On the Evaluation Measures for Machine Learning Algorithms for Safety-critical Systems

(Short Paper)

Mohamad Gharib and Andrea Bondavalli
University of Florence - DiMaI, Viale Morgagni 65, Florence, Italy
{mohamad.gharib,andrea.bondavalli}@unifi.it

***Abstract*—The ability of Machine Learning (ML) algorithms to learn and work with incomplete knowledge has motivated many system manufacturers to include such algorithms in their products. However, some of these systems can be described as Safety-Critical Systems (SCS) since their failure may cause injury or even death to humans. Therefore, the performance of ML algorithms with respect to the safety requirements of such systems must be evaluated before they are used in their operational environment. Although there exist several measures that can be used for evaluating the performance of ML algorithms, most of these measures focus mainly on some properties of interest in the domains where they were developed. For example, Recall, Precision and F-Factor are, usually, used in Information Retrieval (IR) domain, and they mainly focus on correct predictions with less emphasis on incorrect predictions, which are very important in SCS. Accordingly, such measures need to be tuned to fit the needs for evaluating the safe performance of ML algorithms. This position paper presents the authors' view on the inadequacy of existing measures, and it proposes a new set of measures to be used for the evaluation of the safe performance of ML algorithms.**

***Keywords*-Machine learning, Algorithms, Performance Metrics, Safety Measures, Safety-critical Systems**

I. Introduction

Recently, we are witnessing an increasing adoption of Machine Learning (ML) algorithms in many automated systems covering almost all the main domains of our lives [1]. Their ability to learn and work with novel input/incomplete knowledge [2], and their generalization capabilities make them highly desirable solutions for complex problems [3]. This has motivated many system manufacturers to incorporate ML algorithms in their products for performing complex tasks such as pattern recognition, image recognition, and even control [3]. However, some of these systems can be classified as safety-critical systems, where their failure may cause death or injury to humans. Accordingly, the safe performance of such ML algorithms[1] must be evaluated/assessed before they are used in their operational environment.

Generally speaking, an ML algorithm builds a mathematical model of sample data (e.g., training data set), in order to make predictions or decisions without being explicitly programmed to perform such task [4]. This is usually done relying on a classifier that assigns prediction scores to each observation, which indicates the certainty of the classifier that such observation belongs to one of the possible classes [5]. In the case of *binary classifiers*, observations belong to one of only two possible classes (e.g., positive or negative) [4], and the classification decision is usually taken based on the score of observation with respect to the classification threshold (e.g., cut-off point). More specifically, observations with scores higher than the threshold are predicted to belong to the positive class and observations with scores lower than the threshold are predicted to belong to the negative class.

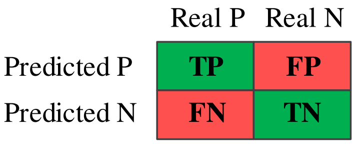
In this context, predictions can be classified into four groups based on the real known class of the observation and the predicted one: *True Positive (TP)*/*True Negatives (TP)* cases refer to the Predicted Positives/Negatives that were correct, while *False Positive (FP)*/*False Negatives (FN)* cases refer to the Predicted Positives/Negatives that were incorrect. These four groups are organized in four cells in the binary contingency table that is shown in Figure 1, where green colored cells contain correct predictions, and incorrect predictions are contained in red color cells. Figure 2 shows a sample distribution of the count of observations against the predicted probability, where we can identify the four main areas corresponding to the four groups of the contingency table.

Taking these groups into consideration, several measures for evaluating the performance of ML algorithms have been used in the literature (e.g., Recall, Precision, F-Factor). However, most of these measures focus mainly on some properties of interest in the domains where they were developed. For instance, Recall, Precision and F-Factor have been used regularly to evaluate the performance of Information Retrieval (IR) algorithms, and they mainly focus on the number of correct positive predictions (e.g., TP cases), i.e., they have less or even no emphasis toward incorrect predictions (FN and FP cases)[2].

