Weighted Mutual Information for Out-Of-Distribution Detection

Giacomo De Bernardi[1,2] Sara Narteni[1[0000−0002−0579−647X]], Enrico Cambiaso[1[0000−0002−6932−1975]], Marco Muselli[1,3], and Maurizio Mongelli[1]

[1] CNR-IEIIT, 16149, Genoa, Italy
[2] Università degli studi di Genova, DITEN Department, 16145, Genova, Italy
[3] Rulex Innovation Labs, Rulex Inc., 16122 Genoa, Italy
name.surname@ieiit.cnr.it

Abstract. Out-of-distribution detection has become an important theme in machine learning (ML) field, since the recognition of unseen data either "similar" or not (in- or out-of-distribution) to the ones the ML system has been trained on may lead to potentially fatal consequences. Operational data compliance with the training data has to be verified by the data analyst, who must also understand, in operation, if the autonomous decision-making is still safe or not. In this paper, we study an out-of-distribution (OoD) detection approach based on a rule-based eXplainable Artificial Intelligence (XAI) model. Specifically, the method relies on an innovative metric, i.e., the weighted mutual information, able to capture the different way decision rules are used in case of in- and OoD data.

Keywords: Out-of-distribution detection · eXplainable AI · mutual information · open data.

1 Introduction

Nowadays out-of-distribution detection (ODD) is an important theme in ML, consisting in a comparison between the training and working conditions of a model. In case of divergence, the autonomous system should trigger an alert because the model may no longer be as performing as it was during the training stage (even if generalization is still acceptable[4]).

The problem constitutes a fundamental challenge in the field of Safe AI, which means individuating all the conditions under which autonomous decisions may lead to hazards. The impact of the ODD is then to make AI systems aware of this, thus understanding under which conditions the model may operate without dangerous effects to humans and/or the environment. In this context, recently,

[4] Generalization bounds, see, e.g., [17], quantify the gap that exists between the empirical risk, calculated on the data actually available (on which the model is trained) and the theoretical risk, calculated on the distribution of probability that represents the data; this probability distribution is unknown in closed-form and, in the ODD context, represents the "in-distribution".

standards were introduced in different fields, such as avionics [6,5], automotive ([22,9] and ISO/IEC) or medical informatics [2], stating the guidelines to identify all the operating conditions that can have an impact on safety. OoD can actually

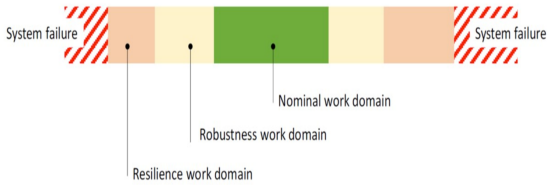


Fig. 1: Illustration of the work domains as reported in [6]. From central green bar to side yellow/orange/red bars, the nominal domain shifts and the severity of the OoD increases in parallel.

occur with different degrees of severity, as depicted through colored bars in Fig. 1 from EASA. First, the in-distribution is realized when the system working conditions are the same seen during training, which is referred to as nominal work domain and depicted through the green bar. When data start to diverge from such a distribution, several levels of OoD can be identified, as well as their relationship with the possible degradation of the underlying machine learning algorithm functioning. Starting from the lower risk level, the yellow bar reflects a situation where the model is still able to perform accurately enough; orange color bar indicates an OoD level in which a failure of the autonomous decision is observed, but the consequences on the system are not dangerous (the surrounding conditions of the environment are still compatible with safe actuations); finally, red bars express a high probability of the autonomous system to encounter dangerous conditions (if guided by the autonomous function). Including all the mentioned severity levels is thus essential in testing autonomous safety-critical systems and the ODD is therefore necessary to properly acknowledge the different zones. In particular, the study of the OoD according to distributional assumption-free and OoD-agnostic methods is an open research problem and is still little investigated [5].

1.1 Related work

Most of the existing solutions to address the problem of the ODD are based on strong distributional assumptions of the feature space [13] or assume the knowledge of the in- and out-of-distribution probability density functions (pdf)

[5] Distributional assumption-free means no closed-form expressions of in- and out-probability distributions are considered. OoD-agnostic means no information about ODD conditions is considered. Another important issue, which is related with the assumption of probabilistic Gaussian or mixed-Gaussian functions, is to avoid calculating the covariance matrix from data, which can be numerically unstable.

