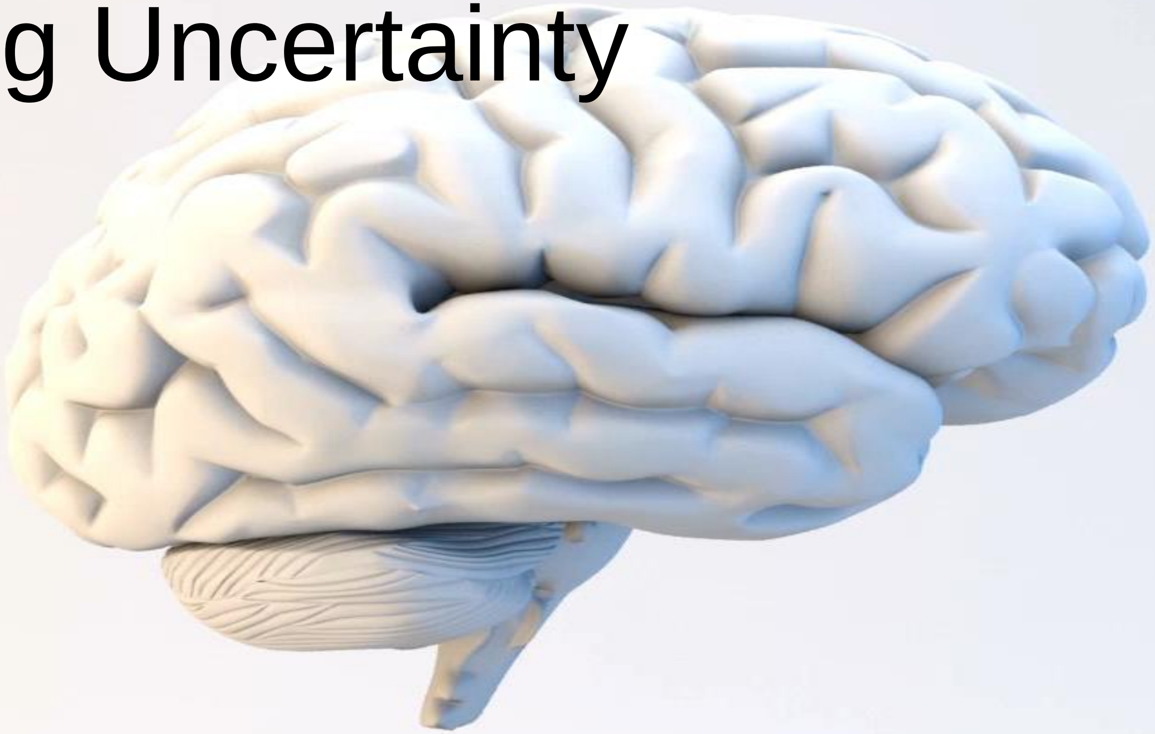
Evaluating the Robustness of Bayesian Neural Networks By Exploiting Uncertainty Estimation



Wasim Essbai – University of Bergamo, Italy

Advisor:

Prof. Angelo Gargantini

Motivations

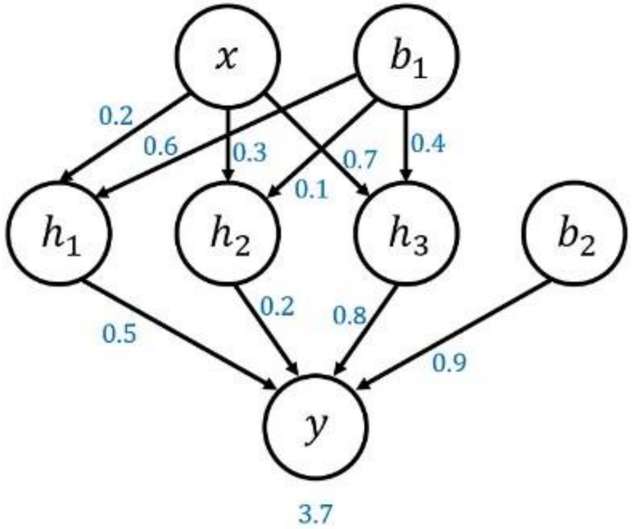
- Machine Learning Models and especially Neural Networks (NNs) are used in data classification, also for performing **critical** tasks
  - Normally used as **black box**
  - Extensive **learning** can be applied
  - But what happens when data are **altered** due to some faults/errors in the image acquisition process?
  - Not only **adversarial examples**
  - **Testing NNs**
- What is their **robustness?**
- See 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)

Outline

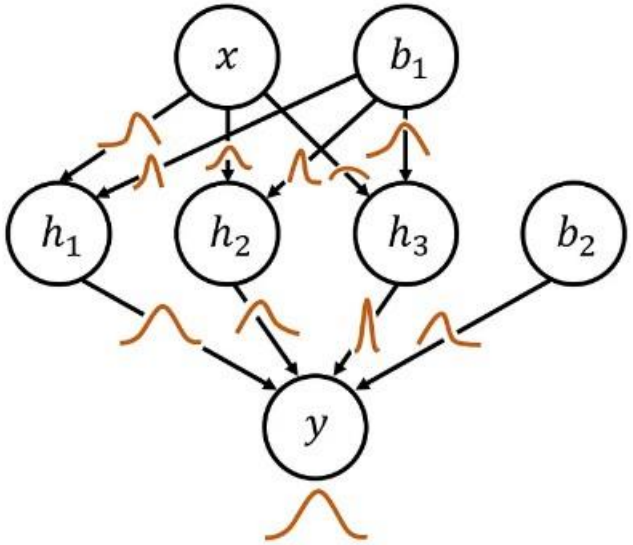
1. Bayesian Neural Networks (BNNs)
2. Classification using uncertainty
3. Alterations
4. Robustness
5. Lesson learned

Bayesian Neural Networks

A standard network



Bayesian network



Training based on Bayes’ Theorem

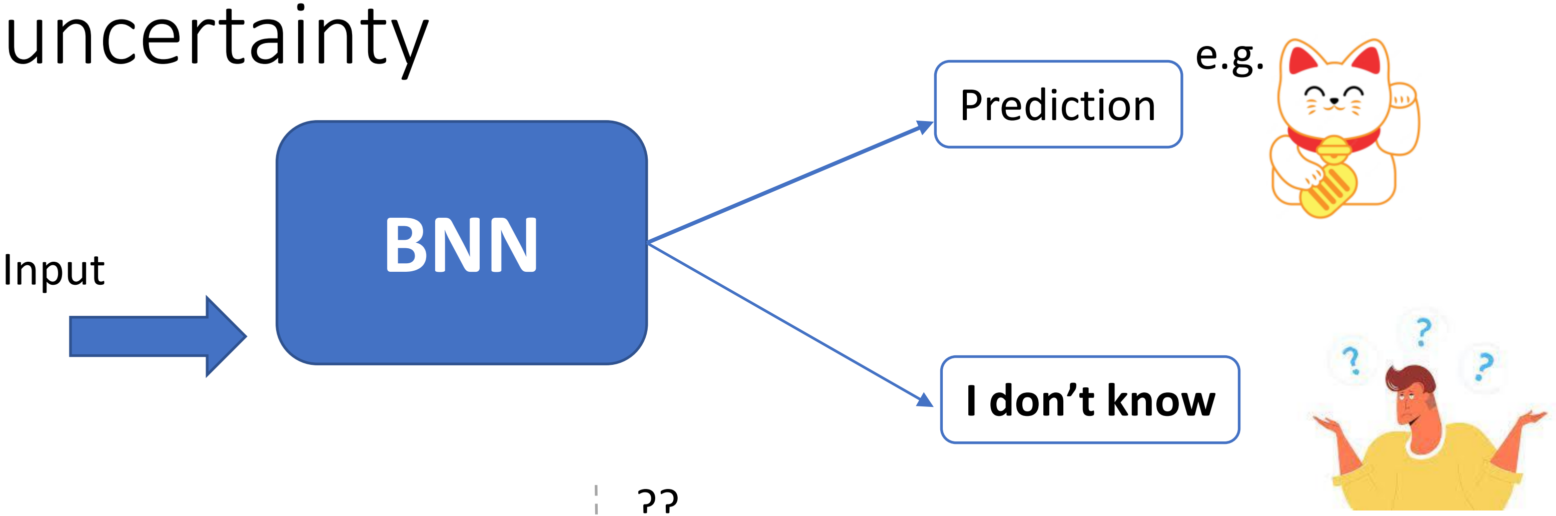
$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{1.1}$$

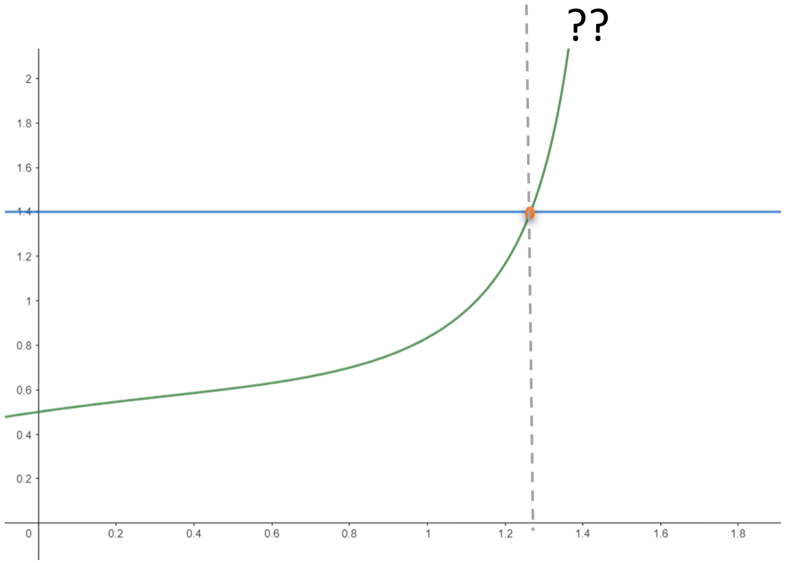
Why?

