Belaid, Mohamed Karim; Rabus, Maximilian; Krestel, Ralf; Hüllermeier, Eyke

Conference Paper — Manuscript Version (Preprint)

Compare-xAI: Toward Unifying Functional Testing Methods for Post-hoc XAI Algorithms into a Multi-dimensional Benchmark

Suggested Citation: Belaid, Mohamed Karim; Rabus, Maximilian; Krestel, Ralf; Hüllermeier, Eyke (2023) : Compare-xAI: Toward Unifying Functional Testing Methods for Post-hoc XAI Algorithms into a Multi-dimensional Benchmark, In: Longo, Luca (Ed.): Explainable Artificial Intelligence, First World Conference, xAI 2023, Lisbon, Portugal, July 26–28, 2023, Proceedings, Part II, Springer Nature, Berlin, pp. 88-109,
https://doi.org/10.1007/978-3-031-44067-0_5

This Version is available at:
http://hdl.handle.net/11108/588

Compare-xAI: Toward Unifying Functional Testing Methods for Post-hoc XAI Algorithms into an Interactive and Multi-dimensional Benchmark

Mohamed Karim Belaid
IDIADA Fahrzeugtechnik GmbH
Munich, Germany
Karim.Belaid@idiada.com

Eyke Hüllermeier
University of Munich (LMU)
Munich, Germany
eyke@if.lmu.de

Maximilian Rabus
Dr. Ing. h.c. F. Porsche AG
Stuttgart, Germany
maximilian.rabus2@porsche.de

Ralf Krestel
ZBW - Leibniz Centre for Economics
& Kiel University
Kiel, Germany
r.krestel@zbw.eu

Abstract

In recent years, Explainable AI (xAI) attracted a lot of attention as various countries turned explanations into a legal right. xAI allows for improving models beyond the accuracy metric by, e.g., debugging the learned pattern and demystifying the AI's behavior. The widespread use of xAI brought new challenges. On the one hand, the number of published xAI algorithms underwent a boom, and it became difficult for practitioners to select the right tool. On the other hand, some experiments did highlight how easy data scientists could misuse xAI algorithms and misinterpret their results. To tackle the issue of comparing and correctly using feature importance xAI algorithms, we propose Compare-xAI, a benchmark that unifies all exclusive functional testing methods applied to xAI algorithms. We propose a selection protocol to shortlist non-redundant functional tests from the literature, i.e., each targeting a specific end-user requirement in explaining a model. The benchmark encapsulates the complexity of evaluating xAI methods into a hierarchical scoring of three levels, namely, targeting three end-user groups: researchers, practitioners, and laymen in xAI. The most detailed level provides one score per test. The second level regroups tests into five categories (fidelity, fragility, stability, simplicity, and stress tests). The last level is the aggregated comprehensibility score, which encapsulates the ease of correctly interpreting the algorithm's output in one easy to compare value. Compare-xAI's interactive user interface helps mitigate errors in interpreting xAI results by quickly listing the recommended xAI solutions for each ML task and their current limitations. The benchmark is made available at https://karim-53.github.io/cxai/

1 Introduction

xAI algorithms are a set of approaches toward understanding black-box models. In recent years, xAI algorithms helped debug manifold issues in ML models, such as exposing underlying wrong patterns in classifying objects [1] or highlighting inequality and bias in decisions [2]. Moreover, given its essential impact on society, legislation in several countries now includes the "Right to explanation" [3] fulfilled by the various xAI tools available in the literature. It is indeed difficult to define the best xAI solution given the number of known evaluation metrics. Moreover, the long evolutionary history of

Preprint. Under review.

specific xAI methods makes it even more difficult to evaluate each version. The Shapley values are an excellent example of this challenge. Sundararajan et al. did state that "...the functional forms of the Shapley value...are sufficiently complex as to prevent direct understanding..." [4]. Indeed, going through the theoretical background of Shapley values [5], its multiple approximations [6, 7], generalizations [8, 4] and final implementations [9, 10] adapted to the AI field might mislead the end-user on the capability of the available tools.

Resulting challenges. Consequently, data scientists face considerable difficulties in accurately evaluating each xAI algorithm and remaining up-to-date on its evolution. This issue yields a clearly visible symptom known as the illusion of explanatory depth [11] in interpreting xAI results [12] as it has been confirmed that data scientists are prone to misuse interpretability tools [13]. Many researchers did address this question by stressing the importance of structuring and documenting xAI algorithms [14, 15], i.e., by highlighting the target end-users of the algorithm, its capability, limitations, and vulnerabilities. Finally, they recommend using quantitative metrics to make claims about explainability.

Functional testing as a solution to stated recommendations. Functional testing aim to verify the end-user's requirement on the xAI algorithm by performing end-to-end tests in a black-box fashion. In other words, every functional test apply the xAI algorithm on a frozen AI model to verify if the output corresponds to the explanation expected by data scientists. A functional test could verify that the explanation accurately reflect the AI model (Fidelity), that it is not sensitive to adversarial attacks (Fragility), that it is stable to small variation in the model (Stability), etc. Functional testing remains unfortunately sparsely used in literature and, thus, provides only a partial evaluation.

Given the unsolved burden of evaluating and correctly interpreting xAI results, we propose Compare-xAI that mitigates these two issues (benchmark xAI results and the illusion of explanatory depth during the interpretation of results) by addressing three research questions:

1. How to select exclusive functional tests from those proposed in the literature?
2. How to score xAI algorithms in a simple way despite the multitude of evaluation dimensions?
3. How to reduce data scientists' potential misuse of the xAI algorithms?

The stated questions are resolved as follows: In Section 3, we propose a benchmark implementation easily scalable to new xAI algorithms and new functional tests. In Section 3.1, we propose a selection protocol for the quantitative evaluation of xAI algorithms. It is then applied to shortlist a selection of exclusive tests, each targeting a distinct end-user requirement. In Section 3.2, we explain the experiments' protocol to mimic the end-user's behavior. In Section 3.3, we propose an intuitive scoring method that scales in detail with the level of expertise of the data scientist: Layman data scientists are invited to manipulate one global score named comprehensibility. Practitioners are invited to compare xAI algorithms given five scores representing five subcategories of the comprehensibility metric. Finally, researchers are invited to study the detailed report (one score per test). In Section 4, we propose a user interface that encapsulates the benchmark's results. We seek to minimize the potential misuse of xAI algorithms by offering quick access to the limitation of each xAI algorithm. Finally, Section 5 is dedicated to the theoretical and practical limitations of the benchmark.

2 Related Work

This section is a survey for xAI evaluation methods. It contains examples contrasting the difference between functional tests and portability tests. Following that, we examine some attempts to regroup them into surveys or benchmarks.