[1], although this is not always true in practice. Moreover, several statistic tests often fail to estimate the real distribution of training data (data are not enough and the pdf are too coarse). Some methods, widely used across OoD detection are the Maximum Softmax Probability [10] which is one of the simplest but strong OOD detection method, being based on the tendency of maximum softmax probability to reach higher values in presence of out-of-distribution samples. Another approach, the OoD detector in neural networks (ODIN, [14]), is based on the observation that using temperature scaling and adding small perturbations to the input can separate the softmax score distributions between in and out. Energy-based OoD Detection [16] uses energy scores to better distinguish in and out than the traditional approach of softmax scores. Other important benchmarks are: model-based methods [25], [3], [8] and threshold-based methods [15], [11], [26]; label shift in deep learning is also considered in [7]. Our method is part of distributional assumption-free hypothesis models, as the work proposed in [24], but with the advantage of the eXplainability on its foundation.

1.2 Contribution

In this paper, we propose a distributional assumption-free method based on the evaluation of the histogram generated by the frequency of validation of a rule-based model by the data themselves. The histogram defined during the training phase constitutes a fingerprint to be verified at runtime. If the data at runtime generate a histogram "significantly different" from the training one, it means that the data are OoD. The purpose of the proposed approach is thus to quantify such a difference by introducing weighted mutual information $W\mu I$, i.e., a modification of mutual information able to consider the structure of the rule-based model. Also, unlike other approaches like K-NN [24] or Neural Networks distance [4], where a single metric is used for the ODD, we here infer the OoD through multiple metrics, namely $W\mu I$ and more traditional vector norms. This offers support to the tests mentioned by EASA, since the proposed method measures incremental cases of departure from in-distribution.

2 Logic Learning Machine

The rule-based model adopted in our work is called Logic Learning Machine (LLM), an efficient implementation of Switching Neural Networks [20], developed and available in Rulex software platform [6]. However, we remark that the methods for OoD detection presented in the paper can be easily extended to any other kind of rule-based model, such as decision tree or tree ensembles like random forests or Skope-Rules [7].
Given some input data, the LLM provides a classification model represented by ruleset $\mathcal{R} = \{r_k\}_{k=1,\dots,N_r}$, with each rule r_k expressed through the following structure: **if** <*premise*> **then** <*consequence*>. The <*premise*> is made up

[6] https://www.rulex.ai
[7] https://github.com/scikit-learn-contrib/skope-rules

of the logical conjunction (AND) of conditions on the input features and the $<consequence>$ constitutes the output of the classification rule.
Rule generation process occurs in three steps. First, a discretization and binarization of the feature space is performed by using the inverse-only-one coding. The resulting binary strings are then concatenated into a single large string representing the considered samples. Shadow clustering is then used to build logical structures, called implicants, which are finally transformed into simple conditions and combined into a collection of intelligible rules [21].
Covering $C(r_k)$ and error $E(r_k)$ metrics can be computed to evaluate the performance of each rule, being defined as:

$$C(r_k) = \frac{TP(r_k)}{TP(r_k) + FN(r_k)}, \qquad E(r_k) = \frac{FP(r_k)}{TN(r_k) + FP(r_k)} \tag{1}$$

where $TP(r_k)$ and $FP(r_k)$ are the number of samples that, respectively, correctly and wrongly verify rule r_k; $TN(r_k)$ and $FN(r_k)$ are the number of samples that, respectively, correctly and wrongly do *not* verify the rule. The covering is also proportional to the relevance of the rule, therefore the larger it is, the higher is the probability that the rule is valid on new unseen samples. On the contrary, the error $E(r_k)$ measures how much wrongly covered is the rule and its maximum value is usually fixed as a model hyper-parameter (by default, it is of 5%).