Uncertainty estimation!

- Aleatoric: Related to the inputs
- Espistemic: Related to the model

Classification using uncertainty





Idea:
Whenever the uncertainty surpasses a defined threshold the network outputs an unknown statement.

How to define the threshold?

- A possible choice is a linear approximation

$$threshold = maxUnc(1 - conf\_level)$$

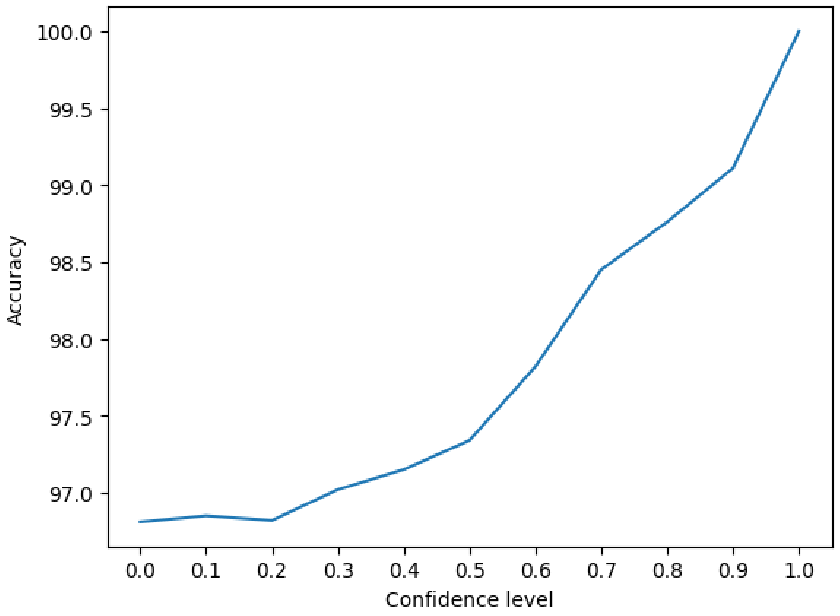
- The accuracy becomes

$$acc(C, D) = \frac{\#correctClassifications}{|D| - \#unknows}$$

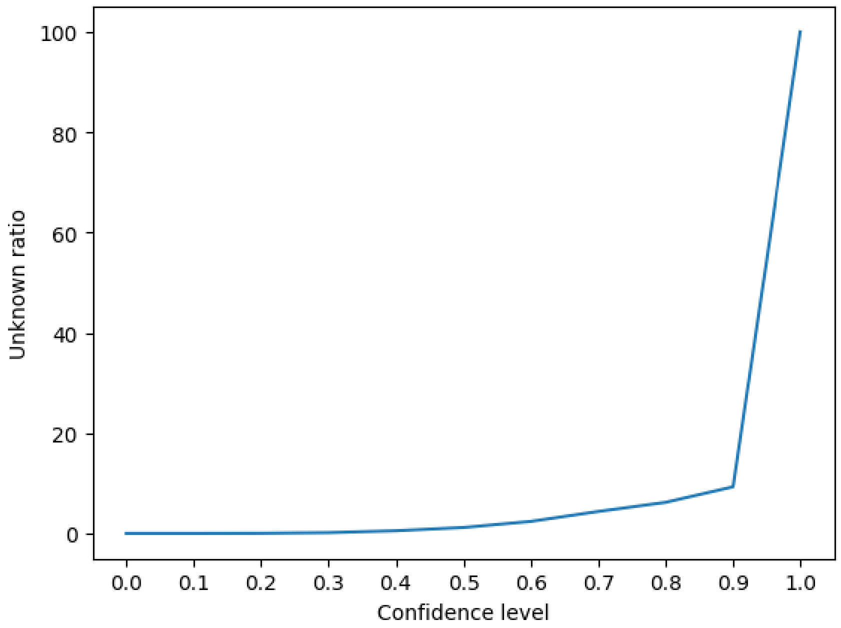
Aleatoric uncertainty

$$aleatoric = diag\{\frac{1}{T}\sum_{t=1}^{T}[diag(\hat{p}_t) - \hat{p}_t^{\otimes 2}]\}$$ → $$threshold = 0.9(1 - conf\_level)$$

Accuracy using aleatoric uncertainty



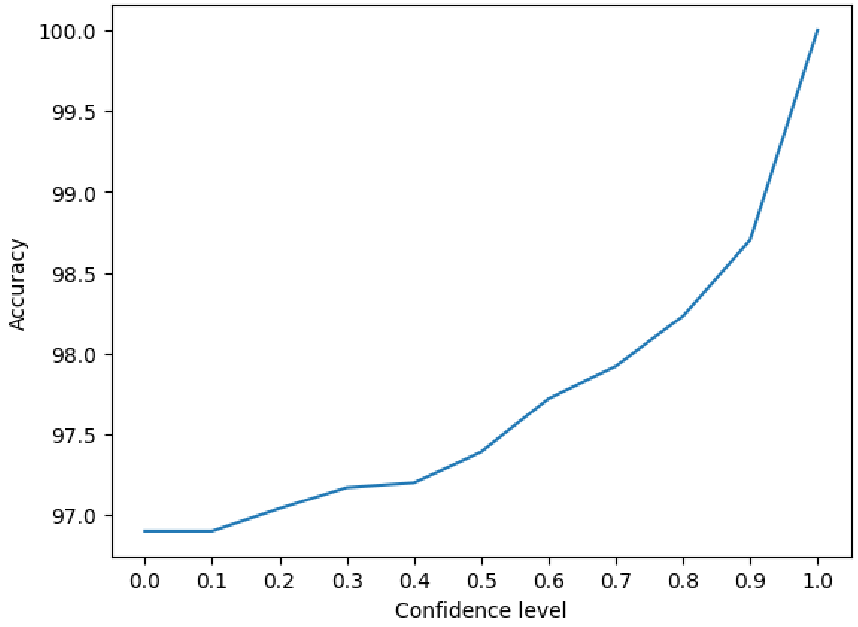
Unknown ration using aleatoric uncertainty



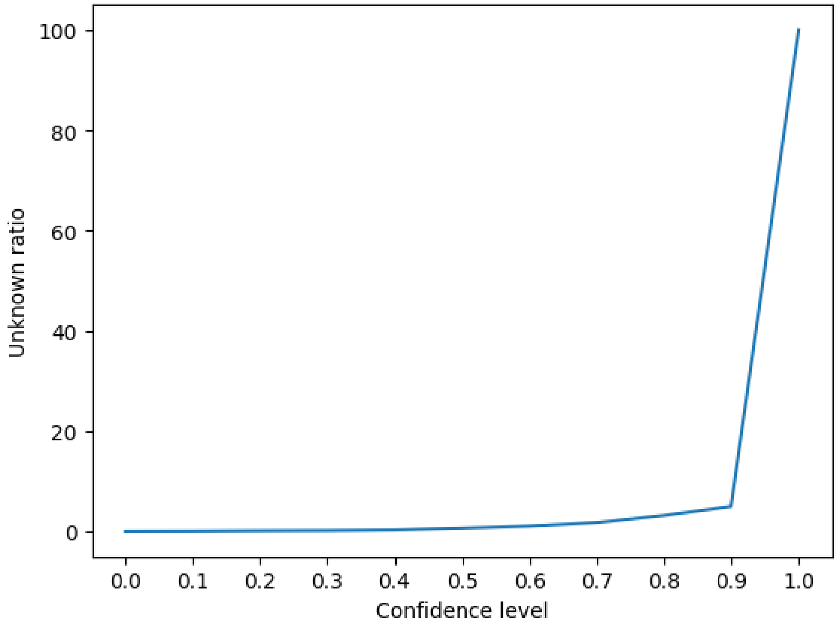
Epistemic uncertainty

$epistemic = diag\{\frac{1}{T}\sum_{t=1}^{T}[\hat{p}_t - \bar{p}]^{\otimes 2}\}$ → $threshold = 0.2(1 - conf\_level)$