2.1 xAI Evaluation Methods: Functional Tests vs. Portability Tests

Researcher in the xAI field often propose a new method along with a set of functional or portability tests that outline the contrast between former work and their contribution.

Functional tests. Functional testing is a popular testing technique for software engineers. The following definition is adapted from the software engineering field to our intended usage in machine learning [16]. Functional tests are created by testers with no specific knowledge of the algorithm's internal modules, i.e., not the developers themselves. Therefore, the algorithm is considered a black-box and is executed from end to end. Each functional test is intended to verify an end-user requirement on the xAI algorithm rather than a specific internal module. Thus, functional tests share the advantage of being able to test different algorithms. On the other hand, failed tests do not inform about the location of the errors but rather attribute it to the entire algorithm. Functional test for xAI algorithms usually exploit tabular synthetic data and few input features, e.g., considering the "cough and fever" test [9], The xAI algorithm is expected to detect symmetry between the two binary features. Simple examples showcase the undeniable limit of certain xAI methods. Nevertheless, specific tests could use real-world data. A good example is the MNIST dataset [17] used as a counterexample for the dummy axiom [18]: Since edge pixels are always black, a multi-layer perceptron will learn not to rely on these constant pixels. As a consequence, the xAI algorithm should confirm that the AI does not use these pixels. Papers proposing new xAI methods remain too short to list all known tests. Furthermore, some of the highlighted issues might be fixed without any publication.

Portability Tests. Portability tests for xAI algorithms evaluate real-word models and demonstrate the robustness of the xAI algorithm against multiple challenges at once (noise, correlated inputs, large inputs, etc.). They are used to claim the potential broad usage of one xAI method rather than demonstrating the quality of the explanation. An example is the verification of the portability across recommendation tasks [19].

Navigating the ocean of tests remains itself a huge challenge. First, many examples in the literature are portability tests which makes comparison between xAI algorithms complex. Second, tests could be redundant to emphasize the frequent occurrence of an issue, e.g., testing interaction detection with different transparent models [19]. Third, researchers could argue the correctness of specific functional tests' ground truth, e.g., causal explanation of the Shapley values [20] has been considered false in certain research [4].

Given the tremendous amount of xAI algorithms and dedicated metrics, surveys [21–24] have trouble providing an in-depth analysis of each algorithm and cannot cope with ongoing implementation updates. Nevertheless, Molnar's online book distinguishes itself with a continuously updated survey about xAI [25]. The initiative of a real-time survey faced great success and acceptance from the data science community.

2.2 Benchmark for xAI Algorithms

There are specialized benchmarks in the literature, like the SVEA benchmark [26]. The latter focuses on computer vision tasks and proposes faster evaluations based on the small mnist-1D dataset [27]. Another benchmark utilizes exclusively human evaluation to assess xAI algorithms on real-world tasks [28]. On the one hand, benchmarking using computer vision and NLP models permits to measure the real success of an xAI tool in helping end-users even though human evaluation could be considered subjective and more costly to obtain. On the other hand, evaluation using real-world tasks does not allow debugging the xAI algorithm, i.e., two algorithms might fail to explain one black-box model for two different reasons.

xAI-Bench [29] evaluates each xAI algorithm on five metrics. Faithfulness measures the Pearson correlation between the feature importance and the approximate marginal contribution of each feature. Of course, one could argue that the ground truth explanation of a model could be slightly different from the marginal contribution of each feature on the observed dataset. The same argument holds for the monotonicity, infidelity, and GT-Shapley metrics. They define a ground truth output, that is, a "better" xAI algorithm is the one outputting a result that is closer to the ground truth. In contrast, functional tests discussed in previous paragraphs evaluate the correctness of the output using a pattern (not an exact ground truth). This paper focuses on functional tests using patterns as evaluation methods. The fifth metric used in xAI-Bench is remove-and-retrain (ROAR). It involves a re-evaluation of the model, which could itself alter the evaluation. Another critical factor affecting the scores and the ranking of the algorithms is the data distribution. The authors did circumvent the issue by testing on different distributions. However, it remains difficult to decide if the algorithm is failing this specific test or if it is generally sensitive to the data distribution. xAI-Bench is an excellent initiative to benchmark the correctness of an xAI algorithm, except that it does not allow a clear debugging and does not propose any final ranking of the xAI algorithm to help practitioners and laymen quickly pick the right tool. Following the analysis of related work, Section 3 details how our proposed benchmark addresses the highlighted issues.

3 Compare-xAI

Compare-xAI is a quantitative benchmark based solely on functional tests. Compare-xAI is able to evaluate any new xAI algorithm or index any new test. Current proof-of-concept evaluates more than 16 post-hoc xAI algorithms on more than 22 functional tests. Compare-xAI outputs a series of scores that allows a multi-dimensional analysis of comparable xAI algorithms. Figure 1 illustrates Compare-xAI as a pipeline with three added values: First, we collect the known functional tests reported in related literature and filter them according to a clear protocol stated in Section 3.1. As a second step, each algorithm is tested automatically on each of the collected tests. Experiments follow a protocol detailed in Section 3.2. Each experiment results in one score ranging from 0 (failing) to 1 (succeeding). Compare-xAI has the exclusive advantage of reporting an intermediate score (between 0 and 1) if the algorithm is partially failing the test. Obtained raw results describe an xAI algorithm by 22 scores, and it is not easy, at this point, to compare two algorithms. Therefore, raw results are aggregated into a hierarchical scoring method, see Section 3.3.

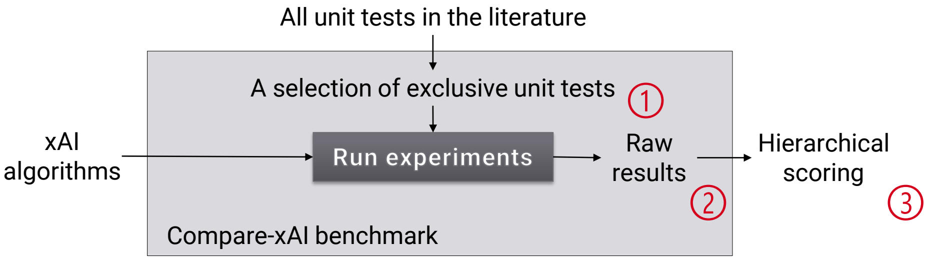


Figure 1: Compare-xAI's pipeline

3.1 Tests Selection Protocol

A functional test consists of a dataset, a model to explain, and an assertion (e.g., an xAI algorithm is expected to detect symmetric input features). Section 2.1 highlighted the diversity of tests used in the literature and the importance of filtering inadequate ones. We propose the following rules to ensure a multi-dimensional evaluation of all post-hoc xAI algorithms:

One end-user requirement per test. Selected tests should identify and debug a clear end-user requirement within an xAI algorithm. This rule ensures that an algorithm will not fail multiple tests for the same reason, and it allows researchers quickly debug the xAI algorithm. Table 1 regroups a non-exhaustive set of unitary end-user requirements that could alter the algorithm's output and let it fail in explaining the model correctly. An example of tests sharing the same end-user requirement is "The failure of the dummy axiom" [4] and "The failure of the linearity axiom" [4]: The first test verifies that specific xAI algorithms cannot detect dummy inputs features when the data distribution is modified. The second test verifies linearity between two AI models' output and their explanations. Each test explains how its respective axiom could fail while both share the same root cause: the effect of data distribution on the xAI algorithm. Therefore, Compare-xAI does not implement "The failure of the linearity axiom" [4]. Nevertheless, another test could be found in the literature to verify exactly this axiom.

