# When AI Difficulty is Easy: The Explanatory Power of Predicting IRT Difficulty

**Fernando Martínez-Plumed**[1,2], **David Castellano**[2], **Carlos Monserrat-Aranda**[2], **José Hernández-Orallo** [2,3]

[1] European Commission, Joint Research Centre
[2] Valencian Research Institute for Artificial Intelligence (VRAIN), Universitat Politècnica de València
[3] Leverhulme Centre for the Future of Intelligence, University of Cambridge
fernando.martinez-plumed@ec.europa.eu, dacasfal@upv.es, cmonserr@dsic.upv.es, jorallo@upv.es

## Abstract

One of challenges of artificial intelligence as a whole is robustness. Many issues such as adversarial examples, out of distribution performance, Clever Hans phenomena, and the wider areas of AI evaluation and explainable AI, have to do with the following question: *Did the system fail because it is a hard instance or because something else?* In this paper we address this question with a generic method for estimating IRT-based instance difficulty for a wide range of AI domains covering several areas, from supervised feature-based classification to automated reasoning. We show how to estimate difficulty systematically using off-the-shelf machine learning regression models. We illustrate the usefulness of this estimation for a range of applications.

## Introduction

As no AI system can be perfect, robustness must be based on knowing where and why it fails, avoiding highly unexpected failures. One key element in this understanding is *instance difficulty*. This effect of difficulty on robustness could be rephrased as follows: is it unexpected that a customary system fails on this instance? Accordingly, difficulty is defined as a metric $\hbar$ that decreases with the expected performance $R$ for a customary system. A good difficulty metric $\hbar$ would maximise the following expected probability:

$$\mathbb{E}_{\boldsymbol{x}_i,\boldsymbol{x}_j,m}[\hbar(\boldsymbol{x}_i) < \hbar(\boldsymbol{x}_j) \Rightarrow R(m,\boldsymbol{x}_i) > R(m,\boldsymbol{x}_j)] \quad (1)$$

with $\boldsymbol{x}_i, \boldsymbol{x}_j$ being feature vectors sampled from $X$ and $m$ sampled from $M$, according to a reference distribution of instances and systems respectively. Note that the choice of $M$ is key here. If $M$ is a human population, our notion of difficulty would be anthropocentric. If $M$ is a collection of representative AI techniques, our notion of difficulty would be AI-centric. Difficulty is hence defined as a notion of conformity. If a capable system fails for an easy instance, we could signal this as non-conformity.

Beyond the above definition, we impose several other desiderata to a method for calculating difficulty:

- *Attribute-based*. Applicable to the observable features of each instance, without the need of comparing it to other instances. Many instance hardness metrics do not meet this property (Smith, Martinez, and Giraud-Carrier 2014) and cannot explain difficulty based on the instance.
- *System-independent*. Many uncertainty or confidence estimation metrics are system-specific (Hendrycks and Gimpel 2016; Jiang et al. 2018; Corbière et al. 2019) or particular to another encoding model's latent variables.
- *Domain-generic*. Applicable to all kinds of tasks and areas in AI (from machine learning to reasoning), and not only for those areas or tasks where specific instance hardness or difficulty metrics exist.

The populational nature of Eq. 1 is convenient in many areas of AI, since we have a wide range of techniques that are usually applied before selecting the system we want to deploy. All these discarded suboptimal systems can be reused for the calculation of difficulty, as we will see in this paper. 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. This may happen if we just calculate the average error of a set of systems for each instance as a proxy for difficulty (Martínez-Plumed et al. 2019).

A solution to this problem was introduced several decades ago, and it is known as item response theory (IRT), 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. However, IRT and other difficulty metrics are derived from previous performance results, but do not depend on the instance space, so we cannot anticipate the difficulty of *new* instances. We present a relatively straightforward solution for this important issue: training regression models with the problem features as input and difficulty as output.

This paper covers a range of problems in AI, derive their IRT difficulties, and train a regression model for each domain—a difficulty estimator—, which we evaluate systematically. For many domains, the estimates for IRT difficulty are very good, according to RMSE and Spearman correlation. We illustrate the explanatory power of these difficulty models on a series of applications:

- *Explainable AI*: understanding what makes instances

hard, or groups of instances (e.g., classes), and explaining whether an error is expected or unexpected.

- *Robust evaluation*: comparing systems using their characteristic curves. Systems that are reliable on all (or most) easy instances should be considered more robust.
- *AI progress*: analysing whether the increase of performance has focused on the low-hanging easy instances or more complex instances through specialisation.
- *Distribution changes and perturbations*: a very capable system failing on a batch of very easy instances may suggest a distributional shift or an adversarial attack. The inverse phenomenon may signal a Clever Hans effect.

The main contributions of this paper are: (1) the first general methodology for training an estimator for IRT difficulties, (2) comprehensive empirical results showing the wide range of domains where it works, and (3) the evidence of its applicability as a powerful explanatory tool.

## Related Work

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. For instance, Smith, Martinez, and Giraud-Carrier (2014) identify instances that are hard to classify through *instance hardness* metrics, which are inferred from measures of density and overlapping.

In computer vision, there are global image properties such as salience, memorability, photo quality and the importance of objects, but there is limited work in extracting image difficulty. We find intrinsic measures of difficulty based on various image features such as clutter (Russakovsky et al. 2015), tone, colour, gradient and texture (Liu et al. 2011). There are also extrinsic (anthropomorphic) approaches, such as estimating the difficulty of an image based on the time needed by a human to segment it (Vijayanarasimhan and Grauman 2009).