Accuracy using epistemic uncertainty



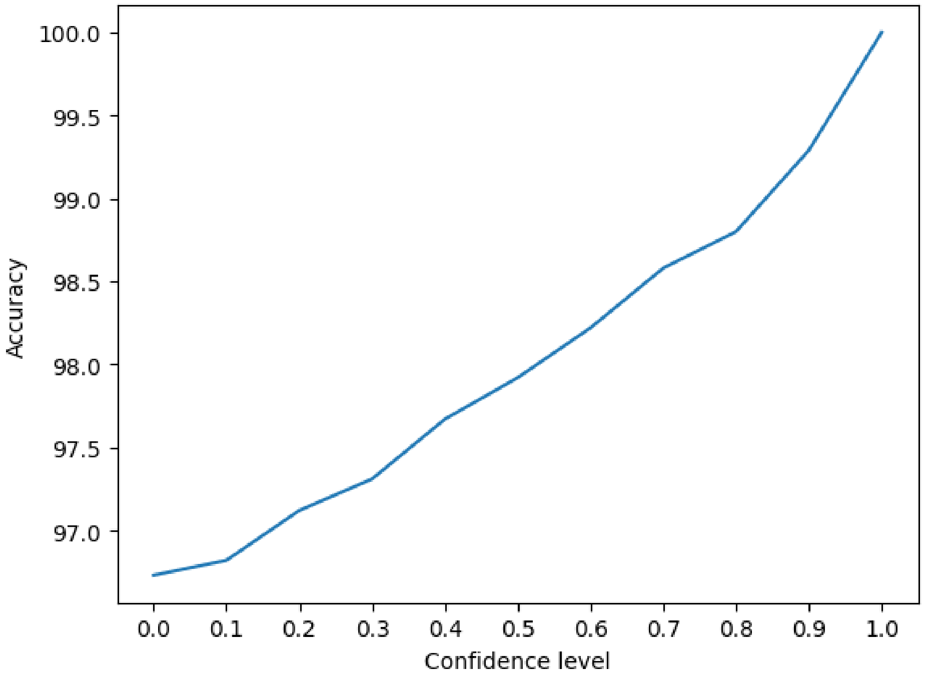
Unknown ration using epistemic uncertainty



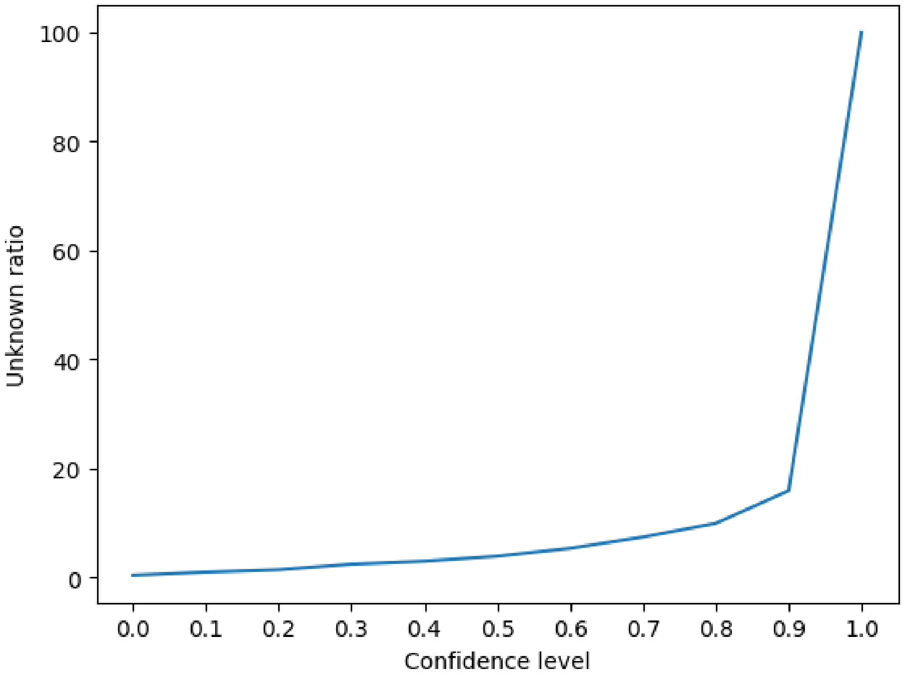
Standard deviation

$$Var[Y] = E[(Y - E[Y])^2] = \sum_{i=0}^{9} (i - E[Y])^2 \cdot p_i = \sigma^2 \Rightarrow threshold = 2.87(1 - conf\_level)$$