Undebatable test. Explanations could target different reasoning. If 2 opposite explanations could be considered correct given the same setup (dataset, AI model), then Compare-xAI would not include such a test. A popular example is the causal explanation of the SHAP values [30].

No exact ground truth explanation. A test contains a scoring function that compares the xAI output to the expected output. The expected output could be an exact set of values, e.g., GT-Shapley's expected output is the Shapley values [29]. The expected output could also be a pattern, e.g., given one specific model, feature A's importance should be the highest,

Table 1: Samples from the shortlisted functional tests.

Category	Grouped functional tests
Fidelity	**Does the algorithm's output reflect the underlying model ?** aka faithfulness [32], consistency [9] • Counterexample for the symmetry axiom [9]. • Test whether features of different importance are represented correctly [9]. • Test the detection of feature interaction based on mathematical terms [19]. • Effect of feature product on local explanations [20]. • Test if one-hot encoded features are explained correctly [33]. • Test if main terms and interaction terms are evaluated correctly [34].
Fragility	**Is the explanation result susceptible to malicious corruption?** • Adversarial attacks can exploit feature perturbation-based xAI algorithms as a vulnerability to lower the importance of specific features [35].
Stability	**Is the algorithm's output too sensitive to slight changes in the data or model?** • Effect of data distribution: Statistical dependence, non-uniform distribution [30] • Effect of feature correlations [29, 13] • Effect of noise in the dataset [4] • Implementation invariance axiom
Simplicity	**Can users look at the explanation and easily reason about the model's behavior?** aka Explicitness/Intelligibility [32] • Counterexample of the dummy axiom [4] • Counterexample of the linearity axiom [4]
Stress	**Can the algorithm explain models trained on big data?** • Test if the xAI algorithm is sensitive to a high number of word tokens (NLP task) [7]. • Detect dummy pixels in the MNIST dataset [18]. • Effect of data sparsity [4].
Other	**Remaining metrics are not integrated into the hierarchical scoring system albeit reported in the final dataset.** • Portability [25] measures the diversity of models' implementation that an xAI algorithm can explain. Portability is tested implicitly by different tests as each one implements a different model/dataset. • The relative execution time is an essential factor in choosing an algorithm, and it is mainly influenced by the stress tests.

or negative, or equal to feature B's. Compare-xAI's shortlisted tests rely only on patterns. Protecting this degree of freedom makes this benchmark compatible with different xAI approaches. For example, in the case of an adversarial attack test, it is expected that the ranking of the feature importance does not get affected by corrupted models, regardless of the exact ranking proposed. Another good example is the test that assesses the symmetry axiom. Its evaluation function verifies the equality between the feature importance values without having an exact value as a reference.

Non-redundant tests. Redundant tests emphasis the failure or the robustness of an xAI. Considering the scoring method of Compare-xAI, redundant tests can not be included. they are manually reported and eliminated from the selection.

Only tests proposed in the related work. The first iteration of the Compare-xAI benchmark is limited to the tests reported in former research papers, as many have been presented, discussed, and heavily criticized in the literature. Compare-xAI takes advantage of this extensive research work to build a consistent benchmark. Identifying not examined end-user requirements and proposing new tests is rewarding field of future research.

Categorizing functional tests. In order to cover a large variety of end-user requirements, we propose to categorize shortlisted functional tests into five common groups [31], see Table 1.

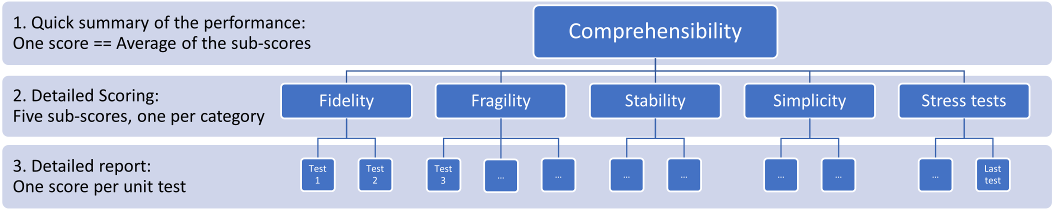


Figure 2: Hierarchical scoring

3.2 Experiments Protocol

An experiment takes one test and one xAI algorithm. First, the test environment is initialized by loading the data and training the model. Then the xAI algorithm is asked to explain the model (Globally or for specific data points). Finally, the explanation is compared to the correct answer, and one final score, a real number between 0 and 1, is returned.

It might seem to end-users that the stated patterns in Table 1 are self-evident and verified for every xAI algorithm. As a matter of fact, the public availability and wide usage of particular xAI tools “swayed several participants to trust the tools without fully understanding them” [13]. For this reason, we score each xAI algorithm by following the most common usage of the xAI algorithms. This policy induces the following rules:

Experiments will only be run once. Recent research revealed “a misalignment between data scientists’ understanding of interpretability tools and these tools’ intended use. Participants misused the tools (either over- or under-used them).” [13]. Compare-xAI is intended for this former group, that is, data scientists not running complementary experiments or repeating the same experiment to test the effect of the noise. Targeting the remaining data scientist, i.e., experts with advanced knowledge of the tools’ limits, is left for future work.

No fixed seed for random number generators. For the same reasons stated above, the seed is not fixed. Shortlisted tests’ underlying noise does not hinder any xAI method from correctly explaining a model, i.e., initialized models always learn the tested patterns independently of the seed.

No parameter tuning. Compare-xAI evaluates algorithms using their default parameters for all tests. Nevertheless, certain xAI algorithm adapts their parameters internally given each task by relying on the model’s structure, the dataset size, and the ML task. Around half of the indexed algorithms have at least one binary parameter, and the performance of some would vary with parameter tuning. As there are no precise methods or public tools to fine-tune parameters of xAI algorithms, we suppose the usage of default parameters, by end-users, to be a common misuse of the xAI algorithm. It is important that Compare-xAI reproduces this behavior in order to calculate the comprehensibility score in practice, that is, the ease of correctly interpreting the algorithm’s output, considering common practice.

