Robustness Testing of Machine Learning Families using Instance-level IRT-Difficulty

Raül Fabra-Boluda[1], Cèsar Ferri[1], Fernando Martínez-Plumed[1] and M. José Ramírez-Quintana[1]

[1]*Universitat Politècnica de València*

Abstract

Performance evaluation of Machine Learning systems have been usually limited to performance measures on curated and clean datasets that may not properly reflect how robustly these systems can operate in real-world situations. One key element in this understanding of robustness is *instance difficulty*. The effect of instance difficulty on robustness could be understood as how unexpected would be that a customary system fails on a particular instance of certain difficulty. In order to provide further understanding on this issue, we estimate IRT-based instance difficulty for an illustrative set of supervised tasks and we implement and test perturbation methods that simulate noise and variability depending on the type of input data. With this, we evaluate the robustness of different families of machine learning models, which we select and characterise according to their behaviour. The preliminary results of this *work in progress* allow us to define a novel taxonomy based on the robustness of the different models and the difficulty of the instances addressed. This study is a significant step towards exposing vulnerabilities of particular families of machine learning models.

Keywords

Machine Learning, Robustness, Robustness Taxonomy, IRT, Noise, Adversarial

1. Introduction

The success of AI and specially Machine Learning (ML) technologies caused these type of systems to spread across many applications from different domains, e.g., medical, financial, social or autonomous transport, among others [1, 2, 3]. These applications form part of our daily life and shapes our lifestyle. They recommend us music to listen or people to establish career/social relationships with. They diagnose our health and monitor our finance. Given this scenario, there is an obvious need of more robust ML systems.

Robustness is defined by the IEEE standard glossary of software engineering terminology [4] as: *The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions.* In the context of ML, robustness measures the resilience of a system towards perturbations in any of its components (the data, the learning program, or the framework) [5]. Earlier works [6] assessed model robustness by perturbing instances in the training set, test set, or both. A general way to perturb instances is adding noise, a method that has been extensively applied in the adversarial ML field for the generation of adversarial examples ([7]). This is why most of the research in ML robustness focused on measuring the robustness of systems with adversarial samples [7, 8].

On the other hand, there are a wide range of factors that can affect the robustness of a model [9] and the difficulty of the instances (intrinsic or extrinsic) [10] is one of the most relevant [11]. Robustness must be based on knowing where and why the model fails, avoiding highly unexpected failure, and difficulty is one key element in this understanding. Therefore, the question we want to analyse is whether the performance of a particular model varies equally distributed across difficulties. We may also analyse model robustness by perturbing data and locating those examples that are more likely to change their predicted label under certain amount of noise, i.e., those instances for which the model is less robust, and hence, more prone to cause a vulnerability. We can expect that the more difficult an instances is, the more likely it is to change their predicted label under minor perturbations.

Estimating the difficulty of instances is another problem we need to address. We may just calculate the average error of a set of systems for each instance as a proxy for difficulty [12]. However, the use of a population of systems entails some risks as well. For instance, if the population contains a non-conformant system (failing on the easy instances and succeeding in some of the hard ones), it may lead to very unstable difficulty metrics. A solution to this problem was introduced several decades ago, and it is known as item response theory (IRT) [13], where difficulty is inferred from a matrix of items (instances) and respondents (systems), giving more relevance to conformant systems. In addition, IRT gives a scaled metric of difficulty that follows a normal distribution and can be compared directly against the *ability* of a system.

In this *work in progress* paper, we present, as a proof of

EBeM'22: Workshop on AI Evaluation Beyond Metrics, July 25, 2022, Vienna, Austria

rafabbo@dsic.upv.es (R. Fabra-Boluda); cferri@dsic.upv.es (C. Ferri); fmartinez@dsic.upv.es (F. Martínez-Plumed); mramirez@dsic.upv.es (M. J. Ramírez-Quintana)

concept, an evaluation setting to analyse the robustness of different ML models empirically, considering the difficulty of the instances. We also perform a hierarchical clustering to derive taxonomies of ML models according to their robustness. The setting is general as we employed datasets from different domains, a wide set of representative ML techniques, and an instance perturbation function that introduce random noise with no specific goal. Therefore, it can be adapted to more specific problems by changing the datasets, models and perturbation function to the domain of interest. In this general evaluation framework, we measure the robustness of a model as the agreement modulo instance difficulty between the output of the model for the original and the perturbed test sets.

The paper is structured as follows. In Section 2 we review part of the literature related to the assessment of robustness in presence of noise, the estimation of instances difficulty and a taxonomy of machine learning techniques derived from a notion of behavioral similarity. Section 3 describes the method we developed to assess model robustness. We apply that method and describe the experiments in Section 4. Finally, 5 concludes the paper.

2. Background

In this section we revisit some key concepts related to model robustness, instance IRT-based difficulty and the definition of behavioural taxonomies of machine learning techniques.

2.1. Robustness in a noisy framework

Robustness is one of the properties of ML systems that characterise their behavior [5], being particularly suitable for checking whether the system behaves as expected to changes in the data. A common way to simulate those changes is injecting noise into the data, given that real world data often contain some degree of noise [14].