FN and FP cases can be of great importance in safety-critical systems. For example, a self-driving vehicle, that is supposed to detect pedestrians, cyclists, etc. and prevent crashing into them, failed to identify a pedestrian (FN), which results in hitting the woman that later died at a hospital [6]. While a FP (i.e., false alarm) in such ML-based detection system may result in automatically applying the breaks of the vehicle to prevent crashing into what the algorithm identifies as a

[1] Their performance with respect to the safety requirements of the incorporating system

[2] More detailed discussion about these metrics in the following section



	Real P	Real N
Predicted P	TP	FP
Predicted N	FN	TN

Fig. 1. The binary contingency table

pedestrian, a cyclist, etc. Although FP is not as critical as hitting a pedestrian (FN), it is still a situation should be avoided since it might lead to life-threatening accidents. To this end, existing measures need to be tuned to fit the needs for evaluating the safe performance of ML algorithms.

The rest of the paper is organized as follows; Section II presents some performance measures for ML algorithms, and we discuss the problem statement and research challenges in Section III. In Section IV, we present and discuss possible solutions. Finally, we conclude the paper in Section V.

II. Performance metrics for ML algorithms

Several measures for evaluating the performance of ML algorithms have been used in the literature. For instance, Precision, Recall and F-Factor have been used regularly to measure the performance of Information Retrieval (IR) algorithms, where Recall (True Positive Rate (TPR)) is the proportion of True Positive (TP) cases that are correctly Predicted Positive (equation 1). While Precision (also called Confidence in Data Mining) denotes the proportion of Predicted Positive cases to the Real Positives (equ.2). F_1-score (also called F_1-measure) is intended to combine Precision and Recall measures into a single measure of search "effectiveness" (equ.3). On the other hand, Sensitivity (called Recall (equ.1) in IR) and Specificity (equ.4) are commonly used in the Behavioral Sciences, and they measure the proportion of real Positive/Negatives that are correctly identified (TPR/ True Negative Rate (TNR)). Finally, the Receiver Operating Characteristics (ROC) graph have been first developed and used in signal detection theory and now it is commonly used in Medical Sciences for evaluating the tradeoff between hit rates (TPR) and false alarm rates (FPR) rates of classifiers [7]. Taking a closer look at these measures, we can conclude that most of them mainly focus on TP and some on TN cases, i.e., they do not focus on FP nor FN cases. Therefore, they need to be tuned to fit the needs for evaluating the safe performance of ML algorithms.

$$\texttt{Recall} = \texttt{Sensitivity} = \texttt{TPR} = \frac{\texttt{TP}}{\texttt{TP} + \texttt{FN}} \tag{1}$$

$$\texttt{Precision} = \texttt{Confidence} = \frac{\texttt{TP}}{\texttt{TP} + \texttt{FP}} \tag{2}$$

$$F_1\texttt{-score} = \frac{2 \times \texttt{Precision} \times \texttt{Recall}}{\texttt{Precision} + \texttt{Recall}} \tag{3}$$

$$\texttt{Specificity} = \texttt{TNR} = \frac{\texttt{TN}}{\texttt{TN} + \texttt{FP}} \tag{4}$$

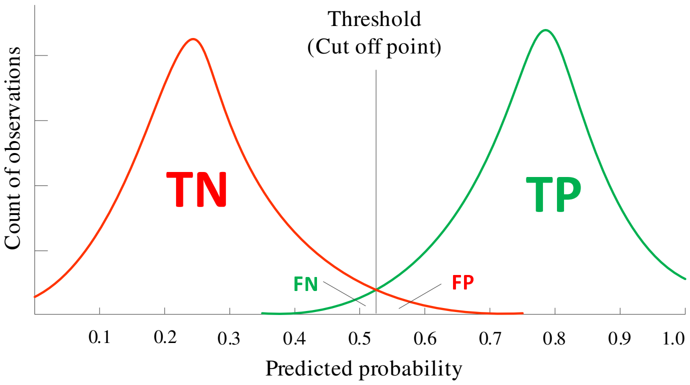


Fig. 2. A distribution of observations' count against the predicted probability

III. Problem Statement and Research Challenges

Consider for example an ML-based system for pedestrian detection, the ML algorithm is said to safely perform when its predictions are correct (TP and TN), i.e., the algorithm correctly identifies a pedestrian as a pedestrian (TP), and it correctly identifies a non-pedestrian as a non-pedestrian (TN). While the ML algorithm may perform unsafely when its predictions are wrong (FN and FP), i.e., the algorithm incorrectly identifies a pedestrian as a non-pedestrian (FN) that may result in catastrophic incident, and it incorrectly identifies a non-pedestrian as a pedestrian (FP) that may result in non significant, marginal, critical, or even catastrophic incident.