3.3 Scoring Protocol

Compare-xAI’s tests result in a set of scores per algorithm, which we call “raw” results. A complete comparison between two algorithms A and B should consider the following four points:

(1)A‘s score for ea ch test is greater than or equal to B‘s score;
(2) For each test, A’s execution time is less than or equal to B’s execution time;
(3) B’s supported models are a subset of A’s (see the portability metric definition in Table 1); and
(4) B’s supported output explanations are a subset of A’s.

As a consequence, sorting xAI algorithms by performance is a multi-metric ranking problem. Comparing even two algorithms using this property is impossible, especially with a considerable number of tests. This formulation of the challenge outlines the difficulty faced by practitioners/laymen who are looking for a quick explanation of their models but cannot decide on which xAI algorithm to pick. This challenge is addressed by proposing a relaxation of the scoring dimensions.

First, let us suppose that each end-user is comparing a subset of xAI algorithms which are all able to explain the model of his/her interest, and they all offer the type of explanation required by the end-user (feature importance, attribution, interaction, etc.). On this account, requirements **(3)** and **(4)**, stated above, might be skipped. Second, looking at the overall distribution of the scores, There is no state-of-the-art algorithm that succeeds in all tests. Thus, we opt to promote the "on-average better explanation" comparison method.

Hierarchical scoring. Figure 2 explains the hierarchical scoring proposed to the end-user to simplify the comparison between xAI algorithms. Level three, the last level in the hierarchical scoring, represents the most detailed report (one score per test) used in the classical ranking method. To simplify it, level two regroups tests per category to propose five scores described in Table 1. Finally, the first level aggregates the scores into one value, which we named the comprehensibility score.

Comprehensibility score. Comprehensibility is defined in the literature as "how much effort is needed for a human to interpret a model correctly?" [36]. Comprehensibility was used as a qualitative metric in related work. To quantify the Comprehensibility metric, we reformulate the definition as follows: For an AI model and an xAI algorithm, the effort needed for an end-user to interpret a model correctly is represented by the number of operations he should perform to obtain a correct explanation from the xAI. An operation could be, for example, multiple runs of the xAI algorithm to obtain a stable average explanation; a check/edition of the test dataset to adapt the distribution to the xAI's needs; or to perform additional checks against potential adversarial attacks exploiting the xAI algorithm's vulnerabilities. We quantify the Comprehensibility metric by linking the number of operations to the number of failed functional tests. In other words, if the xAI fails a specific test, the end-user will have to perform additional operations to obtain a correct explanation. We calculate the Comprehensibility metric as the average over the five subcategories of the second level. Therefore, comparing xAI algorithms becomes more accessible using the Comprehensibility score and the average execution time.

Finally, the evaluation's result is reduced to the comprehensibility score and the average execution time. Remains the dilemma of choosing the fastest algorithm or the most "comprehensible" one from the Pareto front. Depending on his/her level of expertise, the end-user is invited to consult any of the three scoring methods made available via the web user interface.

4 Visualization of the Benchmark

For the proof of concept, we consider a small set of popular xAI methods, implement a user interface to easily explore the benchmark results, and make the results publicly available[1]. Figures 3 and 4 do not represent a final benchmark as the set of tests and xAI algorithms are constantly updated. In this section, the demonstration will focus on feature importance. Nevertheless, the benchmark remains adaptable to many forms of post-hoc xAI methods.

Figure 3 summarizes the performance of the considered set of xAI methods. The best algorithm has the lowest execution time, highest score, and highest portability (bigger dot size). The Pareto front regroups the closest algorithms to the perfect one. End-users could use the filters on the web interface to describe a specific use case: Figure 4 is restricted to the set of model-agnostic xAI algorithms that output (at least) global feature importance. Available filters help the end-user quickly and accurately navigate the massive amount of undocumented properties of available xAI tools.

At this stage, the end-user picks an xAI algorithm that matches his expected performance and execution time requirements. The detailed report is accessible by clicking on the blue dot representing the algorithm: the end-user gets access to information about the supported AI models, the output of the xAI algorithm, additional information required by the xAI algorithm to run correctly, and essentially the score obtained on each test. The detailed report helps the end-user quickly understand the limit of the xAI algorithm before using it. Finally, the end-user can compare multiple xAI algorithms given a set of tests that reflect his usage of AI. Please keep in mind that selected screenshots are for demonstration purposes only and we are not going to discuss individual scores, but only the benchmark itself.