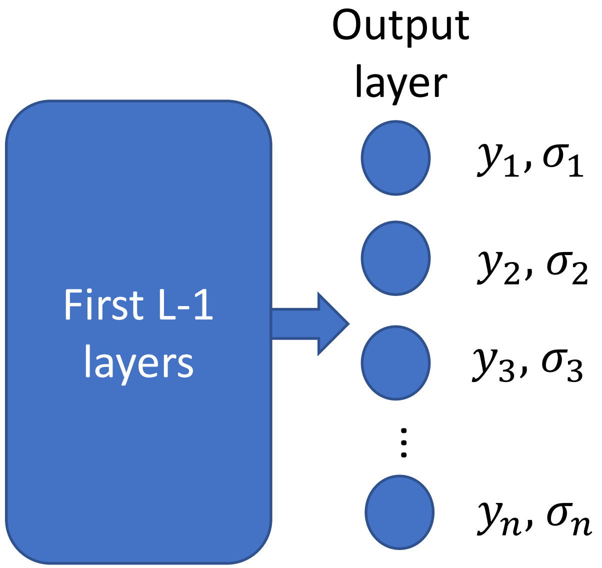
Accuracy using the standard deviation



Unknown ration using the standard deviation

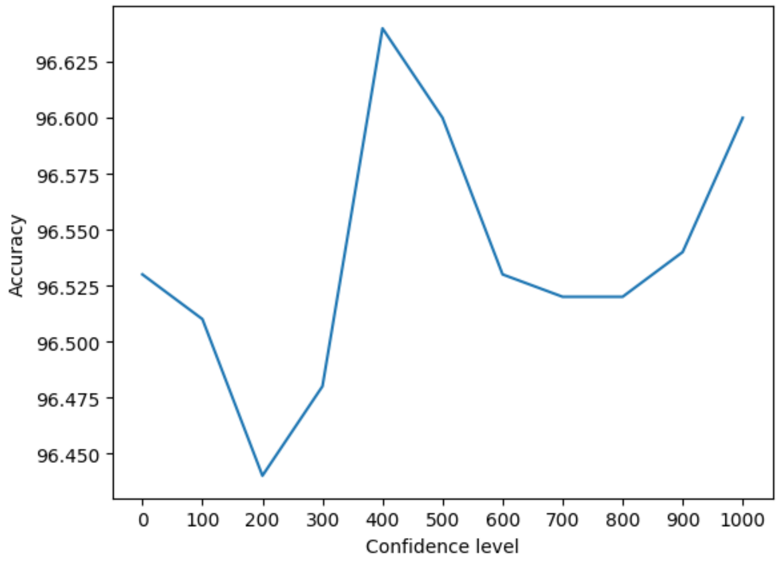


Uncertainty penalization

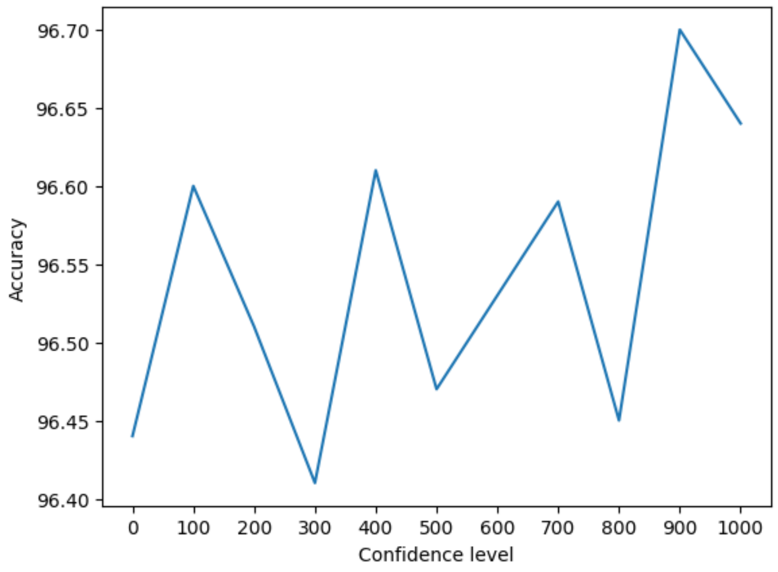


$$\widetilde{y_i} = \frac{y_i}{\sigma_1}$$

Accuracy using epistemic uncertainty penalization

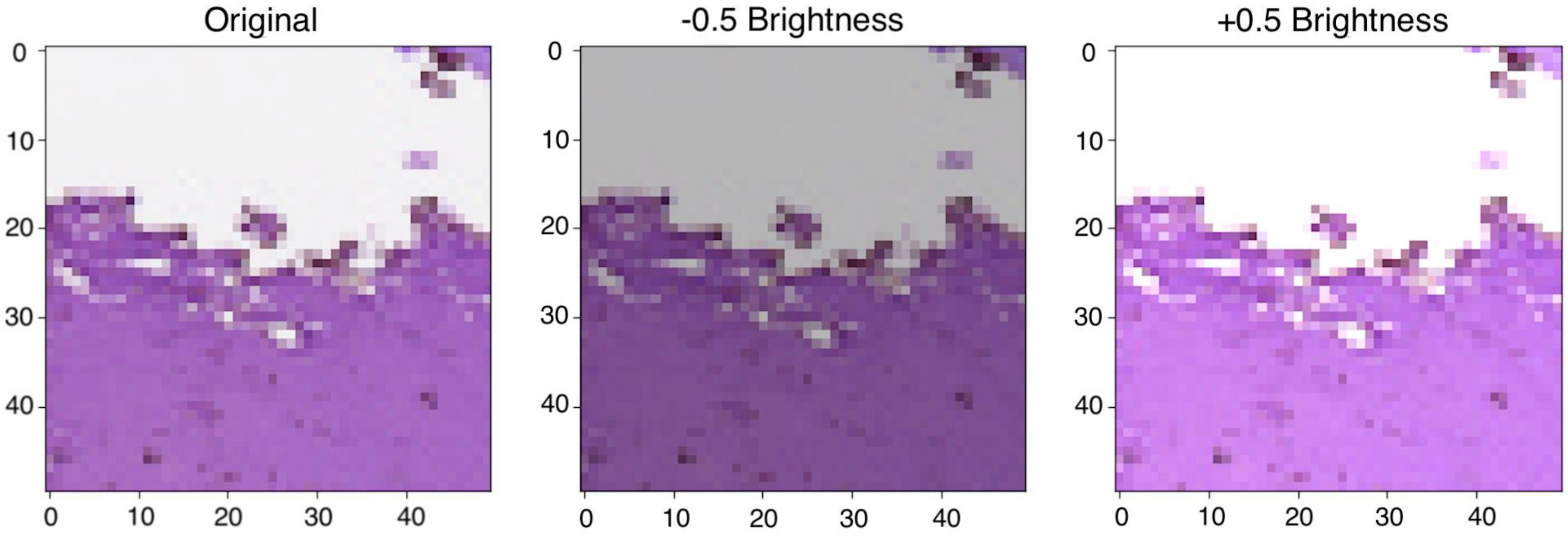


Accuracy using aleatoric uncertainty penalization



Alterations

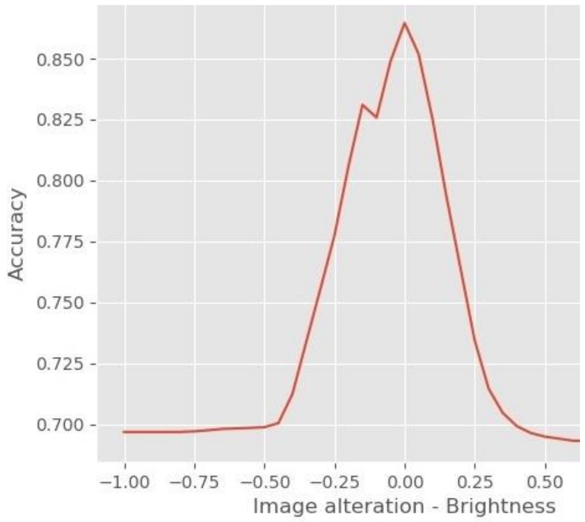
- An **alteration** of type A is a transformation of the input that mimics the possible effects over the inputs when a problem occurs
- Given an alteration of type A, there is a **range** of plausible alterations of type A
  - Example: increase the brightness +- 10%, blur…
- Alterations are defined by **a domain expert**

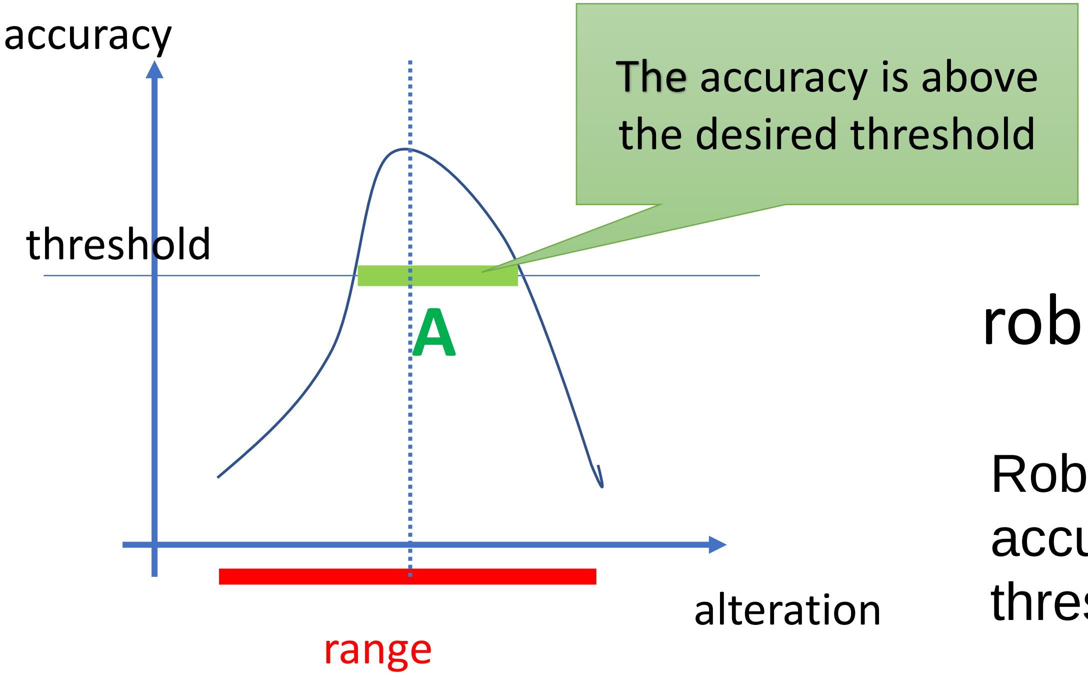


The same concept applies to other types of data: **sounds**, ..

Robustness(1)

- By altering the data, the accuracy (% of samples correctly labeled) decreases
- The accuracy should not go below a given threshold



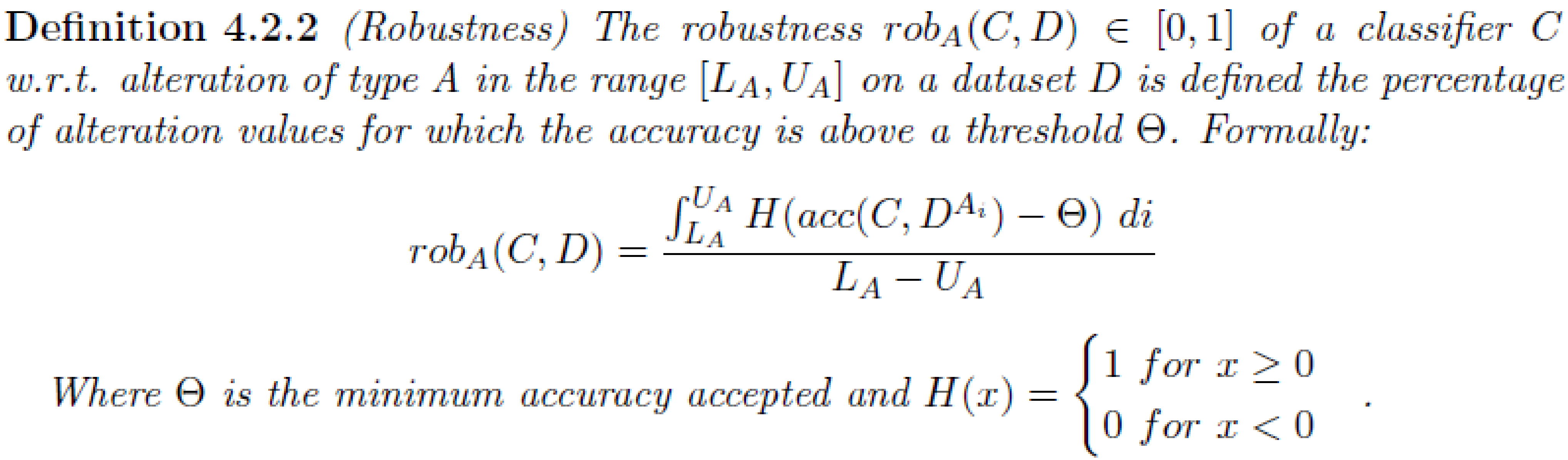


The accuracy is above the desired threshold

$$\text{robustness} = \frac{A}{range}$$

Robustness = 1 means that the accuracy is always above the threshold

Robustness (2)



Definition 4.2.2 *(Robustness) The robustness $rob_A(C, D) \in [0, 1]$ of a classifier C w.r.t. alteration of type A in the range $[L_A, U_A]$ on a dataset D is defined the percentage of alteration values for which the accuracy is above a threshold Θ. Formally:*

$$rob_A(C, D) = \frac{\int_{L_A}^{U_A} H(acc(C, D^{A_i}) - \Theta)\ di}{L_A - U_A}$$

Where Θ is the minimum accuracy accepted and $H(x) = \begin{cases} 1\ for\ x \geq 0 \\ 0\ for\ x < 0 \end{cases}$.

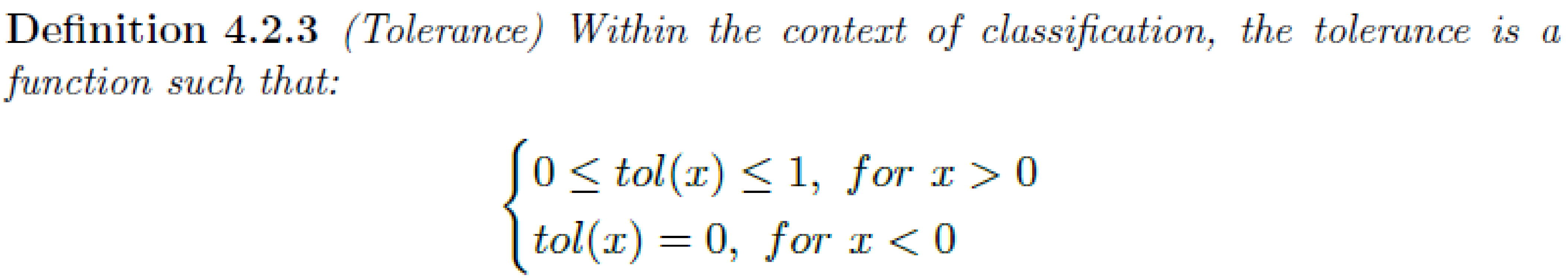
Limitations?