To this end, how can we evaluate the safe performance of an ML algorithm taking into consideration the safety-critical settings that such ML algorithm may perform in? In order to answer this question, we need to tackle the following Research Challenges (RCs):

RC1: *How can we identify when the performance of an ML algorithm is guaranteed to be correct?* As previously mentioned, algorithms make a classification decision relying on the score of the observation with respect to the classification threshold. Based on the distribution of observation predictions that is shown in Figure 2, the performance of an ML algorithm is guaranteed to be correct when its predictions are correct (TP and TN). It can be seen as the union of areas under the green and red lines excluding the area resulting from their intersection, where both FP and FN cases co-locate. Although adjusting the decision threshold to account for misclassification has been used in several works (e.g., [5]), we cannot rely on such solution since adjusting the threshold to decrease FN cases, will increase the FP cases and vice versa. Thus, we need new techniques to identify when the performance of an ML algorithm is guaranteed to be correct.

RC2: *How the performance of an ML algorithm can be safely interpreted in safety-critical settings, where the algorithm may perform?* After clearly identifying when the performance of an ML algorithm is guaranteed to be correct, we need to understand how the results of the overall performance can be safely interpreted by a safety-critical system that relies on such results to make safety-critical decisions.

RC3: *Which measures can be used to evaluate the safe performance of ML algorithms?* As previously discussed, existing measures need to be tuned to fit the needs for evaluating the safe performance of ML algorithms. Therefore, we need to develop new measures specifically designed to be used for the evaluation of the safe performance of ML algorithms.

IV. Towards a method for the evaluation of the safe performance of ML algorithms

In this section, we present and discuss a set of measures that can be used for the evaluation of the safe performance of ML algorithms. In particular, we try to tackle each of the research challenges raised in the previous section:

RC1: *How can we identify when the performance of an ML algorithm is guaranteed to be correct?* As previously mentioned, this problem cannot be solved by adjusting the threshold. However, it can be solved following a commonly used safety principle, namely *safety reserves* [8], which can be used to define safety margins where the predictions of the algorithm are guaranteed to be correct. In particular, instead of adjusting the threshold, we define two thresholds namely, *Safe TP threshold* and *Safe TN threshold*, where the first specifies a threshold that any observation with scores higher than it, is sufficiently guaranteed to be TP, and the last specifies a threshold that any observation with scores lower than it, is sufficiently guaranteed to be TN. In this context, observations with scores higher than the *Safe TP threshold* or lower than the *Safe TN threshold* are sufficiently guaranteed to be correct. Accordingly, observations with scores higher than *Safe TN threshold* and lower than *Safe TP threshold* cannot be guaranteed to be correct. We refer to such observations as No Prediction (NP). We differentiate between NP Positive (NP-P) and NP Negative (NP-N) that refer to positive and negative cases, which cannot be used to make safety-critical decisions. Note that a significantly few numbers of wrong predictions (e.g., FP and FN) might occur because the thresholds should be defined with respect to the Tolerable Hazard Rate (THR) concept [9]. THR is used to guarantee that wrong predictions, which may result from relying on the defined thresholds, will not exceed a predefined level of risk. THR is commonly used in safety standards (e.g., IEC 61508 [9], CENELEC - EN 50129 [10]) as the probabilistic indicator for identifying the related Safety Integrity Level (SIL)[3] that is a measurement of performance required for safety-related functions. For example, the acceptable range of THR for a safety-related function/system classified as SIL4 should be within 10^{-9} < THR < 10^{-8}). The *Safe TP and TN thresholds*, a sample distribution of TP, TN, NP-P, NP-N, FN, and FP are shown in Figure 3.