[1] `https://karim-53.github.io/cxai/`

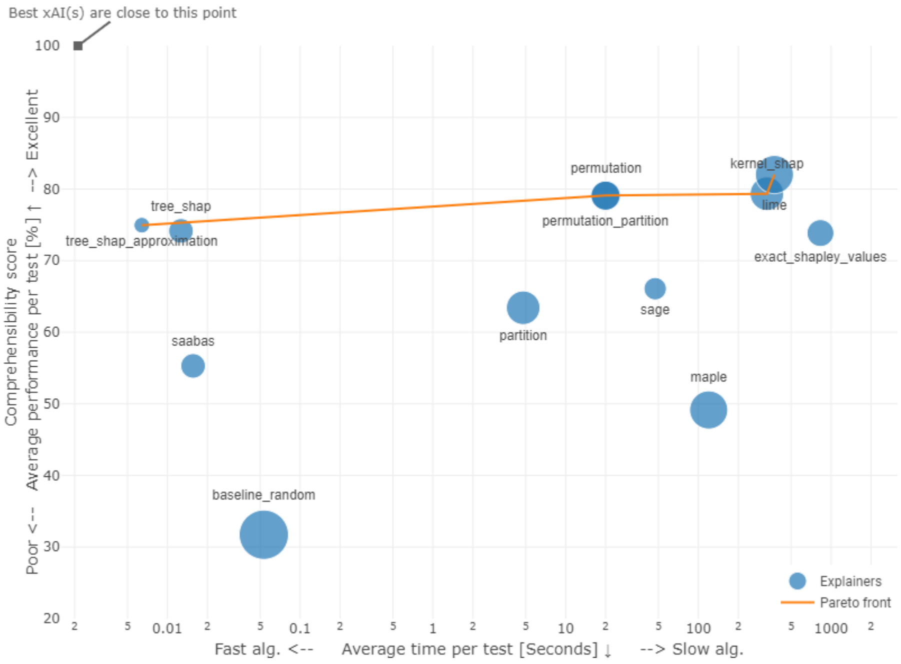


Figure 3: Global overview of the benchmark

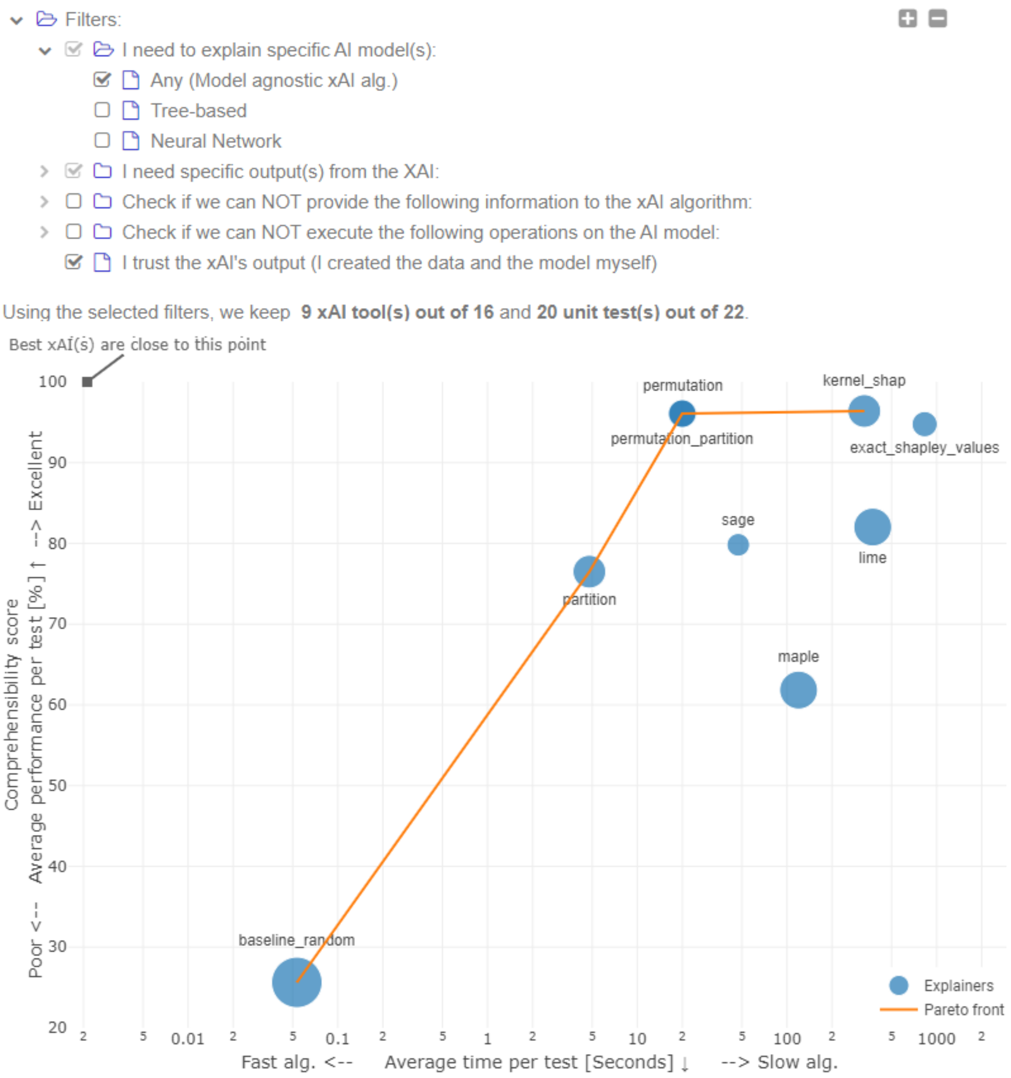
Discussion Compare-xAI is a dynamic benchmark, and its results evolve with the filtered tests/xAI algorithms. Therefore a global analysis of the results is performed without reporting any specific number. First, considering all implemented unit tests without filtering, none of the xAI algorithms did obtain the perfect score. Second, obtained comprehensibility scores are very close (50% of the xAI algorithms obtained a comprehensibility score between 0.70 and 0.85). This clustering reflects the original structure of these xAI algorithms as the majority are permutation-based algorithms. At this level of the analysis, the end-user did save a huge amount of time by understanding which algorithms are almost equivalent and which are relatively faster / explaining better. Same for the scores of the second level, analyzing the five scores quickly locate the weak point of a chosen xAI algorithm, since the average difference between the smallest and biggest score is 71.9% (±31.4%). Finally, end-users can check the detailed report (level 3). They will mainly go through the failed tests (score < 0.05) or partially failed tests ($0.05 \leq$ score < 0.95). On average, an xAI algorithm is eligible to 12.3 tests (± 5) depending on its portability. The distribution of failed tests is skewed, see Figure 5. Thus, the median is more representative. The median percentages of failed tests and partially failed tests are 17.6% and 38.7%, respectively. Thus, checking the detailed report is expected to be fast.

5 Limitations and Future Work

Compare-xAI's weaknesses are classified into design-related and implementation-related limitations.

Design-related limitations. Compare-xAI is a benchmark made exclusively of quantitative metrics. It is objective as it does not include tests based on human evaluation. A common example from is the study of the human mental model like the investigation of users' preferred explanation style [37]). Another example is the study of the information overload, e.g., xAI's additional output information like the confidence interval [18]. Mainly, empirical studies are challenging to quantify [38] and integrate into the comprehensibility score. These and other non-quantifiable advantages/disadvantages will be included in the description of the xAI algorithm, in the future.

implementation-related limitations. The provided proof of concept includes, for now, 22 tests. Currently, none of them cover RL, GAN, or unsupervised learning tasks. The tested form of output is also limited to feature importance (global explanation). Testing feature interaction is still under development.



- Filters:
  - ☑ I need to explain specific AI model(s):
    - ☑ Any (Model agnostic xAI alg.)
    - ☐ Tree-based
    - ☐ Neural Network
  - ☑ I need specific output(s) from the XAI:
  - ☐ Check if we can NOT provide the following information to the xAI algorithm:
  - ☐ Check if we can NOT execute the following operations on the AI model:
  - ☑ I trust the xAI's output (I created the data and the model myself)

Using the selected filters, we keep **9 xAI tool(s) out of 16** and **20 unit test(s) out of 22**.

Best xAI(s) are close to this point
100
90
80
70
60
50
40
30
20
Poor <-- Comprehensibility score Average performance per test [%] ↑ --> Excellent
permutation
permutation_partition
kernel_shap
exact_shapley_values
sage
lime
partition
maple
baseline_random
Explainers
Pareto front
2
5
0.01
2
5
0.1
2
5
1
2
5
10
2
5
100
2
5
1000
2
Fast alg. <-- Average time per test [Seconds] ↓ --> Slow alg.