Natural language processing inherits some metrics of difficulty from linguistics based on lexical readability and richness. *Flesch-Kincaid* lexical readability is based on traditional features of text such as word and sentence length (Kincaid et al. 1975). A widely used lexical measure is the *Type-Token Ratio* (Richards 1987), which is the ratio of the number of unique word tokens to the total number of word tokens in a text. Another related measure of lexical richness is *Hapax richness* (Hoover 2003), defined as the number of words that occur only once divided by the number of total words.

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 (Embretson and Reise 2000), only recently brought to AI (Martínez-Plumed et al. 2016; Lalor 2020). IRT has been used in several areas of AI, where the AI systems are treated as respondents and the tasks as items, including classification (Martínez-Plumed et al. 2016; Martínez-Plumed et al. 2019; Chen and Ahn 2020), regression (Moraes et al. 2020), multi-agent scenarios (Chmait et al. 2017), XAI (Kline et al. 2020) and other AI benchmarks (Martínez-Plumed and Hernández-Orallo 2017; Martínez-Plumed and Hernández-Orallo 2018, 2020).

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. The most widely used functions are of logistic form:

$$P(U_{ji} = 1|\theta_j) = \frac{1}{1 + exp(-a_i(\theta_j - b_i))} \quad (2)$$

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. If *ability* $\theta_j$ 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 the figure).

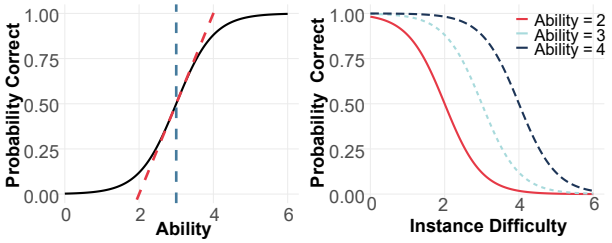


Figure 1: Left: Example of a 2PL IRT ICC curve, with slope $a = 2$ in red and location parameter $b = 3$ in blue. Right: Example of SCC curves with different abilities.

Items can also be characterised by their *discrimination* ($a_i$): the steepness of the function at the location point. It represents the degree to which the item discriminates between respondents in different regions on the latent continuum. An item having $a_i = 1.0$ discriminates fairly well since small increases in ability substantially increase the capacity to respond correctly. IRT models that assume $a_i = 1.0$ are known as 1PL, and only need to inferability $\theta_j$ and difficulty $b_i$. Models estimating $\theta_j$ and both $a_i$ and $b_i$ are known as 2PL, leading to better fit on some occasions but also some overfitting. 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 (Birnbaum 1968), can be used to infer all the IRT parameters. Further coverage of IRT can be found in the appendix (Martínez Plumed et al. 2022).

IRT fulfils two of the desiderata that we outlined in the introduction, and it will be our choice for a *system-independent domain-generic* difficulty metric. 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, and as we will corroborate here (more details in the appendix). However, there is an important limitation of IRT difficulty (and other metrics of difficulty based on average error). For a new instance that has never been evaluated for a set of subjects or systems, how can we estimate its difficulty? Should we run the instance through all the systems that were used in the IRT estimation? Is this realistic and practical? First, we may not have access to all these systems in deployment time, and even if we do, this may be a slow process. Instead, we use the instances for which we have previously calculated the IRT difficulties to train a difficulty estimator $\hat{\hbar}(\boldsymbol{x})$, a regression model, with the instance features as input and difficulty value as output. IRT ignores the features altogether, the estimator does not, as it is *attribute-based*. Given a new, previously unseen, instance in its input feature representation $\boldsymbol{x}$ (a feature vector, an image or a piece of text), we just apply $\hat{\hbar}$ to it. In this way, no previous systems are longer needed and no IRT re-estimation required.

## Methodology

This section presents the choice of datasets and difficulty metrics, and the procedure to build the difficulty estimators[1].

**Benchmarks** In our study, we address a set of 18 illustrative benchmarks from different AI domains, including supervised learning, perception, natural language processing, and reasoning (see Table 1). We put a special emphasis on classification problems, since the difficulty of instances usually depends on neighbouring instances, and we want to explore if an attribute-based estimation model predicting difficulty from a single instance can circumvent this situation. Although some of these benchmarks are not based on machine learning models, we will use the term AI system or *model* indistinctly. The selection is guided by comprehensiveness but also constrained by those benchmarks where there is a sufficiently large number $|I|$ of examples (items) and $|J|$ models (respondents). In IRT, it is recommended to have at least 10-20 responses per item (Wright and Stone 1979). More importantly, we need the *instance-wise results*, i.e., a $|J| \times |I|$ matrix with the performance of each system for each instance. Finding experiments that were not reported in an aggregated way was not an easy task. In the appendix we give further details about how we found or generated the datasets for the 18 benchmarks.

**Reference difficulty metrics: AvE and IRT $n$PL** For each benchmark, we first check there is at least $|J| = 10$ systems or models that are sufficiently diverse (different architectures or technologies). Next, we obtain their responses for *unseen* instances (e.g., in machine learning scenarios 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 $U_{ji}$. When original performance is not binary, we will convert it to binary responses by comparing them with the median result for that item (see benchmarks with $^{\triangle}$ superindex in Table 1).

One simple populational way of deriving a metric of difficulty is *average error*: $\hbar_{\text{AvE}}(i) = 1 - \frac{1}{|J|}\sum_{j=1}^{|J|} U_{ji}$. Because of the many advantages of IRT we also use the **1PL** and **2PL** models. For instance, $\hbar_{\text{1PL}}(i) = b_i$ denotes a difficulty metric for $i$ that is simply the $b_i$ parameter for instance $i$ as obtained by a 1PL IRT model on $U$.