3 Rule-based ODD

3.1 Notation and List of Acronyms

TR	Training set
OP	Operational set
tr_i	i-th training subset
op_i	i-th operational subset
n_s	number of data samples in a split
N_r	Number of rules
N_{tr}	Number of training splits
N_{op}	Number of operational splits
$\mathcal{R}_{tr}$	Training reference ruleset
r_i	i-th rule
h_i^j	j-th hit for the i-th rule
l_p	l_p norm
μI	Mutual information
$W\mu I$	Weighted mutual information
H	Entropy

3.2 Rule hits histograms

Given a training dataset TR, we train the LLM on it and define a reference ruleset $\mathcal{R}_{tr}$. Also, we consider to split TR and the operational set OP into N_{tr} and N_{op} portions, called splits, respectively, thus obtaining $N_h = N_{tr} + N_{op}$ total splits. Let also n_s be the number of data samples present in a single split.

For each split, samples [8] may (or not) verify each rule in $\mathcal{R}_{tr}$ a certain amount of times. We denote this number as the *number of hits* for the considered rule. Therefore we derive N_h vectors, considering this value scaled by the split size n_s:

$$\mathbf{h}^j = \{h_i^j\}, \;\; h_i^j \in [0,1], \; i = 1, \ldots, N_r, \; j = 1, \ldots, N_h \tag{2}$$

Each vector $\mathbf{h}^j$ can be thought as a histogram.

3.3 Data splits and metrics

The N_{tr} splits, available in training for the dataset $TR = \{tr_1, \ldots, tr_{N_{tr}}\}$, become the reference for building the in-distribution histograms, as per Eq. 2. As anticipated, they represent the numbers of hits obtained by testing the rules in $\mathcal{R}_{tr}$ on each considered split. Regarding the operational setting, we consider $N_{op} = 1$, which is a suitable scenario when operational data are scarcely available, besides allowing to simplify the calculations.

The metrics driving ODD are the weighted mutual information $W\mu I$ and both l_1 and l_2 norms. For all metrics, the idea is to compare values computed in operation with the ranges achieved in training (*baseline*). An ODD occurs whenever the value in operation falls outside the baseline. The reason behind the choice of $W\mu I$ with respect to the classical μI is clearly explained in section 7.

3.4 Weighted mutual information

In this Section, we present the algorithm to define Weighted mutual information, first for defining a baseline on the training domain and then to perform the OOD while being at operational.

Table 1: Training numbers of hits table. Each column refers to a training split $tr_i \in TR$ and each row to a rule $r_i \in \mathcal{R}_{tr}$.

	tr_1	....	$tr_{N_{tr}}$
r_1	$h_1^{tr_1}$	...	$h_1^{tr_{N_{tr}}}$
r_2	$h_2^{tr_1}$	...	$h_2^{tr_{N_{tr}}}$
...	...	...	...
...	...	...	...
r_{N_r}	$h_{N_r}^{tr_1}$	...	$h_{N_r}^{tr_{N_{tr}}}$

In the training settings, as to Eq. 2, the matrix-like structure shown in Table 1 arises, where each column refers to a training split, each row represents a rule

[8] Samples can satisfy multiple rules and there may be operational samples satisfying none of the rules.

and the values are the numbers of hits obtained when rules are applied on the splits.

Based on that table, training baselines for weighted mutual information, as well as for l_1 and l_2 norms, are computed as described in Algorithm 1.

Algorithm 1: Weighted Mutual Information and Norms at Training Stage

---

i, j=1,...,N_{tr}, $i \neq j$
Input: Table 1
Output: Training baselines $W\mu I_{base}$ and ${l_p}^{base}$

1a. Define the weight associated with tr_i and tr_j, $\alpha_{i,j} \in (0,1)$, $\forall i$, $\forall j$:

$$\alpha_{i,j} = \frac{1}{N_r} \sum_{r=1}^{N_r} (|h_r^{tr_i} - h_r^{tr_j}|)$$

1b. Compute the weighted entropies $H(tr_i)$, $H(tr_j)$, $H(tr_i, tr_j)$, $\forall i$, $\forall j$:

$$H(tr_i) = -\sum_{r=1}^{N_r} [\alpha_{i,j} P(h_r^{tr_i}) \cdot log(\alpha_{i,j} P(h_r^{tr_i}))]$$
$$H(tr_j) = -\sum_{r=1}^{N_r} [\alpha_{i,j} P(h_r^{tr_j}) \cdot log(\alpha_{i,j} P(h_r^{tr_j}))]$$
$$H(tr_i, tr_j) = -\sum_{r=1}^{N_r} [\alpha_{i,j} P(h_r^{tr_i}, h_r^{tr_j}) \cdot log(\alpha_{i,j} P(h_r^{tr_i}, h_r^{tr_j}))]$$

