$ is the range of plausible alteration levels of type A.

The *robustness* of a CNN is then defined as its ability to correctly classify altered data. Intuitively, given an alteration

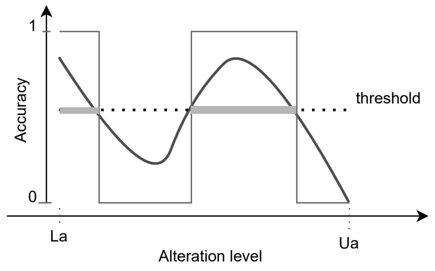


Fig. 1. Robustness

interval $[L_A, U_A]$, the robustness is the portion of that interval in which, if we classify the altered data with the same CNN, the accuracy is still acceptable, i.e., it is above a given threshold Θ. In Fig. 1, the accuracy is the black curve, and the robustness is the total length of the intervals in which the accuracy is above the threshold (gray line), divided by the total length of the alteration interval. The robustness is formally defined as follows.

Definition 2 (Robustness)**.** Let Θ be a threshold representing the minimum accepted accuracy. The *robustness* of a CNN C w.r.t. alteration of type A in the range $[L_A, U_A]$ (using a set of inputs P) is defined as the percentage of alteration values for which the accuracy $acc(C, P^A)$ is above Θ. Formally:

$$rob_A(C,P) = \frac{\int_{L_A}^{U_A} H(acc(C, P^{A_l}) - \Theta)dl}{U_A - L_A}$$

$$\text{where} \quad H(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

Computing precisely the robustness using this formula (where H is the Heaviside function) is very difficult. Indeed, the accuracy function is not known a priori and, therefore, it should be computed for all the alteration levels l in $[L_A, U_A]$ which could be many, if not infinite. Moreover, computing the accuracy of the network C when a single alteration level A_l is applied to the set P, requires a considerable effort, so a method to select suitable alterations is necessary in order to reduce the computation time.

A naive solution is to uniformly sample in $[L_A, U_A]$ and compute the accuracy for the sampled points. We followed this approach in [4], where we proposed a robustness definition exploiting equidistributed discrete points. In that case, we count for how many points (n_{acc}) the accuracy is acceptable, relatively to the total number of sampled points n. Formally:

Definition 3 (Uniform robustness)**.** Given n equidistributed points $SP = \{l_1, \ldots, l_n\}$ sampled in the interval $[L_A, U_A]$, the *uniform robustness* is defined as:

$$rob_A(C,P) = \frac{n_{acc}}{n} = \frac{|\{l \in SP \mid acc(C, P^{A_l}) \geq \Theta\}|}{|SP|} \quad (1)$$

Note that also numerical integration methods for integral calculation rely on sampling of points and, so, they could be used to compute robustness. However, in their application, the user has little or no control over the points to be used to compute the accuracy function, and this may result in oversampling not relevant areas and/or undersampling relevant ones. We will also compare with this approach in our experiments.

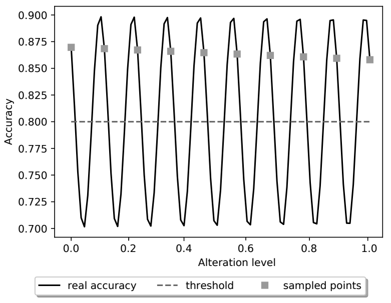


Fig. 2. Error in robustness computation with uniform sampling

A. The limits of the uniform sampling approach

The robustness, as defined in Def. 3, may still require a lot of computational power and time to be evaluated, especially when the alteration levels are many. In fact, given n the number of alteration levels to be applied for an alteration A, k the number of inputs, and t_A the time required for applying the alteration A to a single image, the total time required to perform robustness analysis is $t_{tot_A} = n \cdot k \cdot t_A$. For instance, if $n = 1000$, $k = 1000$ and $t_A = 0.1sec$, the total time required to compute the robustness is approximately 28 hours.

One way to reduce the effort is to reduce n, i.e., the number of sampled levels for the alteration A. This is a viable solution, but has some drawbacks, especially when analyzing networks whose accuracy varies a lot. For example, Fig. 2 shows the evaluated points when uniform sampling of an accuracy function is performed with $n = 10$. Using Def. 3, we would compute a robustness of $rob_A(C,P) = 100\%$. Nonetheless, the real value of the robustness for the analyzed network is significantly lower ($\simeq 50\%$).

Thus, a way to solve these problems is to adaptively select the points to be sampled (possibly not uniformly), as usually done for input values in software testing. In fact, choosing the correct input parameters and the correct values to be tested in a program is challenging because different inputs may lead to different bug discoveries; however, sampling some inputs is required since exhaustive testing can not be performed.

III. Adaptive SAmpling by Parabolic estimation

In order to tackle the limitations of the uniform sampling approach, we propose the ASAP algorithm to automatically select the points where to evaluate the accuracy. The best points to select would be those in which the accuracy curve intersects the threshold Θ. However, since we do not know the analytical form of the curve and we can not compute these intersections, we try to select points as close as possible to Θ.