We follow the recommendations from (Martínez-Plumed et al. 2019) for the application of IRT in. The specific details are explained in the appendix. We will use two further approaches, **AvE (¬abs)** and **2PL (¬abs)**, which are similar to **AvE** and **2PL**, respectively, but the abstruse instances (those with negative discrimination) are eliminated.

**Training and evaluating the difficulty estimator** For each benchmark, once a difficulty metric $\hbar(i)$ has been calculated for each instance $i$ in the whole dataset, we now focus on the problem of building a difficulty estimator for other instances. For each instance $i$ we recover their features $\boldsymbol{x}_i$, and we create a supervised dataset $D = \langle \boldsymbol{x}_i, \hbar(i) \rangle_i$. As IRT difficulties are built to approximately follow a normal distribution with standard deviation 1 but different locations depending on the dataset (but usually with means around $-3$ and 3 (Martínez-Plumed et al. 2019)), we finally chose to remove those instances whose difficulty is out of the range $[-6, 6]$, which are considered outliers. This happened in half of the benchmarks for very easy instances for which all techniques are correct, never affecting more than 0.5% of the instances in the **1PL** case.

With this data, we only need to choose appropriate regression models to estimate $\hat{\hbar}$, so that for any new instance feature vector $\boldsymbol{x}$ we simply get its difficulty as $\hat{\hbar}(\boldsymbol{x})$. The learning algorithms will depend on the data representation (feature-value, bitmap or text). In the case of those benchmarks using a feature-value data representation, common state-of-the art machine learning models will be used. We followed $2 \times 5$-fold cross validation, so we will report average results for the test folds. The choice of techniques, preprocessing and hyperparameters for feature-value, bitmap and NLP problems can be found in the appendix.

We will use Normalised RMSE (Root Mean Squared Error divided by standard deviation), denoted by nRMSE, and Spearman (rank) correlation, as loss functions to assess the quality of our regression models. We normalise the RMSE to facilitate the comparison between different benchmarks with different scales and use Spearman correlation as a metric of how well $\hat{\hbar}(\boldsymbol{x})$ ranks the difficulty of a set of instances.

**Experimental questions** Once the experimental setting is clear, we now want to investigate what difficulty metric is best, how to estimate it and how good the estimation is. For this, we set five experimental questions. Q1: How do difficulties distribute per benchmark for the different difficulty metrics? Q2: What are the most appropriate regression

[1]All the code and results can be found in https://github.com/nandomp/AIdifficulty. Further details in the appendix (Martínez Plumed et al. 2022).

| Domain | Task | Benchmark | $\|I\|$ | Features | $\|J\|$ | Description |
|---|---|---|---|---|---|---|
| Supervised Learning | Classification | diabetes | 768 | 8 | 353 | Diabetes diagnostic |
| | | kc1 | 2110 | 21 | 213 | Software defect prediction |
| | Regression | liver-disorders$^{\triangle}$ | 345 | 6 | 74 | Liver disorder status |
| Audio Processing | Speaker recognition | japaneseVowels | 9961 | 14 | 558 | Records of nine male speakers |
| Computer Vision | Optical Character Recognition | letter | 20000 | 16 | 174 | Letter image recognition Data |
| | | optdigits | 5620 | 64 | 305 | Optical recognition of handwritten digits |
| | | pendigits | 10992 | 16 | 455 | Pen-based recognition of handwritten digits |
| | Image Recognition | satimage | 6430 | 36 | 203 | Multi-spectral satellite images. |
| | | segment | 9901 | 19 | 2310 | Database of outdoor images |
| | | vehicle | 846 | 18 | 847 | Silhouette information of vehicles |
| | | CIFAR-10 | 60000 | 3072 | 156 | 32x32 color images, 10 classes of objects |
| | | MNIST | 70000 | 784 | 1000 | Database of 28x28 handwritten digits |
| | | Fashion-MNIST | 70000 | 784 | 449 | Zalando's 28x28 article images |
| | | UMIST | 575 | 10304 | 53 | Grayscale faces (views) of 20 different people. |
| Automated Reasoning | Automated Theorem Proving | tptp$^{\triangle}$ | 500 | 9 | 17 | The CADE ATP System Competition results |
| | Boolean Satifiability | sat$^{\triangle}$ | 452 | 21 | 16 | The results for solvers of the SAT Competition |
| Natural Language Processing | Sentiment Analysis | IMDb | 872 | 1253.85* | 12 | Movie reviews from IMDb platform |
| | | SST-2 | 872 | 1273.45* | 10 | Stanford Sentiment (analysis) Treebank. |

* Average number of characters $^{\triangle}$ Responses binarised

Table 1: Benchmarks used, categorised by the domain and task, their name, number of instances ($|I|$), number of features and number of AI systems ($|J|$) from which we obtained the matrix of $|J| \times |I|$ responses over the individual instances.

models for difficulty estimation depending on the domain and their representation (feature-based, bitmap or text)? Q3: Does the difficulty estimation model use the same features and representations than the original models? Q4: What is the most reliable and predictable difficulty metric? Q5: How well can we estimate difficulty from the instance attributes and how good the new methodology works in general?

## Results

Fig. 2 shows the **1PL** difficulty distribution per benchmark, with a standard deviation around 1 (as expected). Other approaches have similar distributions but more frequently multimodal and with a higher number of outliers, as we show for **2PL** difficulty and especially **AvE** difficulty (Fig. 7 and Fig. 8 in the appendix). In terms of location, we see that all the methods agree in recognising those benchmarks with more difficult instances (UMIST, CIFAR-10, TPTP, liver-disorders) and those with less difficult instances (pendigits, japaneseVowels and vehicle). In sum, **1PL** seems to have some distributional advantages.