In the literature, there is a large number of approaches for adding artificial noise to datasets [15, 16, 17]. Usually, noise is introduced by perturbing the values of the attribute(s) (attribute noise) or perturbing the class label (label noise). A general technique for the injection of attribute noise consists in perturbing the instance attribute(s) value following a well-known distribution (e.g., a Gaussian distribution) for numerical attributes, or randomly choosing a different value for categorical attributes [14, 18, 19]. This is the method usually used in adversarial ML, where adversarial examples are created by slightly modifying attribute values of examples correctly classified by the model to craft new instances (ideally indistinguishable from the original ones) for which the output of the model changes. Attribute noise has been also used in different approaches to improve the robustness of models to adversarial examples. For instance, [20] shows that the injection of noise in the training dataset results in models more robust to attacks since they are able to detect the perturbed instances beforehand. Similarly, adding adversarial instances to the training set can improve the robustness of neural nets [8], and make a Speech Emotion Recognition system more robust [21]. On the other hand, artificial label noise is useful to simulate wrongly annotated instances or other sources of data corruption. Label noise has been used, for instance, to evaluate robustness in computer vision applications [22]. Additionally, label noise in the training set can be employed to enhance the robustness of models, for instance, by reducing errors derived from overfitting [23].

There has been proposed different ways of assessing the robustness of a model in noise environments. The most general method consists in measuring the correctness loss of models with noise in the data, with respect to the case without noise, regardless of where the noise is located (training set or test set). For noise in the training set, the metrics used to quantify the loss are the standard classification metrics such as accuracy and F-measure [14] and the Equalised Loss of Accuracy (a metric for measuring a classifier's noise robustness [24]). For the sub-field of adversarial robustness (where noise is used in the test set to generate the adversarial examples) there has been used specific metrics such as adversarial accuracy [8]. There are other general techniques for evaluating robustness, including mixed integer programming [25], abstract interpretation [26], and symbolic execution [27, 28, 29].

In this work, we are interested in studying how the robustness of a model is affected by the distortion produced by injecting different levels of noise into test instances considering the difficulty of the perturbed instances. As far as we know, instance difficulty has not yet been taking into account to evaluate the robustness of models.

2.2. Instance difficulty

Difficult instances may cause problems during AI system development, especially for models that are trained. These instances (e.g., usually associated with noise, outliers or decision boundaries) have been blamed for overfitting, lack of convergence or both. Handling these sort of anomalies has been addressed in a number of different ways trying to prevent overfitting. However, these approaches usually try to identify anomalies or mislabeled instances but without defining what characterise them. For instance, in [30] instances that are hard to classify are identified through *instance hardness* metrics. These metrics try to characterise the level of difficulty of each input sample following a (populational) empirical definition based on the classification behaviour over the instances to be evaluated. If we move out of the field of machine learning (e.g., on computer vision or NLP-related tasks), we find an area still to be explored, where the different works are limited to analysing global image properties (e.g., salience, memorability, photo quality, tone, colour, texture, etc.) [31, 32, 33], or, in the case of NLP, they are based on lexical readability and richness [34, 35].

All the approaches above are specific to a domain and in many cases also anthropocentric. A completely different approach is Item Response Theory, a well-developed subdiscipline in psychometrics [36], only recently brought to AI and machine learning [37, 12, 38, 39, 40]. In IRT, the probability of a correct response for an item is a function of the respondent's ability and some item's parameters. The respondent solves the problem and the item is the problem instance itself. We focus on the dichotomous models where the response can be either correct or incorrect.

Let U_{ji} be a binary response of a respondent j to item i, with $U_{ji} = 1$ for a correct response and $U_{ji} = 0$ otherwise. For the basic one-parameter logistic (1PL) IRT model, the probability of a correct response given the examinee's ability is modelled as a logistic function:

$$P(U_{ji} = 1|\theta_j) = \frac{1}{1 + exp(-a_i(\theta_j - b_i))} \tag{1}$$

The parameter θ_j is the ability or proficiency of j and b_i is the difficulty of i. If ability θ equals item difficulty b_i, then there are even odds of a correct answer (cutting the curve as exactly 0.5, as the light blue dashed shows in Fig. 1. For each item, the above model provides an *Item Characteristic Curve (ICC)* (see Fig. 1, left), characterised by *difficulty* (b_i), which is the location parameter of the logistic function.

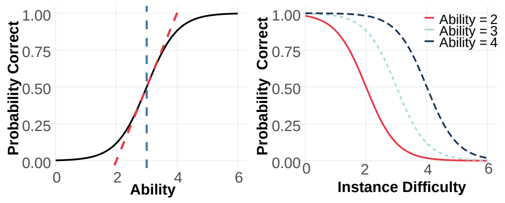

Figure 1: Left: Example of a 2PL IRT ICC curve, with slope $a = 2$ in red and location parameter $b = 3$ in blue. Right: Example of SCC curves with different abilities.

As all IRT models assume one single parameter for the respondent, their dual plots (known originally as person characteristic curves, here renamed as system characteristic curves (SCC), also follow a logistic function (see Fig. 1, right). Respondents who tend to correctly answer the most difficult items will be assigned to high values of ability. Difficult items in turn are those correctly answered only by the most proficient respondents. From this understanding and some common assumptions (ability and difficulty following some particular normal distributions), the latent variables can be inferred from a table of item-respondent pairs U_{ji}. Some two-step iterative variants of maximum-likelihood estimation (MLE), such as Birnbaum's method [41], can be used to infer all the IRT parameters.

IRT difficulty is characterised by being *system-independent* an *domain-generic* unlike the other metrics described above [11]. It also has some advantages over using average performance as a metric of difficulty, in terms of distribution, stability and predictability, as has been studied in the literature of IRT.