Figure 4: Benchmark of model-agnostic xAI algorithms outputting feature importance

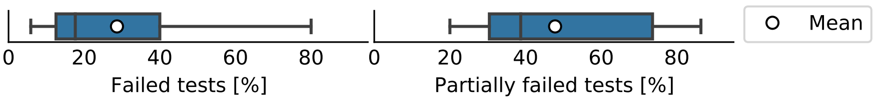


Figure 5: Box plot of test status per xAI algorithm

Despite the stated limitations, Compare-xAI should fulfill its primary objectives: first, helping laymen pick the right xAI method, and second, helping researchers, practitioners, and laymen avoid common mistakes in interpreting its output.

6 Conclusion

Explaining AI is a delicate task, and some end-users are prone to misuse dedicated tools [13]. We propose Compare-xAI a unified benchmark indexing +16 post-hoc xAI algorithms, +22 tests, and +40 research papers. Compare-xAI reproduces experiments following a selection protocol that highlights the contrast between the theoretical claims of the authors in a paper and the practical implementation offered to the end-user. Selected tests measure diverse properties. The authors did not create any xAI algorithm. Therefore, there is no conflict of interest. Compare-xAI proposes to deliver the results using an interactive interface as a solution to mitigate human errors in interpreting xAI outputs by making the limits of each method transparent. Compare-xAI proposes a simple and intuitive scoring method that efficiently absorbs the massive quantity of xAI-related papers. Finally, Compare-xAI proposes a partial sorting of the xAI methods, toward unifying post-hoc xAI evaluation methods into an interactive and multi-dimensional benchmark.

Broader Impact

Compare-xAI is a benchmark with multiple use-cases. It can be seen as a debugging tool for individual xAI algorithms but simultaneously as a global benchmark. Even if Compare-xAI does not offer a total sorting per performance, still it separates comparable algorithms into the Pareto front and the rest. Compare-xAI allows practitioners to quickly and correctly filter xAI algorithms given their needs and to outline the limitations of the selected ones. The end-user, now aware of the limit of the xAI algorithm, would not over-trust the algorithm's output and would avoid common mistakes in explaining a model. On the other hand, Compare-xAI allows researchers to access a detailed scoring and to answer specific questions such as "In which case does this xAI algorithm fail?", "Is it the only one to solve this issue?", "What kind of cases are still not covered by any xAI algorithm?" etc. Compare-xAI continuously re-evaluates indexed xAI algorithms to keep an updated benchmark of the state-of-the-art. Finally, Compare-xAI is more than a benchmark: it is a comprehensive and standardized related work analysis, while it also works as an evaluation method for new research papers in xAI.

Acknowledgments

We thank Dorra El Mekki, Jonas Steinhäuser, Tim Donkiewicz, Marius Daehling, Maximilian Muschalik, Patrick Kolpaczki, and Michael Rapp for their thoughtful feedback on earlier iterations of this work.

References

[1] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[2] Julia Dressel and Hany Farid. The accuracy, fairness, and limits of predicting recidivism. *Science advances*, 4(1):eaao5580, 2018.

[3] Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI magazine*, 38(3):50–57, 2017.

[4] Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. In *International conference on machine learning*, pages 9269–9278. PMLR, 2020.

[5] LS Shapley. Quota solutions op n-person games1. *Edited by Emil Artin and Marston Morse*, page 343, 1953.

[6] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014.

[7] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[8] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[9] Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

[10] Mateusz Staniak and Przemyslaw Biecek. Explanations of model predictions with live and breakdown packages. *arXiv preprint arXiv:1804.01955*, 2018.

[11] Leonid Rozenblit and Frank Keil. The misunderstood limits of folk science: An illusion of explanatory depth. *Cognitive science*, 26(5):521–562, 2002.

[12] Michael Chromik, Malin Eiband, Felicitas Buchner, Adrian Krüger, and Andreas Butz. I think i get your point, ai! the illusion of explanatory depth in explainable ai. In *26th International Conference on Intelligent User Interfaces*, pages 307–317, 2021.

[13] Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna Wallach, and Jennifer Wortman Vaughan. Interpreting interpretability: understanding data scientists' use of interpretability tools for machine learning. In *Proceedings of the 2020 CHI conference on human factors in computing systems*, pages 1–14, 2020.

[14] Matthew L Leavitt and Ari Morcos. Towards falsifiable interpretability research. *arXiv preprint arXiv:2010.12016*, 2020.

[15] Michael Kearns and Aaron Roth. *The ethical algorithm: The science of socially aware algorithm design*. Oxford University Press, 2019.

[16] Boris Beizer. *Black-box testing: techniques for functional testing of software and systems*. John Wiley & Sons, Inc., 1995.

[17] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[18] Ian Covert, Scott M Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, 33:17212–17223, 2020.

[19] Michael Tsang, Sirisha Rambhatla, and Yan Liu. How does this interaction affect me? interpretable attribution for feature interactions. *Advances in neural information processing systems*, 33:6147–6159, 2020.

[20] I Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. Problems with shapley-value-based explanations as feature importance measures. In *International Conference on Machine Learning*, pages 5491–5500. PMLR, 2020.

[21] Sina Mohseni, Niloofar Zarei, and Eric D Ragan. A multidisciplinary survey and framework for design and evaluation of explainable ai systems. *arXiv preprint arXiv:1811.11839*, 2018.

[22] Michael Tsang, James Enouen, and Yan Liu. Interpretable artificial intelligence through the lens of feature interaction. *arXiv preprint arXiv:2103.03103*, 2021.

[23] Plamen P Angelov, Eduardo A Soares, Richard Jiang, Nicholas I Arnold, and Peter M Atkinson. Explainable artificial intelligence: an analytical review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(5):e1424, 2021.

[24] Benedek Rozemberczki, Lauren Watson, Péter Bayer, Hao-Tsung Yang, Olivér Kiss, Sebastian Nilsson, and Rik Sarkar. The shapley value in machine learning. *arXiv preprint arXiv:2202.05594*, 2022.

[25] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[26] Sam Sattarzadeh, Mahesh Sudhakar, and Konstantinos N Plataniotis. Svea: A small-scale benchmark for validating the usability of post-hoc explainable ai solutions in image and signal recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4158–4167, 2021.

[27] Sam Greydanus. Scaling down deep learning. *arXiv preprint arXiv:2011.14439*, 2020.

[28] Sina Mohseni, Jeremy E Block, and Eric D Ragan. Quantitative evaluation of machine learning explanations: A human-grounded benchmark. *arXiv preprint arXiv:1801.05075*, 2020.

[29] Yang Liu, Sujay Khandagale, Colin White, and Willie Neiswanger. Synthetic benchmarks for scientific research in explainable machine learning. *arXiv preprint arXiv:2106.12543*, 2021.