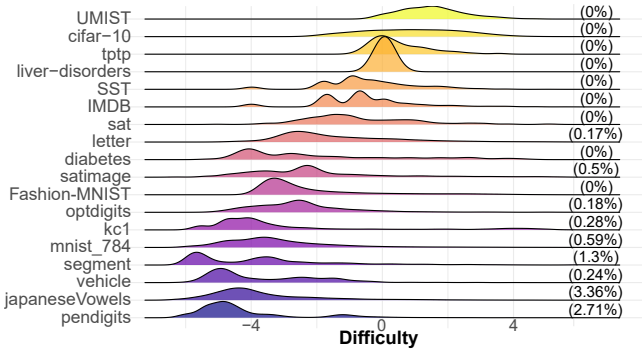


Figure 2: 1PL difficulty distribution per dataset (percentage of difficulties outside the $[-6, 6]$ range indicated in the plots). Benchmarks sorted by average difficulty.

Focusing on estimating **1PL** with a regression model, Table 2a shows nRMSE and Spearman correlation results for those benchmarks in Table 1 following a feature-value representation. As we can see, *rf* is the technique achieving the best results for predicting difficulty. We also compute pairwise comparisons in the test set to identify significant differences between models using Wilcoxon test (Cuzick 1985), showing the overwhelming dominance of *rf*. The results are very good in terms of nRMSE and Spearman correlations. The worst dataset in terms of correlations is tptp, which is also the most difficult in this group as per Fig. 2.

For the datasets following an image-based representation (Table 2b), the results are also very good, especially when using *VGG16* to build the regression model, even better than for the feature-based problems. The only exception is CIFAR-10. We do not find an easy explanation for this behaviour in terms of number of instances, features, distribution or location of difficulties. While it is a difficult benchmark, the results are good with the most difficult one (UMIST). We do not think it is due to a bad choice of hyperparameters either, as the results are poor for the three methods we tried (*VGG16*, *ResNet-50* and *Densenet121*).

Finally, for NLP problems, we see the results in Table 2c. The results are poor for the two datasets we have analysed, IMDb and SST-2, for all techniques (*BERT* is slightly better than the rest, but the correlations are still too low). We make the same considerations here as we did for CIFAR-10, although in this case we have a small number of instances (less than 900 instances available), which made the training of our difficulty estimators a real challenge. Overall, we can only say for certain that three of the six most difficult benchmarks (CIFAR-10, IMDb and SST-2) are also those for which difficulty estimations are poor. This may be a necessary but not sufficient condition for difficulty estimation to fail, so it needs further investigation as future work.

Important insights on this and other questions can come from better understanding the difference between solving the original problem and solving the difficulty estimation problem. To this purpose, we have analysed how the original

(a)

| Dataset | elasticNet | | gbm | | knn | | lm | | rf | |
|---|---|---|---|---|---|---|---|---|---|---|
| | nRMSE | r | nRMSE | r | nRMSE | r | nRMSE | r | nRMSE | r |
| diabetes | .96±.04 | .59 | .93±.06 | .63 | 1.01±.05 | .47 | .97±.04 | .57 | **.86±.05** | **.73** |
| kc1 | .90±.05 | .69 | .90±.05 | .69 | .95±.06 | .61 | .91±.05 | .68 | **.82±.05** | **.79** |
| liver-dis | 1.00±.04 | .64 | 1.01±.05 | .69 | 1.10±.07 | .76 | 1.01±.04 | .64 | **.94±.04** | **.89** |
| japanese | .90±.04 | .72 | .88±.04 | .73 | .96±.05 | .56 | .90±.04 | .71 | **.63±.04** | **.86** |
| letter | .87±.01 | .57 | .86±.01 | .58 | .65±.01 | .89 | .87±.01 | .57 | **.62±.01** | **.90** |
| optdigits | .94±.05 | .05 | .96±.06 | .09 | **.81±.06** | .43 | .94±.05 | -.04 | **.81±.06** | **.65** |
| pendigits | .78±.06 | .47 | .77±.06 | .41 | .55±.06 | .76 | .78±.05 | .47 | **.50±.07** | **.81** |
| satimage | .95±.02 | .48 | .90±.02 | .49 | .84±.02 | .66 | .96±.02 | .48 | **.76±.02** | **.75** |
| segment | .92±.05 | .72 | .89±.05 | .68 | .82±.07 | .69 | .92±.05 | .71 | **.61±.05** | **.82** |
| vehicle | .80±.12 | .49 | .78±.15 | .64 | .84±.11 | .84 | .81±.11 | .48 | **.65±.07** | **.82** |
| tptp | 1.13±.38 | .25 | .91±.05 | .33 | .92±.08 | .42 | 1.10±.33 | .18 | **.82±.05** | **.69** |
| sat | .75±.10 | .74 | .74±.06 | .74 | .70±.07 | .72 | .76±.08 | .73 | **.54±.07** | **.86** |

(b)

| Dataset | VGG16 | | ResNet-50 | | Densenet121 | |
|---|---|---|---|---|---|---|
| | nRMSE | r | nRMSE | r | nRMSE | r |
| CIFAR-10 | 1.98±.54 | .06 | 1.94±.11 | **.12** | **1.91±.02** | .11 |
| MNIST | **.34±.01** | **.91** | .36±.01 | **.91** | .35±.08 | **.91** |
| Fashion-MNIST | **.59±.06** | **.96** | .73±.03 | .92 | .64±.01 | .91 |
| UMIST | **.40±.1** | **.83** | .76±.17 | .46 | .42±.27 | .79 |