ASAP is based on the assumption that, once we have computed the accuracy for two alteration levels A and B, the real accuracy curve between A and B will be included in the

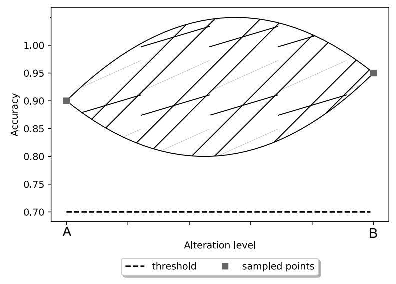
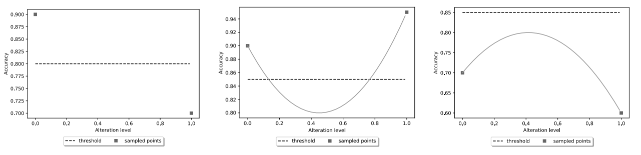


Fig. 3. Area in which the real accuracy curve between A and B will be likely included, identified by two parabolas with concavity depth $\pm\hat{a}$



(a) A is above Θ and B under

(b) The parabola with concavity depth $\hat{a}$ intersects with y=Θ

(c) The parabola with concavity depth -$\hat{a}$ does not intersect with y=Θ

Fig. 4. Different positions of points during recursive accuracy evaluation

area between two parabolas passing through the points A and B, and having concavity depth respectively $+\hat{a}$ and $-\hat{a}$ (see Fig. 3), i.e., with equation $y = ax^2 + bx + c$ with $a = \pm\hat{a}$. If there is an intersection between that area and the threshold Θ, and the distance between A and B is sufficiently large, then we compute the accuracy in the middle point M between A and B, we add it to the sample set, and we recursively apply the same procedure to the two intervals $[A, M]$ and $[M, B]$. In this way, the number of evaluated points is adaptively determined and depends on the value of the parameter $\hat{a}$. Intuitively, the higher is the value of $\hat{a}$, the higher is the number of alteration levels evaluated by the algorithm. Thus, users must choose the $\hat{a}$ value based on the robustness estimation accuracy needed, and on the time available for robustness analysis.

Algorithm 1 describes how the approximation method works. It recursively considers two alteration levels x_A and x_B in $[L_A, U_A]$, and evaluates the accuracy of the model in the two points, using the *getAccuracy* function which applies the selected level of alteration (lines 2-3) to the test set TS.

Then, the algorithm checks, using the function *parabIntsct* (line 4), whether at least one of the two parabolas passing for the two points intersects the threshold. A sufficient condition is that the two accuracy values are opposite w.r.t. the threshold Θ (see an example in Fig. 4(a)): this is checked at line 11. If this is not the case, i.e., both accuracy values are above or under the threshold Θ (see examples in Figs. 4(b)-4(c)), the algorithm computes the parabolas passing for A and B and having concavity depth $\pm\hat{a}$, using the *parabola* function (line 14). Parabola coefficients b and c are obtained with this

Algorithm 1 ASAP: Adaptive SAmpling by Parabolic estimation

```
Require: $x_A$ the first alteration level
Require: $x_B$ the second alteration level
Require: $TS$ the test set including all the input data (e.g. images)
Require: $\Theta$ the threshold to be used for robustness analysis
Require: $\hat{a}$ the concavity depth parameter to be used by ASAP
Require: $minStep$ the minimum step between two alteration levels
Require: $C$ the CNN to be analyzed
Ensure: $RES$ the list of sampled points with their accuracy values
 1: procedure EVAL($x_A$, $x_B$, $TS$, $\Theta$, $\hat{a}$, $minStep$, $RES$, $C$)
 2:     $acc_A \leftarrow getAccuracy(TS, x_A, C, RES)$    ▷ Get the accuracy in $A$
 3:     $acc_B \leftarrow getAccuracy(TS, x_B, C, RES)$    ▷ Get the accuracy in $B$
        ▷ Check whether the parabolas with concavity depth $\pm\hat{a}$ intersect $\Theta$
 4:     $intersected \leftarrow parabIntsct(x_A, acc_A, x_B, acc_B, \hat{a}, \Theta) \vee$
                $parabIntsct(x_A, acc_A, x_B, acc_B, -\hat{a}, \Theta)$
        ▷ Estimate accuracy in the two sub-intervals if they are not too close
 5:     if $intersected \wedge x_B - x_A \geq minStep$ then
 6:         EVAL($x_A$, $\frac{x_A+x_B}{2}$, $TS$, $\Theta$, $\hat{a}$, $minStep$, $RES$, $C$)
 7:         EVAL($\frac{x_A+x_B}{2}$, $x_B$, $TS$, $\Theta$, $\hat{a}$, $minStep$, $RES$, $C$)
 8:     end if
 9: end procedure

10: function $parabIntsct(x_A, acc_A, x_B, acc_B, a, \Theta)$
        ▷ Check whether $A$ and $B$ are opposite w.r.t. $\Theta$
11:     if $(acc_A - \Theta) \cdot (acc_B - \Theta) < 0$ then
12:         return $true$
13:     end if
        ▷ Compute the parabola and its vertex
14:     $b, c \leftarrow parabola(x_A, acc_A, x_B, acc_B, a)$
15:     $(x_v, y_v) = (-\frac{b}{2 \cdot a}, -\frac{b^2 - 4 \cdot a \cdot c}{4 \cdot a})$
16:     return $(x_A \leq x_v \leq x_B) \wedge ((acc_A - \Theta) \cdot (y_v - \Theta) < 0)$
17: end function

18: function $getAccuracy(TS, x, C, RES)$
19:     if $\neg RES.contains(x)$ then
20:         $acc_x \leftarrow ComputeAccuracy(TS, x, C)$
21:         $RES.append(\langle x, acc_x \rangle)$    ▷ save the obtained results
22:     else
23:         $acc_x \leftarrow RES.get(x)$
24:     end if
25:     return $acc_x$
26: end function
```

system of equations (where a is passed as argument):