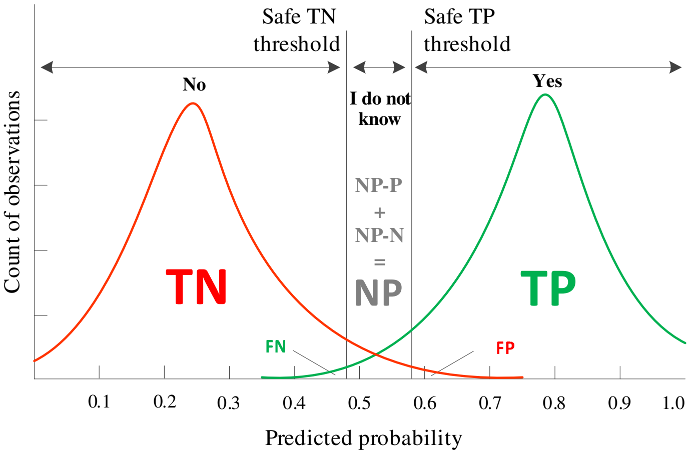


Fig. 3. A distribution of observations' count against the predicted probability with safe TP and TN thresholds

RC2: *How the performance of an ML algorithm can be safely interpreted in safety-critical settings, where the algorithm may perform?* After providing criteria for differentiating the guaranteed correct predictions (TP and TN) and No Predictions (NP-P and NP-N) cases of an ML algorithm[4], we can discuss how such predictions can be safely interpreted in safety-critical settings. In particular, TP cases are mapped to "Yes" decisions with respect to the phenomena under observation. Considering the ML-based algorithm for pedestrian detection, a TP case can be interpreted as identifying a pedestrian as a pedestrian (Yes, it is a pedestrian). TN cases are mapped to "No" decisions with respect to the phenomena under observation. Considering the same example, a TN case can be interpreted as identifying a non-pedestrian as a non-pedestrian (No, it is not a pedestrian). Finally, NP (NP-P and NP-N) cases can be interpreted as "I do not Know", which prevents taking any decisions since we cannot rely on such cases to make a safety-critical decision. More specifically, a *fail-aware* mechanism is adopted to deal with NP cases. To this end, the system either make a safe decision based on TP or TN, or it *fail-aware* and make no decision when it is not guaranteed that such decision will be safe. The mapping between TP, TN, and NP on one hand and "Yes", "No" and "I do not Know" on the other hand is also shown in Figure 3.

RC3: *Which measures can be used to evaluate the safe performance of ML algorithms?* At this point, the

[3] Four SILs are defined (SIL1-4), where SIL 4 is the most dependable

[4] FP and FN are insignificant to be considered

predictions of an ML algorithm can be mainly classified into the following groups: TP, TN, NP-P, NP-N, FP, and FN, which are organized into a new contingency table (shown in Figure 4). In particular, predictions that use to be classified as TP are now classified either as TP or NP-P, and predictions that use to be classified as TN are now classified either as TN or NP-N. FP and FN predictions still exist in the table but their numbers are insignificant to be considered. Therefore, the four main groups of predictions (e.g., TP, TN, NP-P and NP-N) can be used to design the following measures for evaluating the safe performance of ML algorithms:

1. TP rate (TPr) is the percentage of TP that are guaranteed to be correct to the total number of real positives.

$$TPr = \frac{TP}{P} \tag{5}$$

2. TN rate (TNr) is the percentage of TN that are guaranteed to be correct to the total number of real negatives.

$$TNr = \frac{TN}{N} \tag{6}$$

3. Prediction rate (Pr) is the percentage of TP and TN that are guaranteed to be correct to the total number of observations (real positives and real negatives).