(c)

| Dataset | T5 (small) | | BERT (sent.) | | BERT (base) | |
|---|---|---|---|---|---|---|
| | nRMSE | r | nRMSE | r | nRMSE | r |
| IMDb | 1.86±.26 | .01 | **1.48±.11** | **.18** | 1.73±.16 | .03 |
| SST-2 | 1.79 ±.26 | -.01 | **1.50±.11** | **.01** | 1.60±.15 | .00 |

Table 2: Estimation of 1PL difficulties using the input features (feature vector representation (a), pixels (b) and text (c)) for those benchmarks in Table 1. Lower values are better for nRMSE. Statistical significance denoted in bold.

features are used for the letter benchmark (Frey and Slate 1991), whose goal is to identify 26 letters from a feature-vector representation. If we look at most important features of the best difficulty estimator in Table 2 (*rf*), in Fig. 3 (left), we see that the variable `y.ege` (i.e., the mean edge count left to right) is more than twice as important than the following ones: This means that the difficulty of a character image heavily depends on the mean number of edges (at all vertical positions). On the contrary, when analysing the variable importance on the original task in Fig. 3 (right), the sum of the vertical positions of edges (`y.ege`) is the most important one. Also, it is noticeable that the differences between the first five variables by importance are not as prominent as in the previous case. In general, discrepancies can be found in machine learning benchmarks if we look at how the most important features of the difficulty estimator compare to those of the original task. This may partly explain why in some cases the original task is easy but estimating difficulty is hard and vice versa. For instance, the expression or pose of a face may be a distinctive feature for estimating face recognition difficulty but possibly less useful for classifying images. More complete discussion in the appendix.

While we have been showing the results for the **1PL** metric, we also analysed the results for other difficulty metrics. We did the comparison for the feature-value benchmarks and *rf*, as these give good results consistently. The complete nRMSE and correlation results for **2PL**, **AvE**, **2PL (¬abs)** and **AvE (¬abs)** are shown in Tables 5, 6, 7 and 8 in the appendix, respectively. nRMSE values are difficult to compare

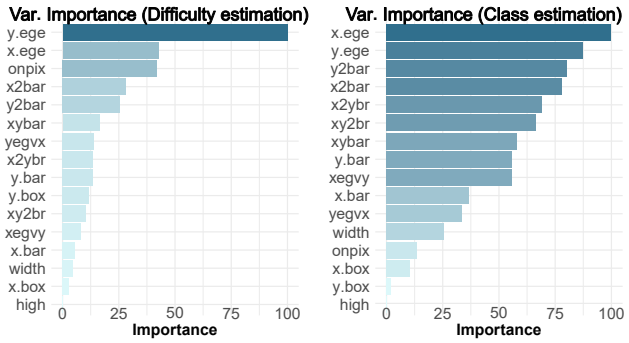


Figure 3: Variable Gini Importance (Breiman 2001) for letter for 1PL prediction (left) and class prediction (right).

between each other and against Table 2a due to the different magnitudes (limited to -6 and 6 for **1PL** and **2PL** and variants, but between 0 and 1 for **AvE**). Instead, here we only show the Spearman correlations of all metrics in Table 3. We see that **1PL** shows better correlations in general than the other difficulty metrics. This may be related to **1PL** more frequently generating unimodal distributions, as we saw in Fig. 2. It may also due to **1PL** having just one parameter per instance and hence being less prone to overfitting. Given these clear results, from now on we use and recommend the use of the **1PL** difficulty metric when estimating difficulty from the original features, as we do in this paper.

| Benchmark | 1PL | 2PL | 2PL (¬abs) | AvE | AvE (¬abs) |
|---|---|---|---|---|---|
| diabetes | **0.73** | 0.67 | 0.72 | 0.6 | 0.57 |
| kc1 | **0.79** | 0.7 | 0.63 | 0.63 | 0.6 |
| liver-disorders | **0.89** | 0.88 | 0.82 | 0.82 | 0.79 |
| japaneseVowels | **0.86** | 0.85 | 0.8 | 0.79 | 0.76 |
| letter | **0.9** | **0.9** | 0.83 | 0.83 | 0.8 |
| optdigits | **0.65** | 0.55 | 0.53 | -0.02 | -0.05 |
| pendigits | **0.81** | **0.81** | 0.74 | 0.74 | 0.71 |
| satimage | **0.75** | 0.69 | 0.62 | 0.62 | 0.59 |
| segment | **0.82** | **0.82** | 0.75 | 0.75 | 0.72 |
| vehicle | 0.82 | **0.88** | 0.82 | 0.75 | 0.72 |
| tptp | **0.69** | 0.59 | 0.44 | 0.44 | 0.41 |
| sat | **0.86** | 0.84 | 0.8 | 0.79 | 0.76 |
| **Average** | **0.79** | 0.76 | 0.71 | 0.65 | 0.62 |

Table 3: Spearman correlation values for each combination of benchmarks and approach for difficulty estimation (including the deletion of abstruse instances). Highest in bold.

Overall, given a task or benchmark, the first step is the collection of results for a set of systems. Here, we have shown that in cases with a low number of systems (e.g., 16 for sat) we can have good results. Then, the construction of the IRT difficulty metrics per instance is followed by the construction of a regression model that estimates the difficulty for new instances using the features as inputs. Whether this could work was an open question, especially for classification tasks, as the difficulty of an instance typically depends on the location of the other instances. In general (15 out of 18 benchmarks), we show that the procedure gives more than satisfactory results, and suggests that more specific efforts in the collection of data or the use of more powerful models may improve the results in the future.