- It does not measure how much the accuracy is above the threshold

→

We can use a different function, but which one?

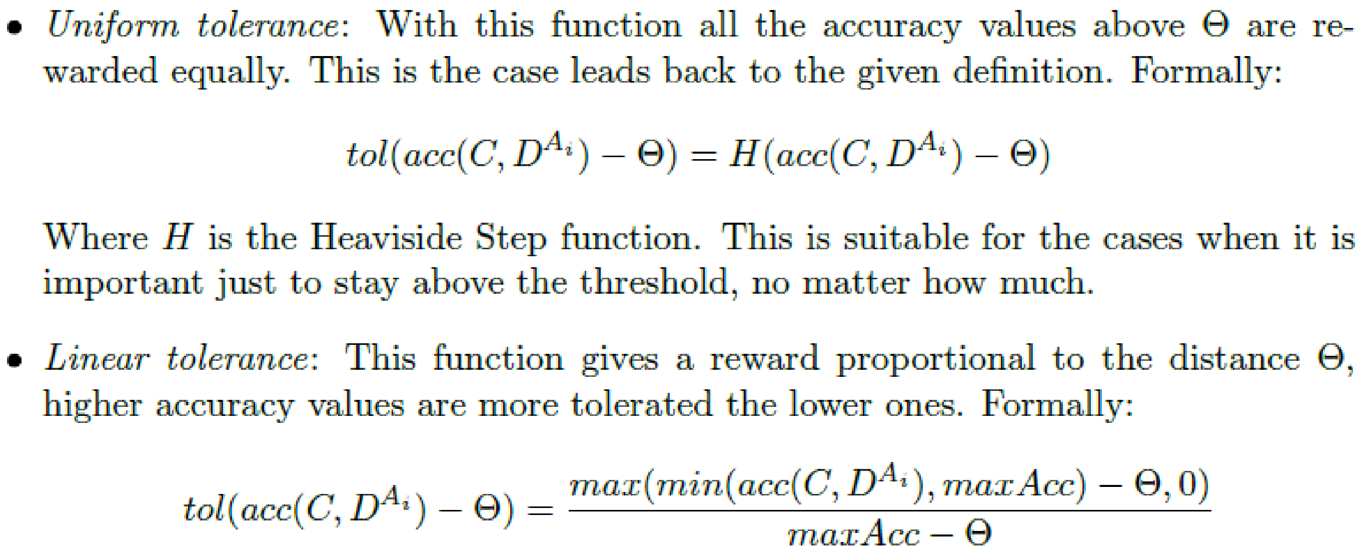
Tolerance



Definition 4.2.3 *(Tolerance) Within the context of classification, the tolerance is a function such that:*

$$\begin{cases} 0 \leq tol(x) \leq 1, \ for \ x > 0 \\ tol(x) = 0, \ for \ x < 0 \end{cases}$$

Some examples:



- *Uniform tolerance*: With this function all the accuracy values above Θ are rewarded equally. This is the case leads back to the given definition. Formally:

  $$tol(acc(C, D^{A_i}) - \Theta) = H(acc(C, D^{A_i}) - \Theta)$$

  Where H is the Heaviside Step function. This is suitable for the cases when it is important just to stay above the threshold, no matter how much.

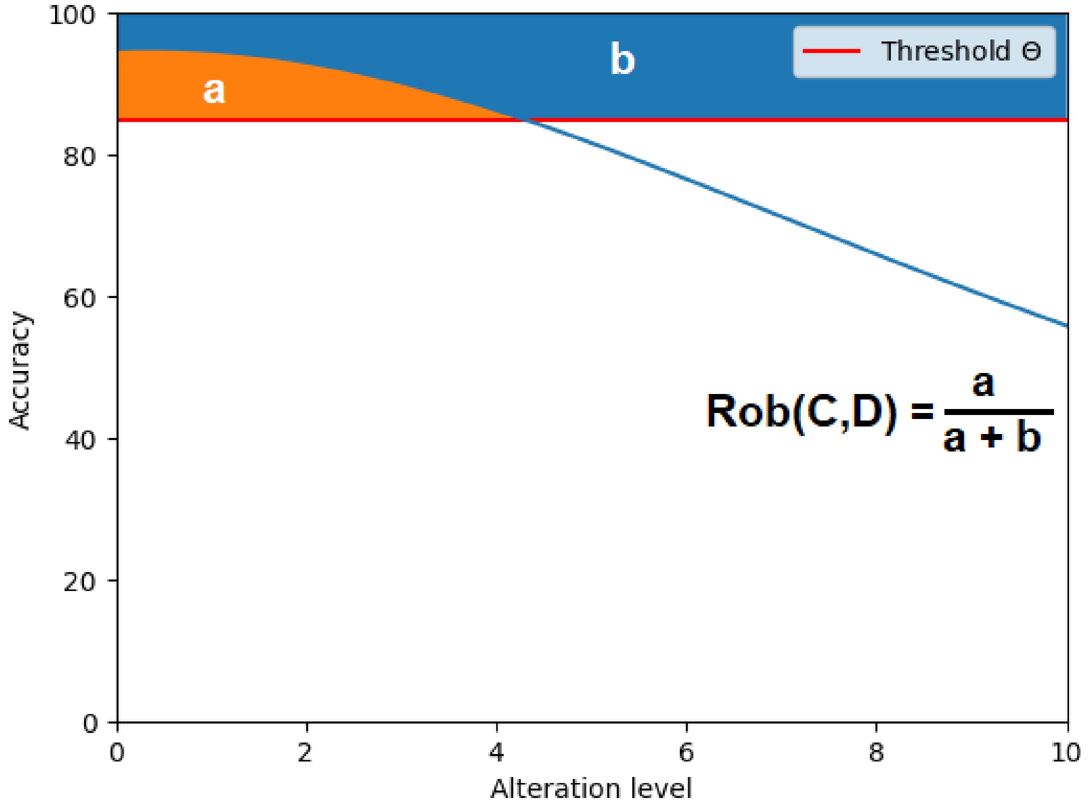
- *Linear tolerance*: This function gives a reward proportional to the distance Θ, higher accuracy values are more tolerated the lower ones. Formally:

  $$tol(acc(C, D^{A_i}) - \Theta) = \frac{max(min(acc(C, D^{A_i}), maxAcc) - \Theta, 0)}{maxAcc - \Theta}$$

Robustness (3)

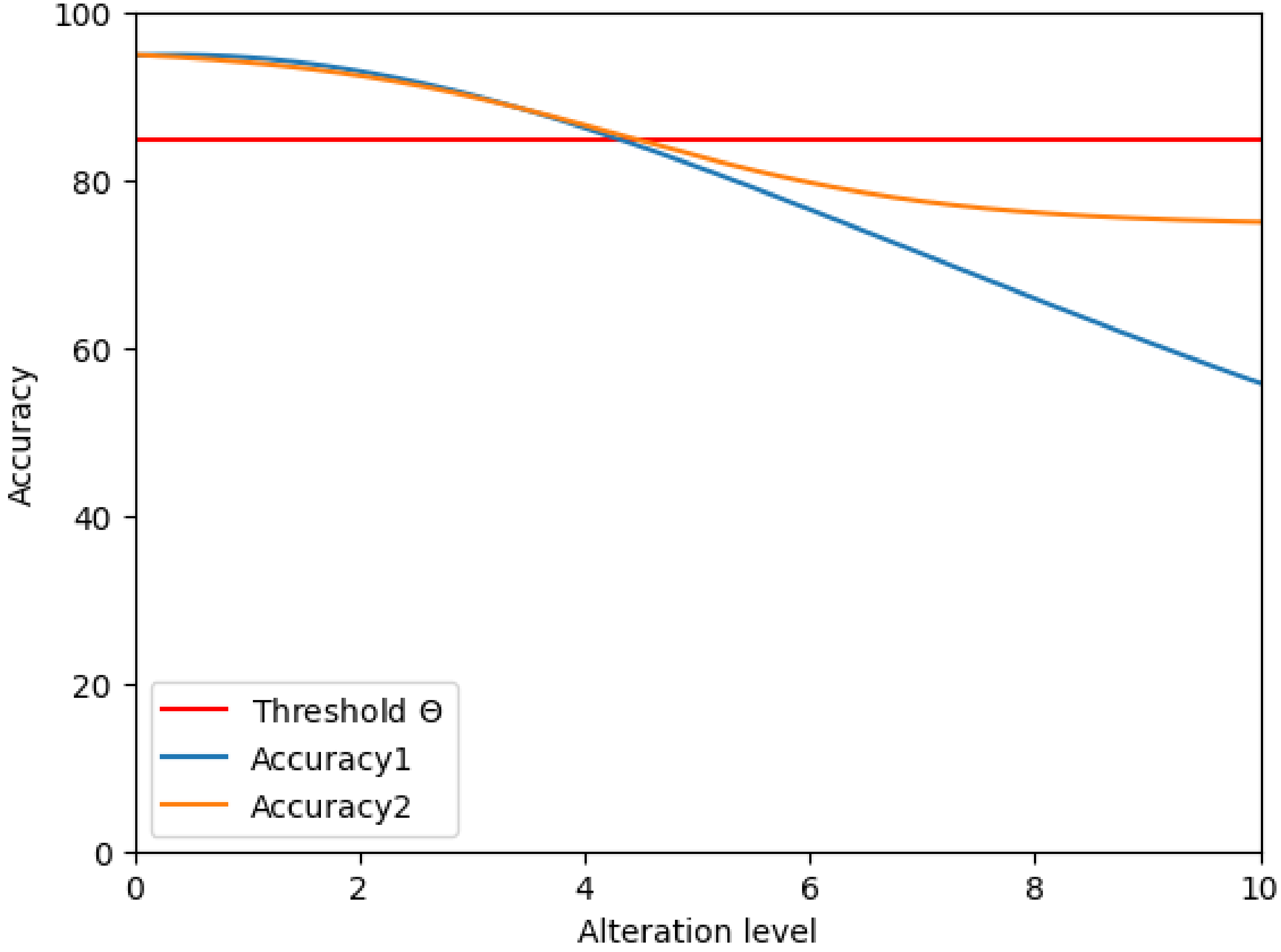
Definition 4.2.4 *(Robustness extension 1). Let C be a classifier and $tol(x)$ a tolerance function. The robustness $rob_A(C, D) \in [0, 1]$ of a classifier C w.r.t. alteration of type A in the range $[L_A, U_A]$ on a dataset D is defined formally as:*