$$Pr = \frac{TP + TN}{P + N} \tag{7}$$

4. TP Lost rate (TPLr) is the percentage of No Prediction Positives (NP-P) to the total number of real positives.

$$TPLr = \frac{P - TP}{P} = \frac{\textit{NP-P}}{P} \tag{8}$$

5. TN Lost rate (TNLr) is the percentage of No Prediction Negatives (NP-N) to the total number of real negatives.

$$TNLr = \frac{N - TN}{N} = \frac{\textit{NP-N}}{N} \tag{9}$$

6. No Prediction rate (NPr) is the percentage of No Prediction cases to the total number of observations.

$$NPr = \frac{\textit{NP-P} + \textit{NP-N}}{P + N} = 1 - Pr \tag{10}$$

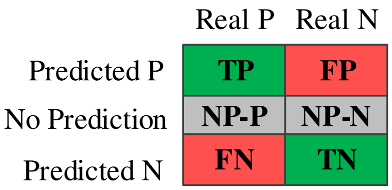
7. NP-P percentage (NP-Pp) is the percentage of NP-P to the total number of NP cases (NP-P and NP-N).

$$\textit{NP-Pp} = \frac{\textit{NP-P}}{\textit{NP-P} + \textit{NP-N}} \tag{11}$$

8. NP-N percentage (NP-Np) is the percentage of NP-N to the total number of NP cases.

$$\textit{NP-Np} = \frac{\textit{NP-N}}{\textit{NP-P} + \textit{NP-N}} \tag{12}$$

Note that measures 11 and 12 can be very useful when the costs of NP-P and NP-N are not equal.



	Real P	Real N
Predicted P	**TP**	**FP**
No Prediction	**NP-P**	**NP-N**
Predicted N	**FN**	**TN**

Fig. 4. Contingency table for the safe performance of ML algorithms

V. Conclusion

We have argued that existing measures need to be tuned to fit the needs for evaluating the safe performance of ML algorithms. Our argument has been structured based on analyzing existing measures and the special needs for evaluating the safe performance of ML algorithms. We formulated this problem as several research challenges. Then, we have discussed a proposed solution for each of these challenges proposing a new set of measures that can be used for the evaluation of the safe performance of ML algorithms. We are planning to demonstrate the applicability and usefulness of the proposed measures by applying them to several data sets concerning ML algorithms that are incorporated in safety-critical systems.

Acknowledgment

This work has received funding from the REGIONE TOSCANA POR FESR 2014-2020 SISTER “SIgnaling & Sensing Technologies in Railway application”, and the European Union’s Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant agreement No. 823788 - ADVANCE project.

References

[1] M. Gharib, P. Lollini, M. Botta, E. Amparore, S. Donatelli, and A. Bondavalli, “On the Safety of Automotive Systems Incorporating Machine Learning Based Components: A Position Paper,” in *Proceedings of the 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops, DSN-W 2018* IEEE, pp. 271–274.

[2] Z. Kurd, T. Kelly, and J. Austin, “Developing artificial neural networks for safety critical systems,” *Neural Computing and Applications*, vol. 16, no. 1, pp. 11–19, oct 2007.

[3] J. Schumann, P. Gupta, and Y. Liu, “Applications of Neural Networks in High Assurance Systems,” in *Neural Networks*, 2010, v. 268, 1–19.

[4] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[5] J. Hernández-Orallo, P. Flach, and C. Ferri, “A Unified View of Performance Metrics: Translating Threshold Choice into Expected Classification Loss,” *Journal of Machine Learning Research*, vol. 13, pp. 2813–2869, 2012.

[6] S. Levin and J. C. Wong, “Self-driving Uber kills Arizona woman in first fatal crash involving pedestrian, Technology, The Guardian,” p. 1, 2018. [Online]. Available: https://goo.gl/DqfQxZ

[7] D. M. Powers, “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation,” *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[8] N. Möller and S. O. Hansson, “Principles of engineering safety: Risk and uncertainty reduction,” *Reliability Engineering and System Safety*, vol. 93, no. 6, pp. 798–805, 2008.

[9] International Electrotechnical Commission (IEC:2010), “IEC 61508: Functional safety of electrical/electronic/programmable electronic safety-related systems,” *IEC, Geneva, Switzerland*, 2010.

[10] C. EN50129, “Railway applications-Communication, signalling and processing systems-Safety related electronic systems for signalling,” *British Standards Institution, United Kingdom. ISBN*, pp. 0580—-4181, 2003.