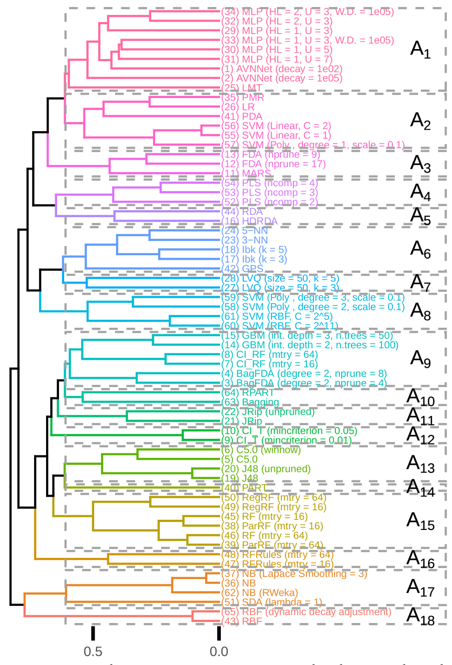

Figure 2: Dendrogram representing the hierarchical clustering (18 groups) of ML models. From [42].

2.3. Behaviour-based Machine Learning families

One classical way of characterising the rich range of machine learning techniques is by defining 'families', according to their formulation and learning strategy (e.g., neural networks, Bayesian methods, etc.) [43, 44, 45]. However, this taxonomy of learning techniques does not take into account the intrinsic behaviour of the models (measured as output agreement), especially considering predictions in sparse zones where insufficient training data was available. If we want to characterise the robustness of ML models, we need to analyse a diverse set of models, as many as possible, under different parameters as well. In this regard, [42] derived a taxonomy of ML techniques for classification, where families are clustered according to their degree of (dis)agreement in behaviour, i.e., the differences between models on how they distribute the output class labels along the feature space. We considered both dense and sparse zones (where training data is scarce or inexistent), using Cohen's kappa statistic [46]. While in dense areas differences between models may be difficult to find, in sparse areas the algorithms diverge significantly, and unveil the characteristic behaviour of the trained models using those techniques.

The methodology was based on comparing the behaviour of 65 different learning models (including hyperparameter variations), performing a pairwise comparison (based on Kappa) and averaging the results obtained for 75 datasets. For grouping in families, authors applied a hierarchical clustering so that the models that presented similar behaviour fell in the same cluster, which is considered a model family (see the 18 different families obtained in Figure 2). This method is useful to objectively quantify how different two models (or model families) are.

3. Empirical Methodology

In this section we describe the experimental methodology performed to obtain a taxonomy of classification algorithms according to their robustness. We start introducing the set of representative datasets and learning models we have employed. Then, we describe how we estimate instance difficulty, the approach followed to introduce noise in the data and, finally, how we define the taxonomy of ML families. All the data, code, complete experiments, plots and results can be found in https://github.com/rfabra/family-robustness.

3.1. Data and Classifiers

In order to estimate IRT-difficulty, we need to find benchmarks that had instance-wise results of a good number of models. It is recommended to have at least 10-20 responses per item [47]. More importantly, we need the *instance-wise results*, i.e., a $|J| \times |I|$ matrix with the performance of each system $j \in J$ for each instance $i \in I$. Finding experiments not reported in an aggregated way was not an easy task. As an exception to the instance-wise result problem, we find platforms such as OpenML [48], a repository in which AI researchers and practitioners can share data sets and results in as much detail as possible. The platform also provides several curated datasets such as OpenML-CC18, from which we address a set of 3 benchmarks for supervised learning (see Table 1). The selection is guided by the illustrative character (with the preliminary objective of testing the effectiveness of our setting), but is also limited by those benchmarks where there is a sufficiently large number $|I|$ of examples (articles) and $|J|$ of models (respondents).

Dataset	# Instances	# Features	# Classes
letter	20000	16	26
optdigits	5620	64	10
wall-robot-navigation	5456	4	4

Table 1
List of datasets for the experiments.

Regarding ML models, we employed a set of 18 ML models from different ML families (see Table 2), derived in [42]. These 18 model families were obtained from a pool of 65 models learned and evaluated on a wide range of datasets for categorisation into different families, as described in the section 2.3. For each family we selected a single model, its centroid (i.e., representing the center of each family cluster), assuming it to be representative of its family. Thus, we can assume that the 18 selected models are diverse enough to provide a wide view of how different model families behave in terms of robustness.

3.2. Estimation of Difficulty

As mentioned in Section 3.1, in order to estimate the difficulty of the instances, we first check that for each benchmark selected from OpenML there is at least 10-20 reponses (model evaluations) per item/feature (e.g., we would need between 640 and 1280 responses for *optdigits*) and that they are sufficiently diverse (different architectures or technologies). Next, we obtain their responses for *unseen* instances (e.g., we will be using the test folds, so it is actually test performance, even if we cover the whole dataset). This will be our $|J| \times |I|$ matrix U with all binary responses U_{ji}.

We follow the recommendations from [12] for the application of IRT. In practice, for generating the IRT models, we used the `MIRT` R package [49], using Birnbaum's method, as explained above. The package `MIRT` (as many other IRT libraries) output indicators about the goodness of fit which can be used to quantify the discrepancy between the values observed in the data (items) and the values expected under the statistical IRT model. Item-fit statistics may be used to test the hypothesis of whether the fitted model could truly be the data-generating model or, conversely, we expect the item parameter estimates to be biased. In practice, an IRT model may be rejected on the basis of bad item-fit statistics, as we would not be reasonably confident about the validity of the inferences drawn from it [50]. In the present case, none of the estimated models were discarded because of bad item-fit statistics or inconsistency in their results.