$$rob_A(C, D) = \frac{\int_{L_A}^{U_A} tol(acc(C, D^{A_i}) - \Theta)\ di}{(U_A - L_A)}$$

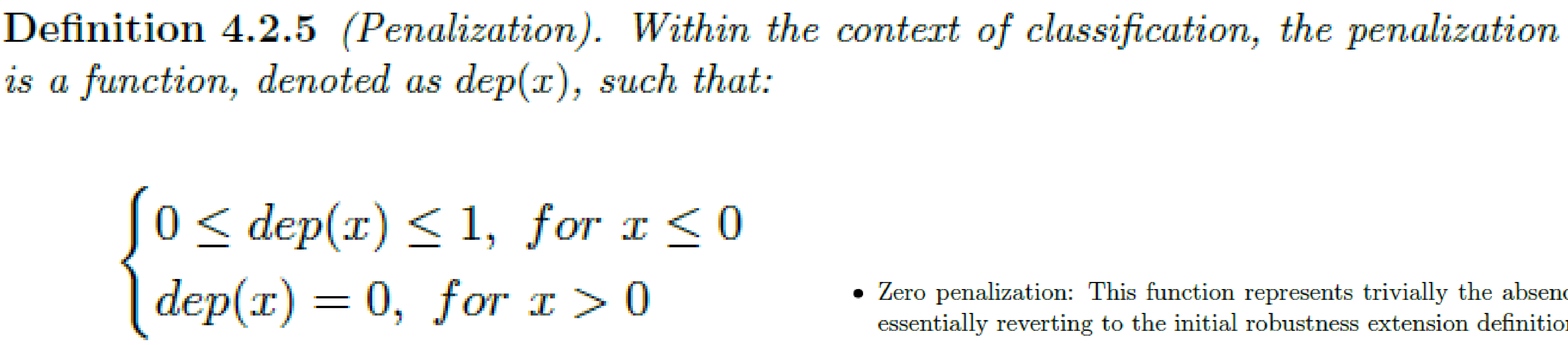


Robustness (4)

What happens after the threshold?



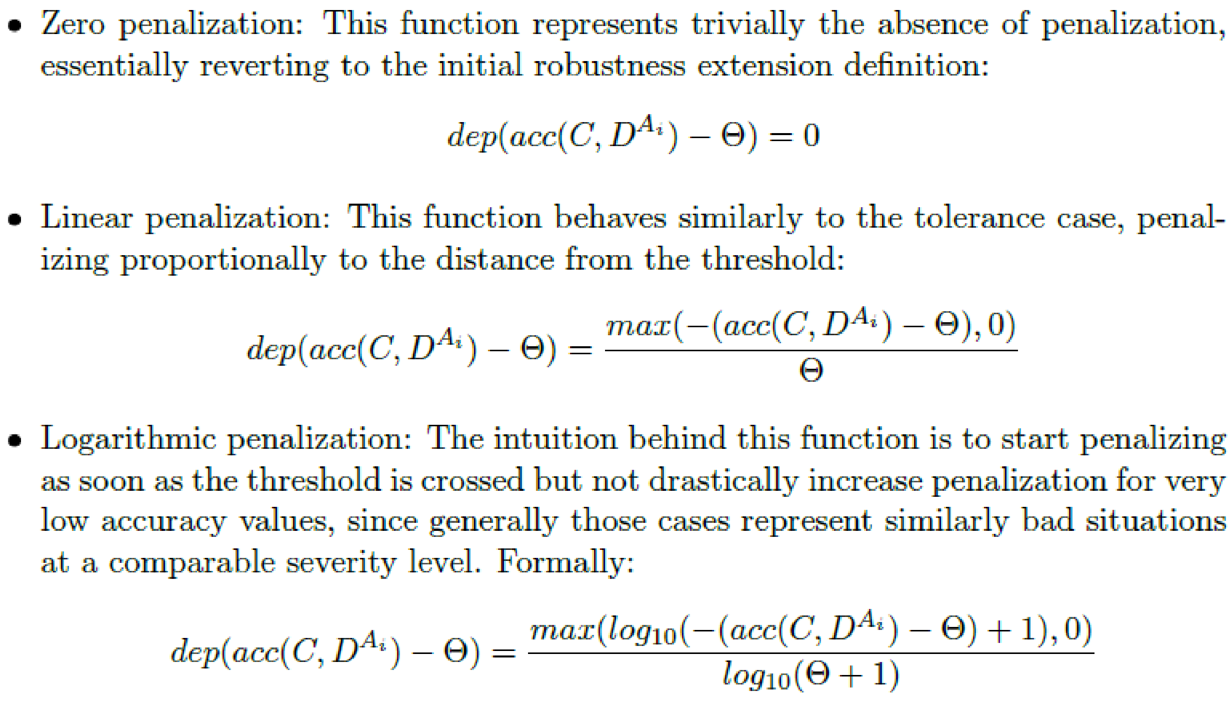
Penalization



Definition 4.2.5 *(Penalization). Within the context of classification, the penalization is a function, denoted as dep(x), such that:*

$$\begin{cases} 0 \leq dep(x) \leq 1, \;\; for\ x \leq 0 \\ dep(x) = 0, \;\; for\ x > 0 \end{cases}$$

Some examples:



- Zero penalization: This function represents trivially the absence of penalization, essentially reverting to the initial robustness extension definition:

$$dep(acc(C, D^{A_i}) - \Theta) = 0$$

- Linear penalization: This function behaves similarly to the tolerance case, penalizing proportionally to the distance from the threshold:

$$dep(acc(C, D^{A_i}) - \Theta) = \frac{max(-(acc(C, D^{A_i}) - \Theta), 0)}{\Theta}$$

- Logarithmic penalization: The intuition behind this function is to start penalizing as soon as the threshold is crossed but not drastically increase penalization for very low accuracy values, since generally those cases represent similarly bad situations at a comparable severity level. Formally:

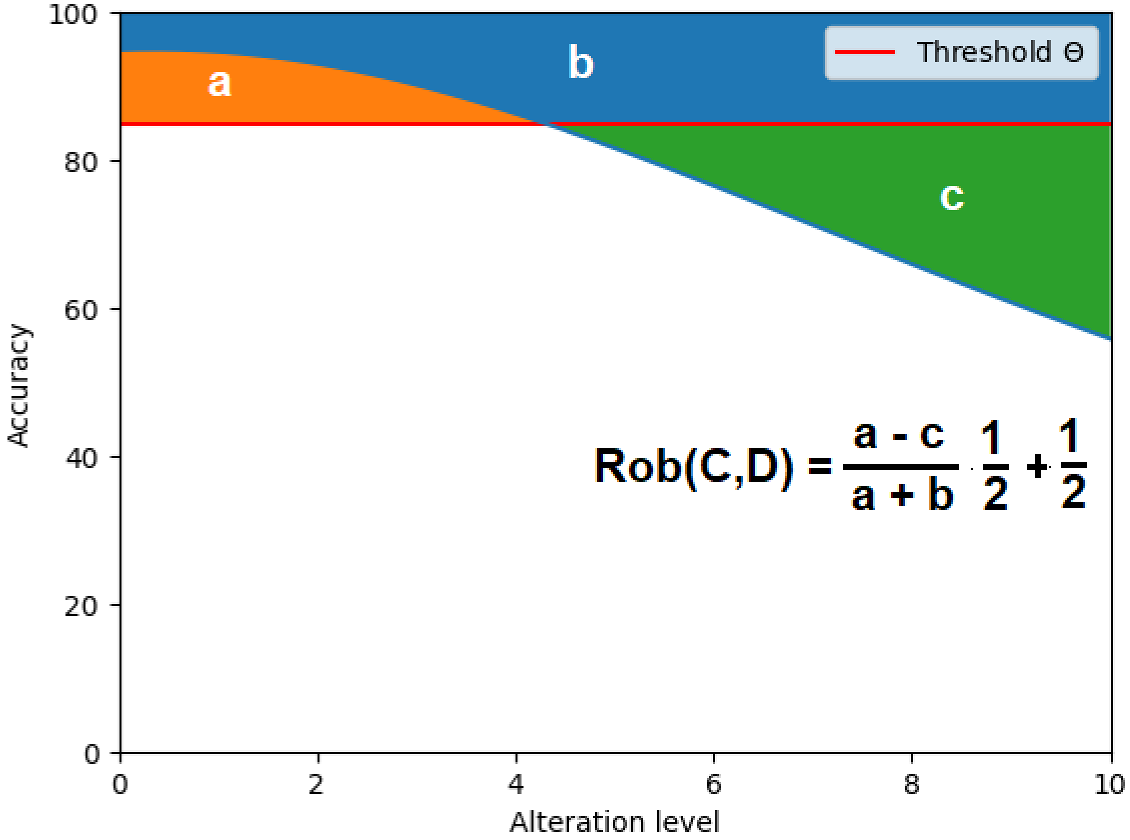
$$dep(acc(C, D^{A_i}) - \Theta) = \frac{max(log_{10}(-(acc(C, D^{A_i}) - \Theta) + 1), 0)}{log_{10}(\Theta + 1)}$$