[30] Dominik Janzing, Lenon Minorics, and Patrick Bl"obaum. Feature relevance quantification in explainable ai: A causal problem. In *International Conference on artificial intelligence and statistics*, pages 2907–2916. PMLR, 2020.

[31] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. Faithful and customizable explanations of black box models. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 131–138, 2019.

[32] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. *Advances in neural information processing systems*, 31, 2018.

[33] Salim I Amoukou, Tangi Salaün, and Nicolas Brunel. Accurate shapley values for explaining tree-based models. In *International Conference on Artificial Intelligence and Statistics*, pages 2448–2465. PMLR, 2022.

[34] Andreas Tilevik. *Two-way anova - the basics*. payhip.com, 2021.

[35] Himabindu Lakkaraju and Osbert Bastani. "" how do i fool you?"" manipulating user trust via misleading black box explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 79–85, 2020.

[36] Freddy Lecue, Krishna Gade, Sahin Geyik, Kenthapadi Krishnaram, Varun Mithal, Ankur Taly, Riccardo Guidotti, and Pasquale Minervini. Explainable ai: Foundations, industrial applications, practical challenges, and lessons learned. In *Proceedings of the AAAI Conference*, pages 131–138, 2020.

[37] Jeya Vikranth Jeyakumar, Joseph Noor, Yu-Hsi Cheng, Luis Garcia, and Mani Srivastava. How can i explain this to you? an empirical study of deep neural network explanation methods. *Advances in Neural Information Processing Systems*, 33:4211–4222, 2020.

[38] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

[39] Giles Hooker, Lucas Mentch, and Siyu Zhou. Unrestricted permutation forces extrapolation: variable importance requires at least one more model, or there is no free variable importance. *Statistics and Computing*, 31(6):1–16, 2021.

[40] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3681–3688, 2019.

[41] Gregory Plumb, Denali Molitor, and Ameet S Talwalkar. Model agnostic supervised local explanations. *Advances in neural information processing systems*, 31, 2018.

[42] Guillermo Owen. Multilinear extensions of games. *Management Science*, 18(5-part-2):64–79, 1972.

[43] Mukund Sundararajan, Kedar Dhamdhere, and Ashish Agarwal. The shapley taylor interaction index. In *International conference on machine learning*, pages 9259–9268. PMLR, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Compare-xAI is a benchmark for feature importance xAI algorithms. It includes a user interface that helps data scientists quickly run through the scores.
   (b) Did you describe the limitations of your work? [Yes] see 5
   (c) Did you discuss any potential negative societal impacts of your work? [No] We discuss the societal impacts in the "Broader impact" section, but we would not qualify it as negative.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] There are no theoretical results.
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments (e.g., for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Our code is available at `https://github.com/Karim-53/Compare-xAI` and the main readme contains the instructions on how to re-run experiments.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Each test has its own training details defined in its class.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] The protocol of the benchmark reproduces a specific behavior of a layman: results are reported after only one run. We aim to target the second group of end-users (those who run experiments multiple times) in future work by elaborating more on what to report (min, max, or average) and how to keep an equitable comparison between stable and noisy algorithms. Nevertheless, the benchmark stays stable because it averages over multiple tests.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See `https://github.com/Karim-53/Compare-xAI/blob/main/README.md#23-computing-ressouces=`
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See `https://github.com/Karim-53/Compare-xAI/blob/main/README.md#reference=`
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] `https://github.com/Karim-53/Compare-xAI/blob/main/LICENSE`
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We do not survey people.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] No crowdsourcing.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] No crowdsourcing.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] No crowdsourcing.

A Tests

For the proof-of-concept, the following list of tests is considered. Note that some tests count twice as they test both feature importance and feature attribution.

cough_and_fever answers the following question: *Can the xAI algorithm detect symmetric binary input features?*. The trained model's equation is [Cough AND Fever]*80. The test utilize **XGBRegressor** model trained on **a synthetic uniform distribution** dataset (total size: 20000). The test procedure is as follows: train a model such that its response to the two features is exactly the same. The xAI algorithm should detect symmetric features (equal values) and allocate them equal importance. The score is calculated as follows: 1 if the xAI detect the two features are symmetric. 0 if the difference in importance is above one unit. The test is classified in the **fidelity** category because it is a simple tree model that demonstrate inconsistencies in explanation [9].

cough_and_fever_10_90 answers the following question: *Can the xAI algorithm detect that 'Cough' feature is more important than 'Fever'?*. The trained model's equation is [Cough AND Fever]*80 + [Cough]*10. Cough should be more important than Fever globally. Locally for the case (Fever = yes, Cough = yes) the feature attribution of Cough should be more important. The test utilize **XGBRegressor** model trained on **a synthetic uniform distribution** dataset (total size: 20000). The test procedure is as follows: train a model with two features with unequal impact on the model. The feature with a higher influence on the output should be detected more important. The score is calculated as follows: Return 1 if Cough is more important otherwise 0. The test is classified in the **fidelity** category because it is a simple tree model that demonstrate inconsistencies in explanation due to the tree structure [9].

x0_plus_x1_distrib_non_uniform_stat_indep answers the following question: *Is the xAI able to explain the model correctly despite a non-uniform distribution of the data?*. The test demonstrate the effect of data distribution / causal inference. The test utilize **XGBRegressor** model trained on **a non-uniform and statistically independent** dataset (total size: 10000). The test procedure is as follows: Check if the explanation change when the distribution change. Check if non-uniform distributions affect the explanation. The score is calculated as follows: returns 1 if the two binary features obtain the same importance. The test is classified in the **stability** category because it assesses the impact of slightly changing the inputs [30].

x0_plus_x1_distrib_uniform_stat_dep answers the following question: *Is the xAI able to explain the model correctly despite a statistically-dependent distribution of the data?*. The test demonstrate the effect of data distribution / causal inference. The example was given in both [39] and [30]. The test utilize **XGBRegressor** model trained on **a uniform and statistically dependent** dataset (total size: 10000). The test procedure is as follows: Check if the explanation change when the distribution change. Check if statistically dependent distributions affect the explanation. The score is calculated as follows: returns 1 if the two binary features obtain the same importance. The test is classified in the **stability** category because To assess the impact of changing the inputs of f... This way, we are able to talk about a hypothetical scenario where the inputs are changed compared to the true features [30].

mnist answers the following question: *Is the xAI able to detect all dummy (constant and useless) pixels?*. The xAI algorithm should detect that important pixels are only in the center of the image. The test utilize **an MLP** model trained on **the MNIST** dataset (total size: 70000). The test procedure is as follows: simply train and explain the MLP model globally for every pixel. The score is calculated as follows: Return the ratio of constant pixels detected as dummy divided by the true number of constant pixels. The test is classified in the **stress** category because of the high number of input features. The test is adapted from [18].