Technique	Parameters	id
C5.0		C5.0
Cond. Inf. Tree	mincriterion = 0.05	CI_T
Flex. Disc. Analysis	degree = 1, nprune = 17	FDA
Stoch. Grad. Boosting Machine	interaction.depth = 2, n.trees = 50	GBM
JRip		JRip
K-Nearest Neighbor	K = 3	3NN
Learning Vector Quant.	size = 50, K = 3	LVQ
MultiLayer Perceptron	1 hidden layer, 7 neurons	MLP
Multinomial Log. Regression		MLR
Naive Bayes		NB
PART		PART
Radial Basis Function Network		RBF
Regularised Discriminant Analysis		RDA
Random Forest	mtry = 64	RF
RPART		RPART
Part. Least Squares	ncomp = 3	PLS
SVM	Poly, degree = 2	SVM
RFRules	mtry = 64	RFRules

Table 2
List of the 18 models employed for the experiments, along with the parameters used.

3.2.1. System Characteristic Curves

One of the most powerful visualisation tools that derives from difficulty is what we call system characteristic curves (SCC) (Fig. 1, right). Inspired by the concept of person characteristic curve previously developed in IRT, a SCC is a plot for the response probability (e.g., accuracy, kappa, etc.) of a particular classifier as a function of the instance difficulty. For producing the SCC, we divide the instances in bins according to difficulty. For each bin, we plot on the x-axis the average difficulty of the instances in the bin and on the y-axis we plot the performance metric selected.

3.3. Introduction of Noise

We need a method to generate noise, representative and general enough, so that the experimental results can be adapted to other noise settings, e.g., to include adversarial attacks. Hence, we will work directly with noise levels, assuming that they are mapped from contexts. Noise is generated randomly by using some well-known probability distributions, following a similar procedure as in [16]. Instances are perturbed by changing their attribute values into a range of possible values. The process to select among the possible values depends on whether the attribute is nominal or numerical:

- **Numerical attributes**: Let v be the level of noise to be injected into a numerical attribute at, and σ the standard deviation of all values of at. Then, a value x in at is modified as $x' \sim N(x, \sigma \cdot v)$, i.e., we follow a normal distribution using x as mean and σ multiplied by the noise level v as standard deviation.
- **Nominal attributes**: Let $\{at_1,...,at_m\}$ be the set of the m possible values of a nominal attribute at, and p the vector that represents the empirical distribution of at, that is, $p = (p_{at_1}, \dots, p_{at_m})$, where, p_i is the frequency of value i. Consider we have an instance of value $x = at_j$ in at, we represent as the vector $t = (t_{at_1}, \dots, t_{at_m})$ with $t_{at_i} = 0 \ \forall i \in \{1..m\}, i \neq j$, and $t_{at_j} = 1$. To insert a noise level v, we calculate $\alpha = 1 - e^{(-v)}$, and then compute a new vector of probabilities $p' = \alpha \cdot p + (1 - \alpha) \cdot t$. Finally, we use p' in order to sample the new value x' of the attribute.

For the experiments, we will generate noisy datasets (test set) using a noise level $v = 0.2$. We vary the proportion of perturbed instances δ in each bin, from $\delta = 0$ (keeping unperturbed the original test set) to $\delta = 1$ (perturbing the whole test set). This is performed under a 5-fold cross validation setting. For each model, we will compare its predictions on the original test set with the predictions of each of the noisy test sets, by means of the Kappa metric, as we describe below.

3.4. Model robustness to noise and difficulty

We compare the behaviour of ML models from different families by classifying the same test set from a particular benchmark, to which we introduce different levels of noise. The more the behaviour of a model changes under noise, the less robust it is. This difference in behaviour can be measured with Cohen's Kappa metric [46]. More concretely, given $\mathcal{T}$ the domain of all data sets we can create from all possible inputs, a test set $T \in \mathcal{T}$, a perturbation function $\phi : \mathcal{T} \rightarrow \mathcal{T}$ to introduce noise into a data set, the perturbed test set $T' = \phi(T)$, the predictions of a model M for the original test $y_M = M(T)$, the predictions of a model M for the perturbed test $y'_M = M(T')$ and two models M_1 and M_2 learned on the same data, the model M_1 is considered more robust than model M_2 if

$$\kappa(y_{M_1}, y'_{M_1}) > \kappa(y_{M_2}, y'_{M_2})$$

Thus, we employ the Kappa as a measure of similarity between the predictions of a model on the original and the perturbed test sets. It is important to notice that we are not accounting for the real class label, since adding noise to the input attributes of an instance implies that the actual class is probably not the same as it was originally. Instead, we compare the model predicted labels for the original test set (without noise) with the ones predicted for the noisy test sets. Our goal is not to determine the well-performance of a model to solve a task, but to assess how the behaviour of the model changes under different levels of noise applied to instances of different levels of difficulty. As we want to analyse whether the model robustness may vary depending on the difficulty of the instances addressed, we estimated the difficulty of each instance in the dataset following the procedure described above. Later, we grouped instances into difficulty bins to analyse the robustness (to produce SCCs), as explained above.

Analysing the data from the SCCs for different models we also derive a ML robustness model taxonomy attending at the different shapes of the SCCs and models' behaviour. In this regard, for each dataset, we built a matrix where each row represents a model and each column represent a combination of difficulty bins and proportion of noisy instances per bin. Each element represents the similarity (i.e., the kappa metric) between the predictions of the model for the original test set and the predictions for each noisy test and bin. By averaging these across all the datasets, we may perform a hierarchical clustering with the aim of obtaining different grouping of models by robustness, showing the similarity between different ML families, in a data-driven fashion.