## Applications

Here we will cover some areas where the mere existence of a metric of difficulty is insightful (extended discussions in the appendix).

**Explainable AI** The use of a *domain-generic* difficulty metric $\hbar$ is very useful to understand where and how a system fails, and can be applied to any area in AI, not only machine learning (Martínez-Plumed et al. 2019). For instance, for problems such as tptp and sat, we can analyse, manually, what makes instances hard, and also see the evolution of instances in different competitions or benchmarks. For this application, it is important that the metric is *system-independent*, i.e., we analyse the problem instances, not a particular system. However, the use of an *attribute-based* $\hat{\hbar}$ increases the applications. First, there is no need to manually extract what makes instances hard, we can inspect which attributes make it hard (as we did with Fig. 3). With a difficulty estimator based on the inputs, we can even play *counterfactually*, asking questions such as "what would the difficulty of this example be, if attribute $x = v_1$ changed to $x = v_2$?". We can also explore this distributionally, e.g., how difficulty would change for new distributions before receiving them, so anticipating or explaining distributional shift. Mapping the whole space of difficulty can also help to understand where the problem is challenging.

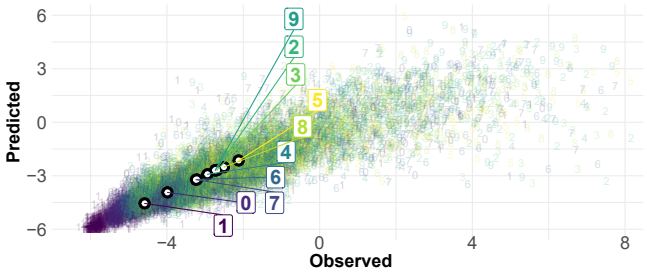


Figure 4: Class difficulty (average) for MNIST.

Understanding groups of instances is a direct application of the difficulty estimator. For instance, Fig. 4 shows the actual $\hbar$ and estimated $\hat{\hbar}$ for a sample in the MNIST dataset. We show the mean location of the 10 classes (digits 0-9). Because of the quality of $\hat{\hbar}$ for this dataset, the ten classes appear in a straight unit line, which represents that $\hat{\hbar}$ is also well calibrated per class. We can see that classes 1 and 0 are easiest, while 5 is the most difficult. While this is well-known for this thoroughly-studied dataset, a similar analysis can be done for other datasets, using classes, clusters or data sources (e.g., if we expect a new dataset with 50% of 5s and 8s, then we can calculate the expected results using Eq. 2).

**Robust evaluation and deployment** One of the most powerful visualisation tools that derives from difficulty is what we call system characteristic curves (SCC) (Fig. 1, right), showing the response probability (e.g., accuracy) of a particular AI system as a function of the instance difficulty. Illustratively, Figure 5 presents the SCC of a subset of classifiers (Table 9 in the appendix) for the letter benchmark

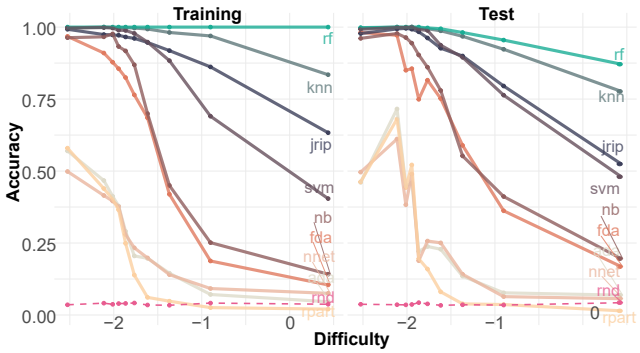


Figure 5: (Left) SCC obtained with the 70% of the letter benchmark and the observed difficulties $\hbar$. (Right) SCC obtained with the test set (30%), using estimated difficulties $\hat{\hbar}$.

using **1PL** difficulty. We split the data into train (difficulty estimated using $\hbar$) and test (difficulty estimated using the best $\hat{\hbar}$ from Table 2a). 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 accuracy.

In this particular example, while *rf* is the best model for all difficulties, the rest have different behaviours: *knn*, *jrip*, *svm*, *nb* and *fda* get good results for easy instances and their performance decreases at different rates and levels when instances become more difficult (with some crossings such as those between *jrip* and *svm*). Looking at the highest slope point, we can distinguish their abilities and see if some degrade sooner than others. Interestingly, Figure 5 also shows overfitting between train and test for the difficult examples. In fact, up to a difficulty of 1.75, the results in both SCCs are very similar, so there is no overfitting for the easy instance. The most striking thing happens in the test set: *svm*, *nb* and *fda* improve for high difficulty instances (while this is not the case for the other techniques).

In general, we would need to inspect the test SCCs as an exercise before selecting and deploying models. For a new dataset, we will first use the difficulty estimator to calculate the average difficulty and choose the best model in that range.

**Analysing AI progress** Progress in many benchmarks is reported as an increase of overall performance for the best method, setting the SOTA for it. But is the performance increase distributed equally among difficulties? Or is research focusing on low-hanging fruits first and sacrificing robustness (failing on easy instances) to get better results for some specific hard instances. We can analyse this using difficulty. Figure 6 presents the SCCs for a subset of CNN architectures ranging from 2012 to 2019 applied to CIFAR-10.