Robustness (5)

Definition 4.2.8 *(Robustness extension 3). Let C be a classifier, $tol(x)$ a tolerance function and $dep(x)$ a penalization function. The robustness $rob_A(C, D) \in [0, 1]$ of a classifier C w.r.t. alteration of type A in the range $[L_A, U_A]$ on a dataset D is defined formally as:*

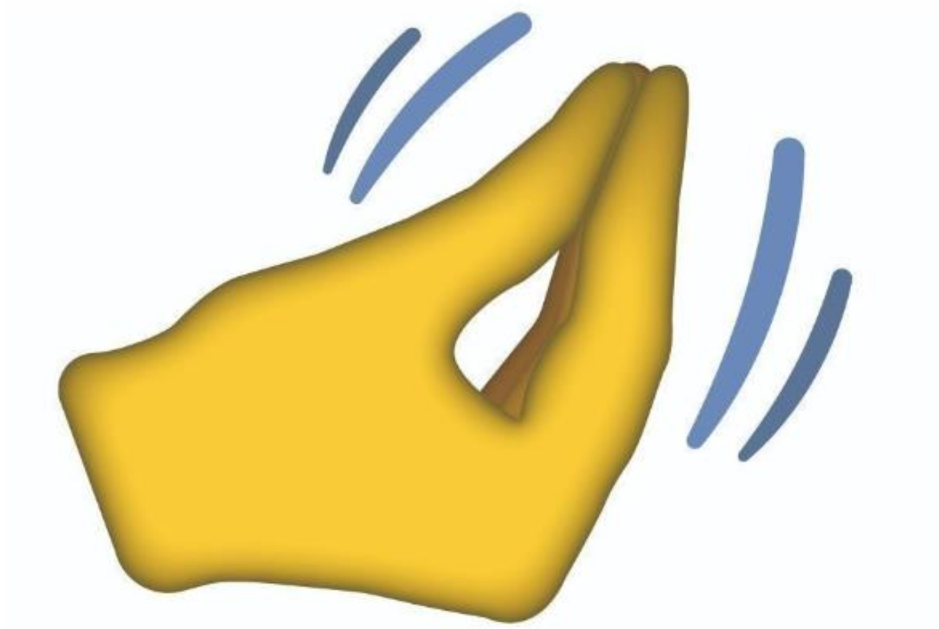
$$rob_A(C, D) = \frac{\int_{L_A}^{U_A} [tol(acc(C, D^{A_i}) - \Theta) - dep(acc(C, D^{A_i}) - \Theta)] \cdot p_A(i)\ di}{2} + \frac{1}{2}$$

where Θ is a threshold referring to minimum accuracy accepted and p_A is probability distribution of the alteration levels.



$$\text{Rob(C,D)} = \frac{a - c}{a + b} \cdot \frac{1}{2} + \frac{1}{2}$$

But where is the uncertainty?..



Robustness against indecision

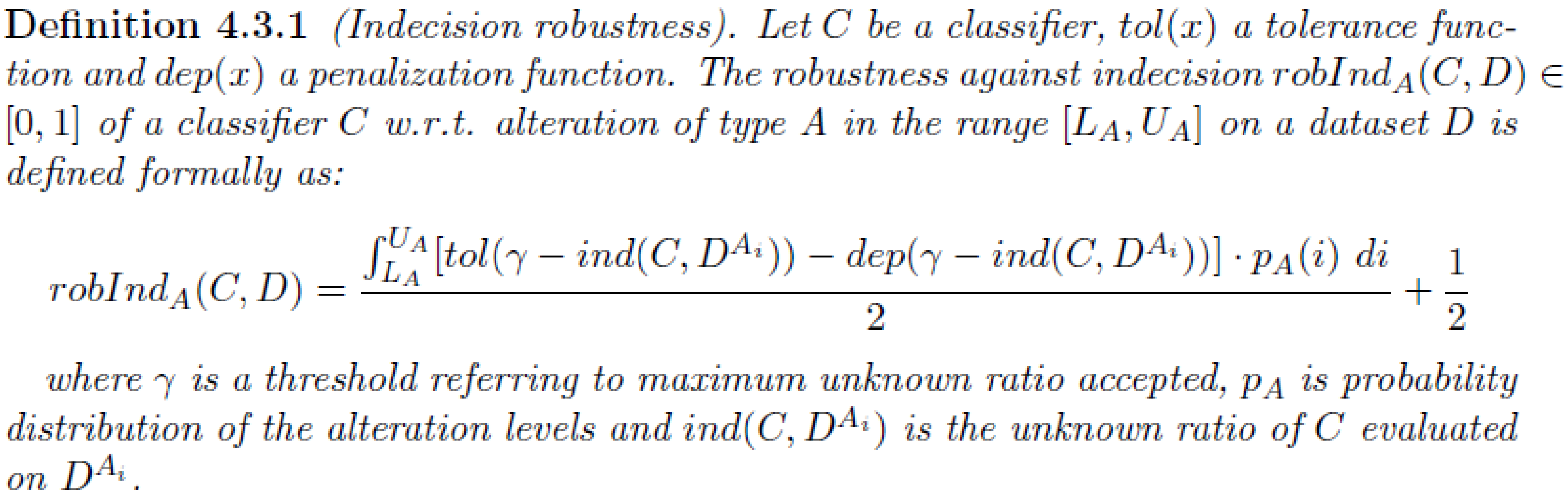
- So far we dealed with approaches applicable always
- In fact, we can directly apply them using the accuracy of the BNN
- The uncertainty is considered during the classification

But…

- What if we have only unknown outputs?
- The uncertainty would be 100%

<ins>**The network is useless!**</ins>

Robustness against indecision



Definition 4.3.1 *(Indecision robustness). Let C be a classifier, $tol(x)$ a tolerance function and $dep(x)$ a penalization function. The robustness against indecision $robInd_A(C,D) \in [0,1]$ of a classifier C w.r.t. alteration of type A in the range $[L_A, U_A]$ on a dataset D is defined formally as:*

$$robInd_A(C,D) = \frac{\int_{L_A}^{U_A} [tol(\gamma - ind(C, D^{A_i})) - dep(\gamma - ind(C, D^{A_i}))] \cdot p_A(i)\ di}{2} + \frac{1}{2}$$

where γ is a threshold referring to maximum unknown ratio accepted, p_A is probability distribution of the alteration levels and $ind(C, D^{A_i})$ is the unknown ratio of C evaluated on D^{A_i}.

Can we combine the two metrics?

The idea is to combine the unknown ratio with the accuracy in someway to get a global metric

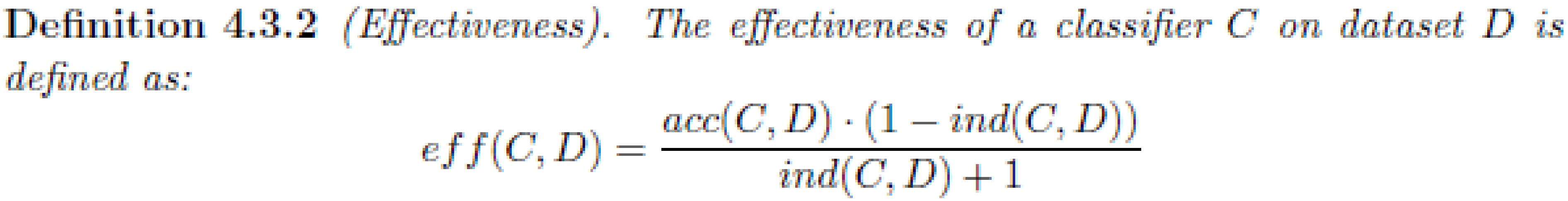
- This new metric should reflect the accuracy When the network has an unknown ratio equal to zero;
- When there are unknown predictions, it should be lower than the accuracy.