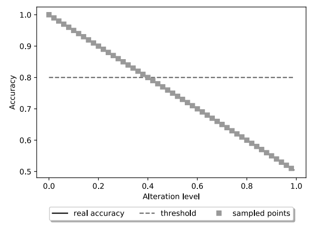
$$\begin{cases} a \cdot {x_A}^2 + b \cdot x_A + c = acc_A \\ a \cdot {x_B}^2 + b \cdot x_B + c = acc_B \end{cases}$$

Then, the parabola vertex $V(x_v, y_v)$ is computed (line 15). The method verifies that the parabola is in the area of interest (line 16), by checking that : (i) $x_v \in [x_A, x_B]$ (first operand of the conjunction), and (ii) the parabola intersects the threshold Θ, i.e., y_v is opposite to acc_A w.r.t. Θ (second operand).

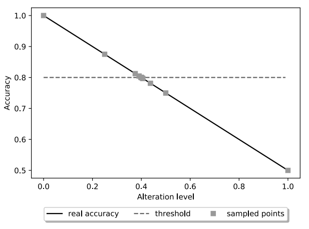
Finally, if one of the parabolas intersects the threshold and the sampled points are not too close (line 5), the computation is recursively repeated in the intervals $[x_A, x_M]$ and $[x_M, x_B]$ (lines 6-7), where x_M is the average alteration level between x_A and x_B.

Starting from the set of computed points with their accuracy values $RES = \{\langle l_1, acc_1\rangle, \ldots, \langle l_n, acc_n\rangle\}$, the robustness is computed by generalizing the formula in Def. 3 as follows:

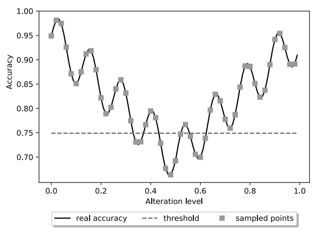
$$rob_A(C, P) = \frac{\sum_{j=2}^{n} H(acc_j - \Theta) \cdot (l_j - l_{j-1})}{U_A - L_A}$$



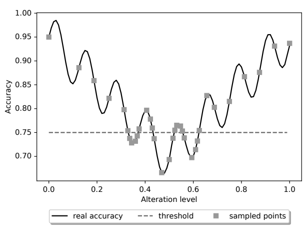
(a) Uniform sampling – Linear func



(b) ASAP – Linear func



(c) Uniform sampling – Irregular func



(d) ASAP – Irregular func

Fig. 5. Examples of robustness computation using synthetic functions

Example 1. Fig. 5 shows the effect of ASAP on two different synthetic functions. In Fig. 5(a), the robustness is computed by using uniform sampling with 50 equidistributed alteration levels, obtaining a robustness of 38.8%, while in Fig. 5(b) only 15 levels are used by ASAP (while $\hat{a}$ equal 256) and a robustness of 39.06% is obtained. Note that the two results are very close and that the real robustness associated with the accuracy plot shown in Fig. 5(a) and Fig. 5(b) is 40.00%. The same behavior can be observed between Fig. 5(c), where robustness 77.9% is obtained by uniform sampling with $n = 50$, and Fig. 5(d), where ASAP uses only 34 alteration levels (focused in the area near the threshold value instead of uniformly distributed ones), obtaining a robustness of 81.2%. Note that, also in this case, the two results are close, and that the real robustness associated with the accuracy plot shown in Fig. 5(c) and Fig. 5(d) is 80.4%. This shows that ASAP uses fewer alteration levels, so it saves time, but it still provides an accurate approximation of the robustness.

A. Maximum error estimation of the computed robustness

Our method provides theoretical guarantees regarding the maximum error that it can do in computing the robustness. To define this, we first need to select from RES (which contains every i-th sampled point $p_i = \langle l_i, acc_i \rangle$) the pairs of two consecutive points p_j and p_{j+1} such that the parabolas passing from them with concavity depth $\pm\hat{a}$ intersect the threshold Θ, i.e.,

$$IP = \left\{ (l_j, l_{j+1}) \;\middle|\; \begin{matrix} \langle l_j, acc_j \rangle, \langle l_{j+1}, acc_{j+1} \rangle \in RES \wedge \\ \begin{pmatrix} parabIntsct(l_j, acc_j, l_{j+1}, acc_{j+1}, \hat{a}, \Theta) \vee \\ parabIntsct(l_j, acc_j, l_{j+1}, acc_{j+1}, -\hat{a}, \Theta) \end{pmatrix} \end{matrix} \right\}$$

Intuitively, a pair of points p_j and p_{j+1} in IP identifies the points between which at least one of the two parabolas intersects the threshold, and, therefore, also the real curve may intersect, but ASAP has quit sampling because the two points have alteration levels sufficiently close ($l_{j+1} - l_j < minStep$).