2. Compute the weighted mutual information ($W\mu I$):

$$W\mu I(tr_i, tr_j) = [H(tr_i) + H(tr_j) - H(tr_i, tr_j)], \forall i, \forall j$$

3. Compute the baseline $W\mu I_{base}$:

$$W\mu I_{base} \doteq [\min_{i,j}(W\mu I(tr_i, tr_j)), \max_{i,j}(W\mu I(tr_i, tr_j))]$$

4. Compute l_p (p=1,2) norms:

$$l_p(tr_i, tr_j) = \left[\sum_{r=1}^{N_r} (|h_r^{tr_i} - h_r^{tr_j}|)^p\right]^{\frac{1}{p}}, \forall i, \forall j$$

5. Compute the baseline ${l_p}^{base}$ (p=1,2):

$$l_p^{base} \doteq [\min_{i,j}(l_p(tr_i, tr_j), \max_{i,j}(l_p(tr_i, tr_j))]$$

Similarly, considering the operational setting we can build the training-operational matrix as in Table 2.

Table 2: Training-operational number of hits table with $N_{op} = 1$. Columns refers to the training splits $tr_i \in TR$ and the operational split op_1, each row to a rule $r_i \in \mathcal{R}_{tr}$.

	tr_1	....	$tr_{N_{tr}}$	op_1
r_1	$h_1^{tr_1}$	...	$h_1^{tr_{N_{tr}}}$	$h_1^{op_1}$
r_2	$h_2^{tr_1}$	...	$h_2^{tr_{N_{tr}}}$	$h_2^{op_1}$
...	...	...	...	...
...	...	...	...	...
r_{N_r}	$h_{N_r}^{tr_1}$	...	$h_{N_r}^{tr_{N_{tr}}}$	$h_{N_r}^{op_1}$

Weighted mutual information and norms are then computed and compared to the training baselines, as described in Algorithm 2.

Algorithm 2: Weighted Mutual Information and Norms at Operational Stage

---

$i = 1, \ldots, N_{tr}$, $p = 1, 2$
Input: Table 2; baseline ranges $W\mu I_{base}$ and ${l_p}^{base}$
Output: ODD through $W\mu I$ and l_p

---

1. Compute the weighted entropies $H(tr_i)$, $H(op_1)$ and $H(tr_i, op_1)$ as done in the **Algorithm 1** (steps 1a-1b) $\forall i$
2. Compute the weighted mutual information ($W\mu I$):
$$W\mu I(tr_i, op_1) = [H(tr_i) + H(op_1) - H(tr_i, op_1)], \quad \forall i$$
3. Compute l_p norms:
$$l_p(tr_i, op_1) = \left[\sum_{r=1}^{N_r}(|h_r^{tr_i} - h_r^{op_1}|)^p\right]^{\frac{1}{p}}, \forall i$$

4.OoD detection:

IF $W\mu I(tr_i, op_1) \notin W\mu I_{base}$ *for the majority of* i THEN flag is *on*
IF $l_p(tr_i, op_1) \notin {l_p}^{base}$ *for the majority of* i THEN flag is *on*
IF *{at least one flag is on}* THEN op_1 is OoD

---

4 Platooning case study

The OoD detection methodologies proposed in this work have been tested on a case study addressing collision avoidance in vehicle platooning [18], which is one of the most celebrated applications in autonomous driving. A group of vehicles is interconnected via wireless, based on the Cooperative Adaptive Cruise Control [23]. The behavior of the platooning system is synthesised by the physical quantities pointed out in table 3. The physical quantities correspond to the features of the problem, which identify potential collisions in advance after a sudden brake.