3.5. Experimental questions

Once the experimental methodology is clear, we now want to investigate the relationship between the robustness of the models and the difficulty of the instances, the latter having been altered with different levels of noise. For this, we set 3 experimental questions. **Q1**: How do difficulties distribute per benchmark for the IRT-difficulty metric estimated? **Q2**. Can we see differences in robustness for different models based on the difficulty of the instances? **Q3**. Can we group models by robustness?

4. Experiments

4.1. Setup

We employed R language with caret package [51] to carry out our experiments, i.e., training and evaluation of the models. All the models were learnt from scratch, so we did not used any pre-trained model. We used the MIRT R package [49] for estimating IRT 1PL models. To feed the IRT method, we obtained the predictions from a wide variety of models by using OpenML API [52]. In total, we employed the predictions of (up to) 2000 evaluations per dataset.

4.2. Results

IRT difficulties are built to approximately follow a normal distribution with standard deviation 1 but different locations depending on the dataset. When it comes to the item difficulty parameters, what you find acceptable depends very much on the purpose of your test and the population of interest. For instance, values around 1 are typical in educational measurement. In health measurement, however, these values are usually much higher and around 4. In our case, when addressing ML benchmarks, difficulty values around -3 and 3 are the norm (see [12]). For this reason, we decided to remove those instances whose difficulty is out of the range $[-6, 6]$, which are considered outliers. This happened in all benchmarks for very easy instances for which all techniques are correct, never affecting more than 0.1% of the instances. Figure 3 shows the IRT-difficulty distribution per benchmark, with a standard deviation around 1 (as expected). In terms of location (**Q1**), the *letter* benchmark contains more difficult instances (mean difficulty of -1.50 ± 0.92) than the others (-1.92 ± 0.67 for *optdigits* and -2.36 ± 0.9 for *wall-robot navigation*). Although the distribution is generally normally distributed, the *wall-robot-navigation* dataset presents a higher number of difficult instances, skewing the distribution to the right. This may be due to the diversity of the population of systems used for the difficulty estimation (similar cases can be observed in [11]).

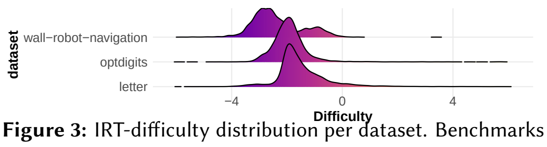

Figure 3: IRT-difficulty distribution per dataset. Benchmarks sorted by average difficulty.

Regarding **Q2**, for each technique in Table 2, we compare its predictions on the original test set (for each dataset in Table 1) with the predictions of each of the noisy test sets, by means of the Kappa metric. The SCCs produced (using Kappa values on the y-axis) are shown in Figure 4. Obviously, Kappa takes values equal to 1 when the test set is not perturbed ($\delta = 0$), since we are comparing the output labels of the different trained model with themselves. As we increase the amount of perturbed instances (the same proportion for each bin of difficulty), we can appreciate differences in the behaviour for the techniques analysed.

As expected, the most difficult instances are those that are more sensitive to noise, and this can be seen in terms of the level of performance of the different techniques for the most difficult instance bins. This behaviour may indicate that these instances are located close to the decision boundary or regions with class overlap, so the behaviour for most techniques is more unpredictable in those regions that in easier ones. In general, we may find some patterns of behaviour for different sets of techniques. First, we identify cases where robustness decreases non-linearly with increasing levels of difficulty. This is the most common case, but with differences in robustness variations for different models and datasets (see, e.g., CI_T, FDA, 3NN, MLP , MLR or SVM). Second, we also see cases

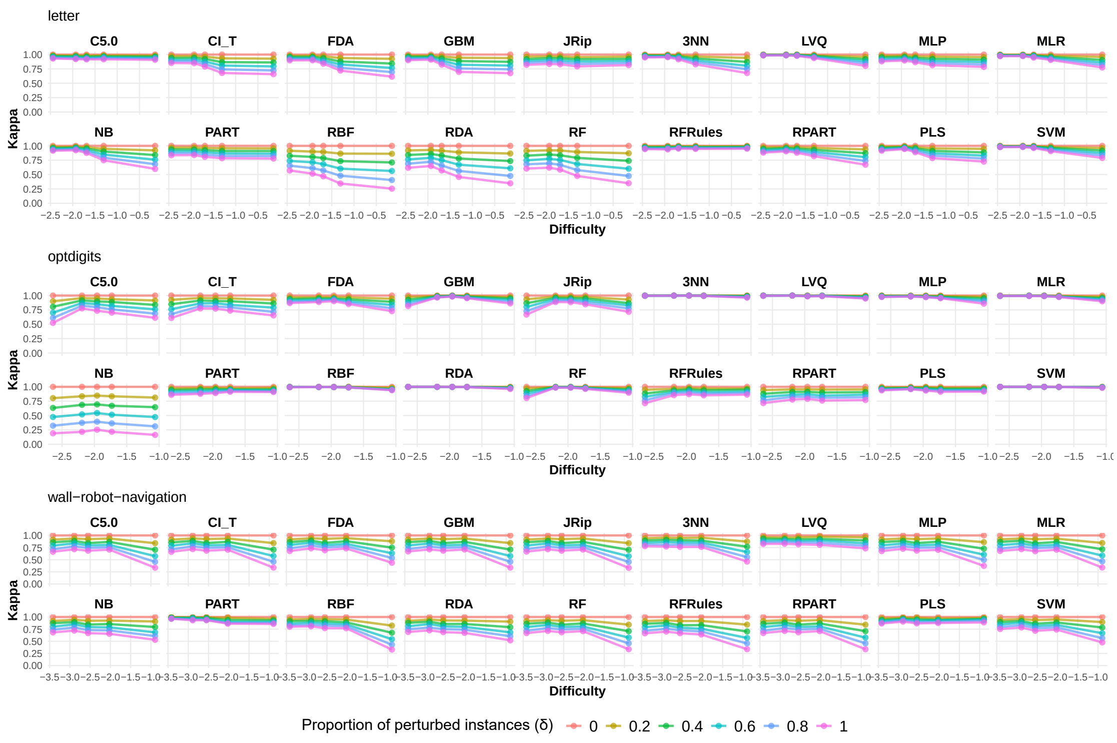

Figure 4: Kappa vs Difficulty for different models and benchmarks, varying the proportion of instances for each difficulty bin.