Assuming that we used an appropriate value $\hat{a}$ for ASAP, the error we can do in computing the robustness only comes from the intervals identified by the points in IP. This intuition is formalized in the following theorem.

Theorem 1. Let C be a CNN and A a given alteration type defined in the range $[L_A, U_A]$. Let rob_A be the robustness computed for C and A by ASAP using a given $\hat{a}$. Let rob_A^O be the *real* robustness value. Under the assumption that $\hat{a}$ is a suitable parameter, i.e., the real accuracy curve is included in the areas of two parabolas with concavity depth $\hat{a}$ (see Fig. 3), the maximum error of the computed robustness has a guaranteed upper bound defined as follows:

$$|rob_A - rob_A^O| \leq \varepsilon_A \quad \text{with } \varepsilon_A = \frac{\sum\limits_{(l_j, l_{j+1}) \in IP} (l_{j+1} - l_j)}{|U_A - L_A|} \tag{2}$$

Proof. The error in robustness computation is due to the cases in which the real curve crosses the threshold line but ASAP fails to find the exact intersection point. Let's consider where this can happen by considering all the sub-intervals $[l_j, l_{j+1}]$ of the points in RES:

- if $(l_j, l_{j+1}) \notin IP$, then the parabolas with concavity depth $\pm\hat{a}$ do not intersect the threshold. Since, by theorem assumption, $\hat{a}$ is a suitable parameter, the real curve is included in the computed parabolas, and so it also does not intersect the threshold. So, no contribution of error in robustness computation comes from these points.
- if $(l_j, l_{j+1}) \in IP$, then we can distinguish two cases:
  - the two points are opposite w.r.t. the threshold line. So, the real curve does intersect the threshold line, but in an unknown point which does not belong to RES.
  - the two points are both below or above the threshold: ASAP ignores the possible intersection of the real curve with the threshold line, since for ASAP the sampled points are close enough.

  In both cases, the maximum absolute error is $l_{j+1} - l_j$.

So, the total error in robustness is given by the sum of the errors for all the pairs of points in IP. Hence, the upper bound of the error is as defined in Eq. 2. $\square$

IV. Experiments Design

Benchmarks

We have chosen the following two different benchmarks to evaluate the efficiency of ASAP.

MNIST Dataset: The MNIST (Modified National Institute of Standards and Technology database) dataset is a well-known dataset containing a lot of images of hand-written number digits [2]. It is shipped with Keras, and we have tested the approaches presented in this paper using the first 1000 images in the test set over a publicly available model [1].

TABLE I
TIME (MINUTES) AND ROBUSTNESS COMPARISON AMONG ORACLE, ASAP, NUMERICAL INTEGRATION, AND UNIFORM SAMPLING WITH THE SAME NUMBER OF POINTS USED BY ASAP (ΔRob [%] AND ΔTIME [%]: DIFFERENCES W.R.T. THE ORACLE VALUES. #P: THE NUMBER OF USED POINTS)

		Oracle		**ASAP**					**Numerical Integration**					**Uniform sampling**				
Alteration	$[U_A, L_A]$	**Time [min]**	**Rob [%]**	**#p**	**Time [min]**	**ΔTime [%]**	**Rob [%]**	**ΔRob$_A$ [%]**	**#p**	**Time [min]**	**ΔTime [%]**	**Rob [%]**	**ΔRob_N [%]**	**#p**	**Time [min]**	**ΔTime [%]**	**Rob [%]**	**ΔRob$_U$ [%]**
MNIST																		
Gaussian Noise	$[0, 1]$	1,638	100.00	34	54	-96.68	100.00	0.00	21	33	-96.91	100.00	0.00	35	55	-94.87	100.00	0.00
Compression	$[0, 1]$	2,365	4.59	17	39	-98.33	4.49	-0.10	441	1021	-37.63	4.59	0.00	18	41	-97.46	5.56	0.97
Vertical Trans.	$[-1, 1]$	1,782	100.00	68	118	-93.35	100.00	0.00	21	36	-98.46	100.00	0.00	69	119	-94.93	100.00	0.00
Horizontal Trans.	$[-1, 1]$	1,546	100.00	65	98	-93.65	100.00	0.00	21	31	-98.22	100.00	0.00	69	104	-94.15	100.00	0.00
Blur	$[0, 1]$	1,086	0.10	16	17	-98.42	0.00	-0.10	21	22	-98.65	0.00	-0.10	16	17	-98.89	6.25	6.15
Brightness	$[-0.5, 0.5]$	1,671	0.10	23	27	-98.42	0.10	0.00	21	34	-98.05	0.00	-0.10	24	39	-97.77	0.04	-0.06
Zoom	$[0, 1]$	1,772	65.85	40	42	-97.64	67.57	1.72	903	945	-46.62	65.92	0.07	41	42	-97.58	65.85	0.00
Breast cancer classification																		
Gaussian Noise	$[0, 1]$	716	43.80	56	39	-94.54	43.26	-0.54	-	timeout	-	-	-	57	39	-94.31	43.86	0.06
Compression	$[0, 1]$	692	100.00	33	22	-97.42	100.00	0.00	21	13	-98.08	100.00	0.00	34	23	-96.78	100.00	0.00
Vertical Trans.	$[-1, 1]$	692	100.00	127	86	-87.56	100.00	0.00	21	14	-97.95	100.00	0.00	128	86	-87.51	100.00	0.00
Horizontal Trans.	$[-1, 1]$	693	100.00	127	86	-87.56	100.00	0.00	21	14	-97.95	100.00	0.00	128	86	-87.50	100.00	0.00
Blur	$[0, 1]$	700	73.66	225	154	-78.01	73.05	-0.61	-	timeout	-	-	-	256	174	-74.77	72.66	-1.00
Brightness	$[-0.5, 0.5]$	693	57.27	49	33	-95.21	57.23	-0.04	861	581	-16.06	56.74	-0.54	52	35	-94.93	57.69	0.42
Zoom	$[0, 1]$	862	100.00	118	92	-88.49	100.00	0.00	21	21	-97.45	100.00	0.00	128	134	-84.34	100.00	0.00