A first idea would be a ratio:

$$eff(C, D) = \frac{acc(C, D)}{ind(C, D) + 1}$$

We call it <u>effectiveness</u>

Effectiveness (1)

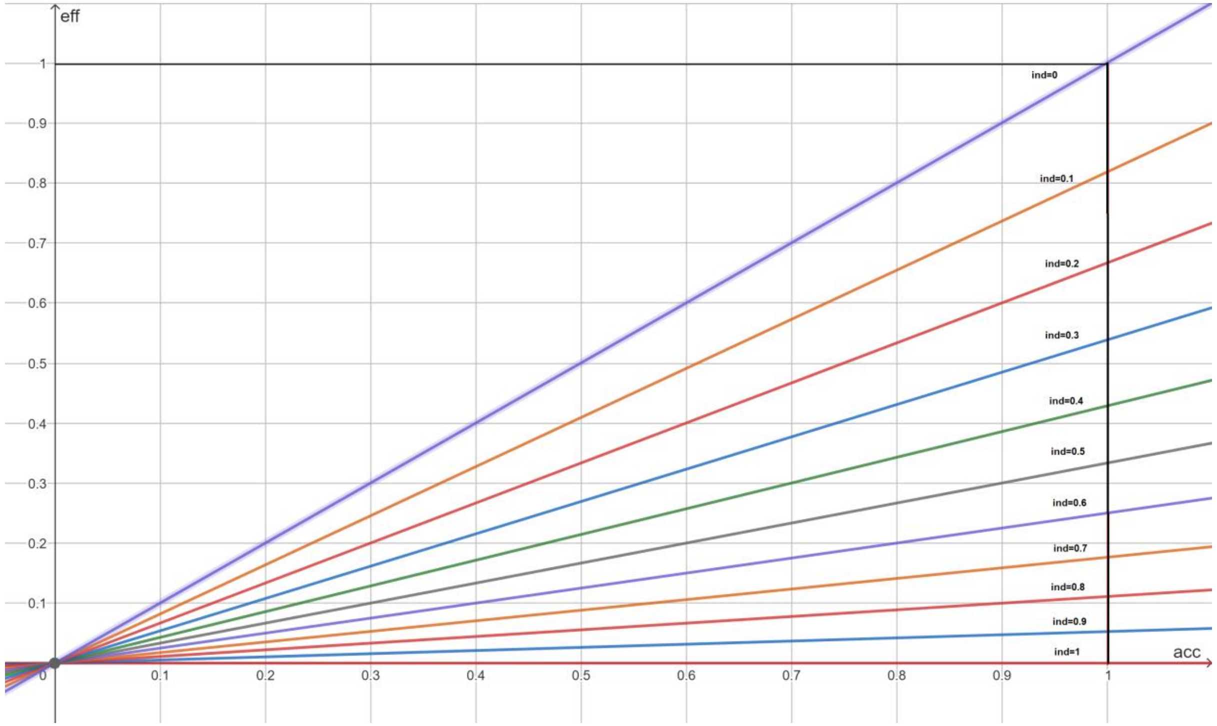


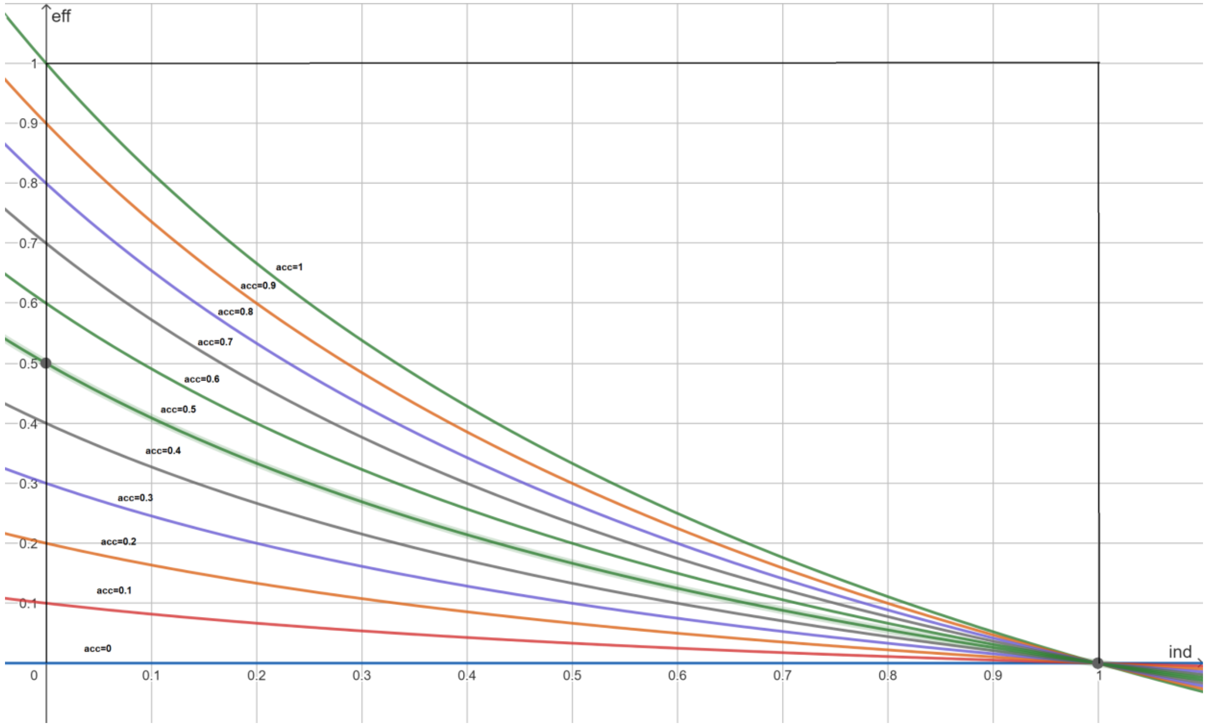
Definition 4.3.2 *(Effectiveness). The effectiveness of a classifier C on dataset D is defined as:*

$$eff(C,D) = \frac{acc(C,D) \cdot (1 - ind(C,D))}{ind(C,D) + 1}$$

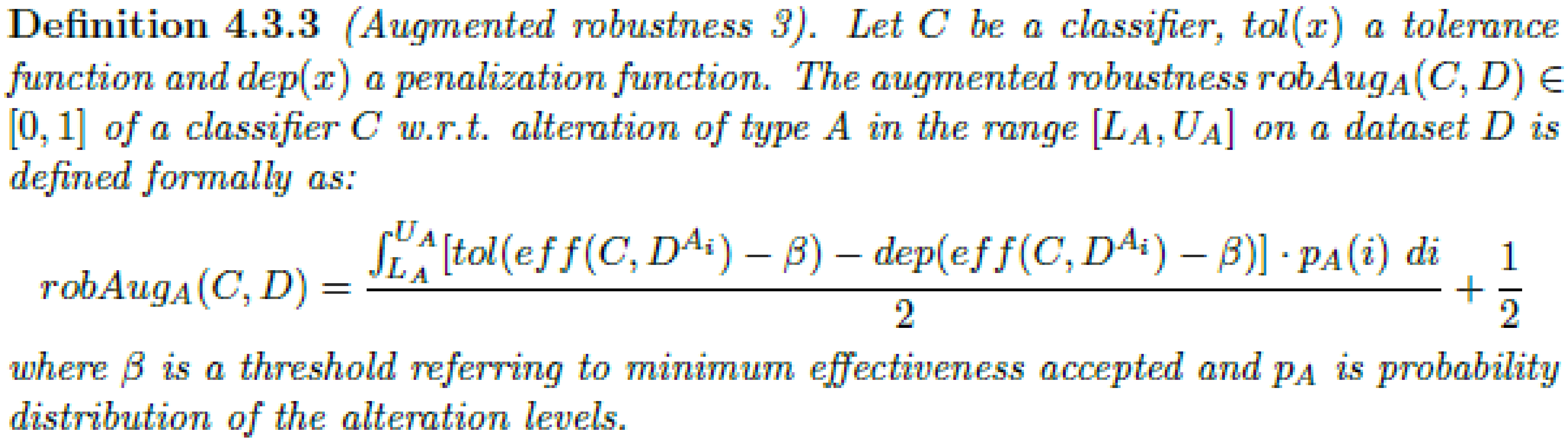
*The added factor serves to push the score to zero when only unknowns are outputted

Effectiveness (2)





Effectiveness (3)



Definition 4.3.3 *(Augmented robustness 3). Let C be a classifier, $tol(x)$ a tolerance function and $dep(x)$ a penalization function. The augmented robustness $robAug_A(C, D) \in [0, 1]$ of a classifier C w.r.t. alteration of type A in the range $[L_A, U_A]$ on a dataset D is defined formally as:*

$$robAug_A(C, D) = \frac{\int_{L_A}^{U_A} [tol(eff(C, D^{A_i}) - \beta) - dep(eff(C, D^{A_i}) - \beta)] \cdot p_A(i) \; di}{2} + \frac{1}{2}$$

where β is a threshold referring to minimum effectiveness accepted and p_A is probability distribution of the alteration levels.

Some lessons learned

- Uncertainty estimation can help in the reduction of wrong predictions, and it is an inherent characteristic that improves the robustness;
- When dealing with uncertainty, it is not enough to consider the accuracy as metric;
- The uncertainty must be included in the evaluation.