fooling_perturbation_alg answers the following question: *Is the xAI affected by an adversarial attack against perturbation-based algorithms?*. Model-agnostic xAI algorithms that use feature perturbation methods might be vulnerable to this attack. The adversarial attack exploits a vulnerability to lower the feature importance of a specific feature. Setup: Let's begin by examining the COMPAS data set. This data set consists of defendant information from Broward Couty, Florida. Let's suppose that some adversary wants to mask biased or racist behavior on this data set. The test utilize **a custom function** model trained on **the**

COMPAS dataset (total size: 4629). The test procedure is as follows: The xAI algorithms need to explain the following corrupted model (custom function): if the input is from the dataset then the output is from a biased model. if not then the output is from a fair model. The score is calculated as follows: Return 1 if Race is the most important feature despite the adversarial attack. Score decreases while its rank decrease. The test is classified in the **fragility** category because fragility includes all adversarial attacks [40].

counterexample_dummy_axiom answers the following question: *Is the xAI able to detect unused input features?*. This is a counter example used in literature to verify that SHAP CES do not satisfy the dummy axiom while BSHAP succeed in this test. The test utilize **a custom function** model trained on **a synthetic** dataset (total size: 20000). The test procedure is as follows: Train a model with one extra feature B that is dummy. The score is calculated as follows: returns 1 if the dummy feature B obtain a null importance. The test is classified in the **simplicity** category because assigning an importance of zero to dummy feature reflect the model behavior (Fidelity) but also helps the data scientist to quickly understand the model (Simplicity).

a_and_b_or_c answers the following question: *Can the xAI algorithm detect that input feature 'A' is more important than 'B' or 'C'?*. This is a baseline test that the xAI should succeed in all cases. Model: A and (B or C). Goal: make sure that A is more important than B, C. Noise effect: even if the model output is not exactly equal to 1 still we expect the xai to give a correct answer. The test utilize **XGBRegressor** model trained on **a synthetic** dataset (total size: 20000). The test procedure is as follows: The model learns the following equation: A and (B or C). The explanation should prove that A is more important. The score is calculated as follows: If A is the most important feature then return 1. If A is the 2nd most important feature then return 0.5 i.e. 1- (1 / nb of feature more important than A). If A is the last one: return 0 (completely wrong). The test is classified in the **fidelity** category because of the same reason as cough and fever 10-90: A's effect on the output is higher than B or C.

B xAI Algorithms

archipelago [19] separate the input features into sets. all features inside a set interact and there is no interaction outside a set. ArchAttribute is an interaction attribution method. ArchDetect is the corresponding interaction detector. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature interaction (local explanation).

baseline_random [29] Output a random explanation. It is not a real explainer. It helps measure the baseline score and processing time. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation), Feature interaction (local explanation).

exact_shapley_values [5] is a permutation-based xAI algorithm following a game theory approach: Iteratively Order the features randomly, then add them to the input one at a time following this order, and calculate their expected marginal contribution [4]. The output is unique given a set of constrains defined in the original paper. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature importance (global explanation). The following information are required by the xAI algorithm: , A reference dataset (input only) , The model's predict function

kernel_shap [7] it approximates the Shapley values with a constant noise [30]. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , A reference dataset (input only) , The model's predict function

lime [1] it explains the model locally by generating an interpretable model approximating the original one. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , A reference dataset (input only) , The model's predict probability function , Nature of the ML task (regression/classification) , The model's predict function

maple [41] is a supervised neighborhood approach that combines ideas from local linear models and ensembles of decision trees [41]. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , AI model's structure , A reference dataset (input only) , The train set , The model's predict function

partition [7] Partition SHAP approximates the Shapley values using a hierarchy of feature coalitions. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , A reference dataset (input only) , The model's predict function

permutation is a shuffle-based feature importance. It permutes the input data and compares it to the normal prediction The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , input features , A reference dataset (input only) , The model's predict function

permutation_partition is a combination of permutation and partition algorithm from shap. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , input features , A reference dataset (input only) , The model's predict function

saabas explain tree based models by decomposing each prediction into bias and feature contribution components The xAI algorithm can explain tree-based models. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , AI model's structure

sage [18] Compute feature importance based on Shapley value but faster. The features that are most critical for the model to make good predictions will have large importance and only features that make the model's performance worse will have negative values.

Disadvantage: The convergence of the algorithm depends on 2 parameters: 'thres' and 'gap'. The algorithm can be trapped in a potential infinite loop if we do not fine tune them. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature importance (global explanation). The following information are required by the xAI algorithm: , True output of the data points to explain , A reference dataset (input only) , The model's predict function

shap_interaction [42] SI: Shapley Interaction Index. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature interaction (local explanation).

shapley_taylor_interaction [43] STI: Shapley Taylor Interaction Index. The xAI algorithm is model agnostic i.e. it can explain any AI model. The xAI algorithm can output the following explanations: Feature interaction (local explanation).

tree_shap [9] accurately compute the shap values using the structure of the tree model. The xAI algorithm can explain tree-based models. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , AI model's structure , A reference dataset (input only)

tree_shap_approximation is a faster implementation of shap reserved for tree based models. The xAI algorithm can explain tree-based models. The xAI algorithm can output the following explanations: Feature attribution (local explanation), Feature importance (global explanation). The following information are required by the xAI algorithm: , AI model's structure , A reference dataset (input only)

C Test Results

Table 2 contains test results without using any filter. **Tests** is the number of completed tests. **Time** is the average execution time per test. It informs the user about the relative difference in execution time between algorithms.

Table 2: Sample of the benchmark scores.

xAI algorithm	Time	Tests	Fidelity	Fragility	Stability	Simplicity	Stress test	Comprehen-
	[Seconds]		[%]	[%]	[%]	[%]	[%]	sibility [%]
baseline_random	< 1	22	12.9	50.0	30.7	0	33.3	31.7
exact_shapley_values	831	12	100.0	11.1	84.3	100.0		73.9
kernel_shap	328	15	100.0	11.1	85.6	100.0	100.0	79.3
lime	373	17	89.0	0	99.7	98.5	40.9	82.0
maple	119	17	60.0	11.1	100.0	0	25.5	49.2
partition	5	15	100.0	11.1	84.3	100.0	21.7	63.4
permutation	20	13	100.0	11.1	84.3	100.0	100.0	79.1
permutation_partition	20	13	100.0	11.1	84.3	100.0	100.0	79.1
saabas	< 1	11	60.0		50.6			55.1
sage	47	10	66.7	11.1	96.9	100.0	55.7	66.1
tree_shap	< 1	11	64.0		84.3			74.2
tree_shap_approximation	< 1	7	100.0		49.9			74.2