Breast cancer classification: Breast cancer diagnoses, in particular for Invasive Ductal Carcinoma (IDC), are based on the analysis of images of histological features of tissue or cells removed with surgery or biopsy. These images are captured by a microscope and examined by pathologists to make a decision about the benignity or the malignity of the suspected cancer. We have performed a robustness analysis using input images coming from a publicly available dataset curated by [11], and the same model we presented in [4].

Oracle definition by uniform sampling

In order to assess the performance of ASAP, we need to check how close the computed robustness values are to the "correct" values. So, in order to define a suitable *oracle* to perform the necessary comparisons, we have computed the robustness for the case studies using the uniform sampling technique with a very large number of points $n = 1024$ (using the tool ROBY [5]). Tab. I reports the alterations applied to both benchmarks, together with the robustness results and the required time of the oracle, when using accuracy threshold $\Theta = 0.8$. Note that the results obtained in this way can be reasonably considered as oracle: indeed, we have observed that, with $n = 1024$, the accuracy function is very stable and the alteration step becomes between 10^{-3} and $2{\cdot}10^{-3}$, which is small enough for the defined alterations.

V. EXPERIMENTAL EVALUATION

To evaluate ASAP, we have performed several experiments on the benchmarks presented in Sect. IV. We have applied it with n=1024 and $\hat{a}$ =128 using the proposed alterations. Moreover, we have also applied, as comparison approaches, numerical integration, and uniform sampling with the same number of points actually used by ASAP.

For each experiment, we have recorded the assessed robustness, the number of points required for the estimation, and the required time (see Tab. I). **ΔTime** is the relative change in percentage of the time taken to compute the robustness by using each analyzed technique w.r.t. the oracle, while **ΔRob** is the difference between the robustness values. Negative values of **ΔTime** indicate a saving of time while a negative value of **ΔRob** means an underestimation of the robustness.

In particular, using the gathered values, we are interested in answering the following research questions:

RQ1 Is ASAP efficient and effective?
RQ2 Is ASAP more effective than:
- **RQ2.1** numerical integration techniques?
- **RQ2.2** uniform sampling with the same number of points?
- **RQ2.3** randomly selecting the alteration levels?

RQ3 Is the number of alteration levels actually used by ASAP correlated with the maximum number of points allowed (n)?
RQ4 How do the values of parameters n and $\hat{a}$ influence the accuracy of the robustness estimation?
RQ5 Is the estimation error provided by Theorem 1 reliable?

RQ1: Is ASAP efficient and effective?

Each alteration entails different accuracy curve shapes for which ASAP may lead to different results in terms of time saving and error in the robustness estimation w.r.t. the oracle. From Tab. I, it can be seen that a significant reduction of the number of points is obtained with both benchmarks and all the alterations. Moreover, the flatter the accuracy alteration curve is, the higher is the decrease in time using ASAP, since it uses fewer points than the oracle and it is more probable that the computed parabolas with the chosen $\hat{a}$ do not intersect the threshold. In terms of accuracy of the robustness estimation, compared to the oracle, the error is always under 2%. In conclusion, ASAP always allows estimating the robustness with

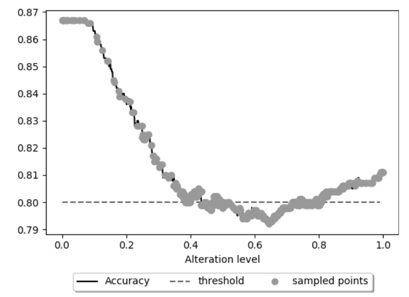


Fig. 6. RQ2.1 – Points automatically sampled by a numerical integration method for robustness computation (for the breast cancer classification benchmark and the blur alteration)

a consistent saving in time and with a really low estimation error.

RQ2.1: Is ASAP more effective than numerical integration techniques?