where robustness is mostly affected by the noise level and less by the difficulty of the instances (see, e.g.,NB, RBF or RDA). Finally, there are cases in which robustness is barely altered by either difficulty or noise level (see, e.g., PLS, PART or LVQ).

On the other hand, if we analyse the results at the dataset level, we see that the behaviour of some techniques changes significantly. For instance, techniques such as C5.0, CI_T and JRip for the dataset *optdigits* exhibit an interesting behaviour. These techniques seem more prone to change theirs predictions in easy instances than medium (even hard) instances. Analysing the results in more detail, we have seen that this is due to the class distributions in those more easy bins: these bins are usually composed of many instances of a single class (usually the majority class), but these instances may be misclassified as we increase the amount of noise, thus reflecting a drop in the Kappa value.

This is the case, for instance, with the JRip model learnt on the *optdigits* dataset. The first bin is composed of 479 instances of the class “6”, without introducing any noise ($\delta = 0$). After perturbing all the instances ($\delta = 1$), only 256 instances in this bin are predicted of class “6”, which explains the observed descend in Kappa. If we focus on the last bin (the most difficult) for this same model and dataset, we can see that it presents similar behaviour compared with the first bin. However, this phenomenon happens for a different reason. In this case, the model predicts 160 instances of class “1” for $\delta = 0$, whereas for $\delta = 1$, the number of instances predicted of this class increased up to 244, i.e., this bin tends to absorb the predictions of class ”1” the more noise is introduced. Both cases may constitute a robustness flaw for a particular model.

Finally, for **Q3**, we derive a taxonomy to group similar techniques in terms of robustness behaviour considering difficulty. To measure the dissimilarity between sets of observations, we employed the Kappa metric computed for each model, aggregated accross all datasets, difficulty bins, and proportions of perturbed instances δ. We performed an agglomerative hierarchichal clustering, employing the euclidean distance and the complete linkage method as a linkage criteria. The result of applying the hierarchical clutering is shown in Figure 5. We found three main clusters. The first cluster show that CI_T, JRip and C5.0 have very similar behaviour, joining with NB. The second cluster is composed of the models GBM, RF, MLR , MLP y FDA, joining with RPART and RFRules at a higher height. The last cluster shows two subgroups. The first one consists of PART, SIMPLS and LVQ. The second subgroup contains RDA, RBF, 3NN and SVM. These results show that models from different ML families may present similar robustness (e.g., the models JRip and CI_T), despite they come from very different techniques.

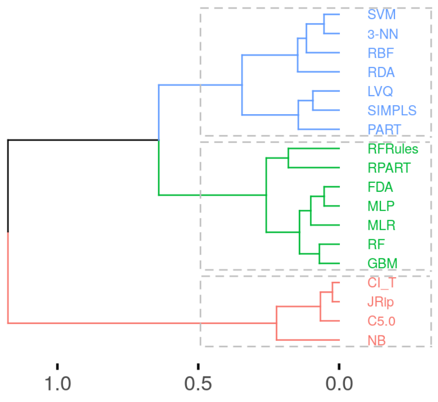

Figure 5: Robustness-based taxonomy for different ML families.

Overall, we have shown that estimating difficulty for analysing robustness may be very useful and insightful. We would need to inspect the test SCCs as an exercise before selecting and deploying models in real-world situations. SCCs can thus be used to select the (set of) best classifier(s) according to the their robustness for different difficulty ranges. Since we may not know the difficulty values of these unseen examples in a test/validation set, we may estimate them in different (an straightforward) ways such as by averaging the difficulty values of the most similar examples in the original set [12] or training a difficulty estimator [11]. We could even do this with small sets or even for single instances, always running the difficulty estimator to determine which model to use for it. If we can predict the difficulty of instances, we could set a threshold to use the system only for the easy instances for which it is robust.

5. Conclusions and Future Work

In this work we propose an evaluation setting to analyse the robustness of different ML models, from different ML families, when addressing noisy instances attending to their difficulty. Furthermore, we established a ML model taxonomy based on the robustness. Our results shown that there are models affected by noise, instance difficulty, or both. Some models are more prone to change their prediction when adding noise to the most difficult instances, while other models also performs similarly with easy instances. This might be caused by the concentration of certain instances of a predicted class in easy bins, which are misclassified after introducing noise. On the other hand, harder bins may absorb certain classes after introducing noise. Given this variety in model' behaviour, we derived a model robustness taxonomy by performing a hierarchical clustering to group items that behave similarly. Our results shown that there are three major clusters. Within each cluster, we can see very different models, from different families, behave very similarly in terms of robustness.