Fig. 2 shows speed (left) and reciprocal distance (right) of three vehicles when braking force changes from 1000 N (no collision) to 1300 N (collision). We consider two datasets: the first one (LOW), is obtained by fixing an upper bound on the communication delay, i.e., $d \leq 0.4$ s; conversely, the second one (HIGH) is characterized by $d > 0.4$ s. LOW is chosen as the training domain (tr_{LOW}, in-distribution) and the reference ruleset is denoted with $\mathcal{R}_{tr_{LOW}}$; HIGH dataset constitutes the operational domain instead. A typical ODD problem is thus posed (between LOW and HIGH) as d has a significant impact on performance.

Table 3: Platooning features.

Symbol	Description
N	Number of vehicles
$F0$	Braking force applied by the leader
PER	Probability of packet loss
$d0$	Initial mutual distance between vehicles
$v0$	Initial speed
d	Communication delay in the inter-connection of vehicles

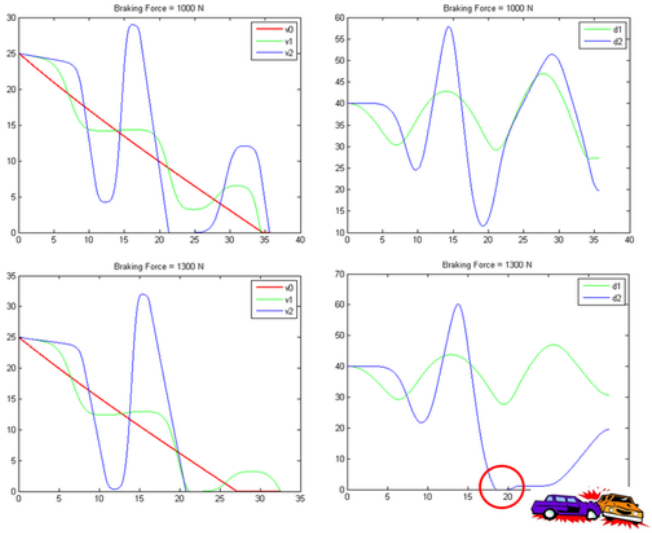


Fig. 2: Trend of the initial speed $v0$ (left) and initial mutual distance $d0$ (right) between three cars, when a different braking force $F0$ is applied. Image from [19].

The ODD has here a safety preserving role as it recognizes if the delay in operation is larger than the one in training. However, the algorithms are unaware that delay is the key to discriminating the datasets and they understand the ODD through the operational hits on $\mathcal{R}_{tr_{LOW}}$.

5 Results

In this Section, we show the results of the proposed approach for the platooning case study. The first row of Tables 4 and 5 reports the baseline ranges of the norms and mutual information metrics, respectively, which constitute the reference to infer possible OoD in operation. An even partial overlap between ranges in training and operation leads to a missed detection, i.e., a false negative (FN). A false positive (FP) consists of a wrong ODD for a training (in-distribution) bunch of samples.

The operational sample (column) of Table 2 constitutes a FN in case no OoD is declared; a sample (column) of Table 1 constitutes a FP in case OoD is declared. Tables are built as follows. Each column refers to a bunch of n_s=500 samples, with values reflecting the number of hits for the reference ruleset. We consider $N_{tr} = 50$ and $N_{op} = 1$ in Table 2. The total repetitions of the experiments for computing FNR and FPR is 2500. Example code and data for the experiments are available at the following link: https://github.com/giacomo97cnr/Rule-based-ODD.

As pointed out from Tables 4 and 5, good performance is registered for both $W\mu I$ and norms while μI runs worse.

Table 4: Algorithms 1 and 2: platooning. Norms.

Couples	l_1	FNR (l_1)	FPR (l_1)	l_2	FNR (l_2)	FPR (l_2)
$tr_{LOW} - tr_{LOW}$	[0.02, 0.12]		0%	[0.01, 0.04]		0%
$tr_{LOW} - op_{HIGH}$	[3.80, 3.90]	0%		[1.37, 1.39]	0%	

Table 5: Algorithms 1 and 2: platooning. μI and $W\mu I$.