In Def. 2, we have presented the general formula for computing robustness using an integral computation of the Heaviside function. One could apply numerical integration techniques that are able to automatically sample the points where to compute the accuracy for computing the integral. For instance, Fig. 6 shows an example of the sampling performed by a numerical integration technique. However, these techniques may not find the right alteration levels and so correctly compute the integral, since the Heaviside function is discontinuous and it hides the accuracy curve as its argument. For this reason, we wanted to assess the feasibility and efficiency of numerical integration techniques to compute the robustness, and compare their performance with ASAP.

We have applied the `scipy.integrate.quad` function which uses adaptive quadrature and allows to set the maximum number of points in which function to be integrated can be evaluated. We have set this number to 1024, in order to allow a fair comparison with ASAP and the oracle.

Tab. I reports the results obtained with the experiments. The cases in which the numerical algorithm cannot compute the integral with the given maximum number of points are considered `timeout`. For the breast cancer benchmark, two alterations gave `timeout` as result. When comparing the results of numerical integration with the ones obtained by ASAP, we can see that, for the breast cancer classification, ASAP always gives an equal or more precise robustness estimation. On the other hand, for the MNIST benchmark, two values of robustness are better predicted by numerical integration, i.e., those for compression and zoom alterations; however, ASAP uses 96% fewer values for the estimation (17 vs 441 and 40 vs 903) and, so, has better performance in terms of the required time.

RQ2.2: Is ASAP more effective than uniform sampling with the same number of points?

One of the main advantages of ASAP is that the method is able to select itself few points, those that are judged relevant, depending on the parameters n and $\hat{a}$. However, users may choose to use the uniform sampling and the same number of points actually used by ASAP (if they knew that number). In order to discover the actual advantage of the proposed approach (in terms of robustness estimation error), we have tried to use the same number of points (or an upper bound, when it is not possible to distribute them uniformly because, for example, the alteration levels are integers) with the uniform sampling approach. Tab. I reports the results we have obtained. In particular, ΔRob_A and ΔRob_U represent the difference between the estimation obtained using the analyzed method (respectively, ASAP and uniform sampling) and the one provided by the oracle. The obtained results highlight that using ASAP always leads to better results, i.e., $|\Delta Rob_A| < |\Delta Rob_U|$, except for the Zoom alteration in the MNIST benchmark, and for the Gaussian Noise in the Breast Cancer classification.

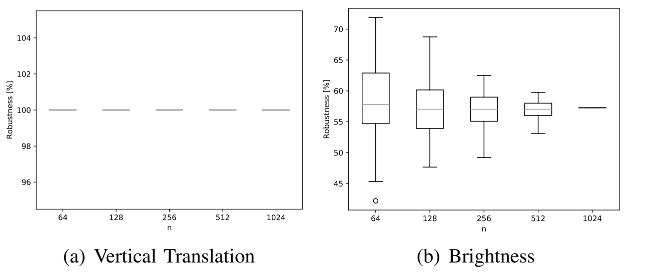


Fig. 7. RQ2.3 – Random sampling for the breast cancer classification benchmark

RQ2.3: Is ASAP more effective than randomly selecting the alteration levels?

Evaluating the robustness using as few alteration levels as possible is of key importance to speed up the estimation process. A possible solution to reduce the number of evaluated alteration levels could be using random sampling and fixing the number of points to be evaluated. We have applied it to the breast cancer benchmark, using different values of n, and repeating each experiment for 100 times. Fig. 7 reports the results for vertical translation and brightness alterations over the analyzed benchmark. The robustness reached applying the vertical translation (see Fig. 7(a)) is 100%. In fact, the variance of the estimation is 0, even when using fewer points. Instead, for the brightness alteration shown in Fig. 7(b), the use of fewer random sampled points leads to a great variance. A similar trend can be observed for all the other alterations. We can conclude that the performance of the random sampling approach depends directly on the robustness w.r.t. the considered alteration. If the robustness is $\simeq 100\%$ or $\simeq 0\%$, randomly choosing the points in which to compute the accuracy leads to accurate results with low variance. In all other cases, random sampling can imply a high variance and less accurate results.

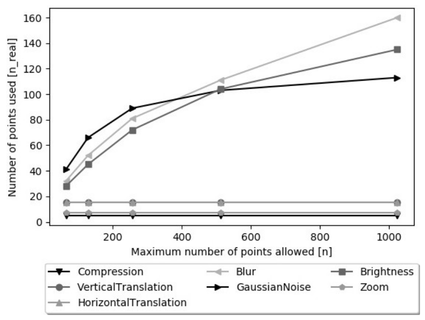


Fig. 8. RQ3 – Relation between n and the actually used points with different alterations, for the breast cancer benchmark with a fixed $\hat{a} = 2$

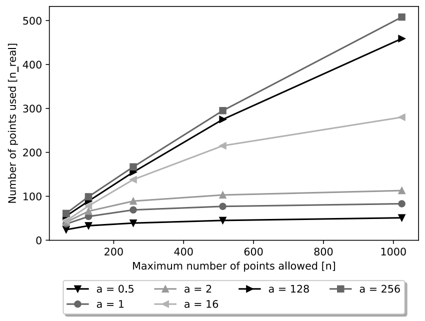


Fig. 9. RQ3 – Relation between n and actually used points for the Gaussian noise alteration in breast cancer benchmark (for different values of $\hat{a}$)

RQ3: Is the number of alteration levels actually used by ASAP correlated with the maximum number of points allowed (n)?