We see different strategies. For instance, *Googlenet* improves on the difficult instances and has a different curve from *Alexnet*, even if the average performance is similar. The next generation of techniques improved more systematically on the range of difficulties, but the latest two techniques, *EfficientNet* and *Densenet*, have again very different behaviours. While *Densenet* is very robust on the easy instances, its performance is poor for difficult instances. On

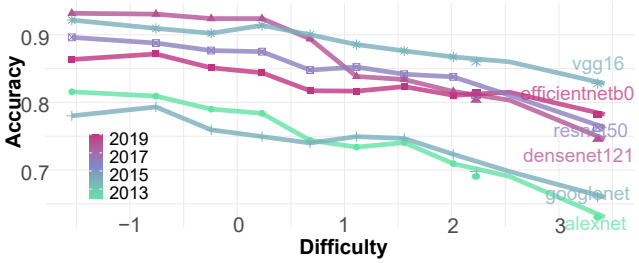


Figure 6: Empirical SCCs of CNN architectures (coloured by their year of development) addressing CIFAR-10.

the contrary, *EfficientNet* is the flattest of all. Apart from important insights of what kinds of instances they are putting the effort on, and the interpretations in terms of overfitting, these plots also open the discussion of whether we would like AI progress to focus on more logistic-shaped curves, being very robust on the easy examples and then falling more sharply on the difficult examples, as expected.

**Distributional and perturbational phenomena** When a high-quality system fails on a very easy instance this is a sign that something strange may be happening. There are many scenarios to be studied here: distributional shift, problem shift, adversarial attacks, distortion and Clever Hans phenomena. As a kind of distributional shift has been covered by looking at the changes of difficulty for different classes in Fig. 4, we study problem shift first. How does the difficulty estimator changes when we apply it to instances that come from a different problem (but shared features)? We can analyse this with 1000 instances from MNIST compared to 1000 instances from Fashion-MNIST (Table 10 in the appendix). The estimator for MNIST, when applied to Fashion-MNIST gives higher difficulty (from -3.10 to -2.75).The comparison of the same estimator for the two problems indicates that if these new 1000 instances were to be labelled with the original MNIST labels, they would be slightly more difficult than the original ones.

Let us now study several kinds of perturbations. To study this problem, we also take the MNIST dataset. We take 1000 examples away randomly ($S_{orig}$), and we make a 70%-30% train-testsplit with the remaining 69000 examples, building a *ResNet50* classifier (He et al. 2016). We estimate the difficulty for $S_{orig}$ (from the best $\hat{h}$ in Table 2(b)) and predict their class using the trained classifier. We transform the 1000 examples into adversarial examples ($S_{advl}$) and estimate their difficulty. We follow a similar procedure for the generation of a sample $S_{hans}$, where we introduce watermarks that help the classifier, and several samples $S^{low}_{blur}$, $S^{med}_{blur}$ and $S^{high}_{blur}$ with increasing degrees of distorsion (see the details for all these modifications in the appendix).

Table 4 sows the results. The original mean difficulty is -3.13, and the estimated mean difficulty is -3.07 for $S_{orig}$. Examples become just slightly more difficult for the adversarial variants ($S_{advl}$), which means that the adversarial attack has little effect on the difficulty estimator, despite the classifier failing for all examples. The difficulty is almost unchanged by the watermark ($S_{hans}$) despite the classifier succeeding for all of them. These discrepancies are so clear that

| | $S_{orig}$ | $S_{advl}$ | $S_{hans}$ | $S^{low}_{blur}$ | $S^{med}_{blur}$ | $S^{high}_{blur}$ |
|---|---|---|---|---|---|---|
| $\hbar$ (mean) | -3.13 | = | = | = | = | = |
| $\hat{h}$ (mean) | -3.07 | -2.77 | -3.09 | -3.05 | -2.75 | -2.45 |
| Error (mean) | 0.20% | 100.00% | 0.00% | 0.80% | 17.03% | 76.50% |

Table 4: Difficulties of a sample of 1000 original MNIST examples ($S_{orig}$) and the same examples altered in different ways: $S_{advl}$ are adversarial versions of the examples, $S_{hans}$ are added a watermark that easily discloses the class (Clever Hans) and $S_{blur}$ are modified with different degrees of blur.

can be used to detect these situations (adversarial attacks and Clever Hans phenomena). Only the levels of blur have an effect that translate on the estimated difficulties and the error. More blur makes images more difficult (from -3.05 to -2.45), as expected, and the classifier has higher error (from 0.80% to 76.5%). In general, if the difficulties change in the new samples as does the performance, we can calculate whether the performance corresponds to expected response given the ability of the model as per Eq. 2. If this deviates from the observed performance significantly, then we can fire an alarm.

While a discrepancy between the expected correct response and actual accuracy may be caused by many reasons, a difficulty-informed discrepancy discards situations where a low or high accuracy is just explained by a batch of difficult or easy instances respectively (as seen in Fig. 5).

## Discussion and Future Work

Estimating instance difficulty in a domain-generic, attribute-based and system-independent way, as we do in this paper for the first time, can have a significant impact on and broad applicability in almost any area of AI. It is of utmost importance to determine, for a new instance —and only from the information in that instance—, whether the system failure or success is in conformity with the difficulty of the instance. This can be extended to groups, classes or distributions to explain or anticipate important phenomena and situations.

Given the extent of a single first paper introducing this methodology, we have performed a systematic coverage of domains and tasks, analysed when the estimators work and how they work best, and suggested potential applications. There are many avenues of future work stemming from this paper. We encourage other researchers to improve the results of our difficulty estimators, and especially those for CIFAR-10, IMDb and SST-2. It is also important to understand those cases where the original task is easy but difficulty estimation is not. Our analysis of feature relevance goes in that direction, but further studies on latent features and representations can shed more light on this issue.