As future work, we will continue to work in this evaluation setting by adding more benchmarks (from different domains) and perturbation functions in our experiments, in order to confirm the results obtained. All this will also add more diversity and generality to our method, thus providing a better insights into the robustness of ML families. Future work may also include the application of our framework in specific use cases. We may focus, for instance, on tasks such as object detection for autonomous vehicles for which we want to evaluate the robustness of a (set of) sytem(s). To do so, we would need a particular benchmark(s) for the detection task, a difficulty estimator for them [11], and a perturbation function to generate invalid inputs, including noise in the captured images or different adversarial attacks [53]. By running our setup, we can potentially analyse which systems(s) are more robust based on the difficulty of the task (e.g, generating also a taxonomy based on similarities), and select the best ones according to the their robustness for different difficulty ranges. As future work, we are interested in exploring other alternative setups of our methodology to gain new insights of the model's behaviour. For instance, we could introduce noise by perturbing only the most relevant attribute/s, instead of all of them, so that we can assess the robustness of the model in relation to those attributes. We could also apply other noise injection methods.

Acknowledgments

This work has been partially supported by the Norwegian Research Council grant 329745 Machine Teaching for Explainable AI, also by the EU (FEDER) and Spanish MINECO grant RTI2018-094403-B-C32 funded by MCIN/AEI/10.13039/501100011033 and by “ERDF A way of making Europe”, Generalitat Valenciana under grant PROMETEO/2019/098, EU’s Horizon 2020 research and innovation programme under grant agreement No. 952215 (TAILOR), and INNEST/2021/317 (Project co-funded by the European Union with the “Programa Operativo del Fondo Europeo de Desarrollo Regional (FEDER) de la Comunitat Valenciana 2014-2020”) and the UPV (Vicerrectorado de Investigación) grant PAI-10-21.

References

[1] J. Grimmer, M. E. Roberts, B. M. Stewart, Machine learning for social science: An agnostic approach, Annual Review of Political Science 24 (2021) 395–419.

[2] F. Zantalis, G. Koulouras, S. Karabetsos, D. Kandris, A review of machine learning and iot in smart transportation, Future Internet 11 (2019) 94.

[3] C. Chen, Y. Zuo, W. Ye, X. Li, Z. Deng, S. P. Ong, A critical review of machine learning of energy materials, Advanced Energy Materials 10 (2020) 1903242.

[4] I. S. C. Committee, et al., Ieee standard glossary of software engineering terminology (ieee std 610.12-1990). los alamitos, CA: IEEE Computer Society 169 (1990) 132.

[5] J. M. Zhang, M. Harman, L. Ma, Y. Liu, Machine learning testing: Survey, landscapes and horizons, IEEE Transactions on Software Engineering (2020).

[6] H. Xu, S. Mannor, Robustness and generalization, Machine learning 86 (2012) 391–423.

[7] J. Rauber, W. Brendel, M. Bethge, Foolbox: A python toolbox to benchmark the robustness of machine learning models, arXiv preprint arXiv:1707.04131 (2017).

[8] O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, A. Criminisi, Measuring neural net robustness with constraints, Advances in neural information processing systems 29 (2016).

[9] J. Lian, L. Freeman, Y. Hong, X. Deng, Robustness with respect to class imbalance in artificial intelligence classification algorithms, Journal of Quality Technology 53 (2021) 505–525.

[10] J. Hernández-Orallo, B. S. Loe, L. Cheke, F. Martínez-Plumed, S. Ó hÉigeartaigh, General intelligence disentangled via a generality metric for natural and artificial intelligence, Scientific reports 11 (2021) 1–16.

[11] F. Martınez-Plumed, D. Castellano-Falcón, C. Monserrat, J. Hernández-Orallo, When AI difficulty is easy: The explanatory power of predicting irt difficulty, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2022.

[12] F. Martínez-Plumed, R. B. Prudêncio, A. Martínez-Usó, J. Hernández-Orallo, Item response theory in AI: Analysing machine learning classifiers at the instance level, Artificial Intelligence 271 (2019) 18–42.

[13] R. K. Hambleton, H. Swaminathan, Item response theory: Principles and applications, Springer Science & Business Media, 2013.

[14] D. Ljunggren, S. Ishii, A comparative analysis of robustness to noise in machine learning classifiers, 2021.

[15] B. D. Ripley, Pattern recognition and neural networks, Cambridge university press, 2007.

[16] C. Ferri, J. Hernández-Orallo, R. Modroiu, An experimental comparison of performance measures for classification, Pattern recognition letters 30 (2009) 27–38.

[17] J. A. Sáez, M. Galar, J. Luengo, F. Herrera, Analyzing the presence of noise in multi-class problems: alleviating its influence with the one-vs-one decomposition, Knowledge and information systems 38 (2014) 179–206.

[18] C.-M. Teng, Correcting noisy data., in: ICML, Citeseer, 1999, pp. 239–248.

[19] X. Zhu, X. Wu, Q. Chen, Eliminating class noise in large datasets, in: Proceedings of the 20th International Conference on Machine Learning (ICML-03), 2003, pp. 920–927.

[20] D. Madaan, J. Shin, S. J. Hwang, Learning to generate noise for multi-attack robustness, in: International Conference on Machine Learning, PMLR, 2021, pp. 7279–7289.

[21] S. Latif, R. Rana, J. Qadir, Adversarial machine learning and speech emotion recognition: Utilizing generative adversarial networks for robustness, arXiv preprint arXiv:1811.11402 (2018).

[22] C. Leistner, A. Saffari, P. M. Roth, H. Bischof, On robustness of on-line boosting-a competitive study, in: 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops, IEEE, 2009, pp. 1362–1369.

[23] J. M. Zhang, M. Harman, B. Guedj, E. T. Barr, J. Shawe-Taylor, Perturbation validation: A new heuristic to validate machine learning models, arXiv preprint arXiv:1905.10201 (2020).