Using ASAP, the user can set the value of n, i.e., the maximum number of points to be used. Increasing its value means that a lower interval $minStep$ (see Alg. 1 for reference) allowed between two adjacent points is used. However, depending on the value of n, and on the $\hat{a}$ fixed, the algorithm may use fewer points than the maximum allowed, since it stops adding points if no intersection between the parabolas and the threshold is found. Thus, we are interested in defining how ASAP reacts to a variation of n in terms of used points. We have analyzed the presented benchmarks and all the alterations using different values of n. We have found that the number of actually used points depends on the alteration and on how the network accuracy curve changes when a defined alteration level is applied: the flatter the curve of accuracy is, the lower the number of points used by ASAP is (Fig. 8). As a rule of thumb, the higher $\hat{a}$ and n are, the higher the number of considered alteration levels is, as shown by Fig. 9, since $minStep$ is lower. Moreover, Fig. 9 shows that the lower $\hat{a}$ is, the lower the advantage of increasing the value of n is since the number of used points for analysis does not increase significantly.

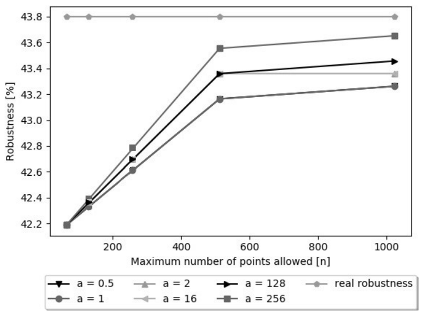


Fig. 10. RQ4 – Relation between n and the robustness estimation for the Gaussian noise alteration in breast cancer benchmark

TABLE II
RQ5 – COMPARISON BETWEEN ASAP ERROR AND UPPER BOUND (ε_A)

	MNIST		**Breast Cancer**	
Alteration	ΔRob_A **[%]**	ε_A	ΔRob_A **[%]**	ε_A
Gaussian Noise	0.00	0.00	-0.54	0.78
Compression	-0.10	0.10	0.00	0.00
Vertical Trans.	0.00	0.00	0.00	0.00
Horizontal Trans.	0.00	0.00	0.00	0.00
Blur	-0.10	0.10	-0.61	7.71
Brightness	0.00	0.00	-0.04	0.10
Zoom	1.72	0.29	0.00	0.00

RQ4: How do the values of n and $\hat{a}$ influence the accuracy of the robustness estimation?

The ASAP method is based on two parameters that are configurable by the user in order to adapt to different networks: n and $\hat{a}$. The former is used to evaluate the distance between two points on the x-axis, i.e., to compute the $minStep$ value. The latter allows to include the evaluation of the distance between points and the threshold Θ. Thus, choosing different values for the two parameters may lead to a different accuracy in the robustness estimation. We have performed several tests on the two benchmarks varying the values of $\hat{a}$ and n, to evaluate their effect on the robustness estimation. The results are shown in Fig. 10. The analysis indicates that the robustness estimation is more accurate with high n and $\hat{a}$. Both the parameters, in fact, contribute to increasing the number of alteration levels evaluated for robustness estimation and, consequently, the accuracy of ASAP.

RQ5: Is the estimation error provided by Theorem 1 reliable?

Theorem 1 gives an upper bound of the possible error done using ASAP, under the assumption that the value of $\hat{a}$ has been chosen correctly. Since for the two benchmarks presented in this paper, we already have the oracle values, we can assess if the error estimations given by the theorem are near to the actual errors. Tab. II shows the error done by ASAP (ΔRob_A) and the upper bound of the error computed by the theorem (ε_A). For almost all of these results, ε_A is very close to $|\Delta Rob_A|$ and $|\Delta Rob_A| < \varepsilon_A$, except for the Zoom alteration on the MNIST benchmark. Since Theorem 1 assures that ε_A is an upper bound for the error under the assumption of the correct choice of $\hat{a}$, having $|\Delta Rob_A| > \varepsilon_A$ means that $\hat{a}$ has not been chosen correctly for that particular case. In fact, increasing $\hat{a}$ and using $\hat{a} = 256$, we have obtained $\Delta Rob_A = 0.03$ which is below the new upper bound $\varepsilon_A = 0.30$.

VI. Related Work

Efficient analysis is an important topic when it comes to neural networks. In fact, training, validation, and testing activities require a lot of time, especially when the models need to be trained and tested over a great number of inputs. A well-known solution adopted for decreasing the test time is the one presented in [7], where the authors have proposed a method to reduce data to train, test, and validate neural networks, which is based on a *stratified sampling* of input data. Other approaches are based on data pre-processing, in order to reduce the dimensionality of input data, such as in [9], where RBF networks are analyzed and a *separability-correlation measure* is introduced to define which inputs are irrelevant for the classification. These approaches are different from the one proposed in this paper, since with ASAP we reduce the sampled points but not the input. Indeed, this is one of the advantages of ASAP, since by not reducing the input data, the user can be sure about testing every possible input feature. Efficient computation of robustness, even if referred to adversarial examples, is performed by several tools such as CNN-Cert [6], using state-of-the-art algorithms (Fast-Lin and CROWN [19]), or PROVEN [17] which exploits CNN-Cert and a probabilistic approach to reduce the time of computing adversarial robustness.