Finally, we would like to end with a recommendation. In many domains, it was not easy for us to find instance-wise results. Experiments with dozens of techniques and hyperparameters are usual with approaches such as AutoML (He, Zhao, and Chu 2021) and the widespread popularity of competitions, but unfortunately the associated papers and the data repositories only report the aggregated results. We need more initiatives such as OpenML (Vanschoren et al. 2014), from which we were able to extract the necessary information for a good proportion of our benchmarks.

## Acknowledgements

This work has been partially supported 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), the Future of Life Institute, FLI, under grant RFP2-152, US DARPA HR00112120007 (RECoG-AI), the AI-Watch and HUMAINT projects by DG CONNECT and DG JRC of the European Commission, and INNEST/2021/317 (Project cofunded by the European Union with the "Programa Operativo del Fondo Europeo de Desarrollo Regional (FEDER) de la Comunitat Valenciana 2014-2020"). We thank the anonymous reviewers for their comments and interaction during the discussion process. We are grateful to María José Ramírez-Quintana and Cèsar Ferri for useful discussions and for their help with questions regarding the automatic reasoning tasks.

## References

Birnbaum, A. 1968. *Statistical Theories of Mental Test Scores*, chapter Some Latent Trait Models and Their Use in Inferring an Examinee's Ability. Reading, MA.: Addison-Wesley.

Breiman, L. 2001. Random forests. *Machine learning*, 45(1): 5–32.

Chen, Z.; and Ahn, H. 2020. Item response theory based ensemble in machine learning. *International Journal of Automation and Computing*, 17(IJAC-2020-02-029): 621.

Chmait, N.; Dowe, D.; Li, Y.-F.; and Green, D. 2017. An information-theoretic predictive model for the accuracy of AI agents adapted from psychometrics. In Everitt, T.; Potapov, A.; and Goertzel, B., eds., *Artificial General Intelligence*, 225–236. Springer.

Corbière, C.; Thome, N.; Bar-Hen, A.; Cord, M.; and Pérez, P. 2019. Addressing failure prediction by learning model confidence. *arXiv preprint arXiv:1910.04851*.

Cuzick, J. 1985. A Wilcoxon-type test for trend. *Statistics in medicine*, 4(1): 87–90.

Embretson, S. E.; and Reise, S. P. 2000. *Item response theory for psychologists*. L. Erlbaum.

Frey, P. W.; and Slate, D. J. 1991. Letter recognition using Holland-style adaptive classifiers. *Machine learning*, 6(2): 161–182.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

He, X.; Zhao, K.; and Chu, X. 2021. AutoML: A Survey of the State-of-the-Art. *Knowledge-Based Systems*, 212: 106622.

Hendrycks, D.; and Gimpel, K. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*.

Hoover, D. L. 2003. Another perspective on vocabulary richness. *Computers and the Humanities*, 37(2): 151–178.

Jiang, H.; Kim, B.; Guan, M. Y.; and Gupta, M. 2018. To trust or not to trust a classifier. *arXiv preprint arXiv:1805.11783*.

Kincaid, J. P.; Fishburne Jr, R. P.; Rogers, R. L.; and Chissom, B. S. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch.

Kline, A.; Kline, T.; Shakeri Hossein Abad, Z.; and Lee, J. 2020. Using Item Response Theory for Explainable Machine Learning in Predicting Mortality in the Intensive Care Unit: Case-Based Approach. *J Med Internet Res*, 22(9): e20268.

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

Liu, D.; Xiong, Y.; Pulli, K.; and Shapiro, L. 2011. Estimating image segmentation difficulty. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, 484–495. Springer.

Martínez Plumed, F.; Castellano Falcón, D.; Monserrat Aranda, C.; and Hernández Orallo, J. 2022. Further Details on Predicting IRT Difficulty. *RiuNet* http://hdl.handle.net/10251/181335.

Martínez-Plumed, F.; and Hernández-Orallo, J. 2017. AI results for the Atari 2600 games: difficulty and discrimination using IRT. *2nd Int. WS Evaluating General-Purpose AI, Melbourne, Australia*.

Martínez-Plumed, F.; and Hernández-Orallo, J. 2018. Analysing Results from AI Benchmarks: Key Indicators and How to Obtain Them. *arXiv preprint arXiv:1811.08186*.

Martínez-Plumed, F.; and Hernández-Orallo, J. 2020. Dual Indicators to Analyse AI Benchmarks: Difficulty, Discrimination, Ability and Generality. *IEEE Transactions on Games*, 12(2): 121–131.

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

Martínez-Plumed, F.; Prudêncio, R. B. C.; Martínez-Usó, A.; and Hernández-Orallo, J. 2016. Making Sense of Item Response Theory in Machine Learning. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, Best Paper Award*, 1140–1148.

Moraes, J. V. C.; Reinaldo, J. T. S.; Prudencio, R. B. C.; and Silva Filho, T. M. 2020. Item Response Theory for Evaluating Regression Algorithms. In *2020 International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Richards, B. 1987. Type/token ratios: What do they really tell us? *Journal of child language*, 14(2): 201–209.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.;

et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252.

Smith, M. R.; Martinez, T.; and Giraud-Carrier, C. 2014. An Instance Level Analysis of Data Complexity. *Mach. Learn.*, 95(2): 225–256.

Vanschoren, J.; Van Rijn, J. N.; Bischl, B.; and Torgo, L. 2014. OpenML: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2): 49–60.

Vijayanarasimhan, S.; and Grauman, K. 2009. 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*, 2262–2269. IEEE.

Wright, B. D.; and Stone, M. H. 1979. *Best test design*. Mesa press.