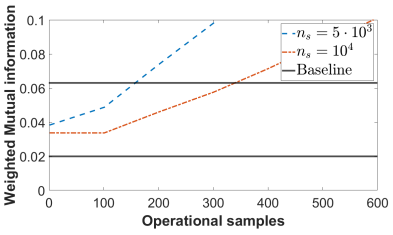
Couples	μI	FNR (μI)	$W\mu I$	FNR ($W\mu I$)	FPR ($W\mu I$)
$tr_{LOW} - tr_{LOW}$	[0.87,2.73]		[0.02,0.06]		0%
$tr_{LOW} - op_{HIGH}$	[0.04,0.97]	8%	[0.51,0.73]	0%	

6 Groupwise in operation

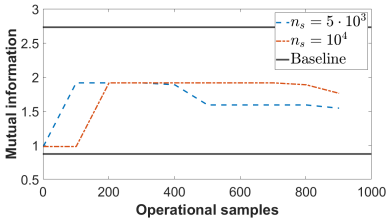
6.1 Incremental technique

The method collects a bunch of operational data before processing and classifying them as in or out of distribution. For this reason, it falls in the category of groupwise methods [12]. Differently from pointwise, groupwise confirms a type of situation (in or out), without relying on a single point that could be a spike in a steady trend. The collection phase in operation does not imply that one would wait for new n_s samples to register a new split and to provide the ODD. Splits are generated continuously, as soon as new samples are collected. Like in incremental techniques, once a new sample is available, a new (operational) bunch of n_s samples is built, by adding the new sample and by disregarding the most far away point in the past (of n_s positions). In turn, the bunch leads to the split collection, by computing the inherent hits on the ruleset. The process assumes a sample-by-sample incremental time window, over which the following operations are performed. A new data bunch is firstly registered, a new split is calculated and a new ODD is then derived.

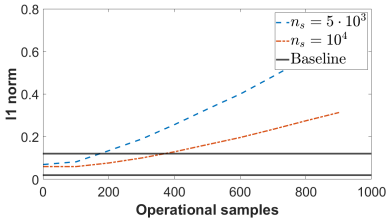
6.2 Incremental groupwise in operation



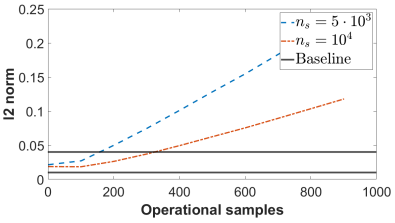
(a) Incremental groupwise - $W\mu I$



(b) Incremental groupwise - μI



(c) Incremental groupwise - l_1



(d) Incremental groupwise - l_2

Fig. 3: Trends of the metrics when operational data arrive sample-by-sample, in an incremental way.

The following experiments highlight the ODD when replacing in-distribution data with out-of, in a sample-by-sample incremental way. The analysis is relevant to the tracking of the OoD drift with both precision and measurement of distributions proximity. Every figure in the following contains the baseline derived at design time; the curves represent the behaviours of the metrics in operation. Increasing time windows with $n_s = 5 \cdot 10^3$ and $n_s = 10^4$ samples are used to emphasize the speed of the drift inference over time. The time size of the windows depends on the time granularity of the arrival of the points in operation; for this reason, the x-axis is not time, but it refers to the progressive identifier of the operational samples. The drift starts at time zero, that means the first operational sample derives from the OoD and previous points (of the window) are compliant with training conditions. As soon as the window collects more data (over the last n_s points), it senses more information about the OoD. As to the $W\mu I$ metric, the results confirm that the shorter the window, the faster the detection. On the other hand, the μI metric experiences a noise that can have different meanings as detailed later on. In the case of platooning $W\mu I$ and both l_p norms (p=1,2) (Fig. 3a, Fig. 3c and Fig. 3d) need at least 150 samples to exit the baseline with $n_s = 5 \cdot 10^3$ and twice as many samples with $n_s = 10^4$; so these metrics match the ODD and are coherent with previous results (table 4), while μI is stuck in the baseline (Fig. 3b).