CNNs are often part of more complex Machine Learning based Systems (MLSs) [15] (e.g., autonomous driving); hence, when testing an MLS, also its embedded CNN must be tested. Since MLSs are complex systems, they must be thoroughly tested; however, literature [15] reports that also the cost of testing MLSs is usually high. As future work, we plan to investigate whether ASAP can improve the efficiency of MLS testing, without degrading its effectiveness.

VII. Conclusions

In this paper, we have introduced ASAP, an efficient way to compute the robustness of a CNN w.r.t. unforeseen (but plausible) input modifications. It exploits the reduction of the number of alteration levels, based on the assumption that the accuracy of the model between two points will likely be included in a parabola-delimited region. Users can adapt the approximation method to their networks using two different parameters: the maximum points n to be computed, and the concavity depth $\hat{a}$ of the evaluated parabolas. We have shown that the proposed solution allows estimating the robustness of a CNN in a more efficient way than the standard (based on uniform sampling or numerical integration) and the random sampling approaches, guaranteeing a small error. As future work, we are investigating solutions for the automatic computation of a suitable value for the parameter $\hat{a}$, in order to avoid errors due to a wrong parameterization of ASAP.

References

[1] Handwritten digit recognition with MNIST and Keras. https://github.com/Curt-Park/handwritten_digit_recognition.

[2] The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.

[3] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938, 2018.

[4] P. Arcaini, A. Bombarda, S. Bonfanti, and A. Gargantini. Dealing with robustness of convolutional neural networks for image classification. In *2020 IEEE International Conference On Artificial Intelligence Testing (AITest)*, pages 7–14, 2020.

[5] P. Arcaini, A. Bombarda, S. Bonfanti, and A. Gargantini. ROBY: a tool for robustness analysis of neural network classifiers. In *2021 IEEE 14th International Conference on Software Testing, Validation and Verification (ICST)*, pages 442–447, 2021.

[6] A. Boopathy, T.-W. Weng, P.-Y. Chen, S. Liu, and L. Daniel. CNN-cert: An efficient framework for certifying robustness of convolutional neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:3240–3247, jul 2019.

[7] G. L. Colmenares and R. Perez. A data reduction method to train, test, and validate neural networks. In *Proceedings IEEE Southeastcon '98 'Engineering for a New Era'*, pages 277–280, 1998.

[8] Q. Fu, F. Yang, J. Zhao, X. Yang, T. Xiang, G. Huai, J. Zhang, L. Wei, S. Deng, and H. Yang. Bioinformatical identification of key pathways and genes in human hepatocellular carcinoma after CSN5 depletion. *Cellular Signalling*, 49:79–86, sep 2018.

[9] X. Fu and L. Wang. Data dimensionality reduction with application to simplifying RBF network structure and improving classification performance. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(3):399–409, 2003.

[10] H. Fujiyoshi, T. Hirakawa, and T. Yamashita. Deep learning-based image recognition for autonomous driving. *IATSS Research*, 43(4):244–252, dec 2019.

[11] A. Janowczyk and A. Madabhushi. Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases. *Journal of Pathology Informatics*, 7(1):29, July 2016.

[12] R. Mangal, A. V. Nori, and A. Orso. Robustness of neural networks: A probabilistic and practical approach. In *Proceedings of the 41st International Conference on Software Engineering: New Ideas and Emerging Results*, ICSE-NIER '19, pages 93–96, Piscataway, NJ, USA, 2019. IEEE Press.

[13] D. Marijan, A. Gotlieb, and M. K. Ahuja. Challenges of testing machine learning based systems. In *2019 IEEE International Conference On Artificial Intelligence Testing (AITest)*, pages 101–102, April 2019.

[14] W. Rawat and Z. Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, 29(9):2352–2449, sep 2017.

[15] V. Riccio, G. Jahangirova, A. Stocco, N. Humbatova, M. Weiss, and P. Tonella. Testing machine learning based systems: a systematic mapping. *Empir. Softw. Eng.*, 25(6):5193–5254, 2020.

[16] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Conference Track Proceedings*, 2014.

[17] L. Weng, P.-Y. Chen, L. Nguyen, M. Squillante, A. Boopathy, I. Oseledets, and L. Daniel. PROVEN: Verifying robustness of neural networks with a probabilistic approach. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6727–6736. PMLR, 09–15 Jun 2019.

[18] X. Xie, L. Ma, F. Juefei-Xu, M. Xue, H. Chen, Y. Liu, J. Zhao, B. Li, J. Yin, and S. See. DeepHunter: A coverage-guided fuzz testing framework for deep neural networks. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2019, pages 146–157, New York, NY, USA, 2019. ACM.

[19] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel. Efficient neural network robustness certification with general activation functions. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[20] J. M. Zhang, M. Harman, L. Ma, and Y. Liu. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering*, pages 1–1, 2020.