[24] J. A. Sáez, J. Luengo, F. Herrera, Evaluating the classifier behavior with noisy data considering performance and robustness: The equalized loss of accuracy measure, Neurocomputing 176 (2016) 26–35.

[25] V. Tjeng, K. Xiao, R. Tedrake, Evaluating robustness of neural networks with mixed integer programming, arXiv preprint arXiv:1711.07356 (2017).

[26] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, M. Vechev, Ai2: Safety and robustness certification of neural networks with abstract interpretation, in: 2018 IEEE Symposium on Security and Privacy (SP), IEEE, 2018, pp. 3–18.

[27] D. Gopinath, K. Wang, M. Zhang, C. S. Pasareanu, S. Khurshid, Symbolic execution for deep neural networks, arXiv preprint arXiv:1807.10439 (2018).

[28] M. Usman, Y. Noller, C. S. Păsăreanu, Y. Sun, D. Gopinath, Neurospf: A tool for the symbolic analysis of neural networks, in: 2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), IEEE,

2021, pp. 25–28.

[29] G. Katz, C. Barrett, D. L. Dill, K. Julian, M. J. Kochenderfer, Reluplex: An efficient smt solver for verifying deep neural networks, in: International conference on computer aided verification, Springer, 2017, pp. 97–117.

[30] M. R. Smith, T. Martinez, C. Giraud-Carrier, An instance level analysis of data complexity, Mach. Learn. 95 (2014) 225–256. URL: https://doi.org/10.1007/s10994-013-5422-z. doi:10.1007/s10994-013-5422-z.

[31] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International journal of computer vision 115 (2015) 211–252.

[32] D. Liu, Y. Xiong, K. Pulli, L. Shapiro, Estimating image segmentation difficulty, in: International Workshop on Machine Learning and Data Mining in Pattern Recognition, Springer, 2011, pp. 484–495.

[33] S. Vijayanarasimhan, K. Grauman, What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations, in: 2009 IEEE conference on computer vision and pattern recognition, IEEE, 2009, pp. 2262–2269.

[34] B. Richards, Type/token ratios: What do they really tell us?, Journal of child language 14 (1987) 201–209.

[35] D. L. Hoover, Another perspective on vocabulary richness, Computers and the Humanities 37 (2003) 151–178.

[36] S. E. Embretson, S. P. Reise, Item response theory for psychologists, L. Erlbaum, 2000.

[37] F. Martínez-Plumed, R. B. C. Prudêncio, A. Martínez-Usó, J. Hernández-Orallo, Making sense of item response theory in machine learning, in: ECAI 2016 - 22nd European Conference on Artificial Intelligence, 2016, pp. 1140–1148. doi:10.3233/978-1-61499-672-9-1140.

[38] F. Martínez-Plumed, J. Hernández-Orallo, Dual indicators to analyse AI benchmarks: Difficulty, discrimination, ability and generality, IEEE Transactions on Games 12 (2020) 121–131.

[39] J. P. Lalor, Learning Latent Characteristics of Data and Models using Item Response Theory, Ph.D. thesis, Doctoral Dissertations, 1842, 2020.

[40] Z. Chen, H. Ahn, Item response theory based ensemble in machine learning, International Journal of Automation and Computing 17 (2020) 621.

[41] A. Birnbaum, Statistical Theories of Mental Test Scores, Addison-Wesley, Reading, MA., 1968.

[42] R. Fabra-Boluda, C. Ferri, F. Martínez-Plumed, J. Hernández-Orallo, M. J. Ramírez-Quintana, Family and prejudice: A behavioural taxonomy of machine learning techniques, in: ECAI 2020, IOS Press, 2020, pp. 1135–1142.

[43] J. Hernández Orallo, C. Ferri Ramírez, M. Ramírez Quintana, Introducción a la Minería de Datos, Pearson Prentice Hall, 2004.

[44] P. Flach, Machine learning: the art and science of algorithms that make sense of data, Cambridge University Press, 2012.

[45] M. Fernández-Delgado, E. Cernadas, S. Barro, D. Amorim, Do we need hundreds of classifiers to solve real world classification problems?, The journal of machine learning research 15 (2014) 3133–3181.

[46] R. Landis, G. Koch, An application of hierarchical kappa-type statistics in the assessment of majority agreement among multiple observers, Biometrics (1977) 363–374.

[47] B. D. Wright, M. H. Stone, Best test design, Mesa press, 1979.

[48] J. Vanschoren, J. N. Van Rijn, B. Bischl, L. Torgo, OpenML: networked science in machine learning, ACM SIGKDD Explorations Newsletter 15 (2014) 49–60.

[49] R. P. Chalmers, mirt: A multidimensional item response theory package for the r environment, Journal of statistical Software 48 (2012) 1–29.

[50] A. Maydeu-Olivares, Goodness-of-fit assessment of item response theory models, Measurement: Interdisciplinary Research and Perspectives 11 (2013) 71–101.

[51] M. Kuhn, Building predictive models in R using the caret package, Journal of Statistical Software, Articles 28 (2008) 1–26. URL: https://www.jstatsoft.org/v028/i05. doi:10.18637/jss.v028.i05.

[52] J. N. van Rijn, B. Bischl, L. Torgo, B. Gao, V. Umaashankar, S. Fischer, P. Winter, B. Wiswedel, M. R. Berthold, J. Vanschoren, OpenML: a collaborative science platform, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2013, pp. 645–649.

[53] B. J. Petit, B. Stottelaar, M. Feiri, F. Kargl, Remote attacks on automated vehicles sensors: Experiments on camera and lidar black hat europe, 2015.