7 Rationale of mutual information modification

When comparing two histograms, mutual information (μI) is useful to identify their dependence but it does not capture the differences among their values. Suppose we get these three histograms A, B and C (Table 6)

Table 6: Example of $W\mu I$ against μI

	A	B	C
r_1	0.156	0.171	0.314
r_2	0.172	0.186	0.329
r_3	0.328	0.314	0.171
r_4	0.349	0.329	0.186

According to μI, histograms A and B are dependent in the same measure as A and C are; but B and C are completely different (they have same values but in different positions). Since each row of the tr and op histograms corresponds to a rule, using μI may have a detrimental effect as the rule hits contain the information peculiar to the OoD. Introducing weights we can overcome this disadvantage both capturing the dependence and the differences between values. The correction consists of weighting the probabilities (used in entropy calculations) through the average of hits differences in each rule/row; this leads to $\alpha_{i,j}$ quantities in algorithm 1. In the case of weighted mutual information the

more the vectors are dependent and similar, the more $W\mu I$ goes towards zero. In the specific case of platooning situations in which similar values of training and operational were discovered in different positions were over 300 times (Table 7).

Table 7: Example of a training and an operational histogram in platooning; some values are roughly the same, but in different positions: r_7-r_6; r_9-r_{13}; r_{11}-r_8; r_{13}-r_3 (evidenced by the colors in the table).

	tr	*op*
r_1	0.1666	0
r_2	0.1689	0
r_3	0.1495	0.2535
r_4	0.1423	0.7914
r_5	0.1382	0.2379
r_6	0.0598	0.0999
r_7	0.0903	0.1918
r_8	0.0845	0.4262
r_9	0.0499	0
r_{10}	0.0303	1
r_{11}	0.4210	0
r_{12}	0.3383	0.0443
r_{13}	0.2516	0.0498
r_{14}	0.1117	0
r_{15}	0.0860	0.0171
r_{16}	0.717	0.0444

8 Conclusion

The paper introduced an innovative technique for the detection of out-of-distribution based on explainable AI. The approach has been validated through measures of proximity between in and out of distribution and it is corroborated by a real-world scenario. Future extensions comprise further testing on image datasets, including alternative ways to infer in-distribution behaviour.

Acknowledgements This work was supported in part by REXASI-PRO H-EU project, call HORIZON-CL4-2021-HUMAN-01-01, Grant agreement ID: 101070028. The work was also supported by Future Artificial Intelligence Research (FAIR) project, Recovery and Resilience Plan ("Piano Nazionale di Ripresa e Resilienza"), Spoke 3 - Resilient AI. G. De Bernardi PhD is partially funded by Collins Aerospace.

References

1. Bitterwolf, J., Meinke, A., Augustin, M., Hein, M.: Breaking down out-of-distribution detection: Many methods based on ood training data estimate a combination of the same core quantities. In: International Conference on Machine Learning. pp. 2041–2074. PMLR (2022)
2. Cabitza, F., Campagner, A.: The need to separate the wheat from the chaff in medical informatics: Introducing a comprehensive checklist for the (self)-assessment of medical ai studies. International Journal of Medical Informatics **153**, 104510 (2021). https://doi.org/https://doi.org/10.1016/j.ijmedinf.2021.104510, https://www.sciencedirect.com/science/article/pii/S1386505621001362
3. DeVries, T., Taylor, G.W.: Learning confidence for out-of-distribution detection in neural networks (2018)
4. Dinh, T.Q., Xiong, Y., Huang, Z., Vo, T., Mishra, A., Kim, W.H., Ravi, S.N., Singh, V.: Performing group difference testing on graph structured data from gans: Analysis and applications in neuroimaging. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(2), 877–889 (2022). https://doi.org/10.1109/TPAMI.2020.3013433
5. Concepts of design assurance for neural networks codann. Standard, European Union Aviation Safety Angency, Daedalean, AG (Mar 2020), also available as https://www.easa.europa.eu/sites/default/files/dfu/EASA-DDLN-Concepts-of-Design-Assurance-for-Neural-Networks-CoDANN.pdf
6. Easa concept paper: First usable guidance for level 1 machine learning applications, a deliverable of the easa ai roadmap. Standard, European Union Aviation Safety Angency, Daedalean, AG (Apr 2021), also available as https://www.easa.europa.eu/easa-concept-paper-first-usable-guidance-level-1-machine-learning-applications-proposed-issue-01pdf
7. Guarrera, M., Jin, B., Lin, T.W., Zuluaga, M.A., Chen, Y., Sangiovanni-Vincentelli, A.: Class-wise thresholding for robust out-of-distribution detection. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 2836–2845 (2022). https://doi.org/10.1109/CVPRW56347.2022.00321
8. Guénais, T., Vamvourellis, D., Yacoby, Y., Doshi-Velez, F., Pan, W.: Bacoun: Bayesian classifers with out-of-distribution uncertainty (2020)
9. Heidecker, F., Breitenstein, J., Rösch, K., Löhdefink, J., Bieshaar, M., Stiller, C., Fingscheidt, T., Sick, B.: An application-driven conceptualization of corner cases for perception in highly automated driving. In: 2021 IEEE Intelligent Vehicles Symposium (IV). pp. 644–651 (2021). https://doi.org/10.1109/IV48863.2021.9575933
10. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. arXiv preprint arXiv:1610.02136 (2016)
11. Hsu, Y.C., Shen, Y., Jin, H., Kira, Z.: Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10948–10957 (2020). https://doi.org/10.1109/CVPR42600.2020.01096
12. Jiang, D., Sun, S., Yu, Y.: Revisiting flow generative models for out-of-distribution detection. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=6y2KBh-0Fd9
13. Lee, K., Lee, K., Lee, H., Shin, J.: A simple unified framework for detecting out-of-distribution samples and adversarial attacks. Advances in neural information processing systems **31** (2018)

14. Liang, S., Li, Y., Srikant, R.: Principled detection of out-of-distribution examples in neural networks. corr abs/1706.02690 (2017). arXiv preprint arXiv:1706.02690 (2017)
15. Liang, S., Li, Y., Srikant, R.: Enhancing the reliability of out-of-distribution image detection in neural networks (2020)
16. Liu, W., Wang, X., Owens, J., Li, Y.: Energy-based out-of-distribution detection. Advances in Neural Information Processing Systems **33**, 21464–21475 (2020)
17. Mirasierra, V., Mammarella, M., Dabbene, F., Alamo, T.: Prediction error quantification through probabilistic scaling. IEEE Control Systems Letters **6**, 1118–1123 (2022). https://doi.org/10.1109/LCSYS.2021.3087361
18. Mongelli, M.: Design of countermeasure to packet falsification in vehicle platooning by explainable artificial intelligence. Computer Communications **179**, 166–174 (2021). https://doi.org/https://doi.org/10.1016/j.comcom.2021.06.026, https://www.sciencedirect.com/science/article/pii/S0140366421002504
19. Mongelli, M., Ferrari, E., Muselli, M., Fermi, A.: Performance validation of vehicle platooning through intelligible analytics. IET Cyber-Physical Systems: Theory & Applications **4**(2), 120–127 (2019)
20. Muselli, M.: Switching neural networks: A new connectionist model for classification (01 2005). https://doi.org/10.1007/11731177_4
21. Parodi, S., Manneschi, C., Verda, D., Ferrari, E., Muselli, M.: Logic learning machine and standard supervised methods for hodgkins lymphoma prognosis using gene expression data and clinical variables. Health Informatics Journal **24** (06 2016). https://doi.org/10.1177/1460458216655188
22. Road vehicles safety of the intended functionality pd iso pas 21448:2019. Standard, International Organization for Standardization, Geneva, CH (Mar 2019)
23. Shladover, S.E., Nowakowski, C., Lu, X.Y., Ferlis, R.: Cooperative adaptive cruise control: Definitions and operating concepts. Transportation Research Record **2489**(1), 145–152 (2015)
24. Sun, Y., Ming, Y., Zhu, X., Li, Y.: Out-of-distribution detection with deep nearest neighbors. arXiv preprint arXiv:2204.06507 (2022)
25. Vernekar, S., Gaurav, A., Abdelzad, V., Denouden, T., Salay, R., Czarnecki, K.: Out-of-distribution detection in classifiers via generation (2019)
26. Zhou, K., Yang, Y., Qiao, Y., Xiang, T.: Mixstyle neural networks for domain generalization and adaptation (2021)